University of Niš
Faculty of Natural Sciences and Mathematics

Sladjana Lj. Miljković

ITERATIVE METHODS FOR COMPUTING GENERALIZED INVERSES OF MATRICES

Ph.D. Dissertation

Niš, January 2012

The complexity of nature and consequently the complexity of everyday life processes often make the mathematical models deterministically unsolvable. Moreover if such solutions do exist, usually a lot of resources are required to find them. Therefore, the idea of approximation has developed as irreplaceable tool for handling many problems.

In the Ph.D. dissertation a special attention is devoted to generalized inverses of matrices, which provide approximate solutions of many problems. They arise in various application in statistics, physics, economy etc. We developed different representation of these inverses and especially by using the idea of nonlinear optimization, we presented many effective algorithms for their computation. Most of the results are original and they are obtained during my studies at the Faculty of Natural Sciences and Mathematics, University of Nish. Some of the results are already published in eminent journals, and others are on their way.

First and foremost I would like to express my sincere gratitude to my supervisor Ph.D. Predrag Stanimirović for all the hope he has put on me, before I thought I could do any research at all. He has been actively interested in my work and has always been available to advise me. I am very grateful for his patience, motivation, enthusiasm, and immense knowledge that helped me in all the time of research and writing of the dissertation.

I also thank Ph.D. Dragan Djordjević and Ph.D. Dragana Cvetković-Ilić for the support and knowledge they gave me while passing the exams, and which provide me tools that turned out to be essential in my Ph.D. research.

Special appreciating is also extending to the other members of the committee and all professors who were supportive of me during the period of my education.
I sincerely thank you all and I'm looking forward to continuing interaction over the years ahead.

Last but not least, I sincerely thank all my friends and colleagues for being here for me anytime I needed.

Finally, I dedicate this dissertation to my parents who provide me with a great support, attention and trust, without which nothing of this would have been realized.

Contents

Chapter 1

Introduction

Generalized inverses of matrices, as their name indicates, are generalization of the notion of the ordinary matrix inverse. While the ordinary matrix inverse exists only for square nonsingular matrices, the generalized inverses exist for a much bigger set of matrices, and at the same time, each matrix has many generalized inverses. The broadest definition for a generalized inverse of a matrix says that it is a matrix which:

- exists for a larger class of matrices than the ordinary inverse does;
- has some properties of the ordinary inverse;
- for a given square nonsingular matrix it reduces to the ordinary inverse.

The idea for the definition of generalized inverses of matrices origins from the necessity of finding a solution of a given system of linear equations, which as a problem appears in many scientific and practical disciplines, such as: statistics, operational research, physics, economy, electrotechnics, and many others. Generalized inverses provide a simple way for obtaining a solution of the so called "ill-conditioned" linear problems.

The concept of generalized inverses is introduced for the first time by the scientist Fredhlom in 1903 [50], although it is considered that Gauss in 1809 implicitly indicated the ideas and the necessity of defining the notion of generalized inverse. Concretely, generalized inverses of matrices has appeared bit later in 1920 in the paper of the scientist E.H.Moore [96]. However, his work was not continued in the next 30 years, first of all because of the way the work was presented and the ambiguous notation. The research on this topic was carried on by the scientist Bjerhammar in 1950, while with the paper published by R. Penrose [106], the real evolution in the development of this area, has started. Also, many monographs has been written [20, 12, 137].

With the purpose of defining a generalized inverse which will have as many properties as the ordinary matrix inverse, different types of generalized inverses are introduced. The first such inverse, named by the scientists who worked on, is the Moore-Penrose inverse. Nowadays, besides the Moore-Penrose inverse, the theory of generalized inverses, recognizes many different types of generalized inverses, such as: the Drazin inverse, the group inverse, the weighted Moore-Penrose inverse, $\{i, j, k\}$-inverses, the Bott-Duffin inverse etc. Except for $\{i, j, k\}$-inverses, the main and one of the most important characteristic of all mentioned generalized inverses, is the fact that, for a given matrix they are the unique matrices which posses the properties typical for them. Additionally, these inverses posses one more good property, i.e., for a given matrix

$A \in \mathbb{C}^{m\times n}$, they can be represented on a unique way, by using the so called $A^{(2)}_{T,S}$-inverses, for appropriate choices of matrices T and S. This inverse is especially interesting, because it enables to foresee how its properties are reflected to a specific generalized inverse.

However, besides the all mentioned good properties of generalized inverses, they are not easily obtainable, especially for large dimensions which usually arise in practical examples. Similarly as the case with the ordinary inverse, it is almost impossible to obtain deterministically a generalized inverse of a matrix.

"All exact science is dominated by the idea of approximation." - Bertrand Russell

Penrose in his paper was the first who showed the close connection between the Moore-Penrose inverse and the least-squares solution problem of a system of linear equations. The last, represents a special case of the nonlinear optimization problems. Additionally, the discovered minimal properties of the solution of a linear system of equations, obtained with the usage of the Moore-Penrose inverse, brought to intensive usage of the optimization methods.

Usually, an optimization method is an iterative method for finding the minimum or maximum of some optimization problem. Namely, given an initial point x_0, an iterative sequence x_k is generated by a given iterative rule, such that the sequence x_k converges to the optimal solution of the problem. A typical behavior of an algorithm which is regarded as acceptable is that the iterates x_k move steadily towards the neighborhood of a local optimizer x, and then rapidly converge to the point x. When a given convergence rule is satisfied, the iteration will be terminated.

The theory of optimization represents a very important mathematical discipline and finds great application, not only in the theory of applied mathematics, but also in many practical disciplines such as: production, aviation, management, sociology, genetic etc. Moreover, the process of evolution, reveals that follows optimization. Although the optimization theory is a part of everyday life for a very long time, this science has faced an important development in the last five decades. The subject is involved in the process of finding optimal solution of problems which are defined mathematically, i.e., given a practical problem, the "best" solution to the problem can be found from lots of schemes by means of scientific methods and tools. It involves the study of optimality conditions of the problems, the construction of model problems, the determination of algorithmic method of solution, the establishment of convergence theory of the algorithms, and numerical experiments with typical problems and real life problems.

It is considered that the very first idea of optimization is presented by Queen Dido in 1000 BC while investigating the so called isoperimetric problems. Though, the first systematic presentation of the optimization theory has appeared in 1694, when John Bernoulli posed the Brachistochrone (Greek for "shortest time") problem. Later, these investigations formed the basis for the numerical optimization developed during and after the Second World War. In 1947 Dantzig proposed the simplex algorithm for solving linear optimization problems. Necessary conditions were presented by Kuhn and Tucker in 1950, and they formed a focal point for the development of the nonlinear optimization theory. Nonlinear optimization will be the most exploited in this Ph.D. dissertation with the purpose of effective calculation of generalized inverses. The most representative monographs in the optimization theory are [42, 85, 100, 130].

Since the optimization methods are iterative, it is our purpose to construct good algorithms for finding their solution. Under "good algorithm" it is assumed that it possesses the following properties:

- Robustness, since it should perform well on a wide variety of problems in their class, for all reasonable choices of the initial variables.
- Efficiency, since it should not require too much computer time or storage.
- Accuracy, since it should be able to identify a solution with precision, without being overly sensitive to errors in the data, or to the arithmetic rounding errors that occur when the algorithm is implemented on a computer.

All these goals are usually conflict, so, tradeoffs between the different types of good properties are central issues in numerical optimization.

The subject of investigation of the Ph.D. dissertation is the calculation of generalized inverses of matrices, as well as the connection between the generalized inverses of matrices with the optimization theory concepts. There are defined new iterative methods for solving optimization problems. A special attention is devoted on defining new iterative methods for generalized inverses calculation. The defined methods enables efficient calculation of generalized inverses, as well an analysis of their properties. The contribution of the Ph.D. dissertation is in the field of generalized inverses, as well, in the field of unconstrained optimization theory. This is claimed by the proposal of the new effective algorithms which can be compared to the most favorable ones in their disciplines. The main papers which served as initial motivation for the Ph.D. dissertation are [8, 43, 111].

1.1 Organization of the Ph.D. dissertaion

Generally, the Ph.D. dissertation is divided on three main parts (Chapter 2, Chapter 3 and Chapter 4). The first one is devoted, only, to the theory and concepts of unconstrained optimization. The second one deals, only, with the definition and properties of generalized inverses of matrices. And finally, the third one presents a unification of the previous parts. Namely, gathering the ideas from unconstrained optimization theory and the theory of generalized inverses of matrices; the third part deals with the calculation of the generalized inverses of matrices. Basically, the calculation is provided by using the optimization theory tools. In addition, in the last chapter, it is presented an application of the generalized inverses in the process of removing blur in images.

The detailed description of the consisting parts of the Ph.D. dissertation is as follows.

The next chapter consists of tree sections. In the first one, the common gradient methods from the theory of nonlinear unconstrained optimization theory are restated, such as: the steepest descent method, the rank-one update, BFGS method, Barzilai-Borwain method (BB) etc. [8, 85, 100]. Also, it is given a short analysis of their performances, efficiency, the reasons for their limiations, as well the possibility of their improvements.

From all previously mentioned methods, as the most important ones for the Ph.D. dissertation, are designated, the rank-one update and the BB method. These two methods belong to the class of the, so called, quasi-Newton methods, i.e., to the class of iterative methods which can be represented with the following iterative scheme

$$x_{k+1} = x_k - B_k g_k, \quad k = 1, 2, \ldots,$$

where B_k is an approximation of the Hessian inverse of the objective function, and g_k is its gradient. According to the rank-one update method, the matrix B_k at each iteration is updated with a rank-one matrix with the following formula

$$B_{k+1} = B_k + \alpha_k z_k z_k^T.$$

According to the BB method, the Hessian inverse is approximated with a scalar matrix $B_k = \gamma_k I$, and γ_k at each iteration is updated with the following formula

$$\gamma_{k+1} = \frac{s_k^T y_k}{y_k^T y_k},$$

where $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$. Although the efficiency for finding a solution with these two methods is incomparable, in favor of the BB method, the ideas from the both methods served as a motivation for obtaining a new iterative method for nonlinear unconstrained optimization [89].
The new method, named as Scalar Correction (SC method), is presented in the second section of this chapter. Namely, in the new method the Hessian inverse is approximated with a scalar matrix $B_k = \gamma_k I$, same as the BB method; but then in each iteration γ_k is rectified with appropriately chosen scalar number $\gamma_{k+1} = \gamma_k + a_k$. With the purpose of complete presentation of the method, in the first section of this chapter it is also presented the methodology of the, so called, the method of nonmonotone line search, which enables a global convergence of the BB method, as well as of the new method. The BB method accompanied with the nonmonotone line search technique is also known as the globalized BB method (GBB method) [111], and represents one of the most important methods in the theory of unconstrained optimization. As a continuation, in the second section, besides the global convergence of the new method, under some conditions it is proven its R-linear convergence. In the end, there are presented numerical examples which reveal the significantly greater efficiency of the new method with respect to the already recognized GBB method.

Motivated by the generalization of the steepest descent method on Hilbert spaces [97], as well as the methods investigated in [43], firstly using the properties of the Fréchet derivative, in the third section of the second chapter, there are presented new results which actually are generalization of the secant equation on Hilbert spaces [91]. This equation is a basis for the two-point stepsize gradient iterative methods (where belong the BB method and the SC method), which present one of the most effective methods in nonlinear optimization. Further, by using these ideas, as well as the ideas of the SC method quoted in the second section, it is constructed an algorithm for finding the Moore-Penrose inverse solution of the operator equation (1.1), where A is an operator between Hilbert spaces. In order to stick to the general organization of the Ph.D. dissertation, in Chapter 2, it is only shown the convergence of the method, without mentioning the Moore-Penrose inverse. The connection with the Moore-Penrose inverse is clarified later in Chapter 4.

In the first section of the third chapter, there are presented definitions and basic notion with respect to the most important types of generalized inverses of matrices, as well as their properties. Taking into account that the generalized inverses of matrices are closely connected to the solutions of a given system of matrix equations, first the results in which are determined the general solutions of matrix equations, are given, with a special stress of the particular case

$$Ax = b, \tag{1.1}$$

i.e., the system of linear equations. If it is not otherwise stated we use $A \in \mathbb{C}^{m\times n}$. Further, it is exposed the well known theory for the minimal properties of the Moore-Penrose inverse solution $A^{\dagger}b$, as well as its characteristic to be the least-squares solution of the system (1.1).
In this section, also are mentioned the Drazin inverse and the Drazin inverse solution of the system (1.1), which in the last years especially attract the attention of the scientists form this area. Namely, in the papers [20, 140, 142], there are presented the properties of the Drazin inverse solution for which, on some way, can be said that they represent an analogues results to the properties of the Moore-Penrose inverse solution. For example, in [20] it is shown that if $b \in \mathcal{R}(A^p)$, where $p = \text{ind}(A)$, the Drazin inverse solution is the unique solution of the system (1.1) which belongs to $\mathcal{R}(A^p)$. Wei in [140, 142], showed that the Drazin inverse solution of the system (1.1) is a solution of minimum P-norm, where P is the Jordan matrix obtained with the Jordan decomposition of the matrix A.

The second section, continues with new results for the properties of the Drazin inverse and the Drazin inverse solution [92, 94]. We start from the very specific case, i.e., we determine the solution $A^D b$ for a given system of linear equations $Ax = b$, where $b \in \mathcal{R}(A^p)$, $p = \text{ind}(A)$, and gradually we progress until we establish a general formula for calculating the matrix A^D. The obtained results are closely related to the minimal properties of the Drazin inverse and represent analogous results to the already known results for the Moore-Penrose inverse.

In the third section of the third chapter, new results are presented which refer to the $A^{(2)}_{T,S}$ inverses and $A^{(2)}_{T,S}$-inverse solutions of the system (1.1). These results consolidate the previously exposed properties of the Moore-Penrose inverse and the Drazin inverse, and on that way they enable those properties to be transferred to the other types of generalized inverses of matrices which can be represented via $A^{(2)}_{T,S}$ inverses.

The representaion of $\{2\}$-inverses, with the general form $F(GAF)^{-1}G$ is frequently applied tool in the numerical calculations. For example, this representation is investigated with the purpose of defining a deterministic representation of $A^{(2)}_{T,S}$ inverses [117], i.e., of the set $A\{2\}_s$ [125]. At the same time this representation has been used for the construction of the successive matrix squaring (SMS) method [126]. Based on these ideas in the last section of the third chapter there are defined full-rank representations of $\{2,3\}$ and $\{2,4\}$-inverses of a given matrix, with a given range and null space, as a special case of the full-rank representation of the $A^{(2)}_{T,S}$-inverse. Also, it is defined a full-rank representations of $A\{2,3\}_s$ i $A\{2,4\}_s$, as a special case of the full-rank representations of the sets $A\{2\}_s$, where $s \leq r$ and r is the rank A [128].

The fourth chapter consists of four sections as follows: In the first section we only give a brief introduction. In the second section, there are presented iterative methods for computing $\{1,3\}$-inverses, the Moore-Penrose inverse, $\{1,3\}$-inverse solution of the system (1.1), as well as its Moore-Penrose inverse solution [91]. In the third section of this chapter, it is presented a method for finding the Drazin-inverse solution of the system (1.1) [92]. And in the fourth section, there are presented iterative methods for finding $A^{(2)}_{T,S}$-inverse solution of the system (1.1).

More precisely, based on the results obtained in Chapter 2 and Chapter 3, as a continuation, it is presented a gradient method for computing $\{1,3\}$-inverses and the Moore-Penrose inverse of a given matrix. In this part, also it is presented the convergence of the introduced methods, as well as numerical results which claim the effectiveness.
Usually, for a given iterative process which converges to $\{1,3\}$-inverse solutions of an equation of the type (1.1), the following question arises: for a given initial iteration, to which $\{1,3\}$-

inverse solution the process would converge. As an addition, in the Ph.D. dissertation we give an answer to the opposite case, i.e., for a given $\{1,3\}$-inverse solution, which initial iteration should be chosen in order the process to converge to that solution.

In this second section, also, it is made a small deviation from the theory of matrices, and is given a short overview of the theory of the Moore-Penrose inverse of linear operators on Hilbert spaces. The theory exposed in this part is necessary for the introduced results. Here, it is presented the well known Steepest descent method for solving linear operator equation on Hilbert spaces [97], which served as a basic motivation for our results.

The properties of the Drazin inverse and the Drazin inverse solution already mentioned, lead to a new iterative process for finding the Drazin inverse solution of the system (1.1). This method is a gradient method, and is first such method for computing the Drazin inverse solution. In the literature, the most famous methods for computing the Drazin inverse solution are: the projection methods, the Krylov subspace methods and the semi-iterative methods.

The fourth section includes new iterative methods, such as method for finding $A^{(2)}_{T,S}$-inverse solution of the system (1.1) which is a generalization of the method for finding the Drazin inverse solution presented in the previous section. Further a method, motivated by [14, 126], so called, the displacement SMS method for finding $A^{(2)}_{T,S}$ inverses of a *Toeplitz* matrix [90]. This method, actually presents a generalization of the results from [14], where by using the, so called, orthogonal displacement operator, it is given a very useful algorithm for computing an ordinary inverse of a given Toeplitz matrix. In the literature, there are different algorithms for finding the Moore-Penrose inverse, the Drazin inverse etc. which use a complicated strategy of choosing an appropriate initial iteration. The importance of the new method is that it enables a unique way for computing different types of generalized inverses of a Toeplitz matrix.

This section, also, includes the obtained results with respect to the computation of $\{2,3\}$ and $\{2,4\}$-inverses [128]. These methods are based on the full-rank representation, introduced in the previous chapters, and the SMS algorithm for computing $A^{(2)}_{T,S}$-inverses introduced in [126]. For all previously mentioned iterative method, the convergence results and numerical examples are presented.

In the fifth chapter we illustrate an application of the Moore-Penrose inverse in the field of image processing. Images are produced to memorize useful information, but unfortunately the presence of the blur is unavoidable. Motion blur is the effect of the relative motion between the camera and the scene during image exposure time.

In this chapter it would be introduced a direct method for restoring images which are blurred by a uniform motion [93], and a computational method for restoring images which are blurred by a uniform or non–uniform motion [129]. It is based on appropriate adaptations of the partitioning and block-partitioning method introduced in [58] and [135]. The performed adjustments of the algorithms and the specific structure of the blurring matrix brought to a numerical results which are competitive to the most favorable algorithms in this field.

In the last chapter it is given a conclusion with respect to the obtained results which are exposed in the Ph.D. dissertation. It is given a short overview of their scientific importance. This chapter is ended with a given outline regarding the ideas for further investigation and directions for a possible results which will arise as an extension.

Chapter 2

Unconstrained optimization

The purpose of this chapter is to present new methods in the theory of unconstrained optimization. Before their presentation in the last two sections, it is given a short overview of the basics and history that lead towards the new methods. Some of the gradient iterative schemes are presented, as well as the characteristics of the non-monotone line search technique. This technique in many situation provides a global convergence of the gradient iterative methods.

2.1 Line search iterative methods

Throughout this chapter, we consider the unconstrained minimization problem

$$\min f(x), \quad x \in \mathbb{R}^n, \tag{2.1}$$

where $\mathbb{R}^n$ denotes the n-dimensional Euclidian space and $f : \mathbb{R}^n \to \mathbb{R}$ is a given objective function that we want to minimize.

For convenience, the following notations are used:

$$g(x) := \nabla f(x),\ g_k = g(x_k) := \nabla f(x_k),\ G_k = G(x_k) := \nabla^2 f(x_k),$$

where $\nabla f(x)$ denotes the gradient of f and $\nabla^2 f(x)$ denotes the Hessian of f in x. For a given vectors $x, y \in \mathbb{R}^n$, by $x^T y$ we denote the Frobenious inner product, and $\|x\| = \sqrt{x^T x}$. By $\mathbb{N}_0$, we denote the set of all non-negative integers.

The most desirable solution of the problem is its global minimizer, i.e., a point x^* for which $f(x^*) \leq f(x)$ for all $x \in \mathbb{R}^n$. However, the fastest optimization algorithms seek only a local solution, i.e., a point at which the objective function is smaller than at all other feasible points in its vicinity.

The mathematical tool used to study minimizers of smooth functions is the Taylor's theorem. This theorem would be central also in the process of defining the quasi-Newton methods. For completness we restate the theorem [85].

Theorem 2.1.1. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and that $\Delta x \in \mathbb{R}^n$. Then we have that*

$$f(x + \Delta x) \approx f(x) + g(x + t\Delta x)^T \Delta x$$

for some $t \in (0,1)$. Moreover, if f is twice continuously differentiable, we have that

$$g(x+\Delta x) \approx g(x) + \int_0^1 G(x+t\Delta x)\Delta x dt \tag{2.2}$$

and that

$$f(x+\Delta x) \approx f(x) + g(x)^T\Delta x + \frac{1}{2}\Delta x^T G(x+t\Delta x)\Delta x.$$

In the sequel, we give some basic theorems for the necessity and sufficiency, a point x^* to be a local minimizer of the function f [85].

Theorem 2.1.2. *(First-order necessary conditions) If x^* is a local minimizer of the function f, which is continuously differentiable in an open neighborhood of x^*, then $g(x^*) = 0$.*

Theorem 2.1.3. *(Second-order necessary conditions) . If x^* is a local minimizer of the function f and G is continuous in an open neighborhood of x^*, then $g(x^*) = 0$ and $G(x^*)$ is positive semi-definite.*

Theorem 2.1.4. *(Second-order sufficient conditions) Suppose that G is continuous in an open neighborhood of x^* and that $g(x^*) = 0$ and $G(x^*)$ is positive definite. Then x^* is a strict local minimizer of the function f, i.e., $f(x^*) < f(x)$ for all x in some neighborhood of x^*.*

The most frequently used general iterative scheme for solving the problem (2.1) is given by

$$x_{k+1} = x_k + t_k d_k, \quad k = 0, 1, \ldots, \tag{2.3}$$

where x_{k+1} is a new iterative point, x_k is a current iterative point, $t_k > 0$ is a steplength, and d_k is a search direction (see, for example, [22, 85, 130]).

The search direction d_k is usually required to satisfy the descent condition

$$g_k^T d_k < 0, \tag{2.4}$$

which guarantees that d_k is a descent direction of $f(x)$ at x_k [3, 100]. In order to ensure the global convergence, it is sometimes required that d_k satisfies the sufficient descent condition

$$g_k^T d_k \leq -c_1\|g_k\|^2, \tag{2.5}$$

as well as the inequality

$$\|d_k\| \leq c_2\|g_k\|, \tag{2.6}$$

where c_1, c_2 are some positive constants [59].

After the search direction d_k is established, we need to determine a steplength t_k which will ensure sufficient decrease of f. For this purpose one possible approach is the line search methodology, which searches along the direction d_k from the current point x_k and determines the steplength t_k. There are two main line search strategies: exact line search methods and inexact line search methods.

In the exact line search the stepsize t_k is chosen according to the one dimensional minimization problem

$$f(x_k + t_k d_k) = \min_{t>0} f(x_k + t d_k). \tag{2.7}$$

In some special cases (for example quadratic problems) it is possible to compute the steplength t_k in (2.7) analytically, but it is usually computed to approximately minimize f along the ray $\{x_k + td_k : t > 0\}$. Since the exact minimization in (2.7) is expensive and of no practical value, the inexact line search algorithms are preferable. This class of algorithms can be divided regarding the changes of the objective function f. In fact, if we force monotone decreasing of the function f in each iteration then we have the monotone line search methods; otherwise, we consider the nonmonotone line search techniques. The first one generates a limited number of trial steplengths until it finds one that provides a sufficient decrease of the objective function f. Many inexact line search algorithms have been proposed: Armijo, Goldstein, Wolfe, Powell, Fletcher and others (see [4, 33, 115, 148]). Among them, the most popular is the algorithm known as backtracking line search (Armijo line search).

The original nonmonotone line search strategy is proposed in [59], and is given in more details further in this chapter.

2.1.1 The steepest descent method

The most obvious direction which satisfies the descent conditions is, of course, the steepest descent or negative gradient direction. It moves along the negative gradient of the function f, i.e.,

$$d_k = -g_k, \tag{2.8}$$

for all $k = 1, 2, \ldots$, and the iterative scheme (2.3) becomes the following iterative scheme

$$x_{k+1} = x_k - t_k g_k, \quad k = 0, 1, \ldots, \tag{2.9}$$

where t_k is obtained by means of the one-dimensional optimization problem (2.7) [22].

In other words, from the point x_k we search along the negative gradient direction $-g_k$, in order to find the minimum of the function, on this line. The method proved to be effective for functions very well conditioned. On the other hand, despite the optimal property (2.7), the steepest descent method converges slowly and it is badly affected by ill-conditioning, and thus being of no practical value (see [1, 49]). Even for quadratic functions the steepest descent method behaves increasingly badly when the conditioning number of the matrix deteriorates. More precisely, by the proof given by Luenberg [85], it follows that for a given strongly convex quadratic function, the steepest descent iterative scheme converges to the minimum of the function at a linear rate, i.e., there exists a constant $M \in (0, 1)$ such that

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq M,$$

for all k sufficiently large. This type of convergence is also called Q-linear convergence [130].

2.1.2 Newton's method

The next step for improving the algorithm of steepest descent is the direction, which besides the information given from the gradient of the function, also, uses the information from its Hessian. The Newton's direction is probably one of the most important directions. It is derived from the second order Taylor's series, from which is obtained the following

$$d_k = -G_k^{-1} g_k, \tag{2.10}$$

The Newton's direction is the most applicable when the original function and its quadratic model, determined from the second order Taylor's series do not differ too much. In order to be satisfied the condition (2.4), the Hessian inverse of the function, should be a positive definite matrix, for in this case we have

$$g_k^T d_k = -g_k^T G_k^{-1} g_k = -\|g_k\|^2_{G_k^{-1}} \leq 0.$$

The usual steplength associated with the Newton's direction is the unit step $t_k = 1$ for each k. However, in the cases when G_k^{-1} is not defined or if the Newton's direction is not a descent direction, to make this method applicable, it is chosen a steplength t_k such that it modifies the direction d_k in order to make it a descent direction, but still pertaining the information from the Hessian.

The Newton's direction methods are very fast and they converge quadratically (or Q-quadratically) to a local minimizer of the function, i.e., there exists a positive constant M such that

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} \leq M,$$

for all k sufficiently large. The main drawback of the Newton's direction is the need to compute the Hessian inverse, which is very expensive operation and acquire a lot of memory space.

2.1.3 Quasi-Newton's method

The weakness of the Newton's direction, discussed previously, is overcome with the intoduction of the quasi-Newton's methods. As we already indicated, this direction is obtained from the second order Taylor series of the function f, i.e., from the equation (2.2) we can obtain

$$g(x + \Delta x) = g(x) + G(x)\Delta x + \int_0^1 (G(x + t\Delta x) - G(x))\Delta x dt.$$

Because the function g is continuous, the last term of the equation is $o(\|\Delta x\|)$ [130]. By setting $x = x_k$ and $\Delta x = x_{k+1} - x_k$, we obtain

$$g_{k+1} = g_k + G_k(x_{k+1} - x_k) + o(\|x_{k+1} - x_k\|),$$

or equivalently

$$G_k(x_{k+1} - x_k) \approx g_{k+1} - g_k. \tag{2.11}$$

The technique used in quasi-Newton's method for choosing a direction, is based on choosing an approximation of the Hessian inverse that follows the property given in (2.11), and which is also called a *secant equation*. Based on the previous consideration, if we denote

$$B_k := G_k,\ s_k := x_{k+1} - x_k,\ y_k := g_{k+1} - g_k$$

we can impose the following condition

$$s_k = B_{k+1} y_k. \tag{2.12}$$

The general iterative scheme (2.3) becomes

$$x_{k+1} = x_k - B_k\, g_k. \tag{2.13}$$

According to the research in [18], we can observe three alternatives for the matrix B_k in (2.13): scalar matrix, corresponding to all cases $B_k = \gamma_k I$, where γ_k is a scalar; diagonal matrix, satisfying $B_k = \text{diag}\,(\gamma_1, \ldots, \gamma_n)$ and full matrix.

- **Symmetric-rank-one update method**

The symmetric-rank-one update method approximates the Hessian inverse by a full matrix of rank one. Namely, the Hessian inverse is corrected by the following formula

$$B_{k+1} = B_k + \frac{(s_k - B_k y_k)(s_k - B_k y_k)^T}{y_k^T (s_k - B_k y_k)^T}. \tag{2.14}$$

The previous formula can be obtained by imposing the condition that the correction of the Hessian inverse should be done by the following formula

$$B_{k+1} = B_k + a_k z_k z_k^T,$$

where the constant a_k and the vector z_k are such that the matrix B_{k+1} satisfies the secant equation.

The main drawback of rank-one-update method, besides calculation of a full matrix, is that the updating formula (2.14) preserves positive definiteness only if $y_k^T(s_k - B_k y_k) > 0$. Also, even if it is positive, it may be small, which can lead to numerical difficulties.

- **BFGS method**

Similarly, BFGS method, named after its inventors, Broyden, Fletcher, Goldfarb, and Shanno, goes one step further and tries to correct the Hessian inverse by adding two symmetric matrices of rank one. The final update is given with the following formula which is defined by

$$B_{k+1} = B_k - \frac{B_k y_k y_k^T B_k}{y_k^T B_k y_k} + \frac{s_k s_k^T}{s_k^T y_k}. \tag{2.15}$$

Whenever the initial approximation B_0 is positive definite, BFGS update generates positive definite approximations and $s_k^T y_k > 0$.

On the contrary of rank-one-update method, the BFGS method, preserves positiveness at each step.

- **Barzilai-Borwein method (BB method)**

It is well-known that the Newton-like methods, which need to store and compute a full matrix B_k at each iteration (for example, symmetric-rank-one update method or BFGS method), are unsuitable to solve large-scale optimization problems in many cases, because their approximations to the Hessian or to its inverse are usually dense. The storage and computational requirements grow in proportion to n^2, and become excessive for large n. It is necessary to modify and extend these methods to make them suitable for large problems [100].

By taking a scalar approximation to the Hessian inverse

$$B_k = \gamma_k I \approx G_k^{-1}, \ \gamma_k > 0, \tag{2.16}$$

the quasi-Newton method (2.13) reduces to the gradient descent iterative scheme

$$x_{k+1} = x_k - \gamma_k g_k.$$

In 1988, Barzilai and Borwein [8] proposed a gradient method (called BB method), in which the steplength along the negative gradient direction is computed from a two-point approximation to the secant equation required in quasi-Newton methods.

This scheme is also analysed in [8], where the steplength γ_k is computed after the minimization

$$\gamma_k = \arg\min_t \|t^{-1}s_{k-1} - y_{k-1}\|^2,$$

which yields

$$\gamma_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}, \tag{2.17}$$

where $s_k := x_{k+1} - x_k$, $y_k := g_{k+1} - g_k$. Observing the symmetric case, Barzilai and Borwein also obtained the following choice for γ_k

$$\hat{\gamma}_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}. \tag{2.18}$$

2.1.4 Non-monotone line search technique

While computing the steplength γ_k it is desirable to accomplish a tradeoff between a choice which would give a substantial reduction of the objective function and which would not require a lot of computational time. A popular inexact line search condition is the Wolfe's condition which stipulates that γ_k satisfies the following inequality

$$f(x_k + td_k) \leq f(x_k) + c_1 t g_k^T d_k, \tag{2.19}$$

for some constant $c_1 \in (0, 1)$. However, the Wolf's condition is not enough by itself to ensure that the algorithm makes reasonable progress. As a complement of this condition it is given the second Wolfe's condition, also called curvature condition

$$g(x_k + td_k)^T d_k \geq c_2 g_k^T d_k, \tag{2.20}$$

for some constant $c_2 \in (c_1, 1)$.

The authors in [59], for the choice of the steplength, imposed that the function value of each new iteration satisfies the Armijo's condition when it comes to the maximum function value achieved in a predefined number of previous iterations. Therefore, this line search procedure can be viewed as a generalization of the Armijo's rule [4] since it allows an increase in the function values without affecting the convergence properties. Due to its practical and theoretical relevance in the global convergence analysis for unconstrained optimization, the nonmonotone line search technique applied to the BB gradient method has attracted considerable attention in recent years (see [38, 39, 48]).

The original nonmonotone line search strategy is based on the usage of a positive integer M. In each iteration the stepsize t is obtained in such a way as to fulfil the inequality

$$f(x_k + td_k) \leq \max_{0 \leq j \leq m(k)} f(x_{k-j}) + \sigma t g_k^T d_k, \tag{2.21}$$

where $m(0) = 0$, $0 \leq m(k) \leq \min\{m(k-1)+1, M-1\}$, and σ is a parameter from the Armijo's rule [4].

Here, we restate the algorithm which will be used as a common line search procedure for both the BB gradient method as well as for the gradient descent method introduced in [89].

Algorithm 2.1.1 The nonmonotone line search.

Input: Objective function $f(x)$, the search direction d_k, numbers $0 < \sigma < 0.5$, $\beta \in]0, 1[$, $a \in \mathbb{R}$ and $m \in \mathbb{N}_0$.

1: $t = a$.

2: While $f(x_k + td_k) > \max\limits_{0 \le j \le m} f(x_{k-j}) + \sigma t g_k^T d_k$, take $t = t\beta$.

3: Return $t_k = t$.

In the case $m \equiv 0$, the nonmonotone line search mentioned above reduces to the Armijo's line search. For our purposes, the real parameter a is an initial trial stepsize obtained by the means of (2.17) for the BB method, or by means of (2.34) for the SC method (obtained later in Section 2).

The authors in [59] prove the global convergence of the nonmonotone line search method for a twice continuously differentiable function where the search direction satisfies conditions (2.5) and (2.6). In his work, Dai gives another proof for the global convergence of the nonmonotone line search method under the conditions (2.5), (2.6) and the Lipschitz continuity of the gradient of the objective function [39]. Moreover, the author shows that it is possible to weaken the conditions (2.5) and/or (2.6) imposed on the search direction d_k and still to preserve the global convergence.

For the sake of completeness we restate the main results from [39].

Definition 2.1.1. *A function f is Lipschitz continuous if there exists $L > 0$ such that*

$$\|f(y) - f(z)\| \le L\|y - z\|, \tag{2.22}$$

for all $y, z \in \mathbb{R}^n$.

Proposition 2.1.1. *Suppose that the function f is bounded below on $\mathbb{R}^n$ and that its gradient g is Lipschitz continuous. Consider any iterative method (2.3), where d_k is a descent direction and t_k is obtained by Algorithm 2.1.1. Then, for any $l \ge 1$,*

$$\max_{1 \le i \le M} f(x_{Ml+i}) \le \max_{1 \le l \le M} f(x_{M(l-1)+i}) + \sigma \max_{0 \le i \le M-1} (t_{Ml+i} g_{Ml+i}^T d_{Ml+i}).$$

Further, we have that

$$\sum_{l \ge 1} \min_{0 \le i \le M-1} \{|g_{Ml+i}^T d_{Ml+i}|, \frac{(g_{Ml+i}^T d_{Ml+i})^2}{\|d_{Ml+i}\|^2}\} < \infty.$$

Proposition 2.1.2. *Suppose that the function f is bounded below on $\mathbb{R}^n$ and that its gradient g is Lipschitz continuous. Consider any iterative method (2.3), where d_k satisfies (2.5) and (2.6), and t_k is obtained by Algorithm 2.1.1. Then there exists a constant c_3 such that*

$$\|g_{k+1}\| \le c_3 \|g_k\|, \quad \textit{for all } k.$$

Further, we have that

$$\lim_{k \to \infty} \|g_k\| = 0.$$

Definition 2.1.2. *Let $x_k \in \mathbb{R}^n$ be any sequence that converges to x^*. If*

$$0 < \limsup_{k\to\infty} \|x_k - x^*\| < 1$$

then x_k is said to be R-linearly convergent to x^*.

The following proposition shows that any iterative method using the nonmonotone line search is R-linear convergent for uniformly convex functions [39].

Proposition 2.1.3. *Suppose that f is a smooth and uniformly convex function. Consider any iterative method (2.3), where d_k satisfies (2.5), (2.6) and t_k is obtained by Algorithm 2.1.1. Then, there exist constants $c_4 > 0$ and $c_5 \in (0, 1)$ such that*

$$f(x_k) - f(x^*) \leq c_4 c_5^k [f(x_1) - f(x^*)].$$

Using a globalization strategy, based on the nonmonotone line search technique introduced in [59], Raydan [111] proved the global convergence of the BB method for nonquadratic functions. The numerical results reported in [111] show that the resulting algorithm is competitive to several well known conjugate gradient algorithms for large scale unconstrained optimization. Due to its simplicity and numerical efficiency, the two-point stepsize gradient method has initiated many studies (see, for example, [37, 110, 111]).

2.2 Scalar correction method (SC method)

2.2.1 Basic ideas

This section aims at presenting a new gradient descent method [89] by the means of the general iterative scheme of the quasi-Newton type, in which a scalar matrix approximates the inverse Hessian. The initial trial stepsize is obtained as a consequence of the secant equation which is based on two successive iterative points. With this in mind, it is proposed a new two-point stepsize gradient descent method; the method is called the Scalar Correction method (the SC method). It is shown that the new algorithm, to which the technique of the nonmonotone line search is applied, is comparable with preferred GBB method, i.e., the BB method accompanied by the nonmonotone line search (introduced by Raydan [111]). Moreover, the presented numerical results demonstrate the fact that the SC algorithm – combined with the nonmonotone line search – outperforms the GBB method.

First, it is introduced a new two-point stepsize gradient descent method, which is obtained from approximation of the Hessian inverse by a scalar matrix and the quasi-Newton property. The initial trial stepsize γ_{k+1} in the new algorithm is determined after updating the inverse Hessian B_k at each step by the following formula

$$B_{k+1} = B_k + a_k I = (\gamma_k + a_k) I = \gamma_{k+1} I.$$

Thus, the correction of the prior trial stepsize γ_k is defined by

$$\gamma_{k+1} = \gamma_k + a_k. \tag{2.23}$$

The parameter $a_k = a_k(s_k, y_k) \in \mathbb{R}$ depends on two successive iterative points $x_k,\ x_{k+1}$ and the corresponding gradients $g_k,\ g_{k+1}$. If we apply the secant equation to the chosen approximations x_k and x_{k+1}, our task is to find a scalar γ_{k+1} satisfying

$$s_k = \gamma_{k+1} y_k. \tag{2.24}$$

This means that the ideal case would be for the vector s_k to be a scalar multiple of y_k. Using (2.23), we can express the last equation in the following way

$$s_k - \gamma_k y_k - a_k y_k = 0.$$

In order to obtain an approximate solution, we search for a vector that minimizes the norm of the left hand side of the previous equation. Similarly as in the BB method, two symmetric solutions can be obtained by minimizing the norms

$$\min_a \|a^{-1}(s_k - \gamma_k y_k) - y_k\|^2 \quad \text{and} \quad \min_{\hat{a}} \|(s_k - \gamma_k y_k) - \hat{a} y_k\|^2. \tag{2.25}$$

Consequently, we get dual solutions

$$a_k = \frac{(s_k - \gamma_k y_k)^T (s_k - \gamma_k y_k)}{(s_k - \gamma_k y_k)^T y_k} \quad \text{and} \quad \hat{a}_k = \frac{(s_k - \gamma_k y_k)^T y_k}{y_k^T y_k}, \tag{2.26}$$

which correspond to the choices (2.17) and (2.18) in the BB method, respectively. Taking into account the advantages of the choice (2.17) with respect to the symmetric case (approved in numerical experiments) [37], we decide to use a_k. In other words, we expect better performances of the induced algorithm with respect to the algorithm based on the choice of $\hat{a}_k$. If $(s_k - \gamma_k y_k)^T y_k = 0$, we choose $a_k = 0$.

After the substitution of a_k in (2.23), in the case $(s_k - \gamma_k y_k)^T y_k \neq 0$ we obtain

$$\gamma_{k+1} = \gamma_k + \frac{(s_k - \gamma_k y_k)^T (s_k - \gamma_k y_k)}{(s_k - \gamma_k y_k)^T y_k} = \gamma_k + \frac{\|s_k - \gamma_k y_k\|^2}{(s_k - \gamma_k y_k)^T y_k}. \tag{2.27}$$

In the opposite case, $(s_k - \gamma_k y_k)^T y_k = 0$, we get $\gamma_{k+1} = \gamma_k$. Let us define an auxiliary vector r_k as follows

$$r_k = s_k - \gamma_k y_k. \tag{2.28}$$

After the substitution in (2.27), we obtain

$$\gamma_{k+1} = \gamma_k + \frac{\|r_k\|^2}{y_k^T r_k} = \frac{s_k^T r_k}{y_k^T r_k}, \tag{2.29}$$

for $y_k^T r_k \neq 0$, where the second equality follows after few algebraic transformations.

Remark 2.2.1. *After the minimization* (2.25) *is performed, the scalar correction* (2.23) *is used in the computation of the parameter* γ_{k+1} *and the corresponding scalar matrix* $B_{k+1} = \gamma_{k+1} I$. *These facts serve as the motivating factor for using the name "scalar correction" for the introduced method.*

Our main idea in choosing the initial trial steplength after each iteration is to ease its selection and choose it to be as large as possible, but still with a steplength produced by the secant property. Small initial trial steplength, in the cases where greater steplength would lead the process closer to the solution, undoubtedly increases the number of iterations. On the other hand, too big trial stepsize choice is prevented by the usage of the nonmonotone line search. For this purpose, we observe additional two trial steplengths as follows

$$\gamma_{k+1}^{BB} = \frac{s_k^T y_k}{y_k^T y_k}, \tag{2.30}$$

$$\gamma_{k+1}^{SS} = \frac{\|s_k\|}{\|y_k\|}. \tag{2.31}$$

The initial trial stepsize γ_{k+1}^{BB} is determined by (2.18), while γ_{k+1}^{SS} follows directly from the secant equation (2.24) after we equalize the norms on the left and the right hand side (see also [116]). It is important to point out that each stepsize given by (2.29) – (2.31) is derived from a two-point approximation to the secant equation, underlying quasi-Newton methods. The following lemma gives a few useful relations among the three stepsizes in order to ensure the most appropriate choice among them.

Lemma 2.2.1. *For trial stepsizes given by* (2.29) − (2.31)*, we have the following inequalities:*

$$\gamma_{k+1}^{BB} \leq \gamma_{k+1}^{SS} \leq \gamma_{k+1}, \qquad y_k^T r_k > 0, \tag{2.32}$$

$$\gamma_{k+1} \leq \gamma_{k+1}^{BB} \leq \gamma_{k+1}^{SS}, \qquad y_k^T r_k \leq 0. \tag{2.33}$$

Proof. If $y_k^T r_k > 0$, using the Cauchy-Schwarz inequality $|y_k^T r_k| \leq \|y_k\| \cdot \|r_k\|$, after a few transformations we get:

$$\begin{aligned}
\gamma_{k+1} &= \gamma_k + \frac{\|r_k\|^2}{y_k^T r_k} \\
&\geq \gamma_k + \frac{\|r_k\|}{\|y_k\|} \\
&= \gamma_k + \frac{\|s_k - \gamma_k y_k\|}{\|y_k\|} \\
&\geq \gamma_k + \frac{\|s_k\| - \gamma_k \|y_k\|}{\|y_k\|} \\
&= \gamma_{k+1}^{SS}.
\end{aligned}$$

Another application of the Cauchy-Schwarz inequality leads to

$$\gamma_{k+1} \geq \gamma_{k+1}^{SS} \geq \frac{s_k^T y_k}{y_k^T y_k} = \gamma_{k+1}^{BB}.$$

In the case of $y_k^T r_k < 0$, since the following holds

$$|y_k^T r_k|^2 \leq \|y_k\|^2 \cdot \|r_k\|^2,$$

taking into account the negative sign of $y_k^T r_k$, we have

$$\frac{y_k^T r_k}{\|y_k\|^2} \geq \frac{\|r_k\|^2}{y_k^T r_k}.$$

Therefore,

$$\begin{aligned}
\gamma_{k+1} &= \gamma_k + \frac{y_k^T r_k}{\|y_k\|^2} + \frac{\|r_k\|^2}{y_k^T r_k} - \frac{y_k^T r_k}{\|y_k\|^2} \\
&= \frac{\gamma_k \|y_k\|^2 + y_k^T r_k}{\|y_k\|^2} + \frac{\|r_k\|^2}{y_k^T r_k} - \frac{y_k^T r_k}{\|y_k\|^2} \\
&= \frac{y_k^T(\gamma_k y_k + r_k)}{\|y_k\|^2} + \frac{\|r_k\|^2}{y_k^T r_k} - \frac{y_k^T r_k}{\|y_k\|^2} \\
&\leq \frac{y_k^T s_k}{y_k^T y_k} = \gamma_{k+1}^{BB} \\
&\leq \gamma_{k+1}^{SS}.
\end{aligned}$$

In the last case, taking into account (2.28), the assumption $y_k^T r_k = 0$ is equivalent to

$$\gamma_k = \frac{y_k^T s_k}{\|y_k\|^2} = \gamma_{k+1}^{BB}.$$

Now, since $a_k = 0$, we have

$$\gamma_{k+1} = \gamma_k = \gamma_{k+1}^{BB} \leq \gamma_{k+1}^{SS},$$

and the proof is completed. □

In accordance with results shown in Lemma 2.2.1, in our method we choose the initial trial stepsize to be equal with the largest values in two different cases (2.32) and (2.33). Therefore, we get

$$\gamma_{k+1}^{SC} := \begin{cases} \frac{s_k^T r_k}{y_k^T r_k}, & y_k^T r_k > 0, \\ \frac{\|s_k\|}{\|y_k\|}, & y_k^T r_k \leq 0. \end{cases} \tag{2.34}$$

Next, we present the general algorithm for the two-point stepsize gradient method in conjunction with the nonmonotone line search, which can be applied for the large scale unconstrained optimization. This algorithm belongs to a class of the quasi-Newton methods with the nonmonotone line search where the approximation of the Hessian inverse is presented by an appropriate scalar matrix.

Algorithm 2.2.1 General gradient algorithm with the nonmonotone line search

Input: Objective function $f(x)$, chosen initial point $x_0 \in dom(f)$, positive integer M and real constants $0 < \xi_1 < \xi_2$.

1: Set $k = 0$, compute $f(x_0)$, $g_0 = \nabla f(x_0)$ and use $\gamma_0 = 1$, $m(0) = 0$.
2: If test criteria is fulfilled, then stop the iteration; otherwise, go to the next step.
3: Find t_k using Algorithm 2.1.1 with input values $d_k = -g_k$, $m = m(k)$ and $a = \gamma_k$.
4: Compute $x_{k+1} = x_k - t_k g_k$, $f(x_{k+1})$, g_{k+1}, $s_k := x_{k+1} - x_k$, $y_k := g_{k+1} - g_k$.
5: Compute the initial trial stepsize γ_{k+1} according to the given method. If $\gamma_{k+1} > \xi_2$ or $\gamma_{k+1} < \xi_1$, set $\gamma_{k+1} = 1$.
6: Set $k = k + 1$, $m(k) = \min\{m(k-1) + 1, M - 1\}$, and go to the step 2.
7: Return x_{k+1} and $f(x_{k+1})$.

In order to keep the sequence γ_k bounded, we use the test criteria as in Step 5.

If we want to embed the BB method in Algorithm 2.2.1, in Step 5 we have to determine γ_{k+1} according to (2.17) or (2.18). We choose (2.17) in accordance with the known fact that the choice (2.18) often performs worse than (2.17) in practical computations [37], which leads to the choice of the stepsize

$$\gamma_{k+1}^{BB} = \frac{s_k^T s_k}{s_k^T y_k}. \tag{2.35}$$

Therefore, we have the following algorithm.

Algorithm 2.2.2 The BB gradient algorithm with the nonmonotone line search

Step 5: Compute the initial trial stepsize γ_{k+1} using (2.35). If $\gamma_{k+1} > \xi_2$ or $\gamma_{k+1} < \xi_1$, set $\gamma_{k+1} = 1$.
Other steps are the same as in Algorithm 2.2.1.

Now, we are in a position to present a new gradient descent algorithm with nonmonotone line search, which is a particular case of the general Algorithm 2.2.1, and differs only in Step 5, where the initial trial stepsize is computed.

Algorithm 2.2.3 SC gradient descent algorithm with the nonmonotone line search

Require: Objective function $f(x)$, chosen initial point $x_0 \in dom(f)$, positive integer M and real constants $0 < \xi_1 < \xi_2$.
Step 5: Compute the initial trial stepsize γ_{k+1} using (2.34). If $\gamma_{k+1} > \xi_2$ or $\gamma_{k+1} < \xi_1$, set $\gamma_{k+1} = 1$.
Other steps are the same as in Algorithm 2.2.1.

Further in the text, the GSC method will stand for the SC method with the nonmonotone line search.

Remark 2.2.2. *Every iteration of the SC method requires only $O(n)$ floating point operations and a gradient evaluation, as in the BB method. In this manner, both the computational and storage complexity of the SC method and the BB method are equivalent.*

2.2.2 Convergence properties

Before proving the global convergence of the new algorithm, we state the following lemma which gives the lower bound for the stepsize t_k.

Lemma 2.2.2. *Suppose that the gradient of $f(x)$ is Lipschitz continuous. For any iterative method* (2.3)*, where $d_k = -g_k$ and t_k is obtained from the nonmonotone line search* (2.21) *after the initial trial stepsize is determined by* (2.34) *– steps* 3 *and* 5 *in Algorithm* 2.2.1 *– we have the following inequality*

$$t_k \geq \min\left\{\xi_1,\ \frac{\beta(1-\sigma)}{L}\right\}, \qquad k = 0, 1, 2, \ldots \tag{2.36}$$

Proof. If the initial trial stepsize $t_k = \gamma_k$ satisfies the nonmonotone condition (2.21), since $\gamma_k \geq \xi_1$, (2.36) holds.

Otherwise, if the initial trial stepsize $t_k = \gamma_k$ does not satisfy the nonmonotone condition (2.21), then we have the following inequalities

$$f\left(x_k + \frac{t_k}{\beta}d_k\right) > \max_{0\leq j\leq m(k)} f(x_{k-j}) + \sigma\frac{t_k}{\beta}g_k^T d_k \geq f(x_k) + \sigma\frac{t_k}{\beta}g_k^T d_k,$$

which imply

$$f\left(x_k + \frac{t_k}{\beta}d_k\right) - f(x_k) > \sigma\frac{t_k}{\beta}g_k^T d_k.$$

After applying the Mean Value Theorem on the left hand side of the above inequality, we get that there exists $\theta \in]0,1[$ such that

$$\frac{t_k}{\beta}g\left(x_k + \theta\frac{t_k}{\beta}d_k\right)^T d_k > \sigma\frac{t_k}{\beta}g_k^T d_k.$$

Hence,

$$g\left(x_k + \theta\frac{t_k}{\beta}d_k\right)^T d_k > \sigma g_k^T d_k.$$

If we subtract $g_k^T d_k$ from both hand sides of the above inequality, and apply the Cauchy-Schwartz inequality as well as the Lipschitz condition (2.22), we have

$$-(1-\sigma)g_k^T d_k < \left(g\left(x_k + \theta\frac{t_k}{\beta}d_k\right) - g_k\right)^T d_k \leq \frac{L}{\beta}t_k\|d_k\|^2,$$

which, after using $d_k = -g_k$, implies

$$t_k > -\frac{\beta(1-\sigma)}{L}\frac{g_k^T d_k}{\|d_k\|^2} = \frac{\beta(1-\sigma)}{L}.$$

The last inequality completes the proof of the Lemma. □

The proof of the global convergence is quite easy to understand and read taking into consideration the particular choice of the search direction $d_k = -g_k$ and the initial trial steplength obtained according to (2.34). Also, we use some results from the global convergence theorems (see [39, 59]) as auxiliary results in the proof.

Theorem 2.2.1. *Suppose that $f(x)$ is bounded below on the level set $L(x_0) = \{x \in \mathbb{R}^n | f(x) \leq f(x_0)\}$ and that its gradient be Lipschitz continuous on an open set C that contains $L(x_0)$. Then, for the iterative method (2.3), where $d_k = -g_k$ is a descent direction and t_k is determined by Algorithm 2.2.3, we have the following*

a) The norm of the gradients vanishes at infinity, i.e.

$$\lim_{k\to\infty}\|g_k\| = 0. \tag{2.37}$$

b) If the number of stationary points of $f(x)$ in $L(x_0)$ is finite, the sequence $\{x_k\}$ converges.

Proof. The authors in [39] (Theorem 2.1 which follows the same conditions) using Lipschitz continuity of $g(x)$, prove that there exists a real constant

$$c_3 = 1 + c_2\xi_2 L > 1$$

such that the following holds

$$\|g_{k+1}\| \leq c_3\|g_k\|, \quad \forall k \in \mathbb{N}. \tag{2.38}$$

For the choice $d_k = -g_k$ we have (2.6) which is satisfied for $c_2 = 1$. Therefore, (2.38) holds for $c_3 = 1 + \xi_2 L$.

Let $l(k)$ be a positive integer such that:

$$\begin{aligned} k - m(k) \leq l(k) \leq k, \\ f(x_{l(k)}) = \max_{0 \leq j \leq m(k)} f(x_{k-j}). \end{aligned} \tag{2.39}$$

The authors in [59] show that the sequence $\{f(x_{l(k)})\}_{k\in\mathbb{N}}$ is non-increasing. Also, they prove the following statement

$$\lim_{k\to\infty} t_{l(k)-1} g_{l(k)-1}^T d_{l(k)-1} = 0.$$

After the replacement $d_k = -g_k$ and usage (2.36), we obtain $\lim_{k\to\infty} \|g_{l(k)-1}\| = 0$. As a straight implication we have

$$\lim_{k\to\infty} \|g_{l(kM)-1}\| = 0. \tag{2.40}$$

Now, according to (2.38) we obtain

$$\|g_{kM+i}\| \leq c_3^{2M+1}\|g_{l(kM)-1}\|, \quad i = 0, 1, \ldots, M, \tag{2.41}$$

which, finally, together with (2.40) implies (2.37), and the proof of a) is completed.

To prove the statement b) we start with the fact that every accumulation point of the sequence $\{x_k\}$ is also a stationary point of $\{f(x)\}$, which follows straight from (2.37) and the fact that $L(x_0)$ is bounded. Since the number of stationary points of $f(x)$ is finite, the relation

$$\|x_{k+1} - x_k\| \leq t_k\|d_k\| \leq \xi_2\|g_k\| \to 0$$

implies that the sequence $\{x_k\}$ converges. □

Theorem 2.2.2. *For smooth and uniformly convex functions, the iterative method* (2.3)*, in which $d_k = -g_k$ is a descent direction and t_k is determined by Algorithm* 2.2.3*, is R-linearly convergent.*

Proof. Follows directly from Proposition 2.1.3. □

2.2.3 Numerical results

In this section, we report some numerical results obtained from testing the new gradient descent method (the SC method) with respect to the BB method, both combined with the nonmonotone line search technique. The codes, based on Algorithms 2.2.2 and 2.2.3, are written in the visual C++ programming language and tested on a Workstation Intel Core duo 1.6 GHz. The parameters used in nonmonotone line search algorithm are $\sigma = 0.0001$ and $\beta = 0.8$ which means that we accept a small decrease in f predicted by linear approximation at the current point. Additional parameters for both algorithms are $\xi_1 = 10^{-5}$, $\xi_2 = 10^5$ and $M = 10$.

The following 40 test functions, given in extended or generalized form, are taken from [2] and used as a large scale test problems. The number of variables considered for each test problem is contained in the set $D = \{100, 500, 1000, 2000, 3000, 5000, 7000, 8000, 10000, 15000\}$. Stopping criteria are:

$$\|g_k\| \leq 10^{-6} \quad \text{and} \quad \frac{|f(x_{k+1}) - f(x_k)|}{1 + |f(x_k)|} \leq 10^{-16}.$$

In the table below, we present the total number of iterations, the total CPU time, and the total number of function evaluations for each test problem, tried out for the above mentioned 10 different numbers of variables. Additionally, in the last two columns the optimal function values, obtained by the both methods in the case when the dimension of the problems equals 100, are given.

Table 2.2.1. Summary numerical results for the *GBB* and *GSC* methods tested on 40 large scale test functions.

Test function	No. of iterations		CPU time		No. of funct. eval.		Func. min.	
	GBB	GSC	GBB	GSC	GBB	GSC	GBB	GSC
Extended Freud. and Roth	417	590	2.98	3.653	1734	1970	0	2E-16
Extended Rosenbrock	858	440	8.995	3.621	4253	1430	0	0
Extended White and Holst	619	580	5.026	5.088	1998	1840	7.66E-13	0
Extended Beale	444	693	5.322	8.261	1028	1676	2E-13	2.45E-13
Extended Penalty	506	501	6.792	7.058	2066	2056	75	75
Perturbed Quadratic	18072	6893	155.231	45.182	62401	18899	4.6E-15	2.6E-15
Raydan 1	8197	3894	52.089	20.557	27040	10556	505	505
Raydan 2	75	75	0.387	0.307	160	160	100	100
Diagonal 1	7066	3313	43.479	21.855	15798	8874	-15706.7	-15706.7
Diagonal 2	10525	4871	111.168	37.292	41163	13778	15.74	15.74
Diagonal 3	8397	3457	83.809	29.903	28014	9501	-4605.8	-4605.8
Hager	950	887	6.808	6.228	2523	2059	-653.08	-653.08
Generalized Tridiagonal 1	280	276	2.169	2.387	640	632	97.21	97.21
Extended Tridiagonal 1	372	359	1.683	2.074	864	808	2.73E-9	2.73E-9
Extended Three Exponential Terms	100	90	0.996	0.715	300	280	127.96	127.96
Diagonal 4	70	65	0.543	0.48	330	320	0	2E-16
Diagonal 5	50	50	0.59	0.495	110	110	69.31	69.31
Extended Himmelblau function	135	120	1.481	1.324	410	380	2.4E-15	0
Generalized PSC1	7344	4688	193.87	105.322	26099	13002	98.72	98.72
Extended PSC1	150	160	2.293	2.447	460	480	38.66	38.66
Extended Powell	10	10	0.03	0.015	30	30	0	0
Extended Block Diagonal BD1	178	166	1.918	1.854	416	392	8.2E-15	8.2E-15

Extended Maratos	200	305	2.058	2.965	730	1000	-50.03	-50.03
Quadratic Diagonal Perturbed	5891	5494	51.934	39.652	23921	17054	3.1E-12	3.1E-12
Quadratic QF1	18175	7407	145.512	45.121	61728	20355	-0.005	-0.005
Extended Quadratic Penalty QP1	171	189	3.714	4.09	740	776	390.06	390.06
Quadratic QF2	13383	7504	140.136	74.198	45740	20972	-1.00	-1.00
Extended EP1	48	48	1.152	1.215	356	356	793.18	793.18
Extended Tridiagonal-2	401	485	1.51	1.808	892	1112	38.58	38.58
ARWHEAD	140	170	4.119	4.293	710	797	-2.1E-14	-2.1E-14
Almost Perturbed Quadratic	20336	7131	170.088	41.511	69489	19472	1.32E-13	1.32E-13
ENGVAL1	333	392	4.963	5.309	836	956	109.09	109.09
QUARTC	212	212	1.09	1.12	474	474	5.97E-9	5.97E-9
Diagonal 6	75	75	0.371	0.262	160	160	1.11E-14	1.11E-14
LIARWHD	579	677	12.104	11.465	2615	2425	0	0
Generalized Quartic GQ1	216	223	2.995	2.714	525	538	1.56E-14	0
Diagonal 7	70	70	0.497	0.465	160	160	-81.68	-81.68
Diagonal 8	70	70	0.729	0.589	190	190	-48.05	-48.05
Diagonal 9	6553	5329	32.528	29.37	22417	14986	-15346.2	-15346.2
HIMMELH	20	20	0.121	0.091	60	60	-62.5	-62.5

We can see in Table 2.2.1 that the GSC method overcomes the GBB method for 21 test functions, both algorithms have the same number of iterations for 9 test functions, while the GBB method requires fewer iterations in 10 cases. Similarly, observing the number of function evaluations needed for the program execution, we have that the GSC method is a better choice in 22 test functions, while the GBB method shows better performances in 9 cases. For 9 test problems both algorithms achieve the same number of function evaluations.

Additionally, we present a table which shows the number of numerical experiments, out of 400, for which the GSC and GBB achieved the minimum number of iterations, the minimum CPU time, and minimum number of function evaluations, respectively.

Table 2.2.3. Comparative performances of the *GSC* and *GBB* methods in 400 numerical experiments.

Performance indicators	GSC	GBB	both
Number of iterations	199	102	99
CPU time (sec)	186	84	130
Number of function evaluations	208	100	92

The last column, named *both*, represents the number of experiments for which the observed indicators of both algorithms have the same values. The great improvement of our new method in comparison to the GBB method in all three observed indicators is more than evident.

Not only is the number of problems in which the GSC overcomes the GBB method greater than in the converse case, but the difference in observing indicators is also significant. Table 2.2.1 shows the functions 'Perturbed Quadratic', 'Raydan 1', 'Diagonal 1, 2, 3', 'Quadratic QF2' and 'Almost Perturbed Quadratic' which serve as an illustration of the above stated fact. It is not difficult to see the substantial difference favoring the GSC method in each of the observing indicators. The GSC method is almost two times better when it comes to the number of iterations as well as about 2.2 times better than the GBB in observing the CPU time and the number of function evaluations. As the confirmation of these facts, the following table

presents the average performances of observed characteristics of the two algorithms obtained from testing 400 problems.

Table 2.2.4. Average numerical outcomes for 40 test functions tried out on 10 numerical experiments in each iteration.

Average performances	GSC	GBB
Number of iterations	169.9	329.2
CPU time (sec)	1.43	3.16
Number of function evaluations	477.7	1123.9

- **Benchmarking of optimization software**

The better performances of the GSC method compared to the GBB method can also be confirmed by using the so-called performance profile of a given metric, introduced in [40]. In the sequel, first, we give the basic ideas of the benchmarking process introduced in [40].

Benchmark results are generated by running a solver on a set of problems and recording information of interest such as the number of function evaluations, the computing time etc. The benchmarking methodology, presented in [40], is based on the, so called, *performance profile* as a means to evaluate and compare the performance of the set of solvers S on a test set P.

Suppose that n_s solvers (algorithms) are compared, on a set of n_p problems. The parameter that is of intrest is the CPU time (number of iterations) as a performance measure. For each problem p and solver s, it is defined the variable

$$t_{p,s} = \text{CPU required to solve problem } p \text{ by solver } s.$$

$$i_{p,s} = \text{number of iterations to solve problem } p \text{ by solver } s.$$

Further, it is compared the performance on problem p by solver s with the best performance by any solver on this problem by using the following ratio:

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}}.$$

This ratio is called the performance ratio (the respective quantity can be defined for the number of iterations $i_{p,s}$). The final assessment is done by measuring the performance of the solver by using the following parameter,

$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\},$$

which is the probability for solver $s \in S$ that a performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio. The function ρ_s is the (cumulative) distribution function for the performance ratio.

Following the notations given in the paper [40] we have that the number of solvers is $n_s = 2$ (the GBB and GSC) and the number of numerical experiments is $n_p = 400$. For the performance metrics we use the CPU time and the number of iterative steps. The quantity $r_{p,s}$ becomes

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in \{GBB, GSC\}\}}$$

Finally, the performance of the solver s is measured with the quantity $\rho_s(\tau)$.

Figure 2.2.1 shows the performance profiles of the methods regarding the CPU time and the number of iterations. The performance profile regarding the number of function evaluations is not illustrated, since it is very similar to the performance profile regarding the number of iterations.

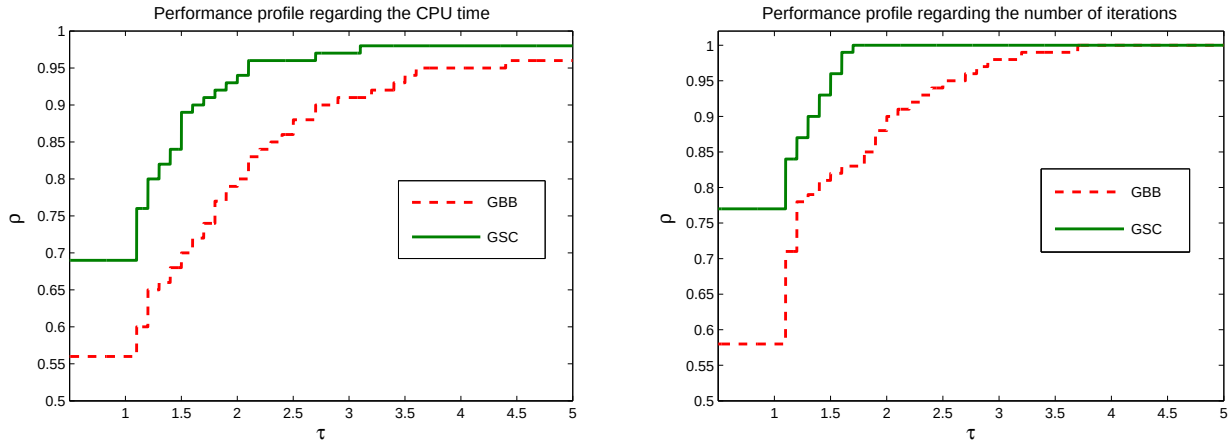

Figure 2.2.1. (Left) Performance profile for the GBB and GSC methods regarding the CPU time.

(Right) Performance profile for the GBB and GSC methods regarding the number of iterations.

It is clear from Figure 2.2.1 that the GSC has better performances. The probability of being the optimal solver regarding the CPU time (number of iterative steps) is in favour of the GSC algorithm, which is confirmed by $\rho_{GSC}(1) = 0.69 > \rho_{GBB}(1) = 0.56$ $(0.77 > 0.58)$. Also, for both metrics and all values of τ, the probability $\rho_{GSC}(\tau)$ of our algorithm is always greater than the probability $\rho_{GBB}(\tau)$ of the GBB method.

The differences between the two algorithms in the growth of the dimensions are shown in Figure 2.2.2. Namely, we propose a quotient as a measurement of the difference between two solvers regarding the dimension d. The quotient, named *ratio*, is defined as follows

$$\text{ratio}(d) = \frac{\sum_p t_{p,GSC,d}}{\sum_p t_{p,GBB,d}}, \quad d \in D,$$

where $t_{p,s,d}$ represents the CPU time (number of iterations) corresponding to the problem p of dimension d and the solver s.

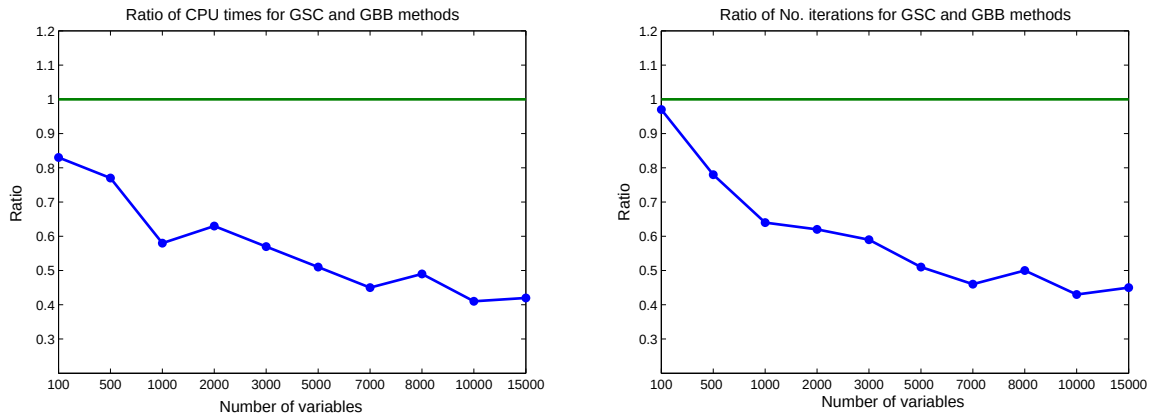

Figure 2.2.2. (Left) The ratio of CPU times for the GSC and GBB methods.
(Right) The ratio of number of iterations for the GSC and GBB methods.

It is obvious from Figure 2.2.2 that, regardless of the dimension of the problem, the GSC algorithm is always better than the GBB algorithm (the ratio is always less than one). Moreover, the ratio for larger dimensions is improved with respect to the ratio for lower dimensions of the problem.

2.3 Least-squares solutions on Hilbert spaces

Solving the system of linear operator equations is an interesting problem. Many different techniques are developed to solve this problem [12, 43, 65, 95, 97, 132]. Since it usually reduces to an optimization problem, it is of our interest and will be considered in the present section.

Let $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$, where $\mathcal{L}(\mathcal{H}, \mathcal{K})$ denotes the space of linear bounded operators between Hilbert spaces $\mathcal{H}$ and $\mathcal{K}$. The equation

$$Ax = b, \ b \in \mathcal{K} \tag{2.42}$$

may or may not have a solution, depending on whether b is in the range $\mathcal{R}(A)$ of A or not. Even if $b \in \mathcal{R}(A)$ the solution need not to be unique. In cases where $b \notin \mathcal{R}(A)$ or the solution is not unique, it is possible to compute vector which minimizes the quadratic functional $q(x) = \frac{1}{2}\|Ax - b\|^2$.

Definition 2.3.1. *A vector $\hat{x} \in \mathcal{H}$ is called a least-squares solution of the operator equation (2.42) if and only if $\|A\hat{x} - b\| = \inf\{\|Ax - b\| : x \in \mathcal{H}\}$.*

Nashed in [97] minimized the functional $q(x)$ in order to find a solution of the operator equation $Ax = b$, where $A \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ is such that $\mathcal{R}(A)$ is closed. The minimization of the functional $q(x)$ is accomplished by using the iterative scheme

$$x_{k+1} = x_k - \gamma_k g_k, \tag{2.43}$$

where

$$g_k = A^*Ax_k - A^*b \quad \text{and} \quad \gamma_k = \frac{\|g_k\|^2}{\|Ag_k\|^2}.$$

Since this method is actually the steepest descent method, the stepsize α_k is chosen according to the strategy that guarantees the most rapid decrease of $\|Ax_{k+1} - b\|$. The linear convergence of the method to a least-squares solution of the equation $Ax = b$ is established in [97], for an arbitrary initial approximation $x_0 \in \mathcal{H}$.

As it is also stressed before, despite the optimal property, the steepest descent method behaves poorly, except for very well conditioned functions and converges slowly (see [1, 49]). In some particular cases, such as quadratic functions, it is possible to compute the steplength γ_k analytically. Even for these functions the steepest descent method behaves increasingly badly when the condition number of the matrix deteriorates. The authors in [112] stressed out that the poor behavior of the steepest descent method is due to the optimal Cauchy choice of the steplength γ_k and not to the choice of the search direction (direction of the negative gradient).

The two-point stepsize gradient method introduced in [8] is proved to be more effective and thus, preferable over the classical steepest descent method both in theoretical investigations and in real computations.

In this section, we develop a two-point stepsize gradient descent method for finding least-squares solution of an operator equation. It is based on the idea of secant equation, introduced by now for finite dimensional spaces [91]. In order to generalize this notion we use the idea of Fréchet differentiable operator on Hilbert spaces.

For the sake of completeness, we restate main known facts about the Fréchet derivative from [34, 78].

2.3.1 Fréchet derivative

Definition 2.3.2. *Let $\mathcal{H}$ and $\mathcal{K}$ be Hilbert spaces and $\mathcal{U} \subset \mathcal{H}$ is an open set. Let $f : \mathcal{U} \to \mathcal{K}$ be an operator and $x \in \mathcal{U}$. If there is a bounded linear operator $g : \mathcal{H} \to \mathcal{K}$ such that*

$$\lim_{\|h\| \to 0} \frac{\|f(x+h) - f(x) - g(h)\|}{\|h\|} = 0,$$

$h \in \mathcal{H}$, we say that f is Fréchet differentiable at x, or simply differentiable at x; g is called the (Fréchet) derivative of f at x and we will denote it by $Df(x) \in \mathcal{L}(\mathcal{H}, \mathcal{K})$.

Definition 2.3.3. *An operator f is $(n+1)$-times differentiable on $\mathcal{U}$ if it is n times differentiable on $\mathcal{U}$ and for each x in $\mathcal{U}$ there exists a continuous multilinear map g of $(n+1)$ arguments such that the limit*

$$\lim_{\|h_{n+1}\| \to 0} \frac{\|D^n f(x+h_{n+1})(h_1, \ldots, h_n) - D^n f(x)(h_1, \ldots, h_n) - g(h_1, \ldots, h_{n+1})\|}{\|h_{n+1}\|} = 0 \qquad (2.44)$$

exists uniformly for $h_1, h_2, \ldots, h_n$ in bounded sets in $\mathcal{H}$. In this case, g is the $(n+1)$st derivative of f at x.

Proposition 2.3.1. *Let D be a convex subset of $\mathcal{H}$ and f is $(n+1)$-times Frechet differentiable operator on D. Then if x and $x+p$ are given elements in D we have*

$$f(x+p) = \sum_{k=0}^{n} \frac{1}{k!} D^{(k)} f(x) \underbrace{(p, p, \ldots, p)}_{k \text{ times}} + w(x, p)$$

where

$$\|w(x,p)\| \leq \frac{1}{(n+1)} \sup_{t\in[0,1]} \|D^{(n+1)}f(x+tp)\|\|p\|^{n+1}.$$

Definition 2.3.4. [86] *Let $\mathcal{H}$ be Hilbert space, $\mathcal{U} \subset \mathcal{H}$ is an open set and $f : \mathcal{U} \to \mathbb{R}$ is a given differentiable functional. The gradient of the functional f is the linear map*

$$\nabla f : \mathcal{U} \to \mathcal{H} \quad \textit{such that} \quad \langle \nabla f(x), h\rangle = Df(x)(h),$$

where $Df(x)(h)$ means the linear map $Df(x)$ applied to the vector $h \in \mathcal{H}$.

The existence and uniqueness of such linear map follows straight from the application of the Riesz representation theorem of the linear bounded operator $Df(x) : \mathcal{H} \to \mathbb{R}$.

Definition 2.3.5. *Let $\mathcal{H}$ be Hilbert space, $\mathcal{U} \subset \mathcal{H}$ is an open set and $f : \mathcal{U} \to \mathbb{R}$ is a given twice differentiable functional. The Hessian of the functional f is the linear operator*

$$\nabla^2 f \in \mathcal{L}(\mathcal{U} \times \mathcal{H}, \mathcal{H}) \quad \textit{such that} \quad \langle \nabla^2 f(x,p), h\rangle = D^2 f(x)(p,h),$$

where $D^2f(x)(p,h)$ means the linear map $D^2f(x)(p)$ applied to the vector $h \in \mathcal{H}$.

The existence and uniqueness of such linear map follows straight from the Riesz representation theorem for the linear bounded operator $D^2f(x)(p) : \mathcal{H} \to \mathbb{R}$.

Since the necessary results have been established. next we are going to obtain the analog to the secant equation for an operator on a Hilbert space.

2.3.2 Secant equation on Hilbert spaces

Let $\mathcal{H}$ and $\mathcal{K}$ be given Hilbert spaces and $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ be given operator such that $\mathcal{R}(A)$ is closed. Let the functional $q : \mathcal{U} \to \mathbb{R}$ be defined by

$$q(x) = \frac{1}{2}\|Ax - b\|^2 = \frac{1}{2}\langle Ax - b, Ax - b\rangle. \tag{2.45}$$

Based on Proposition 2.3.1 we have that the first order Taylor series expansion for the operator Dq is

$$Dq(x) = Dq(x_{k+1}) + D^2q(x_{k+1})(x - x_{k+1}) + w(x, x - x_{k+1}), \tag{2.46}$$

which is equivalent to

$$Dq(x)(h) = Dq(x_{k+1})(h) + D^2q(x_{k+1})(x - x_{k+1}, h) + w(x, x - x_{k+1})(h) \quad \text{for all} \quad h \in \mathcal{H}.$$

From Definition 2.3.4 and Definition 2.3.5 we have

$$\langle \nabla q(x) - \nabla q(x_{k+1}), h\rangle - \langle \nabla^2 q(x_{k+1}, x - x_{k+1}), h\rangle = w(x, x - x_{k+1})(h) \quad \text{for all} \quad h \in \mathcal{H}.$$

Taking $h = \nabla q(x) - \nabla q(x_{k+1}) - \nabla^2 q(x_{k+1}, x - x_{k+1})$ and having in mind that $w(x, x - x_{k+1})$ stands for second order residual we get

$$\nabla q(x) - \nabla q(x_{k+1}) \approx \nabla^2 q(x_{k+1})(x - x_{k+1}).$$

Setting $x = x_k$, $g_k = \nabla q(x_k)$, $H_k = \nabla^2 q(x_k)$, $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$, we get

$$H_{k+1}(s_k) \approx y_k, \tag{2.47}$$

which is analogue to the secant equation on $\mathbb{R}^n$.

2.3.3 SC method for solving linear operator equation on Hilbert spaces

For the purpose of minimizing the functional q defined by (2.45), on an open set $\mathcal{U} \subset \mathcal{H}$, we analyze the gradient iterative scheme (2.43) where $\gamma_k > 0$ is a stepsize, which will be appropriately determined. Taking into account that $g_k = \nabla q(x_k) = A^*(Ax_k - b)$, which is not difficult to show from (2.45), we consider an iterative process given in the following general form

$$x_{k+1} = x_k - \gamma_k A^*(Ax_k - b), \tag{2.48}$$

for the purpose of finding least-squares solutions of the equation $Ax = b$. The importance in choosing appropriate stepsize in order to obtain convergence as well as good computational performance is obvious.

In the rest of this section, we use the stepsize determined according to the idea of the scalar correction method, introduced in [89]. For the sake of completeness, we extend the basic ideas for SC method with respect to Hilbert spaces. In order to obtain an appropriate stepsize we use the information of the stepsize obtained in the previous step and try to correct it by adding some scalar. To determine that scalar properly, the idea of a two-point approximation to the secant equation is used, similarly as in BB method. The final choice for the stepsize is done by relaxing the stepsize as much as it is possible in view of two additional steplengths, which are also obtained from the first order secant equation. Now, we want to approximate H_{k+1} with some identical operator $I : \mathcal{H} \to \mathcal{H}$ multiplied by the real parameter $\frac{1}{\gamma_{k+1}}$ such that $H_{k+1}(h) = \frac{1}{\gamma_{k+1}} h$ holds for all $h \in \mathcal{H}$.

After the steplength γ_k is computed, we observe the following correction

$$\gamma_{k+1} = \gamma_k + a_k. \tag{2.49}$$

in order to find the stepsize γ_{k+1} for the next iteration. According to the secant equation (2.47) we get

$$s_k - \gamma_k y_k - a_k y_k \approx 0.$$

Hence, we have the problem of minimizing the function

$$\min_a \|a^{-1}(s_k - \gamma_k y_k) - y_k\|^2, \tag{2.50}$$

which yields the solution

$$a_k = \frac{\langle s_k - \gamma_k y_k, s_k - \gamma_k y_k \rangle}{\langle s_k - \gamma_k y_k, y_k \rangle}, \tag{2.51}$$

in the case $\langle s_k - \gamma_k y_k, y_k \rangle \neq 0$. Otherwise, we choose $a_k = 0$. After the substitution of (2.51) in (2.49), applying the notation $r_k = s_k - \gamma_k y_k$ and few algebraic transformations we obtain

$$\gamma_{k+1} = \gamma_k + \frac{\|r_k\|^2}{\langle y_k, r_k \rangle} = \frac{\langle s_k, r_k \rangle}{\langle y_k, r_k \rangle}, \tag{2.52}$$

if $\langle r_k, y_k \rangle \neq 0$ and $\gamma_{k+1} = \gamma_k$, otherwise. Finally, comparing this steplength with two additional steplengths

$$\gamma_{k+1}^{BB} = \frac{\langle y_k, s_k \rangle}{\|y_k\|^2}, \qquad \gamma_{k+1}^{SS} = \frac{\|s_k\|}{\|y_k\|}, \quad k \geq 0,$$

we do the following choice (see also [89])

$$\gamma_{k+1}^{SC} = \begin{cases} \frac{\langle s_k, r_k\rangle}{\langle y_k, r_k\rangle}, & \langle y_k, r_k\rangle > 0 \\ \frac{\|s_k\|}{\|y_k\|}, & \langle y_k, r_k\rangle \leq 0 \end{cases}, \quad k \geq 0. \tag{2.53}$$

Corresponding algorithm is defined as follows.

Algorithm 2.3.1 SC method for computing least-squares solutions

Input: An operator $A : \mathcal{H} \to \mathcal{K}$ such that $\mathcal{R}(A)$ is closed, chosen initial point $x_0 \in \mathcal{H}$ and real positive constants $0 < \varepsilon \ll 1, \quad 0 < \xi_1 \ll \frac{2(1-\varepsilon)}{\|A\|^2}$.

1: Set $k = 0$, compute $q(x_0)$, g_0 and use $\gamma_0 = 1$.
2: If test criteria are fulfilled then go to Step 7; otherwise, go to the next step.
3: Compute x_{k+1} using (2.48), $q(x_{k+1})$, g_{k+1}, $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$.
4: Determine $\xi_2^{(k+1)} = 2(1-\varepsilon)\frac{\|g_{k+1}\|^2}{\|Ag_{k+1}\|^2}$.
5: Compute the stepsize γ_{k+1} using (2.53). If $\gamma_{k+1} < \xi_1$ or $\gamma_{k+1} > \xi_2^{(k+1)}$, set $\gamma_{k+1} = \xi_2^{(k+1)}$.
6: Set $k := k + 1$ and go to Step 2.
7: Return x_{k+1} and $q(x_{k+1})$.

Proposition 2.3.2. *Algorithm* 2.3.1 *is well defined, i.e. the interval* $(\xi_1, \xi_2^{(k)})$, $k \geq 1$ *is non-empty, for a chosen real constant* ξ_1. *Additionally, the sequence of stepsizes* $(\gamma_k)_k$ *is positive and it is bounded by real constants.*

Proof. Since

$$\xi_2^{(k)} = 2(1-\varepsilon)\frac{\|g_k\|^2}{\|Ag_k\|^2} \geq \frac{2(1-\varepsilon)}{\|A\|^2}$$

one can always choose constant ξ_1 such that

$$0 < \xi_1 \ll \frac{2(1-\varepsilon)}{\|A\|^2} \leq \xi_2^{(k)}.$$

Thus, the interval $(\xi_1, \xi_2^{(k)})$ is non empty in each iteration.

Taking into account that $\|Ax\| \geq j(A) \cdot \|x\|$, where $j(A) = \inf_{\|x\|=1} \|Ax\|$, it is not difficult to show that $\xi_2^{(k)} \leq 2 \cdot j(A)^{-2}$, $k \geq 1$. Thus, the following sequence of inequalities holds

$$0 < \xi_1 \leq \gamma_k \leq \xi_2^{(k)} \leq 2 \cdot j(A)^{-2}, \quad k \geq 1,$$

which is a verification of the second statement. □

2.3.4 Convergence properties

The following theorem shows the convergence of the method given by Algorithm 2.3.1.

Theorem 2.3.1. *Let* $\mathcal{H}$ *and* $\mathcal{K}$ *be given Hilbert spaces and* $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ *be an operator such that* $\mathcal{R}(A)$ *is closed. The sequence* $(x_k)_k$ *determined by Algorithm* 2.3.1 *converges.*

Proof. For $x_{k+1} = x_k - \gamma_k A^*(Ax_k - b)$, we compute

$$\begin{aligned} q(x_{k+1}) &= \frac{1}{2}\|Ax_{k+1} - b\|^2 \\ &= \frac{1}{2}\left(\langle Ax_k - b, Ax_k - b\rangle - 2\langle Ax_k - b, \gamma_k Ag_k\rangle + \langle \gamma_k Ag_k, \gamma_k Ag_k\rangle\right) \\ &= q(x_k) - \gamma_k\|g_k\|^2 + \frac{1}{2}\gamma_k^2\|Ag_k\|^2 \\ &= q(x_k) - \gamma_k\|g_k\|^2\left(1 - \frac{1}{2}\gamma_k\frac{\|Ag_k\|^2}{\|g_k\|^2}\right). \end{aligned} \tag{2.54}$$

Based on the steps 4 and 5 from Algorithm 2.3.1 immediately follows

$$\gamma_k \leq 2(1-\varepsilon)\frac{\|g_k\|^2}{\|Ag_k\|^2},\ k \geq 1,$$

and one can simply verify

$$1 - \frac{1}{2}\gamma_k\frac{\|Ag_k\|^2}{\|g_k\|^2} \geq \varepsilon > 0, \quad k \geq 1. \tag{2.55}$$

Therefore, it is clear that the functional q is strictly monotone decreasing and it is also bounded below with zero. Thus, it follows that the sequence $q(x_k)$ converges to its minimum.

Taking into account that

$$\begin{aligned} 0 &= \lim_{k\to\infty} \mid q(x_{k+1}) - q(x_k) \mid \\ &= \lim_{k\to\infty} \gamma_k\|g_k\|^2\left(1 - \frac{1}{2}\gamma_k\frac{\|Ag_k\|^2}{\|g_k\|^2}\right), \end{aligned}$$

as well as the fact that the sequences (γ_k) and $\left(1 - \frac{1}{2}\gamma_k\frac{\|Ag_k\|^2}{\|g_k\|^2}\right)$ are bounded by real constants, one can conclude that

$$\lim_{k\to\infty}\|g_k\|^2 = 0,$$

which finalizes the proof.

□

Chapter 3

Generalized inverses

The third chapter is completely devoted to the theory of generalized inverses. In the beginning after the initiation of the basic matrix theory, we introduce the main types of generalized inverses of matrices, as well as their properties. After recalling the basics of the theory of generalized inverses, presented in the first section of this chapter, we continue with presentation, of other very important properties of the Drazin-inverse solution and $A^{(2)}_{T,S}$-inverse solution. These properties are the initial point for developing iterative methods for computing the Drazin-inverse solution and $A^{(2)}_{T,S}$-inverse solution, which are presented in the next chapter. Finally, this chapter is finished with the theory of full-rank factorization of generalized inverses. Here, we also included new full-rank presentations of $\{2,3\}$ and $\{2,4\}$-inverses, which are further explored in the next chapter in conjunction with the SMS method for computing $A^{(2)}_{T,S}$ inverses.

3.1 Basic definitions and properties

Let $\mathbb{C}^{m\times n}$ and $\mathbb{C}^{m\times n}_r$ denote the set of all complex $m \times n$ matrices and all complex $m \times n$ matrices of rank r, respectively. I denotes the unit matrix of appropriate order. For a given matrix A, by A^*, $\mathcal{R}(A)$, rank(A) and $\mathcal{N}(A)$ we denote the conjugate transpose, the range, the rank and the null space of $A \in \mathbb{C}^{m\times n}$.

As previously mentioned, the main idea of defining generalized inverses originates from the need to solve the problem of finding a solution of the following system

$$Ax = b, \tag{3.1}$$

where $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^m$.

Next we give definitions for the notions which are usually related to a given matrix and which are frequently used further in the text [6, 12, 87].

First of all, it is well known that a matrix $A \in \mathbb{C}^{m\times n}$ represents a matrix form of a linear map from $\mathbb{C}^n$ to $\mathbb{C}^m$ with respect to the standard basis of $\mathbb{C}^n$ and $\mathbb{C}^m$. So, the symbol A will be used interchangeably, in order to denote a matrix or a linear map.

Definition 3.1.1. *A square matrix $A \in \mathbb{C}^{n\times n}$ ($A \in \mathbb{R}^{n\times n}$) is*

1) Hermitian (self-adjoint) if $A^* = A$ ($A^T = A$),

2) normal, $A^*A = AA^*$ $(A^TA = AA^T)$,

3) lower-triangular, if $a_{ij} = 0$ *for* $i > j$,

4) upper-triangular, if $a_{ij} = 0$ *for* $i < j$,

5) positive semi-definite, if $\mathrm{Re}\ (x^*Ax) \geq 0$ *for all* $x \in \mathbb{C}^{n\times 1}$. *Additionally, if it holds* $\mathrm{Re}\ (x^*Ax) > 0$ *for all* $x \in \mathbb{C}^{n\times 1} \setminus \{0\}$, *then the matrix* A *is positive definite.*

Definition 3.1.2. *Let* $A \in \mathbb{C}^{n\times n}$. *A real or complex scalar* λ *which satisfies the following equation*

$$Ax = \lambda x, \quad i.e., \quad (A - \lambda I)x = 0,$$

is called an eigenvalue of A, *and* x *is called an eigenvector of* A *corresponding to* λ.

The eigenvalues and eigenvectors of a matrix, play a very important role in matrix theory. They represent a tool which enables to understand the structure of a matrix. For example, if a given square matrix of complex numbers is self-adjoint, then there exist basis of $\mathbb{C}^m$ and $\mathbb{C}^n$, consisting of distinct eigenvectors of A, with respect to which the matrix A can be represented as a diagonal matrix. Since not every matrix has enough distinct eigenvectors to enable its good decomposition, in order to resolve this problem, the following definition is given as a generalization of the previous one.

Definition 3.1.3. *Let* $A \in \mathbb{C}^{n\times n}$ *and* λ *is an eigenvalue of* A. *A vector* x *is called generalized eigenvector of* A *of grade* p *corresponding to* λ, *or* λ*-vector of* A *of grade* p, *if it satisfies the following equation*

$$(A - \lambda I)^p x = 0.$$

Namely, for each matrix there exists a basis of generalized eigenvectors with respect to which, a matrix can be represented in the Jordan form, as it is stated in the following proposition.

Proposition 3.1.1. *(The Jordan decomposition). Let the matrix* $A \in \mathbb{C}^{n\times n}$ *has* p *distinct eigenvalues* $\{\lambda_1, \lambda_2, \ldots, \lambda_p\}$. *Then* A *is similar to a block diagonal matrix* J *with Jordan blocks on its diagonal, i.e., there exists a nonsingular matrix* X *such that*

$$AX = XJ = X \begin{bmatrix} J_{k_1}(\lambda_1) & 0 & \ldots & 0 \\ 0 & J_{k_2}(\lambda_2) & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & J_k(\lambda_p) \end{bmatrix}$$

where the Jordan block

$$J_{k_i}(\lambda_i) = \begin{bmatrix} \lambda_i & 1 & 0 & \ldots & 0 \\ 0 & \lambda_i & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \\ 0 & 0 & \ldots & \lambda_i & 1 \end{bmatrix}$$

and the matrix J *is unique up to a rearrangement of its blocks.*

As it can be seen from the previous proposition, on this way we obtain a decomposition of the matrix A with respect to same basis of $\mathbb{C}^n$. The following definition and proposition, give us an alternative way to obtain even simpler decomposition of the matrix A, than the one given with the Jordan decomposition, but with respect to different basis of $\mathbb{C}^n$.

Definition 3.1.4. *Let $A \in \mathbb{C}^{m\times n}$. Let $\{\lambda_1, \lambda_2, \ldots, \lambda_p\}$ be the nonzero eigenvalues of AA^*. The singular values of A, denoted by $\sigma_i(A)$, $i = 1, \ldots, p$ are defined on the following way:*

$$\sigma_i(A) = \sqrt{\lambda_i(AA^*)}, \qquad i = 1, \ldots, p.$$

Proposition 3.1.2. *(Singular value decomposition) Let $A \in \mathbb{C}^{m\times n}$ be a matrix with singular values $\{\sigma_1, \ldots, \sigma_r\}$. Then there exist orthonormal matrices U and V such that*

$$A = U\Sigma V^*,$$

where

$$\Sigma = \begin{bmatrix} \sigma_1 & & & \vdots & 0 \\ & \ddots & & \vdots & 0 \\ & & \sigma_r & \vdots & \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ & 0 & & \vdots & 0 \end{bmatrix}.$$

Further, we state facts related to the inverse of a given square matrix.

Definition 3.1.5. *For a given matrix $A \in \mathbb{C}^{n\times n}$, the inverse of the matrix A is a square matrix A^{-1} such that it satisfies the following equalities*

$$AA^{-1} = I \qquad \text{and} \qquad A^{-1}A = I.$$

Proposition 3.1.3. *A square matrix $A \in \mathbb{C}^{n\times n}$ has a unique inverse if and only if $\det(A) \neq 0$, in which case we say that the matrix A is nonsingular matrix.*

Remark 3.1.1. *In order to distinguish between generalized inverses, the inverse of a matrix defined with Definition 3.1.5 will be called the ordinary inverse.*

In the case when the matrix A from the system (3.1) is nonsingular, the vector

$$x = A^{-1}b,$$

provides a solution of the system (3.1). However, many problems that usually arise in practice, reduce to a problem of the type (3.1), where the matrix A is singular, and moreover, in many cases it is not even a square matrix.

To overcome the previous problem scientists analyzed the ordinary inverse of a matrix, in order to understand its good properties which enable a solution of many different problems. On that way, it is possible to link different sets of properties to different problems, and maybe to define a matrix which would not have all properties of the ordinary inverse, but it will have only the set of properties which are crucial for finding a solution of a given specific problem.

The most important properties of the ordinary inverse are summarized in the following proposition

Proposition 3.1.4. *Let $A \in \mathbb{C}^{n\times n}$ be a given nonsingular matrix, then it holds*

1) $(A^{-1})^{-1} = A$;

2) $(A^*)^{-1} = (A^{-1})^*$*;*

3) $(AB)^{-1} = B^{-1}A^{-1}$*;*

4) A vector x *is an eigenvector of* A *corresponding to the eigenvalue* $\lambda \neq 0$ *if and only if* x *is an eigenvector of* A^{-1} *corresponding to the eigenvalue* λ^{-1}*.*

5) A vector x *is a* λ*-vector of* A *of grade* p *if and only if* x *is a* λ^{-1}*-vector of* A^{-1} *of grade* p*.*

We finish the introduction part for matrices with the definition of Kronecker product of two matrices, which we will be used in order to prove some new results obtained in [94].

Let $A = [a_{ij}]_{i=\overline{1,m},j=\overline{1,n}} \in \mathbb{C}^{m\times n}$ be given matrix. By $a = \text{vec}(A) \in \mathbb{C}^{mn}$ we denote the vector obtained by listing the elements of A, by rows.

Definition 3.1.6. *The Kronecker product* $A \otimes B$ *of two matrices* $A = [a_{ij}] \in \mathbb{C}^{m\times n}$*,* $B \in \mathbb{C}^{p\times q}$ *is the* $mp \times nq$ *matrix expressible in partitioned form as*

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \dots & \dots & \dots & \dots \\ a_{m1}B & a_{m2}B & \dots & a_{mn}B \end{bmatrix}.$$

The properties of the Kronecker product are summarized in the following proposition.

Proposition 3.1.5. [12] *Let* A, B, E, F *be matrices of appropriate dimensions. Then the following hold:*

1) $(A \otimes B)(E \otimes F) = AE \otimes BF$*,*

2) For any $q \in \mathbb{N}$ *it holds* $(A \otimes I)^q = A^q \otimes I$*,*

3) If $\text{ind}(A) = k$*, then* $\text{ind}(A \otimes I) = k$*.*

4) If A *is a square nonsingular matrix, then the matrix* $A \otimes I$ *is nonsingular and* $(A \otimes I)^{-1} = A^{-1} \otimes I$*,*

An important application of the Kronecker product is rewriting a matrix equation

$$AXB = D, \tag{3.2}$$

as a vector equation of the form

$$(A \otimes B^T)vec(X) = vec(D).$$

For simplicity, further in text, we denote $A_B = A \otimes B$.

3.1.1 Matrix equations and $\{i, j, \ldots, k\}$-inverses

It is well known the fact that the system (3.1), always has at least one solution if and only if $b \in \mathcal{R}(A)$. This can be translated into: $b \in \mathcal{R}(A)$ if and only if there exists a matrix X such that $x = Xb$ is a solution of the system. In order to find such matrix, there were established the four so called Penrose equations:

(1) $AXA = A$ **(general condition)**

(2) $XAX = X$ **(reflexive condition)**

(3) $(AX)^* = AX$ **(normalized condition)**

(4) $(XA)^* = XA$ **(reversed normalized condition).**

For a subset $\mathcal{S}$ of the set $\{1, 2, 3, 4\}$, the set of all matrices obeying the conditions contained in $\mathcal{S}$ is denoted by $A\{\mathcal{S}\}$.

Definition 3.1.7. *Any matrix from $A\{\mathcal{S}\}$ is called $\mathcal{S}$-inverse of A and is denoted by $A^{(\mathcal{S})}$.*

On this way we come to the notion of $\{i, j, \ldots, k\}$-inverses, where $i, j, k \in \mathcal{S}$. For example, for a given matrix $A \in \mathbb{C}^{m\times n}$, if there exists a matrix such that it satisfies only the first Penrose equation, then this matrix is called $\{1\}$-inverse of the matrix A, it is denoted by $A^{(1)}$ and we write $A^{(1)} \in A\{1\}$. Similarly, if it satisfies the first and the third Penrose equations, it is $\{1, 3\}$-inverse of A, denoted by $A^{(1,3)}$ and $A^{(1,3)} \in A\{1, 3\}$.

It is obvious that if the matrix A is nonsingular square matrix, then its ordinary inverse satisfies all Penrose equations.

For a given subspaces T and S from $\mathbb{C}^n$ by $P_{T,S}$ we denote a projector from $\mathbb{C}^n$ on T along S. If $S = T^\perp$, i.e., if S is orthogonal complement of T, then P_T is orthogonal projector from $\mathbb{C}^n$ on T. The matrix which corresponds to a linear map which is a projector, is idempotent matrix. The matrix which corresponds to a linear map which is orthogonal projector, is a Hermitian idempotent matrix. In the sequel, we restate the main properties of $\{i, j, \ldots, k\}$-inverses, without proof (see also [12, 137]).

Lemma 3.1.1. *For a given matrix $A \in \mathbb{C}_r^{m\times n}$ the following holds*

1) $(\lambda A)^{(1)} = \lambda^\dagger A^{(1)}$, *where* $\lambda \in \mathbb{C}$ *and* $\lambda^\dagger = \begin{cases} \frac{1}{\lambda}, & \lambda \neq 0 \\ 0, & \lambda = 0 \end{cases}$;

2) $AA^{(1)}$ *is a projection from* $\mathbb{C}^m$ *on* $\mathcal{R}(A)$, *i.e.,* $AA^{(1)} = P_{\mathcal{R}(A),S}$ *where* $S \in \mathbb{C}^m$ *is such that* $\mathcal{R}(A) + S = \mathbb{C}^m$;

3) $I - A^{(1)}A$ *is a projection from* $\mathbb{C}^n$ *on* $\mathcal{N}(A)$, *i.e.,* $I - A^{(1)}A = P_{\mathcal{N}(A),T}$ *where* $T \in \mathbb{C}^n$ *is such that* $T + \mathcal{N}(A) = \mathbb{C}^n$;.

4) $\operatorname{rank}(A^{(1)}) \geq \operatorname{rank}(A)$;

5) $A^{(1)}A = I_n$ *if and only if* $r = n$;

6) $AA^{(1)} = I_m$ *if and only if* $r = m$;

7) If $X \in A\{1\}$*, then* $X \in A\{1,2\}$ *if and only if* $\operatorname{rank}(A) = \operatorname{rank}(X)$*;*

8) $(A^*A)^{(1)}A^* \in A\{1,2,3\}$*;*

9) $A^*(AA^*)^{(1)} \in A\{1,2,4\}$*;*

The next results establish the relationship between $\{i,j,\ldots,k\}$-inverses and the solutions of a given matrix equation [12, 137].

Proposition 3.1.6. *Let* $A \in \mathbb{C}^{m\times n}$*,* $B \in \mathbb{C}^{p\times q}$*,* $D \in \mathbb{C}^{m\times q}$*. Then the matrix equation*

$$AXB = D$$

is consistent if and only if, for some $A^{(1)}, B^{(1)}$*, it holds*

$$AA^{(1)}DB^{(1)}B = D$$

in which case the general solution is

$$X = A^{(1)}DB^{(1)} + Y - A^{(1)}AYBB^{(1)}$$

for arbitrary $Y \in \mathbb{C}^{n\times p}$*.*

Corollary 3.1.1. *Let* $A \in \mathbb{C}^{m\times n}$*,* $A^{(1)} \in A\{1\}$*. Then*

$$A\{1\} = \{A^{(1)} + Z - A^{(1)}AZAA^{(1)} : Z \in \mathbb{C}^{n\times m}\}.$$

Corollary 3.1.2. *Let* $A \in \mathbb{C}^{m\times n}$*,* $b \in \mathbb{C}^m$*. Then the system* (3.1) *is consistent if and only if for some* $A^{(1)}$ *it holds,*

$$AA^{(1)}b = b,$$

in which case the general solution of the system (3.1) *is*

$$x = A^{(1)}b + (I - A^{(1)}A)y,$$

for arbitrary $y \in \mathbb{C}^n$*.*

Proposition 3.1.7. *Let* $A \in \mathbb{C}^{m\times n}$*,* $X \in \mathbb{C}^{n\times m}$*. Then* $X \in A\{1\}$ *if and only if, for all* $b \in \mathcal{R}(A)$*,* $x = Xb$ *is a solution of the system* (3.1)*.*

Proposition 3.1.8. *The set* $A\{1,3\}$ *consists of all solutions* X *of the system*

$$AX = AA^{(1,3)},$$

where $A^{(1,3)}$ *is an arbitrary element of* $A\{1,3\}$*.*

Proposition 3.1.9. *Let* $A \in \mathbb{C}^{m\times n}$*,* $A^{(1,3)} \in A\{1,3\}$*. Then*

$$A\{1,3\} = \{A^{(1,3)} + (I - A^{(1,3)}A)Z : Z \in \mathbb{C}^{n\times m}\}.$$

Proposition 3.1.10. *The set* $A\{1,4\}$ *consists of all solutions* X *of the system*

$$XA = A^{(1,4)}A,$$

where $A^{(1,4)}$ *is an arbitrary element of* $A\{1,4\}$*.*

Corollary 3.1.3. *Let* $A \in \mathbb{C}^{m\times n}$*,* $A^{(1,4)} \in A\{1,4\}$*. Then*

$$A\{1,4\} = \{A^{(1,4)} + Y(I - AA^{(1,4)}) : Y \in \mathbb{C}^{n\times m}\}.$$

These results give us a powerful tool for finding a solution of a given consistent system of linear equation, and as it will be discussed later, for finding an approximate solution of an inconsistent system of linear equations.

3.1.2 The Moore-Penrose inverse

The most important result related to the Penrose equations, which was shown by Penrose [106] in 1955, is that for a given matrix A, there always exists a unique matrix which satisfies the four Penrose equations. This matrix is called the Moore-Penrose inverse, denoted by $A^\dagger$.

Although $\{1\}$-inverses and $\{1,3\}$-inverses provide a solution of a given matrix equation, it is the Moore-Penrose inverse which most resemble to the ordinary inverse. This is justified by its uniqueness and the properties listed in the following two lemmas. Also, we should bear on mind that, since the Moore-Penrose inverse is $\{1\}$-inverse, the properties from Lemma 3.1.1 are also valid.

Lemma 3.1.2. *Let $A \in \mathbb{C}^{m\times n}$ be an arbitrary matrix. Then the following properties are valid.*

1) $(A^\dagger)^\dagger = A$, $(A^\dagger)^* = (A^*)^\dagger$;

3) $(AA^*)^\dagger = (A^*)^\dagger A^\dagger$, $(A^*A)^\dagger = A^\dagger (A^*)^\dagger$;

4) $A^\dagger AA^* = A^* = A^*AA^\dagger$;

5) $A^\dagger = (A^*A)^\dagger A^* = A^*(AA^*)^\dagger$;

6) $\mathcal{N}(AA^\dagger) = \mathcal{N}(A^\dagger) = \mathcal{N}(A^*)$

7) $\mathcal{R}(AA^*) = \mathcal{R}(AA^{(1)}) = \mathcal{R}(A)$, $\operatorname{rank}(AA^{(1)}) = \operatorname{rank}(A^{(1)}A) = \operatorname{rank}(A)$;

8) $AA^\dagger = P_{\mathcal{R}(A),\mathcal{N}(A^*)}$ *and* $A^\dagger A = P_{\mathcal{R}(A^*),\mathcal{N}(A)}$.

Lemma 3.1.3. *Let $A \in \mathbb{C}^{m\times n}$ be an arbitrary matrix. Then the matrix A can be written in the following way:*

$$A \sim \begin{bmatrix} A_1 & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A^*) \\ \mathcal{N}(A) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A) \\ \mathcal{N}(A^*) \end{bmatrix}, \tag{3.3}$$

where A_1 is invertible. Hence,

$$A^\dagger \sim \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A) \\ \mathcal{N}(A^*) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^*) \\ \mathcal{N}(A) \end{bmatrix}.$$

The form (3.3) of the matrix A, can be easily obtained by Singular value decomposition of A, i.e., the matrix A_1 represents a diagonal matrix, where the singular values of A are actually its diagonal elements.

If the system (3.1) is such that $b \notin \mathcal{R}(A)$, then we search for an approximate solution of the system (3.1) by trying to find a vector x for which the norm of the vector $Ax - b$ is minimal.

Definition 3.1.8. *Let $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^m$. A vector $\hat{x}$, which satisfies the equality*

$$\|A\hat{x} - b\|^2 = \min_{x\in\mathbb{C}^n} \|Ax - b\|^2. \tag{3.4}$$

is called a least-squares solution of the system (3.1).

The next lemma gives a characterization of all least-squares solutions of the system (3.1).

Lemma 3.1.4. *The vector x is a least-squares solution of the system* (3.1) *if and only if x is a solution of the normal equation, defined by*

$$A^*Ax = A^*b. \tag{3.5}$$

The following proposition [12] shows that $\|Ax - b\|$ is minimized by choosing $x = A^{(1,3)}b$, thus establishing a relation between the $\{1,3\}$-inverses and the least-squares solutions of the system (3.1).

Proposition 3.1.11. *Let $A \in \mathbb{C}^{m\times n}$, $b \in \mathbb{C}^m$. Then $\|Ax - b\|$ is smallest when $x = A^{(1,3)}b$, where $A^{(1,3)} \in A\{1,3\}$. Conversely, if $X \in \mathbb{C}^{n\times m}$ has the property that, for all b, $\|Ax - b\|$ is smallest when $x = Xb$, then $X \in A\{1,3\}$.*

Since $A^{(1,3)}$-inverse of a matrix is not unique, consequently a system of linear equations can have many least-squares solutions. However, it is shown that among all least-squares solutions of a given system of linear equations, there exists only one such solution of minimum norm.

Definition 3.1.9. *Let $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^m$. A vector $\hat{x}$, which satisfies the equality*

$$\|\hat{x}\|^2 = \min_{x\in\mathbb{C}^n} \|x\|^2. \tag{3.6}$$

is called a minimum-norm solution of the system (3.1).

The next proposition establishes a relation between $\{1,4\}$-inverses and the minimum-norm solutions of the system (3.1).

Proposition 3.1.12. *Let $A \in \mathbb{C}^{m\times n}$, $b \in \mathbb{C}^m$. If $Ax = b$ has a solution for x, the unique solution for which $\|x\|$ is smallest is given by $x = A^{(1,4)}b$, where $A^{(1,4)} \in A\{1,4\}$. Conversely, if $X \in \mathbb{C}^{n\times m}$ is such that, whenever $Ax = b$ has a solution, $x = Xb$ is the solution of minimum-norm, then $X \in A\{1,4\}$.*

Joining the results from Proposition 3.1.11 and Proposition 3.1.12 we are coming to the most important result in the theory of the Moore-Penrose inverse.

Corollary 3.1.4. *[106]. Let $A \in \mathbb{C}^{m\times n}$, $b \in \mathbb{C}^m$. Then, among the least-squares solutions of $Ax = b$, $A^\dagger b$ is the one of minimum-norm. Conversely, if $X \in \mathbb{C}^{n\times m}$ has the property that, for all b, Xb is the minimum-norm least-squares solution of $Ax = b$, then $X = A^\dagger$.*

The next proposition, characterizes the set of all least-squares solutions of a given system of linear equations.

Proposition 3.1.13. [97, 98] The set S of all least-squares solutions of the system $Ax = b$ is given by

$$S = A^\dagger b \oplus \mathcal{N}(A) = \{A^\dagger b + (I - A^\dagger A)y |\ y \in \mathbb{C}^n\},$$

where $\mathcal{N}(A)$ denotes the null space of A.

One generalization of the Moore-Penrose inverse is the, so called, weighted Moore-Penrose inverse which is introduced with the following definition.

Definition 3.1.10. *Let $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^m$, M and N be Hermitian positive definite matrices of orders m and n respectively. The matrix $X \in \mathbb{C}^{n\times m}$ which despite equations* (1) *and* (2)*, also satisfies the equations*

(**3M**) $(MAX)^* = MAX$

(**4N**) $(NXA)^* = NXA$

is called weighted Moore-Penrose inverse of A and is denoted with $A^\dagger_{M,N}$.

3.1.3 The Drazin inverse

We saw that the Moore-Penrose inverse is a very good replacement for the ordinary inverse, when a solution of a given matrix equation is needed. It exists for all matrices, and when the matrix is nonsingular it reduces to the ordinary inverse. But, unfortunately we can not say that it satisfies the properties of the ordinary inverse, characterized with the fourth and fifth item from Lemma 3.1.4. To be defined such inverse, the set of Penrose equations is enlarged with two more:

($\mathbf{1^p}$) $A^pXA = A^p$ **(general p condition)**

($\mathbf{5}$) $AX = XA$ **(commutativity condition)**

where $p = \mathrm{ind}(A)$.

Definition 3.1.11. *Let $A \in \mathbb{C}^{n\times n}$ and $p = \mathrm{ind}(A)$. The matrix $X \in A\{1^p, 2, 5\}$ is called the Drazin inverse of A, denoted by A^D. If $p = 1$ then the Drazin inverse is called group inverse, denoted by $A^\#$.*

Since the eigenvalues and eigenvectors are defined only for square matrices, the Drazin inverse, also, is defined only for square matrices. In the next lemma we give the main properties of the Drazin inverse.

Lemma 3.1.5. *Let $A \in \mathbb{C}^{n\times n}$ and $p = \mathrm{ind}(A)$*

1) $A^lXA = A^l$ for all $l \geq p$

2) $\mathcal{R}(A^l) = \mathcal{R}(A^{l+1})$, $\mathcal{N}(A^l) = \mathcal{N}(A^{l+1})$ and $\mathrm{rank}(A^l) = \mathrm{rank}(A^{l+1})$, for all $l \geq p$. Moreover, p is the smallest integer for which the equalities hold.

3) The matrix A can be written in following way:

$$A \sim \begin{bmatrix} A_1 & 0 \\ 0 & N \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A^p) \\ \mathcal{N}(A^p) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^p) \\ \mathcal{N}(A^p) \end{bmatrix}, \tag{3.7}$$

where A_1 is invertible, and N is nilpotent matrix. Hence,

$$A^D \sim \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A^p) \\ \mathcal{N}(A^p) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^p) \\ \mathcal{N}(A^p) \end{bmatrix};$$

4) for all $\lambda \neq 0$, a vector x is a λ^{-1}-vector of A^D of grade s if and only if it is a λ-vector of A of grade s, and x is a 0-vector of A^D if and only if it is a 0-vector of A (without regard to grade.

The form (3.7) of the matrix A, can be easily obtain by the Jordan decomposition of A.

Despite the spectral properties, the Drazin inverse, in some cases, it also provides a solution of a given system of linear equations. Namely for $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$, as it was shown in [20], A^Db is a solution of the following system

$$Ax = b, \quad \text{where } b \in \mathcal{R}(A^p)\ , p = \mathrm{ind}(A). \tag{3.8}$$

and we call it the *Drazin-inverse solution* of the system (3.8). Also, since this is the only case, when the Drazin-inverse provides a solution to the given system, we call the system (3.8), a *Drazin-consistent system.*

The Drazin inverse has many applications in the theory of finite Markov chains as well as in the study of differential equations and singular linear difference equations [20], cryptography [83] etc.

Establishing a relation between the Drazin inverse and the solutions of a given system of linear equations, naturally imposed the idea of exploring minimal properties of the Drazin inverse. Next we present results from the paper [142], where are established respective results for the Drazin-inverse solution, to the ones presented in the previous section for the Moore-Penrose inverse solution.

Theorem 3.1.1. *Let $A \in \mathbb{R}^{n\times n}$ with $p = \mathrm{ind}(A)$. Then $A^D b$ is the unique solution in $R(A^p)$ of the system*

$$A^{p+1}x = A^p b. \tag{3.9}$$

Theorem 3.1.2. *Let $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and $p = \mathrm{ind}(A)$. The set of all solutions of the equation* (3.9) *is given by*

$$x = A^D b + \mathcal{N}(A^p). \tag{3.10}$$

Since (3.9) is analogous to (3.5), we shall call it the *generalized normal equations* of (3.8).

Let $A = PJP^{-1}$ be the Jordan decomposition of the matrix A. We denote $\|x\|_P = \|P^{-1}x\|$.

Theorem 3.1.3. *[142] Let $A \in \mathbb{R}^{n\times n}$ with $p = \mathrm{ind}(A)$. Then $\hat{x}$ satisfies*

$$\|b - A\hat{x}\|_P = \min_{u\in\mathcal{N}(A)+\mathcal{R}(A^{p-1})} \|b - Ax\|_P$$

if and only if $\hat{x}$ is the solution of the equation

$$A^{p+1}x = A^p b, \quad x \in \mathcal{N}(A) + \mathcal{R}(A^{p-1}).$$

Moreover, the Drazin-inverse solution $x = A^D b$ is the unique minimal P-norm solution of the generalized normal equations (3.9).

Corollary 3.1.5. *[142] Let $A \in \mathbb{C}^{n\times n}$, $p = \mathrm{ind}(A)$ and $b \in \mathcal{R}(A)$. Then for all solutions x of the system* (3.9) *the inequality $\|x\|_P \geq \|A^D b\|_P$ holds, i.e., $A^D b$ is the unique solution of the equation* (3.9) *of minimum P-norm.*

Analogously to the weighted Moore-Penrose inverse we introduce the notion of weighted Drazin inverse.

Definition 3.1.12. *Let $A \in \mathbb{C}^{m\times n}$, $W \in \mathbb{C}^{n\times m}$. Then the matrix $X \in \mathbb{C}^{m\times n}$ satisfying*

1) $(AW)^{p+1}XW = (AW)^p$; *(for some nonnegative integer p)*

2) $XWAWX = X$

3) $AWX = XWA$

is called W-weighted Drazin inverse of A, and is denoted by $X = A^{D,W}$.

3.1.4 The $A^{(2)}_{T,S}$-inverse

Recall that, for an arbitrary matrix $A \in \mathbb{C}^{m\times n}$, the set of all outer inverses (or also called $\{2\}$-inverses) is defined by the following

$$A\{2\} = \{X \in \mathbb{C}^{n\times m} | XAX = X\}. \tag{3.11}$$

With $A\{2\}_s$ we denote the set of all outer inverses of rank s and the symbol $A^{(2)}$ stands for an arbitrary outer inverse of A.

Definition 3.1.13. *Let $A \in \mathbb{C}^{m\times n}_r$, T is a subspace of $\mathbb{C}^n$ of dimension $t \leq r$ and S is a subspace of $\mathbb{C}^m$ of dimension $m - t$, then A has a $\{2\}$-inverse X such that $\mathcal{R}(X) = T$ and $\mathcal{N}(X) = S$ if and only if $AT \oplus S = \mathbb{C}^m$, in which case X is unique and it is denoted by $A^{(2)}_{T,S}$.*

Lemma 3.1.6. *Let $A \in \mathbb{C}^{m\times n}$ be an arbitrary matrix, T is a subspace of $\mathbb{C}^n$ and S is a subspace of $\mathbb{C}^m$ such that $AT \oplus S = \mathbb{C}^m$. Then the matrix A can be written in the following way:*

$$A \sim \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} : \begin{bmatrix} T \\ \mathcal{N}(A^{(2)}_{T,S}A) \end{bmatrix} \to \begin{bmatrix} AT \\ S \end{bmatrix}, \tag{3.12}$$

where A_1 is invertible. Moreover,

$$A^{(2)}_{T,S} \sim \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} AT \\ S \end{bmatrix} \to \begin{bmatrix} T \\ \mathcal{N}(A^{(2)}_{T,S}A) \end{bmatrix}.$$

The outer generalized inverse with prescribed range T and null-space S is a generalized inverse of special interest in matrix theory. The reason of the importance of this inverse is the fact that: the Moore-Penrose inverse $A^\dagger$, the weighted Moore-Penrose inverse $A^\dagger_{M,N}$, the Drazin inverse A^D, the group inverse $A^\#$, the Bott-Duffin inverse $A^{(-1)}_{(L)}$ and the generalized Bott-Duffin inverse $A^{(+)}_{(L)}$; are all $\{2\}$-generalized inverses of A with prescribed range and null space.

Lemma 3.1.7. *Let $A \in \mathbb{C}^{m\times n}_r$ and $p = \mathrm{ind}(A)$. Then the following representations are valid*

1) $A^\dagger = A^{(2)}_{\mathcal{R}(A^*),\mathcal{N}(A^*)}$,

2) $A^\dagger_{M,N} = A^{(2)}_{\mathcal{R}(N^{-1}A^*M),\mathcal{N}(N^{-1}A^*M)}$,

3) $A^D = A^{(2)}_{\mathcal{R}(A^p),\mathcal{N}(A^p)}$,

4) $A^\# = A^{(2)}_{\mathcal{R}(A),\mathcal{N}(A)}$ *if and only if* $p = 1$,

5) $A^{D,W} = A^{(2)}_{\mathcal{R}(A(WA)^p),\mathcal{N}(A(WA)^p)}$.

3.2 Further properties of the Drazin inverse

3.2.1 Least-square properties of the Drazin-inverse solution

The results from this section [92] are complement to the results investigated in [142], and which are presented in Section 3.1.3. Namely, they are motivated form the idea of defining a gradient iterative method for computing the Drazin-inverse solution of the system (3.8). The goal is achieved by establishing a relation between the Drazin-inverse solution of the system (3.8) and its $\{1,3\}$-inverse solutions. Later, in the subsequent chapters will be presented the usefulness of the considered results, by the definition of gradient iterative methods as a tool for computing the Drazin-inverse solution.

- **Case 1: The system of linear equations is Drazin consistent, i.e.,** $b \in \mathcal{R}(A^p)$.

Theorem 3.2.1. *Each solution of* (3.8) *is also a solution of* (3.9) *but the contrary does not hold.*

Proof. If $Ax = b$ then obviously $A^p(Ax - b) = 0$.
On the other hand, Wei in [140] shows that the general solution of (3.8) is given by

$$x = A^D b + A^{p-1}(I - A^D A)z, \tag{3.13}$$

where z is an arbitrary vector. Based on this, the opposite statement is not valid since not every element from $\mathcal{N}(A^p)$ can be represented as $A^{p-1}(I - A^D A)z$, $z \in \mathbb{C}^n$ is arbitrary. Consequently, not every solution of (3.9) is a solution of the equation (3.8) nor a solution of the equation (3.1). □

Lemma 3.2.1. *Let* $A \in \mathbb{C}^{n\times n}$ *and* $p = \mathrm{ind}(A)$. *Then for each* l *such that* $l \geq p$ *the following holds*

$$A^l(A^{l+1})^{(1)} = A^p(A^{p+1})^{(1)} = A^D P_{\mathcal{R}(A^p),S},$$

where S *is a matrix satisfying* $\mathcal{R}(A^p) + S = \mathbb{C}^n$. *Moreover, the following statements are valid*

a) *if* $\mathcal{N}((A^p)^*) \subseteq \mathcal{N}(A^p)$ *then* $A^D = A^p(A^{p+1})^\dagger$,

b) $A^D = A^p(A^{p+1})^\#$.

Proof. For each $l \geq p$ we have

$$A^l(A^{l+1})^{(1)} = A^D A^{l+1}(A^{l+1})^{(1)} = A^D P_{\mathcal{R}(A^{l+1}),S} = A^D P_{\mathcal{R}(A^p),S}.$$

From here, also, follows that

a) if $\mathcal{N}((A^p)^*) \subseteq \mathcal{N}(A^p)$ then $A^p(A^{p+1})^\dagger = A^D P_{\mathcal{R}(A^p),\mathcal{N}((A^p)^*)} = A^D$.

b) $A^p(A^{p+1})^\# = A^D P_{\mathcal{R}(A^p),\mathcal{N}(A^p)} = A^D$.

□

Corollary 3.2.1. *Let $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ be such that $b \in \mathcal{R}(A^p)$ where $p = \mathrm{ind}(A)$. Then the vector defined by*

$$x = A^p(A^{p+1})^{(1,3)}b, \tag{3.14}$$

is the Drazin-inverse solution of the given system (3.8).

Proof. According to (3.14), from the previous lemma we have

$$x = A^p(A^{p+1})^{(1,3)}\, b = A^D P_{\mathcal{R}(A^p),S}\, b,$$

where $\mathcal{R}(A^p) + S = \mathbb{C}^n$. Since $b \in \mathcal{R}(A^p)$, immediately follows that

$$x = A^D b = A^p(A^{p+1})^{(1,3)}b, \tag{3.15}$$

which completes the proof. □

Remark 3.2.1. *In the case when A is invertible, equality* (3.14) *reduces to $x = A^{-1}b$.*

Campbell in [20] shows that $A^D b$ is a solution of (3.8) if and only if $b \in \mathcal{R}(A^p)$, and points out that $A^D b$ is the unique solution of (3.8) provided that $x \in \mathcal{R}(A^p)$.

Therefore, we assume that $b \in \mathcal{R}(A^p)$. Based on the previous considerations, we are going to analyze the following problem:

$$\min_x f(x) = \min_x \frac{1}{2}\|A^{p+1}x - b\|^2. \tag{3.16}$$

Theorem 3.2.2. *Let $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ be such that $b \in \mathcal{R}(A^p)$ where $p = \mathrm{ind}(A)$. Let $\hat{x}$ be the minimizer of the function* (3.16) *then*

$$A^p\hat{x} = A^D b. \tag{3.17}$$

Proof. From [97, 98], it follows that a least-squares solution of the system $A^{p+1}x = b$, which is also its solution, is given by

$$\hat{x} = (A^{p+1})^{(1,3)}b + \left(I - (A^{p+1})^{(1,3)}A^{p+1}\right) y,$$

where $y \in \mathbb{C}^n$ is an arbitrary vector. Then

$$\begin{aligned} A^p\hat{x} &= A^p(A^{p+1})^{(1,3)}b + A^p(I - (A^{p+1})^{(1,3)}A^{p+1})y \\ &= A^D b + A^p(I - (A^{p+1})^{(1,3)}A^{p+1})y. \end{aligned} \tag{3.18}$$

In the subsequent part of the proof we use the next statement from [12]: let A and B be an arbitrary matrices, then $B(AB)^{(1)}AB = B$ if and only if $\mathrm{rank}(AB) = \mathrm{rank}(B)$. Since $\mathrm{rank}(A^{p+1}) = \mathrm{rank}(A^p)$, we have

$$A^p(A^{p+1})^{(1,3)}A^{p+1} = A^k(AA^p)^{(1,3)}AA^p = A^p.$$

Finally, from the second part of equation (3.18) immediately follows $A^p\hat{x} = A^D b$. □

Therefore, the problem of finding the Drazin-inverse solution is reduced to the problem of finding the minimum of the function (3.16).

- **Case 2: The system of linear equations is an arbitrary system, i.e., $b \in \mathbb{C}^n$.**

As it is well known, the Drazin inverse always exists for a square matrix, although it provides a solution of the system (3.1) only in the cases when $b \in \mathcal{R}(A^p)$. In this section, our purpose is to explore properties of the vector of the form $A^D b$, for arbitrary square matrix A and arbitrary vector b of appropriate dimensions. For convenience we will also call this vector a Drazin-inverse solution. In order to achieve this goal, we introduce the notion of *modified Drazin normal equation* which is given by

$$A^{2p}x = A^p b. \tag{3.19}$$

Theorem 3.2.3. *Let $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and $p = \mathrm{ind}(A)$. The set of all solutions of the equation (3.19) is given by*

$$x = (A^p)^D b + \mathcal{N}(A^p). \tag{3.20}$$

Proof. Since $A^p b \in \mathcal{R}(A^p) = \mathcal{R}(A^{2p})$ we have that the system (3.19) is consistent. Following the result from [20] we have that $(A^p)^D b = (A^p)^{\#} b$ is the unique solution of the analyzed system that belongs to $\mathcal{R}(A^p)$. Additionally, if x is a solution of (3.19) it follows that $A^p x - b \in \mathcal{N}(A^p)$. Based on the fact that the set of all solutions of the homogeneous system $A^{2p}x = 0$ is the same as $\mathcal{N}(A^{2p}) = \mathcal{N}(A^p)$, which we show that is nonempty, we finally obtain (3.20). □

The following theorem gives the initial idea for finding the Drazin-inverse solution in general case.

Theorem 3.2.4. *Let $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and $p = \mathrm{ind}(A)$. Then $\hat{x}$ satisfies*

$$\|b - A^p\hat{x}\|_P^2 = \min_x \|b - A^p x\|_P^2 \tag{3.21}$$

if and only if $\hat{x}$ is a solution of the equation (3.19).

Proof.

$$\begin{aligned} b &= \left(P_{\mathcal{R}(A^p)} + P_{\mathcal{N}(A^p)}\right) b \\ b - A^p x &= (P_{\mathcal{R}(A^p)} b - A^p x) + P_{\mathcal{N}(A^p)} b \\ \|b - A^p x\|_P^2 &= \|P_{\mathcal{R}(A^p)} b - A^p x\|_P^2 + \|P_{\mathcal{N}(A^p)} b\|_P^2 \end{aligned} \tag{3.22}$$

Evidently $\|b - A^p x\|_P^2$ attains minimum for

$$A^p x = P_{\mathcal{R}(A^p)} b = AA^D b. \tag{3.23}$$

We will show that the equations (3.19) and (3.23) are equivalent. If x satisfies (3.21) and thus (3.23), evidently, multiplying the equation (3.23) by A^p from the left on the both hand sides, we obtain that it satisfies (3.19).

Conversely, let us suppose that x satisfies (3.19), i.e., $A^{2p}x = A^p b$. By multiplying with $(A^p)^D$ from the left on the both hand sizes, and using the facts that $(A^p)^D A^p = A^D A$ and $A^D A^{p+1} = A^p$ we immediately obtain (3.23). □

Corollary 3.2.2. *Let $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and $p = \mathrm{ind}(A)$. If $\hat{x}$ is a P-norm least-squares solution of the system $A^p x = b$, then*

$$A^{p-1}\hat{x} = A^D b + A^{p-1}\mathcal{N}(A^p).$$

Moreover, if $\hat{x} \in \mathcal{R}(A^p)$, then $A^{p-1}\hat{x} = A^D b$.

Proof. If $\hat{x}$ satisfies (3.21), then it is a solution of the equation (3.19), i.e., it is of the form

$$\hat{x} = (A^p)^D b + \mathcal{N}(A^p).$$

Then

$$\begin{aligned} A^{p-1}\hat{x} &= A^{p-1}(A^p)^D b + A^{p-1}\mathcal{N}(A^p) = A^{p-1}(A^{p-1})^D A^D b + A^{p-1}\mathcal{N}(A^p) \\ &= AA^D A^D b + A^{p-1}\mathcal{N}(A^p) = A^D b + A^{p-1}\mathcal{N}(A^p). \end{aligned}$$

Additionally, if $\hat{x} \in \mathcal{R}(A^p)$, then $\hat{x} = (A^p)^D b$ which implies

$$A^{p-1}\hat{x} = A^{p-1}(A^p)^D b = A^D b,$$

and thus, completes the proof. □

Considering these results, we are moving to the problem of computing the vector $A^D b$. For that purpose let us focus to the following problem

$$\min_x f(x) = \min_x \|A^{2p}x - b\|_P^2, \tag{3.24}$$

whose solution we want to find. The characterization of the Drazin-inverse solution in general case, given in terms of the minimizer of the problem (3.24), is stated in the following theorem.

Theorem 3.2.5. *Let $A \in \mathbb{C}^{n\times n}$, $b \in \mathbb{C}^n$ and let $\hat{x}$ be the minimizer of the functional* (3.24)*. Then*

$$A^{2p-1}\hat{x} = A^D b. \tag{3.25}$$

Proof. Since $\hat{x}$ is the minimizer of the functional (3.24), then it is a P-norm least-squares solution of the system $A^{2p}x = b$. Then $\hat{y} = A^p\hat{x}$ is the minimizer of the functional

$$\min_y \|A^p y - b\|_P^2$$

which belongs to $\mathcal{R}(A^p)$. According to Corollary 3.2.2, it follows that

$$A^{p-1}\hat{y} = A^{p-1}A^p\hat{x} = A^{2p-1}\hat{x} = A^D b,$$

which finishes the proof. □

Remark 3.2.2. *With the previous result we established an analogy to the result given by Theorem* 3.2.2*.*

If for simplicity we put $A_1 = P^{-1}A^{2p}$ and $b_1 = P^{-1}b$ then we obtain the equivalent form of the problem (3.24) given in terms of the 2-norm

$$\min_x f(x) = \min_x \|A_1 x - b_1\|^2. \tag{3.26}$$

Obviously, the method for finding the Drazin-inverse solution in general case is reduced to the method for finding a minimum of the functional given by (3.24) or (3.26).

3.2.2 Least-squares properties of the Drazin inverse

Let $A,\ B,\ G \in \mathbb{C}^{n\times n}$ be given matrices. The intent of the present section is to provide a method for computing the Drazin-inverse solution of the matrix equation $AXB = G$, of the form A^DGB^D. Namely, the problem of computing the Drazin-inverse solution is reduced to the problem of finding a minimum of a matrix function. This reduction is obtained according to the proposed relationship between the Drazin-inverse solution of the matrix equation and appropriate $\{1,3\}$-inverses. Finally, multiplying the obtained limit point by an appropriate matrices, appears to be getting exactly the Drazin-inverse solution of the matrix equation. Later using the obtained results, we derive several consequences related to the representation of the Drazin inverse of a given square matrix. The sole process of determining a solution of the matrix equation $AXB = D$ will reveal the minimal properties of the Drazin inverse.

For the purpose of the present section we use the following notation:

$$\mathcal{N}(A) = \{x \in \mathbb{C}^n | Ax = 0\}, \quad \mathcal{R}(A) = \{Ax | x \in \mathbb{C}^n\},$$

$$\widetilde{\mathcal{N}}(A) = \{X \in \mathbb{C}^{n\times n} | AX = 0\}, \quad \widetilde{\mathcal{R}}(A) = \{AX | X \in \mathbb{C}^{n\times n}\}.$$

Proposition 3.2.1. *Let $A,\ B \in \mathbb{C}^{n\times n}$. Then the following two statements are equivalent:*

1) $\mathcal{R}(B) \subset \mathcal{R}(A)$

2) $B \in \widetilde{\mathcal{R}}(A)$.

Let us consider the following matrix equation

$$AXB = G, \quad \text{where } \mathcal{R}(G) \subset \mathcal{R}(A^{p_1}),\ \mathcal{N}(B^{p_2}) \subset \mathcal{N}(G),\ p_1 = \text{ind}(A),\ p_2 = \text{ind}(B). \tag{3.27}$$

As it was shown in [138], the matrix $X = A^DGB^D$ is the unique solution of (3.27) such that $\mathcal{R}(X) \subset \mathcal{R}(A^{p_1})$ and $\mathcal{N}(B^{p_2}) \subset \mathcal{N}(X)$, and we call it the Drazin-inverse solution of the matrix equation (3.27). As a consequence, it seems reasonable to call the matrix equation (3.27) by *Drazin consistent matrix equation.*

In the present section, we consider the problem of finding the matrix A^DGB^D in general case

$$AXB = G,\ G \text{ is an arbitrary matrix from } \mathbb{C}^{n\times n}. \tag{3.28}$$

Although, in the case when $\mathcal{R}(G) \nsubseteq \mathcal{R}(A_1^p)$ or $\mathcal{N}(B^{p_2}) \nsubseteq \mathcal{N}(G)$, the vector A^DGB^D is not a solution of (3.28), for convenience we also call it the Drazin-inverse solution of the matrix equation (3.28).

Let $A = PJP^{-1}$ and $B = QJQ^{-1}$ be the Jordan decompositions of the matrices A and B respectively, and let us denote

$$\|X\|_P = \|P^{-1}X\|_F \quad \|X\|_{PQ} = \|P^{-1}XQ\|_F \quad \text{and } \|x\|_P = \|P^{-1}x\|_2.$$

where $X \in \mathbb{C}^{n\times n}$ and $x \in \mathbb{C}^n$. And, let us recall the notation $A_B = A \otimes B^T$ from the first section of this chapter.

Remark 3.2.3. *The norm $\|\cdot\|_{PQ}$ is not multiplicative norm.*

Proposition 3.2.2. *Let $A \in \mathbb{C}^{n\times n}$, P be the Jordan basis of A and let J be a Jordan matrix of A. Then $P_I J_I P_I^{-1} = A_I$. Moreover, $\|X\|_P = \|vec(X)\|_{P_I}$.*

Proof. From the presumption, it follows that $A = PJP^{-1}$. Now from the first property of Proposition 3.1.5, we obtain,

$$P_I J_I P_I^{-1} = (PJP^{-1})_I = A_I.$$

Finally,

$$\begin{aligned}\|X\|_P = \|P^{-1}X\|_F = \|(P^{-1}\otimes I)\mathrm{vec}(X)\|_2 = \|(P\otimes I)^{-1}\mathrm{vec}(X)\|_2\\ = \|P_I^{-1}\mathrm{vec}(X)\|_2 = \|\mathrm{vec}(X)\|_{P_I},\end{aligned}$$

which completes the proof. □

Proposition 3.2.3. *Let $A \in \mathbb{C}^{n\times n}$, P be the Jordan basis of the matrix A and $p = \mathrm{ind}(A)$. Then*

$$\widetilde{\mathcal{R}}(A^p) \oplus \widetilde{\mathcal{N}}(A^p) = \mathbb{C}^{n\times n}.$$

Moreover, the spaces are orthogonal with respect to the P-norm.

Proof. Let $X \in \mathbb{C}^{n\times n}$ be arbitrary matrix. Then $A^pX \in \widetilde{\mathcal{R}}(A^p) = \widetilde{\mathcal{R}}(A^{2p})$. Then there exists a matrix $V \in \mathbb{C}^{n\times n}$ such that $A^{2p}V = A^pX$. Let $U = A^pV \in \widetilde{\mathcal{R}}(A^p)$. Then $A^pU = A^pX$, i.e., $X - U \in \widetilde{\mathcal{N}}(A^p)$. From $X = U + X - U$, $A^p(X - U) = 0$, and $U \in \widetilde{\mathcal{R}}(A^p)$, it follows $\widetilde{\mathcal{R}}(A^p) + \widetilde{\mathcal{N}}(A^p) = \mathbb{C}^{n\times n}$.

Now let us suppose that $X \in \widetilde{\mathcal{R}}(A^p) \cap \widetilde{\mathcal{N}}(A^p)$. Then $X = A^pY$ for some $Y \in \mathbb{C}^{n\times n}$ and $A^pX = 0$. From here, we obtain $A^{2p}Y = A^pX = 0$. Consequently, $Y \in \widetilde{\mathcal{N}}(A^{2p}) = \widetilde{\mathcal{N}}(A^p)$, from which follows $X = A^pY = 0$. Therefore, we obtain $\widetilde{\mathcal{R}}(A^p) \oplus \widetilde{\mathcal{N}}(A^p) = \mathbb{C}^{n\times n}$.

Let $X \in \widetilde{\mathcal{R}}(A^p)$ and $Y \in \widetilde{\mathcal{N}}(A^p)$. Then

$$X = A^pZ, \text{ for some } Z \in \mathbb{C}^{n\times n} \text{ and } A^pY = 0.$$

Using the Kronecker product, the previous equalities can be converted to

$$x = A_I^p z \text{ and } A_I^p y = 0,$$

where $x = \mathrm{vec}(X), y = \mathrm{vec}(Y), z = \mathrm{vec}(Z)$. Consequently $x \in \mathcal{R}(A_I^p)$ and $y \in \mathcal{N}(A_I^p)$. From the second and third property of Proposition 3.1.5 follows the orthogonality of the spaces $\mathcal{R}(A_I^p)$ and $\mathcal{N}(A_I^p)$. Thus, it follows that

$$\|x + y\|_{P_I}^2 = \|x\|_{P_I}^2 + \|y\|_{P_I}^2.$$

Using Proposition 3.2.2 we obtain

$$\|X + Y\|_P^2 = \|x + y\|_{P_I}^2 = \|x\|_{P_I}^2 + \|y\|_{P_I}^2 = \|X\|_P^2 + \|Y\|_P^2.$$

which completes the proof. □

Now, we are going to develop a methodology for finding the matrix of the form A^DGB^D, for arbitrary square matrices A, B and G of appropriate dimensions and, in the final instance, for the choices $B = I, G = I$ to determine the Drazin inverse of the matrix A. In order to achieve this goal, we introduce the notion of *modified Drazin normal matrix equation* which is given by

$$A^{2p_1}XB^{2p_2} = A^{p_1}GB^{p_2}. \tag{3.29}$$

Theorem 3.2.6. *Let $A, B, G \in \mathbb{C}^{n\times n}$ be such that $p_1 = \mathrm{ind}(A), p_2 = \mathrm{ind}(B)$. The set of all solutions of the equation* (3.29) *is given by*

$$X = (A^{p_1})^D G (B^{p_2})^D + Y - A^D A Y B B^D, \quad Y \in \mathbb{C}^{n\times n}. \tag{3.30}$$

Proof. First we show that (3.30) is a solution of the system

$$\begin{aligned} A^{2p_1} X B^{2p_2} &= A^{2p_1}(A^{p_1})^D G (B^{p_2})^D B^{2p_2} + A^{2p_1} Y B^{2p_2} - A^{2p_1} A^D A Y B B^D B^{2p_2} \\ &= A^{p_1} A A^D G B^D B B^{p_2} + A^{2p_1} Y B^{2p_2} - A^{2p_1} Y B^{2p_2} \\ &= A^{p_1} G B^{p_2}. \end{aligned}$$

Moreover, let $Y \in \mathbb{C}^{n\times n}$ be arbitrary solution of (3.29), i.e., let $A^{2p_1} Y B^{2p_2} = A^{p_1} G B^{p_2}$. We can write

$$Y = (A^{p_1})^D G (B^{p_2})^D + Y - (A^{p_1})^D G (B^{p_2})^D.$$

Since,

$$\begin{aligned} (A^{p_1})^D G (B^{p_2})^D &= ((A^{p_1})^D)^2 A^{p_1} G B^{p_2} ((B^{p_2})^D)^2 \\ &= (A^{2p_1})^D A^{2p_1} Y B^{2p_2} (B^{2p_2})^D = A^D A Y B B^D, \end{aligned}$$

we complete the proof. □

The following theorem gives the initial idea for finding the Drazin-inverse solution of the matrix equation (3.2).

Theorem 3.2.7. *Let $A, B, G \in \mathbb{C}^{n\times n}$ be such that $p_1 = \mathrm{ind}(A)$, $p_2 = \mathrm{ind}(B)$ and let P be the Jordan basis of the matrix A and Q be the Jordan basis of matrix B. If $\hat{X}$ satisfies*

$$\|G - A^{p_1} \hat{X} B^{p_2}\|_{PQ}^2 = \min_X \|G - A^{p_1} X B^{p_2}\|_{PQ}^2 \tag{3.31}$$

then $\hat{X}$ is a solution of the equation (3.29)*. Moreover, if $\mathcal{R}(G|_{\mathcal{N}(B^{p_2})}) \subset \mathcal{N}(A^{p_1})$ the opposite statement is also valid.*

Proof. Let

$$P^{-1} A^{p_1} P = \begin{bmatrix} J_A & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A^{p_1}) \\ \mathcal{N}(A^{p_1}) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^{p_1}) \\ \mathcal{N}(A^{p_1}) \end{bmatrix}; \quad Q^{-1} B^{p_2} Q = \begin{bmatrix} J_B & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(B^{p_2}) \\ \mathcal{N}(B^{p_2}) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(B^{p_2}) \\ \mathcal{N}(B^{p_2}) \end{bmatrix},$$

be the Jordan matrices of A^{p_1} and B^{p_2}, respectively. Let

$$P^{-1} G Q = \begin{bmatrix} G_1 & G_2 \\ G_3 & G_4 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(B^{p_2}) \\ \mathcal{N}(B^{p_2}) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^{p_1}) \\ \mathcal{N}(A^{p_1}) \end{bmatrix}$$

and

$$P^{-1} X Q = \begin{bmatrix} X_1 & X_2 \\ X_3 & X_4 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(B^{p_2}) \\ \mathcal{N}(B^{p_2}) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^{p_1}) \\ \mathcal{N}(A^{p_1}) \end{bmatrix}.$$

Since

$$\begin{aligned} P^{-1} P_{\mathcal{R}(A^{p_1})} G P_{\mathcal{R}(B^{p_2})} Q &= P^{-1} A^D A G B B^D Q = P^{-1} A^D A P P^{-1} G Q Q^{-1} B B^D Q \\ &= \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} G_1 & G_2 \\ G_3 & G_4 \end{bmatrix} \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} G_1 & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(B^{p_2}) \\ \mathcal{N}(B^{p_2}) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A^{p_1}) \\ \mathcal{N}(A^{p_1}) \end{bmatrix}, \end{aligned}$$

we have that,

$$\begin{aligned} P^{-1}(G - A^{p_1}XB^{p_2})Q &= \begin{bmatrix} G_1 & G_2 \\ 0 & 0 \end{bmatrix} - P^{-1}A^{p_1}XB^{p_2}Q + \begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix} \\ &= P^{-1}A^DAGBB^DQ - P^{-1}A^{p_1}XB^{p_2}Q + \begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix}. \end{aligned} \tag{3.32}$$

Obviously, $\mathcal{R}\left(\begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix}\right) \subset \mathcal{N}(A^{p_1})$. Also,

$$\mathcal{R}\left(P^{-1}A^DAGBB^DQ - P^{-1}A^{p_1}XB^{p_2}Q + \begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix}\right) = \mathcal{R}\left(\begin{bmatrix} G_1 - J_AX_1J_B & G_2 \\ 0 & 0 \end{bmatrix}\right) \subset \mathcal{R}(A^{p_1}).$$

Following the results from Proposition 3.2.3 and by using (3.32) we get

$$\begin{aligned} \|G - A^{p_1}XB^{p_2}\|_{PQ}^2 &= \left\| P^{-1}A^DAGBB^DQ - P^{-1}A^{p_1}XB^{p_2}Q + P^{-1}P\begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix} + P^{-1}P\begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix} \right\|_F^2, \\ &= \left\| A^DAGBB^DQ - A^{p_1}XB^{p_2}Q + P\begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix} + P\begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix} \right\|_P^2, \\ &= \left\| A^DAGBB^DQ - A^{p_1}XB^{p_2}Q + P\begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix} \right\|_P^2 + \left\| P\begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix} \right\|_P^2. \end{aligned}$$

Or equivalently,

$$\|G - A^{p_1}XB^{p_2}\|_{PQ}^2 = \left\| A^DAGBB^D - A^{p_1}XB^{p_2} + P\begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix}Q^{-1} \right\|_{PQ}^2 + \left\| P\begin{bmatrix} 0 & 0 \\ G_3 & G_4 \end{bmatrix}Q^{-1} \right\|_{PQ}^2.$$

Evidently $\|G - A^{p_1}XB^{p_2}\|_{PQ}^2$ attains its minimum for

$$A^{p_1}XB^{p_2} = A^DAGBB^D + P\begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix}Q^{-1}. \tag{3.33}$$

Therefore, we prove that the initial statement given by (3.31) is equivalent with the identity (3.33).

In what follows we show that from equation (3.33) follows the equation (3.29). If X satisfies (3.33), evidently, multiplying the equation (3.33) by A^{p_1} from the left and by B^{p_2} on the right, on the both hand sides, we obtain

$$A^{2p_1}XB^{2p_2} = A^{p_1}A^DAGBB^DB^{p_2} + P\begin{bmatrix} J_A & 0 \\ 0 & 0 \end{bmatrix}P^{-1}P\begin{bmatrix} 0 & G_2 \\ 0 & 0 \end{bmatrix}Q^{-1}Q\begin{bmatrix} J_B & 0 \\ 0 & 0 \end{bmatrix}Q^{-1} = A^{p_1}GB^{p_2}.$$

which proves that it satisfies (3.29).

Let us suppose that $\mathcal{R}(G|_{\mathcal{N}(B^{p_2})}) \subset \mathcal{N}(A^{p_1})$. Then since $G_2 : \mathcal{N}(B^{p_2}) \to \mathcal{R}(A^{p_1})$ it follows that $G_2 = 0$, i.e., the equality (3.33) becomes

$$A^{p_1}XB^{p_2} = A^DAGBB^D. \tag{3.34}$$

Let X satisfies (3.29), i.e., $A^{2p_1}XB^{2p_2} = A^{p_1}GB^{p_2}$. By multiplying with $(A^{p_1})^D$ from the left and with $(B^{p_2})^D$ on the right, on the both hand sides, and using the facts that $(A^{p_1})^DA^{p_1} = A^DA$, $(B^{p_2})^DB^{p_2} = B^DB$, $A^DA^{p_1+1} = A^{p_1}$ and $B^DB^{p_2+1} = B^{p_2}$ we obtain (3.34). □

Corollary 3.2.3. *Let $A, B, G \in \mathbb{C}^{n\times n}$ be such that $p_1 = \mathrm{ind}(A)$, $p_2 = \mathrm{ind}(B)$ and let P be the Jordan basis of the matrix A and Q be the Jordan basis of matrix B. If $\hat{X}$ is a PQ-norm least-squares solution of the matrix equation $A^{p_1}XB^{p_2} = G$, then*

$$A^{p_1-1}\hat{X}B^{p_2-1} = A^DGB^D + A^{p_1-1}YB^{p_2-1} - A^DA^{p_1}YB^{p_2}B^D, \quad Y \in \mathbb{C}^{n\times n}.$$

Moreover, if $\mathcal{R}(\hat{X}) \subset \mathcal{R}(A^{p_1})$ and $\mathcal{N}(B^{p_2}) \subset \mathcal{N}(\hat{X})$ then $A^{p_1-1}\hat{X}B^{p_2-1} = A^DGB^D$.

Proof. If $\hat{X}$ satisfies (3.31), then it is a solution of the equation (3.29), i.e., it is of the form

$$\hat{X} = (A^{p_1})^DG(B^{p_2})^D + Y - A^DAYBB^D, \quad Y \in \mathbb{C}^{n\times n}.$$

Then

$$\begin{aligned} A^{p_1-1}\hat{X}B^{p_2-1} &= A^{p_1-1}(A^{p_1})^DG(B^{p_2})^DB^{p_2-1} + A^{p_1-1}YB^{p_2-1} - A^{p_1-1}A^DAYBB^DB^{p_2-1} \\ &= A^DGB^D + A^{p_1-1}YB^{p_2-1} - A^DA^{p_1}YB^{p_2}B^D \end{aligned}$$

Additionally, let $\mathcal{R}(\hat{X}) \subset \mathcal{R}(A^{p_1})$ and $\mathcal{N}(B^{p_2}) \subset \mathcal{N}(\hat{X})$, then it follows that $\mathcal{R}(A^{p_1-1}YB^{p_2-1}) \subset \mathcal{R}(A^{p_1})$ and $\mathcal{N}(B^{p_2}) \subset \mathcal{N}(A^{p_1-1}YB^{p_2-1})$. Consequently,

$$P_{\mathcal{R}(A^{p_1}),\mathcal{N}(A^{p_1})}A^{p_1-1}YB^{p_2-1} = A^{p_1-1}YB^{p_2-1}$$

and

$$A^{p_1-1}YB^{p_2-1}P_{\mathcal{R}(B^{p_2}),\mathcal{N}(B^{p_2})} = A^{p_1-1}YB^{p_2-1}.$$

Finally we obtain,

$$\begin{aligned} A^DA^{p_1}YB^{p_2}B^D &= A^{p_1-1}AA^DYBB^DB^{p_2-1} = P_{\mathcal{R}(A^{p_1}),\mathcal{N}(A^{p_1})}A^{p_1-1}YB^{p_2-1}P_{\mathcal{R}(B^{p_2}),\mathcal{N}(B^{p_2})} \\ &= A^{p_1-1}YB^{p_2-1} \end{aligned}$$

and thus, $A^{p_1-1}\hat{X}B^{p_2-1} = A^DGB^D$, which completes the proof. □

Considering these results, we are moving to the problem of computing the vector A^DGB^D. For that purpose let us focus to the following problem

$$\min_X f(X) = \min_X \|A^{2p_1}XB^{2p_2} - G\|^2_{PQ}, \tag{3.35}$$

whose solution we want to find. The characterization of the Drazin-inverse solution of the matrix equation (3.2), given in terms of the minimizer of the problem (3.35), is stated in the following theorem.

Theorem 3.2.8. *Let $A, B, G \in \mathbb{C}^{n\times n}$ be such that $p_1 = \mathrm{ind}(A), p_2 = \mathrm{ind}(B)$. Let P be the Jordan basis of the matrix A and Q be the Jordan basis of matrix B. If $\hat{X}$ is a minimizer of the functional* (3.35)*, then the following holds*

$$A^{2p_1-1}\hat{X}B^{2p_2-1} = A^DGB^D. \tag{3.36}$$

Proof. Let the matrix $\hat{X}$ be a minimizer of the functional (3.35), then it is a PQ-norm least-squares solution of the system $A^{2p_1}XB^{2p_2} = G$. Thus $\hat{Y} = A^{p_1}\hat{X}B^{p_2}$ is the minimizer of the functional

$$\min_{Y} \|A^{p_1}YB^{p_2} - G\|_{PQ}^2$$

such that $\mathcal{R}(\hat{Y}) \subset \mathcal{R}(A^{p_1})$ and $\mathcal{N}(B^{p_2}) \subset \mathcal{N}(\hat{Y})$. According to Corollary 3.2.3, it follows that

$$A^D G B^D = A^{p_1-1}\hat{Y}B^{p_2-1} = A^{p_1-1}A^{p_1}\hat{X}B^{p_2}B^{p_2-1} = A^{2p_1-1}\hat{X}B^{2p_2-1},$$

which completes the proof. □

Corollary 3.2.4. *Let $A, B \in \mathbb{C}^{n\times n}$ be such that $p_1 = \mathrm{ind}(A)$, $p_2 = \mathrm{ind}(B)$, let P be the Jordan basis of the matrix A and Q be the Jordan basis of the matrix B.*

a) *If $\hat{X} \in \mathbb{C}^{n\times n}$ is a minimizer of the functional*

$$f(X) = \|A^{2p_1}X - I\|_P^2 = \|P^{-1}A^{2p_1}X - P^{-1}\|_F^2.$$

Then

$$A^D = A^{2p_1-1}\hat{X}. \tag{3.37}$$

b) *If $\hat{X} \in \mathbb{C}^{n\times n}$ is a minimizer of the functional*

$$f(X) = \|XB^{2p_2} - I\|_{IQ}^2,$$

Then

$$B^D = \hat{X}B^{2p_2-1}. \tag{3.38}$$

c) *If $\hat{X}$ be the minimizer of the functional*

$$f(X) = \|A^{2p_1}XB^{2p_2} - I\|_{PQ}^2.$$

Then

$$A^D B^D = A^{2p_1-1}\hat{X}B^{2p_2-1}. \tag{3.39}$$

Proposition 3.2.4. *[137] Let $S \in \mathbb{C}^{m\times n}$, $T \in \mathbb{C}^{k\times s}$ and $Q \in \mathbb{C}^{m\times s}$. Then the matrix $S^\dagger Q T^\dagger$ minimizes the function*

$$g(X) = \|SXT - Q\|_F^2. \tag{3.40}$$

Theorem 3.2.9. *Let $A, B \in \mathbb{C}^{n\times n}$ be such that $p_1 = \mathrm{ind}(A), p_2 = \mathrm{ind}(B)$, P be the Jordan basis of A and Q be the Jordan basis of A.*

a) $A^D = A^{2p_1-1}(P^{-1}A^{2p_1})^\dagger P^{-1}$.

b) $B^D = Q(B^{2p_2}Q)^\dagger B^{2p_2-1}$.

c) $A^D G B^D = A^{2p_1-1}(P^{-1}A^{2p_1})^\dagger P^{-1}GQ(B^{2p_2}Q)^\dagger B^{2p_2-1}$.

Proof.

a) The minimum-norm least-squares solution of the function

$$f(X) = \|A^{2p_1}X - I\|_P^2 = \|P^{-1}A^{2p_1}X - P^{-1}\|_F^2,$$

is the vector $(P^{-1}A^{2p_1})^\dagger P^{-1}$. The rest is obvious from Theorem 3.2.8.

b) The minimum-norm least-squares solution of the function

$$f(X) = \|XB^{2p_2} - I\|_{IQ}^2 = \|XB^{2p_2}Q - Q\|_F^2,$$

is the vector $Q(B^{2p_2}Q)^\dagger P^{-1}$. The rest is obvious from Theorem 3.2.8.

c) Similarly.

□

3.3 Least-square properties of $A^{(2)}_{T,S}$-inverse solutions

The purpose of the present section, is to generalize the results which are joint for the Moore-Penrose inverse solution and the Drazin-inverse solution. This, of course, can be done with the $A^{(2)}_{T,S}$- inverse solution, since many generalized inverses can be represented via $A^{(2)}_{T,S}$- inverses. On this way, the minimality results will naturally transfer to the weighted Moore-Penrose inverse, the Bott-Duffin inverse etc.

For the purpose of this section, we use the following notation:
The matrices $A \in \mathbb{C}^{m\times n}$ and $R \in \mathbb{C}^{n\times m}$ are such that

$$\operatorname{rank}(AR) = \operatorname{rank}(RA) = \operatorname{rank}(R) \quad \text{and} \quad A\mathcal{R}(R) \oplus \mathcal{N}(R) = \mathbb{C}^m.$$

The vector $b \in \mathbb{C}^n$, and where appropriate for simplicity we use

$$T := \mathcal{R}(R), \quad S := \mathcal{N}(R) = \mathcal{N}(AR), \quad X := A^{(2)}_{T,S} = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}.$$

With this notation, clearly $A(T) = \mathcal{R}(AR)$ and $\mathcal{N}(XA) = \mathcal{N}(RA)$.
For a given invertible matrix $B \in \mathbb{C}^{n\times n}$ we denote $\|x\|_B = \|B^{-1}x\|$.

Lemma 3.3.1. *The set of all solutions of the equation*

$$RAx = Rb, \tag{3.41}$$

is given by

$$x = A^{(2)}_{T,S}b + \mathcal{N}(RA). \tag{3.42}$$

Proof. Since $Rb \in \mathcal{R}(R) = \mathcal{R}(RA)$ we have that the system (3.41) is consistent (see also [97]). The set of all solutions of the homogeneous system $RAx = 0$ is the same as $\mathcal{N}(RA)$. Moreover, since

$$RAA^{(2)}_{T,S} = RP_{A(T),S} = RP_{\mathcal{R}(AR),\mathcal{N}(R)} = R,$$

it follows that $A^{(2)}_{T,S}b$ is a particular solution of the analyzed system, which finalizes the proof. □

Corollary 3.3.1. *The vector $A^{(2)}_{T,S}b$ is the unique solution of* (3.41) *that belongs to T.*

Proof. Since $A^{(2)}_{T,S}b \in T$, and also $\mathbb{C}^n = T \oplus \mathcal{N}(RA)$, the proof follows immediately from Lemma 3.3.1, i.e., from equation (3.42). □

As we stated earlier the matrix A can be represented on the following way:

$$A \sim \begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} : \begin{bmatrix} T \\ \mathcal{N}(XA) \end{bmatrix} \to \begin{bmatrix} A(T) \\ S \end{bmatrix}, \tag{3.43}$$

where A_1 is invertible, and

$$A^{(2)}_{T,S} \sim \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} A(T) \\ S \end{bmatrix} \to \begin{bmatrix} T \\ \mathcal{N}(XA) \end{bmatrix},$$

Let $Q = \{q_1, \ldots, q_n\}$ and $W = \{w_1, \ldots, w_m\}$ be the basis of $\mathbb{C}^n$ and $\mathbb{C}^m$ respectively, with respect to which the matrix A has the representation given with (3.43).

Corollary 3.3.2. *For all solutions $\hat{x}$ of the system of linear equations* (3.1) *the inequality $\|\hat{x}\|_Q \geq \|A^{(2)}_{T,S}b\|_Q$ holds.*

Proof. Let $\hat{x}$ be a solution of the system $Ax = b$. Then $\hat{x}$ is a solution of the system (3.41). From Lemma 3.3.1, $\hat{x}$ can be represented as

$$\hat{x} = A^{(2)}_{T,S}b + y \in A^{(2)}_{T,S}b + \mathcal{N}(XA) \subseteq T \oplus \mathcal{N}(XA).$$

So, clearly from the orthogonality of the spaces T and $\mathcal{N}(XA)$, we have that

$$\|\hat{x}\|^2_Q = \|A^{(2)}_{T,S}b\|^2_Q + \|y\|^2_Q.$$

From which follows that $\|\hat{x}\|_Q > \|A^{(2)}_{T,S}b\|_Q$ unless $\hat{x} = A^{(2)}_{T,S}b$. □

Corollary 3.3.3. *The vector $A^{(2)}_{T,S}b$ is the unique solution of the equation* (3.41) *of minimum Q-norm.*

Corollary 3.3.4. *If $b \in \mathcal{R}(AR)$ then $A^{(2)}_{T,S}b$ is the unique solution of the system* (3.1) *that belongs to the subspace T.*

Lemma 3.3.2. *If $b \in \mathcal{R}(AR)$ then the vector*

$$x = R(AR)^{(1,3)}b$$

is $A^{(2)}_{T,S}$-inverse solution of the given system (3.1).

Proof. Since $b \in \mathcal{R}(AR) \subseteq \mathcal{R}(A)$ the given system has a solution and:

$$Ax = AR(AR)^{(1,3)}b = P_{\mathcal{R}(AR),S}b = b,$$

where $\mathcal{R}(AR) \oplus S = \mathbb{C}^n$. So, $x = R(AR)^{(1,3)}b$ is a solution of the system (3.1) and also $x \in T$. According to the previous lemma $A^{(2)}_{T,S}$-inverse solution is the unique solution of the system (3.1) that belongs in T. Consequently,

$$x = A^{(2)}_{T,S}b = R(AR)^{(1,3)}b \tag{3.44}$$

which completes the proof □

Therefore, we assume that $b \in \mathcal{R}(AR)$. Based on the previous considerations, we are going to analyze the following problem:

$$\min_x f(x) = \min_x \frac{1}{2}\|ARx - b\|^2. \tag{3.45}$$

Theorem 3.3.1. *Let $A \in \mathbb{C}^{m\times n}$ and $b \in \mathbb{C}^n$ be such that $b \in \mathcal{R}(AR)$. Let $\hat{x}$ be the minimizer of the function* (3.45) *then*

$$R\hat{x} = A^{(2)}_{T,S}b = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}b. \tag{3.46}$$

Proof. From [97, 98], it follows that a least-squares solution of the system $ARx = b$, which is also its solution, is given by

$$\hat{x} = (AR)^{(1,3)}b + \left(I - (AR)^{(1,3)}AR\right)y,$$

where $y \in \mathbb{C}^n$ is an arbitrary vector. Then

$$\begin{aligned} R\hat{x} &= R(AR)^{(1,3)}b + R(I - (AR)^{(1,3)}AR)y \\ &= A^{(2)}_{T,S}b + R(I - (AR)^{(1,3)}AR)y. \end{aligned} \tag{3.47}$$

Since $\operatorname{rank}(AR) = \operatorname{rank}(R)$, we have that

$$R(AR)^{(1,3)}AR = R.$$

Finally, from the second part of equation (3.47) immediately follows $R\hat{x} = A^{(2)}_{T,S}b = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}b$. □

So similarly as previous, we showed that the problem of finding the $A^{(2)}_{T,S}$-inverse solution can be reduced to the problem of finding a minimum of the function (3.45).

3.4 Full-rank factorization of generalized inverses

The representaion of (2)-inverses, with the general form $F(GAF)^{-1}G$ is frequently applied tool in the numerical calculations. For example, this representation is investigated with the purpose of defining a deterministic representation of $A^{(2)}_{T,S}$ inverses [117], i.e., of the set $A\{2\}_s$ [125]. At the same time this representation has been used for the construction of the successive matrix squaring (SMS) method [126]. Based on these ideas in this section we define the full-rank representation of $\{2,3\}$ and $\{2,4\}$-inverse of a given matrix, with a given range and null space, as a special case of the full-rank representation of the $A^{(2)}_{T,S}$-inverse. Also, it is defined a full-rank representation of $A\{2,3\}_s$ i $A\{2,4\}_s$, as a special case of the full-rank representation of the sets $A\{2\}_s$, where $s \leq r$ and r is the rank A.

There exist many full-rank representations for different generalized inverses of prescribed rank. For convenience some of them are comprised in the following proposition [45, 108, 117, 122, 123, 124].

Proposition 3.4.1. *Let $A \in \mathbb{C}_r^{m\times n}$ be an arbitrary matrix, $0 < s \leq r$ and $A = MN$ is a full-rank factorization of A. The following general representations for some classes of generalized inverses are valid:*

$$\begin{aligned}
A\{2\}_s &= \{F(GAF)^{-1}G \mid F\in\mathbb{C}^{n\times s},\, G\in\mathbb{C}^{s\times m},\ \mathrm{rank}(GAF)=s\};\\
A\{2\} &= \cup_{s=0}^{r} A\{2\}_s;\\
A\{2,4\}_s &= \left\{(VA)^{\dagger}V \mid V\in\mathbb{C}^{s\times m},\ VA\in\mathbb{C}_s^{s\times n}\right\};\\
A\{2,3\}_s &= \left\{U(AU)^{\dagger} \mid U\in\mathbb{C}^{n\times s},\ AU\in\mathbb{C}_s^{m\times s}\right\};\\
A\{1,2\} &= \{F(GAF)^{-1}G \mid F\in\mathbb{C}^{n\times r},\, G\in\mathbb{C}^{r\times m},\ \mathrm{rank}(GAF)=r\} = A\{2\}_r;\\
A\{1,2,4\} &= \{N^*(VAN^*)^{-1}V \mid V\in\mathbb{C}^{r\times m},\ \mathrm{rank}(VAN^*)=r\} = \left\{(VA)^{\dagger}V \mid VA\in\mathbb{C}_r^{r\times n}\right\};\\
A\{1,2,3\} &= \{U(M^*AU)^{-1}M^* \mid U\in\mathbb{C}^{n\times r},\ \mathrm{rank}(M^*AU)=r\} = \left\{U(AU)^{\dagger} \mid AU\in\mathbb{C}_r^{m\times r}\right\};\\
A^{\dagger} &= N^*(M^*AN^*)^{-1}M^*;\\
\textit{if } m=n,\quad A^D &= F_{A^l}(G_{A^l}\ A\ F_{A^l})^{-1}G_{A^l},\ A^l = F_{A^l}G_{A^l},\ l\geq \mathrm{ind}(A),\ A^l = F_{A^l}G_{A^l}.
\end{aligned}$$

It is known that the sets $A\{2\}_0$, $A\{2,3\}_0$, $A\{2,4\}_0$ and $\mathrm{A}\{2,3,4\}_0$ are identical and contain a single element, the $n\times m$ zero matrix. For this purpose, it suffices to consider only positive s.

Full-rank representation of $\{2\}$-inverses with prescribed range and null space is determined in the next proposition, which originated in [117].

Proposition 3.4.2. [117] *Let $A\in\mathbb{C}_r^{m\times n}$, T be a subspace of $\mathbb{C}^n$ of dimension $s\leq r$ and let S be a subspace of $\mathbb{C}^m$ of dimensions $m-s$. In addition, suppose that $R\in\mathbb{C}^{n\times m}$ satisfies $\mathcal{R}(R)=T, \mathcal{N}(R)=S$. Let R has an arbitrary full-rank decomposition, that is $R=FG$. If A has a $\{2\}$-inverse $A_{T,S}^{(2)}$, then:*

(1) *GAF is an invertible matrix;*

(2) *$A_{T,S}^{(2)} = F(GAF)^{-1}G$.*

Representation of outer inverses, in the general form $F(GAF)^{-1}G$, is applicable in numerical calculations. For example, such a representation has been exploited to define the determinantal representation of $A_{T,S}^{(2)}$ inverse in [117] or the set $A\{2\}_s$ in [125]. Also, this representation has been used in the construction of the general successive matrix squaring algorithm for computing $A_{T,S}^{(2)}$ [126] or in the block representation of the set $A\{2\}_s$ [122].

3.4.1 Full-rank factorization of $\{2,4\}$ and $\{2,3\}$-inverses

The general representations of $\{2,4\}$ and $\{2,3\}$-inverses of the form $(VA)^{\dagger}V$ and $U(AU)^{\dagger}$, respectively, are not widely exploited in the literature. Several modifications of the hyper-power method are used in computation of $\{2,3\}$ and $\{2,4\}$-inverses in [123]. Various representations of $\{2,3\}$ and $\{2,4\}$-inverses with prescribed range and null space has been investigated [21, 117, 149, 150]. The expressions for $\{2,3\}$ and $\{2,4\}$-inverses of a normal matrix by its Schur decomposition are discussed in [151]. But, these representations are not exploited in developing of some effective computational procedures.

For this reason, our main goal is to determine full-rank representations of $\{2,4\}$ and $\{2,3\}$-inverses with prescribed range and null space as particular cases of the full-rank representation for generalized inverses $A_{T,S}^{(2)}$ [128]. We also define full-rank representations of the sets $A\{2,4\}_s$ and $A\{2,3\}_s$ as particular cases of the full-rank representation of the set $A\{2\}_s$.
In Lemma 3.4.1 we exactly distinguish sets $A\{2,4\}_s$ and $A\{2,3\}_s$ as subsets of the set $A\{2\}_s$.

Lemma 3.4.1. *Let $A\in\mathbb{C}_r^{m\times n}$ be the given matrix and $0<s\leq r$ a chosen integer. Assume that V and U are two arbitrary matrices satisfying* $\mathrm{rank}(VA)=\mathrm{rank}(V)$ *and* $\mathrm{rank}(AU)=\mathrm{rank}(U)$. *Then the following statements are valid:*

$$\begin{aligned}
&\text{(a)}\quad A\{2,4\}_s=\left\{(VA)^*\left(VA(VA)^*\right)^{-1}V\mid V\in\mathbb{C}_s^{s\times m}\right\};\\
&\text{(b)}\quad A\{2,3\}_s=\left\{U\left((AU)^*AU\right)^{-1}(AU)^*\mid U\in\mathbb{C}_s^{n\times s}\right\};\\
&\text{(c)}\quad A\{1,2,4\}=\left\{(VA)^*\left(VA(VA)^*\right)^{-1}V\mid V\in\mathbb{C}_r^{r\times m}\right\}=A\{2,4\}_r;\\
&\text{(d)}\quad A\{1,2,3\}=\left\{U\left((AU)^*AU\right)^{-1}(AU)^*\mid U\in\mathbb{C}_r^{n\times r}\right\}=A\{2,3\}_r.
\end{aligned}$$

Proof. The proof for the parts (a), (b) follows immediately from

$$(VA)^*(VA(VA)^*)^{-1}V=(VA)^*\left((VA)^*\right)^\dagger(VA)^\dagger V=(VA)^\dagger V$$

and

$$U((AU)^*AU)^{-1}(AU)^*=U(AU)^\dagger((AU)^*)^\dagger(AU)^*=U(AU)^\dagger$$

together with the general representations of the sets $A\{2,4\}_s$ and $A\{2,3\}_s$ from Proposition 3.4.1.

Representations (c) and (d) follows from well-known fact that $X=F(GAF)^{(1)}G\in A\{1\}$ if and only if $\mathrm{rank}(GAF)=r$ (see, for example [136], Theorem 1.3.7). □

Therefore, we conclude that $\{2,4\}$-inverses can be derived from the set of outer inverses in the particular case $F=(VA)^*$ and $G=V$. Similarly, $\{2,3\}$-inverses can be derived in the particular case $F=U$ and $G=(AU)^*$. Practically, the following general representations hold:

$$\begin{aligned}
A\{2,4\}_s&=\{A^{(2)}\in\mathbb{C}_s^{n\times m}\mid\ A^{(2)}=F(GAF)^{-1}G,\ F=(VA)^*,G=V,\ V\in\mathbb{C}^{s\times m},\ VA\in\mathbb{C}_s^{s\times n}\};\\
A\{2,3\}_s&=\{A^{(2)}\in\mathbb{C}_s^{n\times m}\mid\ A^{(2)}=F(GAF)^{-1}G,\ G=(AU)^*,F=U,\ U\in\mathbb{C}^{n\times s},\ AU\in\mathbb{C}_s^{m\times s}\}.
\end{aligned}$$

In the rest of this section we give an answer to the problem of finding appropriate values for the matrix R which lead to $\{2,4\}$ and $\{2,3\}$-inverses with prescribed range and null space. It follows from the following two statements.

Motivated by Proposition 3.4.2, we find alternative representations of $\{2,4\}$ and $\{2,3\}$-inverses with prescribed range and null space in the general form which characterizes the set of outer inverses: $F(GAF)^{-1}G$, where $F\in\mathbb{C}^{n\times s}$, and $G\in\mathbb{C}^{s\times m}$, $s\leq\mathrm{rank}(A)$.

Theorem 3.4.1. *For arbitrary matrix $A\in\mathbb{C}_r^{m\times n}$ and arbitrary integer s satisfying $0<s\leq r$ we have*

$$\text{(a)}\quad A\{2,4\}_s=\left\{A^{(2,4)}_{\mathcal{N}(VA)^\perp,\mathcal{N}(V)}\mid V\in\mathbb{C}_s^{s\times m},\ \mathrm{rank}(VA)=\mathrm{rank}(V)\right\}\tag{3.48}$$

$$\text{(b)}\quad A\{2,3\}_s=\left\{A^{(2,3)}_{\mathcal{R}(U),\mathcal{R}(AU)^\perp}\mid U\in\mathbb{C}_s^{n\times s},\ \mathrm{rank}(AU)=\mathrm{rank}(U)\right\}.\tag{3.49}$$

Proof. Let us choose an arbitrary element $X\in A\{2,4\}_s$. According to Proposition 3.4.1, X is of the form

$$X=(VA)^\dagger V,\ V\in\mathbb{C}_s^{s\times m},\ \mathrm{rank}(VA)=\mathrm{rank}(V).$$

Since

$$\mathrm{rank}(X)=\mathrm{rank}((VA)^\dagger)=\mathrm{rank}(VA)=\mathrm{rank}(V)$$

we conclude

$$\mathcal{R}(X) = \mathcal{R}((VA)^*) = \mathcal{N}(VA)^\perp,$$
$$\mathcal{N}(X) = \mathcal{N}(V)$$

and verify

$$X \in \left\{ A^{(2,4)}_{\mathcal{N}(VA)^\perp,\mathcal{N}(V)} |\ V \in \mathbb{C}^{s\times m}_s,\ \operatorname{rank}(VA) = \operatorname{rank}(V) \right\}.$$

To verify the opposite inclusion, assume that

$$X = A^{(2,4)}_{\mathcal{N}(VA)^\perp,\mathcal{N}(V)}$$

for the selected matrix $V \in \mathbb{C}^{s\times m}_s$ satisfying $\operatorname{rank}(VA) = \operatorname{rank}(V)$. According to Proposition 3.4.2, X is of the form

$$X = F(GAF)^{-1}G, \quad \mathcal{R}(F) = \mathcal{N}(VA)^\perp = \mathcal{R}\left((VA)^*\right), \quad \mathcal{N}(G) = \mathcal{N}(V).$$

For example, it is possible to choose $F = (VA)^*$, $G = V$. According to Lemma 3.4.1, we get $X \in A\{2,4\}_s$.

The part (a) of the proof is completed. The dual statement for $\{2,3\}$-inverses can be verified in a similar way. □

A correlation between outer inverses with prescribed range and null space with $\{2,4\}$ and $\{2,3\}$-inverses with prescribed range and null space is determined in the next statement.

Corollary 3.4.1. *Let $A \in \mathbb{C}^{m\times n}_r$ be a given matrix and $R \in \mathbb{C}^{n\times m}_s$, $s \le r$ be arbitrary, but fixed matrix. Assume that $F \in \mathbb{C}^{n\times s}_s$ and $G \in \mathbb{C}^{s\times m}_s$ form the full-rank factorization $R = FG$. Let $\mathcal{R}(R) = T$ be a subspace of $\mathbb{C}^n$ of dimension $s \le r$ and $\mathcal{N}(R) = S$ be a subspace of $\mathbb{C}^m$ of dimensions $m - s$. Then the following statements are satisfied:*

(1) *In the case $R = (VA)^*V$, (or $F = (VA)^*$, $G = V$, $V \in \mathbb{C}^{s\times m}$ is an arbitrary matrix), the outer inverse $A^{(2)}_{T,S} = F(GAF)^{-1}G$ reduces to the $\{2,4\}$-inverse*

$$A^{(2,4)}_{\mathcal{R}((VA)^*),S} = A^{(2,4)}_{\mathcal{N}(VA)^\perp,\mathcal{N}(V)} = (VA)^*(VA(VA)^*)^{-1}V.$$

(2) *In the case $R = U(AU)^*$, (or $F = U$, $G = (AU)^*$, $U \in \mathbb{C}^{n\times s}$ is an arbitrary matrix), the outer inverse $A^{(2)}_{T,S} = F(GAF)^{-1}G$ reduces to the $\{2,3\}$-inverse*

$$A^{(2,3)}_{T,\mathcal{N}((AU)^*)} = A^{(2,3)}_{\mathcal{R}(U),\mathcal{R}(AU)^\perp} = U((AU)^*AU)^{-1}(AU)^*.$$

Chapter 4

Iterative methods for computing generalized inverses

4.1 Intoduction

It is well known, that there is not an easy way to obtain, deterministically, the ordinary inverse of a given nonsingular square matrix. As a result, scientists stepped into a different approach, i.e., approximate determination of the inverse. The same logic is also followed for the generalized inverses. Namely, it is almost impossible to obtain generalized inverse of a matrix, deterministically, especially when we talk about solutions of practical problems, where the arisen matrices are usually of very large dimensions.

Penrose in his paper was the first who showed the close connection between the Moore-Penrose inverse and the least-squares solution problem of a system of linear equations. The last represents a special case of the nonlinear optimization problems. Additionally, the discovered minimal properties of the solution of a system of linear equations, obtained with the usage of the Moore-Penrose inverse, brought to intensive utilization of the optimization methods.

At the same time, the classical iterative methods for computing the ordinary inverse of a given matrix, were extensively modified, for the purpose to serve as a tool for computing the generalized inverses of matrices.

Also, what is worth mentioning is that scientists do not, only, strive to determine one universal algorithm for generalized inverses calculation. Rather, when it is possible, they explore the structure of matrices, via defining groups of matrices with characteristics properties, and define methods which are appropriate only for one group. On that way, the theory of structured matrices, such as: Toeplitz matrices, Hankel matrices, Vandermonde matrices, Cauchy matrices etc., recognizes special algorithms for computing the generalized inverses of matrices. These algorithms beside the good properties of the iterative schemes, they also benefit from the specific structure of the matrices. As a result, the obtained algorithms are more competitive to the classical ones, for this type of matrices.

Nowadays, the theory of generalized inverses is consisted of vast spectra of different methods for approximate determination of generalized inverses of matrices. Some of them served as a motivation for the results presented in the Ph.D. dissertation. For the sake of completeness, we restate some of the well-known methods which are related to the new results.

4.2 The Moore-Penrose inverse

The purpose of this section is to present gradient iterative methods, for computing $\{1,3\}$-inverse solutions, $\{1,3\}$-inverses, the Moore-Penrose inverse solution and the Moore-Penrose inverse. The main idea for these methods comes from the *steepest descent method for singular linear operator equation* introduced in [97], together with the idea of SC method introduced in the second chapter. In the beginning we restate the method from [97], as well as necessary notions related to the Moore-Penrose inverse on Hilbert spaces.

4.2.1 Steepest descent method for singular linear operator equation

Let $\mathcal{H}$ and $\mathcal{K}$ be Hilbert spaces over the same scalars (real or complex). For any subspace S, we denote by $S^{\perp}$ the orthogonal complement of S and $\bar{S}$ the closure of S. Let A be a bounded linear operator on $\mathcal{H}$ into $\mathcal{K}$, and let A^* denote the adjoint of A, i.e., for all $x \in \mathcal{H}$, $y \in \mathcal{K}$,

$$< Ax, y >=< x, A^*y > .$$

Let $\mathcal{R}(A)$ and $\mathcal{N}(A)$ denote the range and null space of A respectively.

The steepest descent method for singular linear operator equation, presents an iterative scheme which converges to the solution of the normal equation

$$A^*Ax = A^*b, \tag{4.1}$$

of the system

$$Ax = b, \tag{4.2}$$

where A is bounded linear operator from a Hilbert space $\mathcal{H}$ to a Hilbert space $\mathcal{K}$, and $b \in \mathcal{K}$. The presented results which are related to the steepest descent method for singular linear operator equation, are taken form [97], only the notation is appropriately modified.

It is well known [131] that the results from the following proposition hold

Proposition 4.2.1. *Let $\mathcal{H}$ and $\mathcal{K}$ be Hilbert spaces, and $A : \mathcal{H} \to \mathcal{K}$ a linear bounded operator*

(1) $\mathcal{H} = \mathcal{N}(A) \oplus \mathcal{N}(A)^{\perp}$,

(2) $\mathcal{K} = \mathcal{N}(A^*) \oplus \mathcal{N}(A^*)^{\perp}$,

(3) $\overline{\mathcal{R}(A)}^{\perp} = \mathcal{N}(A^*)$, $\overline{\mathcal{R}(A^*)}^{\perp} = \mathcal{N}(A)^{\perp}$,

(4) $\mathcal{R}(A)$ *is closed if and only if* $\mathcal{R}(A^*)$ *is closed,*

(5) $\mathcal{N}(A^*A) = N(A)$, $\overline{\mathcal{R}(A)} = \overline{\mathcal{R}(AA^*)}$.

Let us recall Definition 2.3.1: A vector $\hat{x} \in \mathcal{H}$ is called a least-squares solution of the operator equation (4.2) if and only if $\|A\hat{x} - b\| = \inf\{\|Ax - b\| : x \in \mathcal{H}\}$.

Proposition 4.2.2. *[88] The vector $\hat{x}$ is a least-squares solution of the* (4.2) *if and only if $\hat{x}$ is a solution of the normal equation* (4.1).

Proposition 4.2.3. *Let $\mathcal{R}(A)$ be closed. Then the set S of all least-squares solutions of the system (4.2) is nonempty closed convex set, and there is a unique element y of minimal norm [131],i.e.,*

$$\|Ay - b\| \leq \|Ax - b\| \quad \text{for all} \quad x \in \mathcal{H},$$

and

$$\|y\| \leq \|\hat{x}\| \quad \text{for all} \quad \hat{x} \in S, \quad \hat{x} \neq y.$$

Definition 4.2.1. *The operator which assigns, to each $b \in \mathcal{K}$, the unique least-squares solution of minimal norm of (4.2) is called the Moore-Penrose inverse of A and is denoted by $A^{\dagger}$. $A^{\dagger}$ is linear and bounded operator.*

Lemma 4.2.1. *If $\mathcal{R}(A)$ is closed, then $\mathcal{R}(A^*) = \mathcal{R}(A^*A)$.*

Lemma 4.2.2. *If $\mathcal{R}(A)$ is closed, then the set S of all least-squares solutions of (4.2) is given by*

$$S = A^{\dagger}b \oplus \mathcal{N}(A).$$

Theorem 4.2.1. *Let A be a bounded linear operator on a Hilbert space $\mathcal{H}$ into $\mathcal{H}$ such that $\mathcal{R}(A)$ is closed. Let $x_0 \in \mathcal{H}$ be an initial approximation to a least-squares solution of the equation (4.2). Then the steepest descent iterative scheme given with*

$$x_{k+1} = x_k - \alpha_k A^*(Ax_k - b),$$

where

$$\alpha_k = \frac{\|A^*(Ax_k - b)\|}{\|AA^*(Ax_k - b)\|},$$

i.e., α_k is such that it minimizes the functional

$$f(x_{k+1}) = \frac{1}{2}\|Ax_{k+1} - b\|^2$$

at each step; converges to a least-squares solution of the system (4.2). Moreover, if $x_0 \in \mathcal{R}(A^*)$ then the steepest descent iterative scheme converges to $A^{\dagger}b$.

Inspired by the Nashed steepest descent method, the authors in [43] considered the iterative scheme of the same form in more general spaces, $\mathcal{C}^*$ algebras. The established proof for the linear convergence of the method differs from the one from [97], not only with respect to the observed spaces but also with respect to the used norms. Namely, the presented convergence theorem in [43] is given in terms of an operator norm while the convergence of the Nashed steepest descent scheme is given in terms of the Frobenius norm.

4.2.2 Application of the SC method for finding the Moore-Penrose inverse solution of an operator equation

In order to achieve the final goal, i.e., computation of the Moore-Penrose inverse of a given matrix, the analysis of the Moore-Penrose inverse of an operator on Hilbert space is inevitable. Namely, via generalization of a method for finding the Moore-Penrose inverse on Hilbert spaces, by treating a matrix as a linear map, enables efficient calculation of its Moore-Penrose inverse.

With the next theorem, we show that the Algorithm 2.3.1 presented in Chapter 2, not only converges, as it is already shown in Theorem 2.3.1, but it also converges to a least-squares solution of the analyzed system.

Theorem 4.2.2. *Let $\mathcal{H}$ and $\mathcal{K}$ be given Hilbert spaces and $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ be an operator such that $\mathcal{R}(A)$ is closed. The sequence $(x_k)_k$ determined by Algorithm 2.2.1 converges to a least-squares solution of the equation $Ax = b$.*

For an arbitrary initial approximation $x_0 \in \mathcal{H}$ the limit $\lim_{k\to\infty} x_k$ satisfies

$$\lim_{k\to\infty} x_k = A^\dagger b + (I - A^\dagger A)x_0.$$

Additionally, $x_0 \in \mathcal{R}(A^*)$ if and only if $\lim_{k\to\infty} x_k = A^\dagger b$.

Proof. In Chapter 2 we showed that for the given sequence it holds that

$$\lim_{k\to\infty} \|g_k\|^2 = 0.$$

Now, from $\lim_{k\to\infty} \|g_k\| = 0$ we have

$$\lim_{k\to\infty} A^*Ax_k = A^*b.$$

Using known fact that u is a least-squares solution of $Ax = b$ if and only if u is a solution of the "normal" equation $A^*Ax_k = A^*b$ (see [88]), we conclude that the sequence x_k converges to a least-squares solution of the operator equation $Ax = b$.

For the second part of the proof see the convergence theorem for steepest descent from [97]. □

The following proposition gives a characterization of least-squares solutions of the operator equation $Ax = b$ obtained by an arbitrary gradient method given by (2.48) which converges to the minimum of the functional defined by (2.45).

Proposition 4.2.4. [97] *Let $\mathcal{H}$ and $\mathcal{K}$ be given Hilbert spaces and $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ is chosen operator such that $\mathcal{R}(A)$ is closed. Let the iterative process defined by (2.48) converges to a least-squares solution of the operator equation $Ax = b$. Then the obtained least-squares solution is completely determined by an arbitrary chosen initial approximation $x_0 \in \mathcal{H}$ and has the following representation*

$$\lim_{k\to\infty} x_k = A^\dagger b + (I - A^\dagger A)x_0, \tag{4.3}$$

where $I \in \mathcal{L}(\mathcal{H})$ is the identity operator. Consequently, $x_0 \in \mathcal{R}(A^*)$ if and only if

$$\lim_{k\to\infty} x_k = A^\dagger b. \tag{4.4}$$

Similar representation of least-squares solutions in C^* algebras obtained by the process (2.48) is established in [43].

Problem 4.2.1. *It seems interesting to find explicit solution of the following problem: for an arbitrary chosen least-squares solution of the equation $Ax = b$, find corresponding vector $x_0 \in \mathcal{H}$ such that the limit $L(x_0)$ of the iterative process (2.48) is just equal to this lss. The solution of the problem is given in the rest of this section.*

Let us denote the limiting value of the iterative process (2.48) which starts with the initial vector x_0 by

$$L(x_0) \equiv \lim_{k\to\infty} x_k = A^\dagger b + (I - A^\dagger A)x_0.$$

The following auxiliary results will be used to get the answer to the stated problem.

Lemma 4.2.3. *Let $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ have a closed range, where $\mathcal{H}, \mathcal{K}$ are Hilbert spaces. Then*

a) $A^\dagger A A^{(1,3)} = A^\dagger$.

b) *Let $b \in \mathcal{K}$. Then $\|Ax - b\|$ is smallest when $x = A^{(1,3)}b$. Conversely, if $X \in \mathcal{L}(\mathcal{K}, \mathcal{H})$ has the property that for all b the norm $\|Ax - b\|$ is smallest for $x = Xb$, then $X \in A\{1,3\}$.*

Proof. A has the following matrix form:

$$A = \begin{bmatrix} A_1 & 0 \\ 0 & 0 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A^*) \\ \mathcal{N}(A) \end{bmatrix} \to \begin{bmatrix} \mathcal{R}(A) \\ \mathcal{N}(A^*) \end{bmatrix},$$

where A_1 is invertible. Hence,

$$A^\dagger = \begin{bmatrix} A_1^{-1} & 0 \\ 0 & 0 \end{bmatrix}, \quad A^{(1,3)} = \begin{bmatrix} A_1^{-1} & 0 \\ U & V \end{bmatrix},$$

where U, V are arbitrary linear and bounded.

a) An easy computation shows that $A^\dagger A A^{(1,3)} = A^\dagger$ holds.

b) Let

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \begin{bmatrix} \mathcal{R}(A^*) \\ \mathcal{N}(A) \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \in \begin{bmatrix} \mathcal{R}(A) \\ \mathcal{N}(A^*) \end{bmatrix}$$

be arbitrary elements from $\mathcal{H}$ and $\mathcal{K}$ respectively. We see that

$$\min \|Ax - b\|^2 = \min \|A_1 x_1 - b_1\|^2 + \|b_2\|^2 = \|b_2\|^2$$

is attained for $x_1 = A_1^{-1} b_1$. Hence, all least-squares solutions of the equation $Ax = b$ have the form $\begin{bmatrix} A_1^{-1} b_1 \\ x_2 \end{bmatrix}$, which is the result proved in [44]. Let

$$x = A^{(1,3)} b = \begin{bmatrix} A_1^{-1} b_1 \\ U b_1 + V b_2 \end{bmatrix},$$

whence x is a least-squares solution. Conversely, for all b,

$$x = Xb = \begin{bmatrix} X_1 & X_2 \\ X_3 & X_4 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} X_1 b_1 + X_2 b_2 \\ X_3 b_1 + X_4 b_2 \end{bmatrix} = \begin{bmatrix} A_1^{-1} b_1 \\ x_2 \end{bmatrix}$$

since x is a least-squares solution, from which follows that $X_1 = A_1^{-1}$ and $X_2 = 0$, we get $X \in A\{1,3\}$.

□

Lemma 4.2.4. *Let $\mathcal{H}, \mathcal{K}, \mathcal{M}$ be Hilbert spaces. If $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ has a closed range and $F \in \mathcal{L}(\mathcal{M}, \mathcal{H})$ satisfies $\mathcal{R}(F) = \mathcal{N}(A)$, then there exists some $G \in \mathcal{L}(\mathcal{H}, \mathcal{M})$ such that $FG = I - A^\dagger A$.*

Proof. We keep the notations from Lemma 4.2.3. Since $\mathcal{R}(F) = \mathcal{N}(A)$, we conclude that F has the following form:

$$F = \begin{bmatrix} 0 \\ F_1 \end{bmatrix} : \mathcal{M} \to \begin{bmatrix} \mathcal{R}(A^*) \\ \mathcal{N}(A) \end{bmatrix},$$

where $F_1 : \mathcal{M} \to \mathcal{N}(A)$ is onto, hence it is right invertible. There exists some $G_1 : \mathcal{N}(A) \to \mathcal{M}$, such that $F_1 G_1 = I$. Now, let us consider the operator

$$G = \begin{bmatrix} 0 & G_1 \end{bmatrix} : \begin{bmatrix} \mathcal{R}(A^*) \\ \mathcal{N}(A) \end{bmatrix} \to \mathcal{M}.$$

Then

$$FG = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} = I - A^\dagger A,$$

which was our original attention. □

We firstly derive a particular solution to Problem 4.2.1.

Theorem 4.2.3. *For an arbitrary given operator $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ with closed range and an arbitrary chosen least-squares solution s of the operator equation $Ax = b$ the following holds*

$$s = L(s - A^\dagger b). \tag{4.5}$$

Proof. According to Lemma 4.2.3, part **b)**, we can characterize the set of all least-squares solutions of the operator equation $Ax = b$ by $\{A^{(1,3)}b + (I - A^{(1,3)}A)y |\ y \in \mathcal{H}\}$, where $A^{(1,3)}$ is arbitrary but fixed element. For example, it is possible to choose $A^{(1,3)} = A^\dagger$. Let $F \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ be such that $\mathcal{R}(F) = \mathcal{N}(A)$ (for example we can take $F = \mathcal{I}_{\mathcal{H}}|_{\mathcal{N}(\mathcal{A})}$). Since $\mathcal{R}(I - A^\dagger A) = \mathcal{N}(A)$ from Lemma 4.2.4 we have that there exists an operator $G \in \mathcal{L}(\mathcal{H}, \mathcal{H})$ such that $FG = I - A^\dagger A$. Now it is clear that we can reduce the characterization set to the following one

$$\{A^\dagger b + Fy |\ y = Gz, z \in \mathcal{H}\},$$

Now we obtain

$$s = A^\dagger b + FGz = A^\dagger b + (I - A^\dagger A)z.$$

One can verify the following

$$s = A^\dagger b + (I - A^\dagger A)(s - A^\dagger b),$$

taking into account that $s = Sb$ where S is some $\{1, 3\}$ inverse of A. Therefore, it is possible to choose $z = s - A^\dagger b$, which implies

$$x_0 = s - A^\dagger b,$$

and completes the proof. □

In the next theorem we get a general solution to the stated problem.

Theorem 4.2.4. *For an arbitrary given linear operator $A \in \mathcal{L}(\mathcal{H}, \mathcal{K})$ and an arbitrary least-squares solution s of the equation $Ax = b$ the following holds*

$$s = L(s - A^\dagger b + A^\dagger A y),\ y \in \mathcal{H}. \tag{4.6}$$

Proof. Let us start with the least-squares solution s of the equation $Ax = b$, obtained by (2.48)

$$s = A^{\dagger}b + (I - A^{\dagger}A)x_0.$$

In order to find the vector x_0 in terms of the vector s we consider the following operator equation

$$Cx_0 = d, \tag{4.7}$$

where $C = I - A^{\dagger}A$, $d = s - A^{\dagger}b$. Since C is idempotent and Hermitian (thus the orthogonal projector), it follows $C = C^{\dagger}C$. According to Theorem 4.2.3 we conclude that the equation $Cx_0 = d$ has a solution. Following the general form of the least-squares solution (which is a solution) of the equation $Cx_0 = d$ (see [98]), we obtain

$$x_0 = C^{\dagger}d + (I - C^{\dagger}C)y, \quad y \in \mathcal{H}. \tag{4.8}$$

After applying the equality $A^{\dagger}As = A^{\dagger}b$ which is not difficult to check and $I - C^{\dagger}C = A^{\dagger}A$ we obtain

$$\begin{aligned} x_0 &= (I - A^{\dagger}A)(s - A^{\dagger}b) + A^{\dagger}Ay \\ &= s - A^{\dagger}b - A^{\dagger}As + A^{\dagger}b + A^{\dagger}Ay \\ &= s - A^{\dagger}b + A^{\dagger}Ay, \end{aligned} \tag{4.9}$$

which completes the proof. □

4.2.3 Application of the SC method for finding the Moore-Penrose inverse of a matrix

The results from this section are inspired from Proposition 3.1.11 as well as from the next result (see, for example [12]): $X \in A\{1,3\}$ if and only if X is a least-squares solution of $AX = I_m$, i.e., X minimizes the norm $\|AX - I\|_F$.

Let $\mathcal{H} = \mathbb{C}^{n\times m}$ and $\mathcal{K} = \mathbb{C}^{m\times m}$ be regarded as Hilbert spaces. Here, on the space of complex matrices we consider the Frobenius scalar product, $\langle A, B\rangle = Tr(A^*B)$, and the Frobenious norm $\|A\|_F = \sqrt{\langle A, A\rangle}$, where $\mathrm{Tr}(A)$ denotes the trace of the matrix A.

Any matrix $A \in \mathbb{C}^{m\times n}$ defines a mapping from $\mathcal{H}$ to $\mathcal{K}$ by

$$A(X) = AX.$$

In this way, we can establish an analogy to the results of the previous sections with respect to the functional $Q(X) = \frac{1}{2}\|AX - I\|_F^2$. Consequently, we obtain the iterations

$$X_{k+1} = X_k - \gamma_k G_k = X_k - \gamma_k A^*(AX_k - I), \quad k \geq 0, \tag{4.10}$$

which are of the form (2.48) and the corresponding stepsizes for SC method is given by

$$\gamma_{k+1}^{SC} = \begin{cases} \frac{\langle S_k, R_k\rangle}{\langle Y_k, R_k\rangle}, & \langle Y_k, R_k\rangle > 0 \\ \frac{\|S_k\|}{\|Y_k\|}, & \langle Y_k, R_k\rangle \leq 0 \end{cases}, \quad k \geq 0, \tag{4.11}$$

where $S_k = X_{k+1} - X_k$, $Y_k = G_{k+1} - G_k$ and $R_k = S_k - \gamma_k Y_k$.

Finally, we define the following algorithm for computing $\{1,3\}$-inverses (and particularly the Moore-Penrose inverse) of complex matrices.

Algorithm 4.2.1 SC method for computing $\{1,3\}$-inverses of a matrix

Input: Complex matrix $A \in \mathbb{C}^{m\times n}$, initial approximation matrix $X_0 \in \mathbb{C}^{n\times m}$ and real positive constants $0 < \varepsilon \ll 1, \quad 0 < \xi_1 \ll \frac{2(1-\varepsilon)}{\|A\|^2}$.

1: Set $k = 0$, compute $Q(X_0)$, G_0 and use $\gamma_0 = 1$.
2: If test criteria are fulfilled then go to Step 7; otherwise, go to the next step.
3: Compute X_{k+1} using (4.10), $Q(X_{k+1})$, G_{k+1}, $S_k = X_{k+1} - X_k$, $Y_k = G_{k+1} - G_k$.
4: Determine $\xi_2^{(k+1)} = 2(1-\varepsilon)\frac{\|G_{k+1}\|^2}{\|AG_{k+1}\|^2}$.
5: Compute the stepsize γ_{k+1} using (4.11). If $\gamma_{k+1} < \xi_1$ or $\gamma_{k+1} > \xi_2^{(k+1)}$, set $\gamma_{k+1} = \xi_2^{(k+1)}$.
6: Set $k := k + 1$ and go to Step 2.
7: Return X_{k+1} and $Q(X_{k+1})$.

It is clear that the iterations (4.10) can be considered as a general gradient method for computing $\{1,3\}$-inverses of a given matrix. Taking into account the equation (2.47), we can also consider BB method as a kind of a two-point stepsize gradient method and additionally we observe the steepest descent method which is a gradient descent method. The stepsizes for these methods are computed according to the following formulae respectively

$$\gamma_{k+1}^{BB} = \frac{\langle Y_k, S_k\rangle}{\langle Y_k, Y_k\rangle} \quad \text{and} \quad \gamma_k^{SD} = \frac{\|G_k\|^2}{\|AG_k\|^2} \quad k \geq 0. \tag{4.12}$$

It is known that BB method for any dimensional quadratic function is R-linearly convergent [37] as well as that the steepest descent method converges to a least-squares solution of the matrix equation $AX = I$ [97].

The algorithms (BB and steepest descent) for computing $\{1,3\}$-inverses and the Moore-Penrose inverse of a matrix would be almost the same as Algorithm 4.2.1. The only difference is that SC method is implemented using the restrictions imposed in Step 5 of the algorithm on the parameter γ_k, while BB and the steepest descent methods do not make use of these restrictions. Let us indicate to a significant difference between the Algorithm 4.2.1 and the corresponding BB method for the pseudoinverse computation. The BB method considered here is actually nonmonotone gradient method (the positiveness of the stepsize is not mandatory). On the other hand, the SC method is a strictly monotone gradient descent method (similarly as the steepest descent method).

- **Convergence properties**

Remark 4.2.1. *Let us consider SC method, BB method and the steepest descent methods as gradient methods for computing a least-squares solution of the matrix equation $AX = I$ and the iterative methods introduced in* [43]. *For any initial approximation X_0 these methods converge to the $A^{(1,3)}$ inverse, given by $A^\dagger + (I - A^\dagger A)X_0$. Particularly, in the case $X_0 \in R(A^*)$ we have that these methods converge to $A^\dagger$.*

For this purpose, it is realistic to expect that the general iterative scheme (4.10) possesses the same convergence properties.

Corollary 4.2.1. *Let the matrix $X_0 \in \mathbb{C}^{n\times m}$ be any initial approximation, $A \in \mathbb{C}^{m\times n}$ be a given matrix and $I \in \mathbb{C}^{m\times m}$ be the identity matrix. If the sequence $(X_k)_k$ given by the gradient*

method (4.10) *converges to a least-squares solution of the matrix equation* $AX = I$*, then this solution is given by*

$$\lim_{k\to\infty} X_k = A^\dagger + (I - A^\dagger A)X_0. \tag{4.13}$$

Particularly, $X_0 \in \mathcal{R}(A^*)$ *if and only if*

$$\lim_{k\to\infty} X_k = A^\dagger. \tag{4.14}$$

Proof. Follows straight from Proposition 4.2.4. □

Remark 4.2.2. *According to Corollary* 4.2.1*, we conclude that the* $\{1,3\}$*-inverse which is achieved by the iterative process* (4.10) *(as a least-squares solution) is completely determined by the initial approximation* X_0*, and it is given by* $A^\dagger + (I - A^\dagger A)X_0$*. At this point, we denote the limit of the iterative process* (4.10) *determined by* X_0 *as* $L(X_0)$*. Now we consider algebraic properties of the set* $\{L(X_0) \mid X_0 \in \mathbb{C}^{n\times m}\}$*.*

Corollary 4.2.2. *Let* $A \in \mathbb{C}^{m\times n}$ *be given complex matrix. The following statement holds*

$$L = \left\{L(X_0) \mid X_0 \in \mathbb{C}^{n\times m}\right\} = A\{1,3\}. \tag{4.15}$$

Proof. We use the following characterization of the set $A\{1,3\}$ from [12]:

$$A\{1,3\} = \left\{A^{(1,3)} + (I - A^{(1,3)}A)Z : Z \in \mathbb{C}^{n\times m}\right\},$$

for arbitrary but fixed $A^{(1,3)} \in A\{1,3\}$. For example, it is possible to use $A^{(1,3)} = A^\dagger$. In this case, the inclusion $L \subseteq A\{1,3\}$ is evident. To verify the opposite inclusion let us choose an arbitrary $A^{(1,3)} \in A\{1,3\}$. It is of the form $A^{(1,3)} = A^\dagger + (I - A^\dagger A)Z$, $Z \in \mathbb{C}^{n\times m}$. If the initial iteration in (4.10) is chosen as $X_0 = Z$ we obtain $A^{(1,3)} = L(X_0)$, which implies $A\{1,3\} \subseteq L$. □

Let us consider the analogous problem with respect to Problem 4.2.1 considering the matrix equation $AX = I$; for an arbitrary chosen $\{1,3\}$-inverse $A^{(1,3)}$ find corresponding matrix X_0 such that the limit $L(X_0)$ of the iterative process (4.13) is just equal to $A^{(1,3)}$.

In the following corollary we present the general solution of the stated problem.

Corollary 4.2.3. *For an arbitrary given matrix* $A \in \mathbb{C}^{m\times n}$ *and an arbitrary chosen* $S \in A\{1,3\}$ *the following holds*

$$S = L(S - A^\dagger + A^\dagger AY),\ Y \in \mathbb{C}^{n\times m}. \tag{4.16}$$

It is possible to derive an alternative characterization for the convergence of (4.10) using main principle from [107]. If L is the desired limit matrix and X_k is the k-th estimate of L, then the convergence properties of the examined algorithm can be studied with the aid of the error matrix $E_k = X_k - L$. If an iterative algorithm is expressible as a simple matrix formula, E_{k+1} is a sum of several terms:

- zero-order term consisting of a matrix which does not depend upon E_k,
- one or more first-order matrix terms in which E_k or its conjugate transpose E_k^* appears only once,
- higher-order terms in which E_k or E_k^* appears at least twice.

All suitable algorithms have a zero-order term equal to 0. Hence the first-order terms determine the terminal convergence properties [107]. The calculation of the first-order terms $error_1$ begins by substituting $X_k = A^\dagger + E$ and expanding the resulting formula.

Using this approach, in the following statement we verify the linear convergence of our method (4.10).

Theorem 4.2.5. *Iterative method* (4.10) *converges to the Moore-Penrose inverse* $X = A^\dagger$ *linearly, where the first-order and the second-order terms, corresponding to the error estimation of* (4.10) *are equal to:*

$$error_1 = (I - \gamma_k A^* A)E_k, \quad error_2 \ = 0, \tag{4.17}$$

respectively.

Proof. Putting $X_k = A^\dagger + E_k$ in (4.10) it is not difficult to verify that the error matrix E_{k+1} is equal to

$$E_{k+1} = E_k - \gamma_k A^* A A^\dagger - \gamma_k A^* A E_k + \gamma_k A^*,$$

which confirms the statements in (4.17). □

4.2.4 Numerical Results

According to the convergence properties of gradient methods (which include the steepest descent method, SC and BB method), investigated in previous subsection, it seems reasonable to compare these methods in computation of the ordinary inverse, the Moore-Penrose inverse and various $\{1,3\}$-inverses. The code for the three methods (the steepest descent, BB and SC method) is written in the MATLAB programming package and tested on a Workstation Intel Core duo 1.6 GHz. We selected 5 different types of matrices as test problems. For each test matrix we have considered five different dimensions which are chosen according to the nature of the test problem. For each test problem we compared two indicators: number of iterations and the accuracy of the obtained result, i.e., the difference between the exact pseudoinverse and its approximation (obtained by the algorithm) given in terms of the matrix norm. Stopping conditions are:

$$\|X_{k+1} - X_k\|_F \leq \varepsilon = 10^{-8} \ \text{ and } \ |f_{k+1} - f_k| \leq \varepsilon = 10^{-8}.$$

Example 4.2.1. *In this example the inverse of the nonsingular symmetric matrix* Z_n *of order* $n = 2, 4, 6, \ldots$, *taken from* [158], *which is given by*

$$z_{i,j} = \begin{cases} a-1, & i = j, \ i \text{ even} \\ a+1, & i = j, \ i \text{ odd} \\ a, & \text{otherwise} \end{cases} \tag{4.1}$$

is computed. For computing the ordinary inverse of the parametric matrix Z_n *we use* $a = 2$.

Table 4.2.1. Numerical results for computing the inverse of the matrix Z_n where $a = 2$

	Number of iterations			$\|Z^{-1} - X\|_F$		
Dim	SC	BB	Steepest	SC	BB	Steepest
10	5	8	5	1.4e-11	6.9e-07	2.9e-09
20	5	13	5	4.6e-10	1.9e-09	3.4e-06
30	5	13	5	3.2e-09	6.6e-09	3.4e-05
40	5	13	5	6.6e-09	9.2e-08	2.7e-04
50	7	8	5	3.9e-08	3.5e-05	7.9e-04

Following the results from Table 4.2.1 it is evident that SC method as well as the steepest descent method outperform BB method approximately twice, in the number of iterations. Also, according to the accuracy, the SC method proves oneself as the best. It is important to say that the matrix Z_n is well conditioned matrix and therefore the steepest descent method is competitive with the two-point stepsize methods.

Example 4.2.2. *The structured (Toeplitz) test matrix*

$$A_n = \text{toeplitz}[(1, 1/2, \ldots, 1/(n-1), 1/n)]$$

is taken from [14] *and the numerical results for computing its inverse are presented.*

Table 4.2.2. Numerical results for computing the inverse of the Toeplitz matrix A_n

	Number of iterations			$\|A^{-1} - X\|_F$		
Dim	SC	BB	Steepest	SC	BB	Steepest
10	83	72	618	4.6e-07	2.2e-08	2.8e-07
20	117	108	903	2.4e-07	1.9e-08	5.1e-07
30	124	127	1237	4.1e-07	1.4e-07	7.1e-07
40	150	137	1533	5.6e-07	1.4e-06	8.5e-07
50	168	162	1789	6.9e-07	6.5e-07	9.8e-07

From the results for the computation of the inverse of the well conditioned matrix A_n we conclude that the steepest descent method is not competitive with the two-point stepsize methods regarding the number of iterations. Additionally, the BB method performs slightly better then SC method. Also, there is no big difference between the accuracy for all three observed methods.

Example 4.2.3. *In this example we consider the symmetric test matrix S_n of order $n = 3, 5, 7, \ldots$ and of rank $n-1$, taken from* [158]*, which is given by*

$$s_{i,j} = \begin{cases} a-1, & i = j,\ i \text{ even} \\ a+1, & i = j,\ i \text{ odd} \\ a+1, & |i-j| = n-1 \\ a, & \text{otherwise.} \end{cases} \tag{4.2}$$

For this ill-conditioned matrix whose condition number is large $cond_F(S_n) \approx |a|^2(n^2 - 3n/2)$, $|a| \gg 2$, the Moore-Penrose inverse is determined. The presented numerical results are obtained after we made the choice $a = 2$.

Table 4.2.3. Numerical results for computing the Moore-Penrose inverse of the matrix S_n

	Number of iterations			$\|S^\dagger - X\|_F$		
Dim	SC	BB	Steepest	SC	BB	Steepest
9	12	17	216941	1.6e-04	1.8e-06	2.6e-04
11	12	17	315545	4.6e-05	2.9e-06	4.9e-04
13	9	17	449687	1.9e-05	3.0e-05	8.7e-04
15	9	17	629469	1.0e-05	1.2e-04	0.0014
17	9	17	863949	6.3e-06	4.1e-04	0.0022

According to the results from Table 4.2.3 it is clear that two-point stepsize methods behave significantly better with respect to the steepest descent method not only in the number of iterations but also in the accuracy The enormous number of iterations corresponding to the steepest descent method confirms the fact that the steepest descent is very badly affected by ill conditioning. Additionally, SC method outperform BB method observing the number of iterations almost as twice.

Example 4.2.4. *The tridiagonal square test matrix B_n of the order n and* $\operatorname{rank}(B_n) = n - 1$ *taken from* [158]

$$B_n = \begin{bmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & \mathbf{0} & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & \mathbf{0} & & -1 & 2 & -1 \\ & & & & -1 & 1 \end{bmatrix}$$

is considered. The numerical results for computing the Moore-Penrose inverse $B_n^\dagger$ as well as the $B_n^{(1,3)}$ inverse are presented. The matrix B_n is ill-conditioned matrix with the spectral condition number equal to $cond_2(B_n) = 4n^2/\pi^2$.

Table 4.2.4. Numerical results for computing the Moore-Penrose inverse of the matrix B_n

	Number of iterations			$\|B^\dagger - X\|_F$		
Dim	SC	BB	Steepest	SC	BB	Steepest
5	71	42	585	7.7e-07	4.6e-07	3.3e-07
10	340	170	11369	5.8e-06	1.8e-07	5.8e-06
15	924	671	55215	1.7e-05	1.5e-05	2.9e-05
20	2130	2058	168083	2.3e-04	8.6e-07	9.5e-05
30	3125	4669	801031	7.4e-04	2.1e-07	4.8e-04

To compute the $B_n^{(1,3)}$ inverse we made the choice $X_0 = I$ for the initial approximation. Each of the observed methods converges to $B^\dagger + I - B^\dagger B$, which is completely determined by $X_0 = I$.

Table 4.2.5. Numerical results for computing $B^{(1,3)}$ inverse of the matrix B_n

	Number of iterations			$\|B^{(1,3)} - X\|_F$		
Dim	SC	BB	Steepest	SC	BB	Steepest
5	63	44	629	6.3e-08	3.5e-10	3.3e-07
10	365	413	11253	2.2e-06	1.1e-05	5.8e-06
15	847	517	54843	6.7e-05	3.4e-06	3.0e-05
20	2254	1813	167147	2.5e-04	2.3e-04	9.6e-05
30	4096	3716	797241	1.0e-03	1.3e-03	4.9e-04

SC and BB method show significantly better results with respect to the steepest descent method for computing the Moore-Penrose inverse as well as the $\{1,3\}$ inverse of the matrix B_n, as it is excepted. Although BB method outperform SC method observing the number of iterations, in some cases such as the case $n = 30$ from Table 4.2.4 we see a big difference in the number of iterations in favor of SC method.

Example 4.2.5. *The test matrix A_n is constructed by the Matrix Market generator. The matrix A_n is nonsymmetric square sparse (40% zero elements) random matrix which is filled out by uniformly distributed elements from the interval $[0, 5]$.*

Table 4.2.6. Numerical results for computing the Moore-Penrose inverse of the matrix A_n

	Number of iterations			$\|A^{\dagger} - X\|_F$		
Dim	SC	BB	Steepest	SC	BB	Steepest
5	57	42	253	4.5e-07	1.5e-07	2.6e-07
10	478	485	23763	3.7e-05	1.5e-06	2.4e-05
15	445	419	13567	8.9e-06	3.0e-05	1.2e-05
20	384	300	8955	1.7e-05	1.4e-05	9.0e-06
30	1532	1305	137873	2.0e-04	1.4e-04	1.7e-04

For the random sparse matrix A_n, according to presented numerical results, again we see the ascendancy of the two point stepsize methods over the steepest descent method. Also, we distinguish the slightly better performance of BB method with respect to SC method.

4.3 The Drazin inverse

The problem of finding the solution of the form $A^D b$ for the Drazin-consistent system given by

$$Ax = b, \quad \text{where } b \in \mathcal{R}(A^p)\ ,\ p = \text{ind}(A), \tag{4.3}$$

is very common in the literature and many different techniques were developed in order to solve it.

The method of conjugate gradients (CG) [70] is applicable in the case when A is a Hermitian positive semidefinite matrix and the linear system is consistent. The authors in [30] developed a semi-iterative method for finding the Drazin-inverse solution of the given system, but only in the case when the matrix A has a real spectrum. They extended the idea from [61] where the authors proposed a semi-iterative method for finding a solution of an inconsistent system of linear equations in the case when the spectrum of A is real and nonnegative and $\text{ind}(A) = 1$. One

of the Krylov subspace methods (denoted by DGCR), modeled after the generalized conjugate residual method (GCR), is considered in [120]. It is shown that all of the approximations produced by DGCR exist, and as its implementation a GMRES-like algorithm, denoted by DGMRES, is derived. This algorithm is also analyzed in [154, 155].

The authors in [121] developed a BI-CG type Krylov subspace method that gave a Lanczos type method and a bi-conjugate gradient (Bi-CG)-type algorithm suitable for the general case. Zhang and Wei [153], Zhou and Wei in [157] presented a preconditioned Krylov subspace method for the Drazin-inverse solution of (3.1) with unit index. Other projection methods, including Krylov methods, for solving the given problem which consider also the case when the system is inconsistent are treated in [5, 46, 47, 70, 82, 156] etc. A unified framework for Krylov subspace methods of arbitrary system of linear equations is given in [51, 119, 144].

Index splitting methods for computing the Drazin-inverse solution are presented in [140, 142]. The Cramer rule for computing the Drazin-inverse solution is given in [139].

The intent of this section is to give a different approach for computing the Drazin-inverse solution, despite previously mentioned. Using the properties of the Drazin-inverse solution, including its least-squares properties, we propose iterative methods for computing the Drazin-inverse solution. Namely, the problem of computing the Drazin-inverse solution is reduced to the problem of finding a minimum of a quadratic function. This reduction is obtained according to the proposed relationship between the Drazin-inverse solution and appropriate $\{1,3\}$-inverses, presented in Section 3.2.1. In order to find the minimum of a quadratic function, we use BB method, introduced in the first chapter.

4.3.1 Gradient methods for computing the Drazin-inverse solution

- **Case 1: The system of linear equations is Drazin-consistent, i.e.,** $b \in \mathcal{R}(A^p)$.

Let us recall from Chapter 3 that the objective problem we analyze in this case, is given by

$$\min_x f(x) = \min_x \frac{1}{2}\|A^{p+1}x - b\|^2. \tag{4.4}$$

where $p = \text{ind}(A)$.
It can be easily checked that the gradient of the function given by (4.4) is

$$g(x) = \left(A^{p+1}\right)^* (A^{p+1}x - b). \tag{4.5}$$

We analyze the general iterative scheme (2.3) in which the search direction is chosen to be the negative gradient $d_k = -g_k$, i.e.,

$$x_{k+1} = x_k - \gamma_k g_k. \tag{4.6}$$

For the purposes of presenting the functionality of the algorithm we use the stepsize γ_k determined according to the BB method [8]. Therefore, we get

$$\gamma_i = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} = \frac{\|s_{k-1}\|^2}{\|A^{p+1}s_{k-1}\|^2}, \tag{4.7}$$

since $y_{k-1} = g_k - g_{k-1} = (A^{p+1})^* A^{p+1} s_{k-1}$.

Now we present an algorithm, for computing the Drazin-inverse solution of the problem (3.8).

Algorithm 4.3.1 The BB method for computing the Drazin-inverse solution

Input: A matrix $A \in \mathbb{C}^{n\times n}$, a vector $b \in \mathbb{C}^n$ such that $b \in \mathcal{R}(A^p)$, chosen initial point x_0 and real positive constant $0 < \varepsilon \ll 1$.

1: Set $k := 0$, compute g_0 and use $\gamma_0 = 1$.
2: If stopping conditions are satisfied then go to Step 6.
3: Compute x_{k+1} using $x_{k+1} = x_k - \gamma_k g_k$ and g_{k+1} according to (4.5).
4: Compute the stepsize γ_{k+1} using (4.7)
5: Set $k := k + 1$ and go to Step 2.
6: Return $\hat{y} = A^p x_{k+1}$.

Corollary 4.3.1. *Let $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ such that $b \in \mathcal{R}(A^p)$ where $p = \mathrm{ind}(A)$. The iterative scheme, defined with Algorithm 4.3.1 converges, and the output is the vector $A^D b$.*

Proof. From the convergence of the BB method, it follows that the sequence $(x_k)_k$ defined with Algorithm 4.3.1 converges to the minimum of the function (4.4). Finally, the last statement is a direct consequence of Theorem 3.2.2. □

- **Case 2: The system of linear equations is an arbitrary system, i.e.,** $b \in \mathbb{C}^n$.

In this case our objective is the following problem:

$$\min_x f(x) = \min_x \|A^{2p}x - b\|_P^2, \tag{4.8}$$

i.e.,

$$\min_x f(x) = \min_x \|A_1 x - b_1\|^2, \tag{4.9}$$

where $A_1 = P^{-1}A^{2p}$ and $b_1 = P^{-1}b$.

Similarly as in the previous part, in order to find the minimizer of the given functional we use the two-point stepsize gradient iterative scheme

$$x_{k+1} = x_k - \gamma_k g_k. \tag{4.10}$$

The gradient of the functional (4.8) and (4.9) is

$$g(x) = (A_1)^*(A_1 x - b_1) = (P^{-1}A^{2p})^* P^{-1}(A^{2p}x - b). \tag{4.11}$$

Thus, the stepsize which is determined according to the BB method becomes

$$\gamma_i = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} = \frac{\|s_{k-1}\|^2}{\|A^{2p}s_{k-1}\|_P^2}. \tag{4.12}$$

According to Theorem 3.2.5 and the iterative method for computing the minimum of the functional given by (4.8) (or (4.9)) we are in a position to present an algorithm for computing the vector $A^D b$, in the case where A is an arbitrary square matrix and b is an arbitrary vector, both of appropriate dimensions.

Algorithm 4.3.2 The BB method for computing the Drazin-inverse solution in general case

Input: A matrix $A \in \mathbb{C}^{n\times n}$, a vector $b \in \mathbb{C}^n$, chosen initial point x_0 and real positive constant $0 < \varepsilon \ll 1$.

1: Set $k := 0$, compute g_0 and use $\gamma_0 = 1$.
2: If stopping conditions are satisfied then go to Step 6.
3: Compute x_{k+1} using $x_{k+1} = x_k - \gamma_k g_k$ and g_{k+1} according to (4.11).
4: Compute the stepsize γ_{k+1} using (4.12)
5: Set $k := k + 1$ and go to Step 2.
6: Return $\hat{y} = A^{2p-1}x_{k+1}$.

Corollary 4.3.2. *Let $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ such that $b \in \mathcal{R}(A^p)$ where $p = ind(A)$. The iterative scheme, defined with Algorithm 4.3.2 converges, and the output is the vector $A^D b$.*

Proof. Similarly as in Corollary 4.3.1, the proof follows from the convergence of the BB method and Theorem 3.2.5. □

- **Numerical examples**

In this section we report some numerical results obtained by executing Algorithm 4.3.1 for computing the Drazin-inverse solution of the consistent system (3.8) and Algorithm 4.3.2 for computing the Drazin-inverse solution in general case. The test examples are given to illustrate that the method works and to demonstrate some good numerical properties. The code is written in the programming package MATLAB and tested on a Workstation Intel Core duo 1.6 GHz. Stopping conditions are:

$$\|x_{k+1} - x_k\| \le 10^{-8} \qquad \text{and} \qquad \frac{|f(x_{k+1}) - f(x_k)|}{1 + |f(x_k)|} \le 10^{-16}.$$

The test matrices which are presented in the following examples are taken from the papers [30, 55, 143]. During the execution of algorithms we observed and later reported two important indicators: the number of iterations and the estimation error given in terms of the norm $\|A^D b - \hat{y}\|$ where the vector $\hat{y}$ presents an approximation to the Drazin-inverse solution. This approximation is obtained according to $\hat{y} = A^p\hat{x}$ and $\hat{y} = A^{2p-1}\hat{x}$, in the case of a Drazin consistent system and in general case, respectively. The Drazin inverse of the matrix A is computed based on the known formula $A^D = A^p\left(A^{2p+1}\right)^{\dagger}A^p$. Additionally, in the case of a Drazin consistent system the estimation error given in terms of the norm $\|A\hat{y} - b\|$ is presented, since it is expected that this norm tends to zero. The vector b is chosen to be a random vector of appropriate dimensions and is constructed using the MATLAB function *randint*(). In the case of a consistent system given by (3.1) the random vector b is chosen such that it satisfies $b \in \mathcal{R}(A^p)$.

Example 4.3.1. *We consider two examples of system $Ax = b$ whose matrices (A_i, $i \in \{1, 2\}$) and random selected column vectors (b_{Con}, b_{Gen}) are given as follows*

$$A_1 = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 1 & -1 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & -1 & 0 & 2 & -1 \\ -1 & -1 & 0 & -1 & -1 & 2 \end{bmatrix}, \qquad b_{Con} = \begin{bmatrix} -14 \\ 14 \\ -22 \\ 22 \\ 81 \\ -28 \end{bmatrix}, \qquad b_{Gen} = \begin{bmatrix} -30 \\ -19 \\ 5 \\ -27 \\ -8 \\ 8 \end{bmatrix}.$$

$$A_2 = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 & -1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \quad b_{Con} = \begin{bmatrix} -32 \\ 32 \\ -8 \\ 8 \\ -102 \\ 106 \\ 80 \\ -84 \end{bmatrix}, \quad b_{Gen} = \begin{bmatrix} -6 \\ -5 \\ 9 \\ 21 \\ -8 \\ -5 \\ 6 \\ 4 \end{bmatrix}.$$

The vectors b_{Con} are chosen for the purpose of computing the Drazin-inverse solution in the case of a consistent system, i.e., a system in which the condition $b \in \mathcal{R}(A^p)$ is satisfied. On the other side, in the general case (when the condition $b \in \mathcal{R}(A^p)$ is not imposed) the random vectors b_{Gen} are chosen arbitrary. In what follows we present the results obtained by computing the Drazin-inverse solution of the system $Ax = b$ in both cases: the consistent system and the general case.

Table 4.3.7. Numerical results for computing the Drazin-inverse solution

			BB Con			BB Gen	
Matrix	index	cond	No of iter	$\|A^D b - \hat{y}\|$	$\|A\hat{y} - b\|$	No of iter	$\|A^D b - \hat{y}\|$
A_1	2	Inf	59	$1 \cdot 10^{-9}$	$2 \cdot 10^{-9}$	65	$1 \cdot 10^{-8}$
A_2	4	Inf	27	$2 \cdot 10^{-10}$	$4 \cdot 10^{-10}$	45	$7 \cdot 10^{-9}$

For each of the considered test examples the index and the condition number of the matrix are presented in the first part of the table, denoted by *index* and *cond*, respectively. The condition number is computed by the MATLAB function *cond*(). The next part of the table includes the results related to the consistent system, while the last part includes the results related to the general case.

Example 4.3.2. *In this test example two additional systems $Ax = b$ are considered for computing the Drazin-inverse solutions. Systems are determined by the following matrices and arbitrary vectors columns:*

$$A_1 = \begin{bmatrix} 5 & -1 & -1 & -1 & -1 & 0 & -1 \\ 1 & 3 & -1 & -1 & -1 & 0 & -1 \\ 0 & 0 & 3 & -1 & -1 & 0 & -1 \\ 0 & 0 & 1 & 1 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -10 \end{bmatrix}, \quad b_{Con} = \begin{bmatrix} -348 \\ 420 \\ -156 \\ 12 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad b_{Gen} = \begin{bmatrix} -9 \\ 19 \\ -30 \\ -22 \\ -18 \\ -18 \\ 6 \end{bmatrix}.$$

$$A_2 = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad b_{Con} = \begin{bmatrix} 12 \\ -30 \\ 12 \\ 0 \end{bmatrix}, \quad b_{Gen} = \begin{bmatrix} -20 \\ 30 \\ -4 \\ 10 \end{bmatrix}.$$

Through this example the same notations are used as in the previous example.

Table 4.3.8. Numerical results for computing the Drazin-inverse solution

			BB Con			BB Gen	
Matrix	index	cond	No of iter	$\|A^D b - \hat{y}\|$	$\|A\hat{y} - b\|$	No of iter	$\|A^D b - \hat{y}\|$
A_1	3	$1 \cdot 10^{17}$	32	$8 \cdot 10^{-9}$	$1 \cdot 10^{-8}$	54	$1 \cdot 10^{-8}$
A_2	1	Inf	4	$5 \cdot 10^{-15}$	0	8	$5 \cdot 10^{-15}$

The presented results clearly show that the methods converge to the Drazin-inverse solution with a sufficient accuracy.

4.4 The $A^{(2)}_{T,S}$-inverse

4.4.1 Gradient methods for computing the $A^{(2)}_{T,S}$-inverse solution

Let $A \in \mathbb{C}^{m\times n}$ and $R \in \mathbb{C}^{n\times m}$ be such that

$$\operatorname{rank}(AR) = \operatorname{rank}(RA) = \operatorname{rank}(R) \quad \text{and} \quad A\mathcal{R}(R) \oplus \mathcal{N}(R) = \mathbb{C}^m,$$

and let $b \in \mathcal{R}(AR)$.

As one can suppose, by using similar techniques as in the previous section, we can use the gradient methods for finding $A^{(2)}_{T,S}$-inverse solution of a given system of linear equation. Hence, recalling Lemma 3.1.7, we obtain gradient methods as a tool for computing generalized-inverse solutions for any type of generalized inverses mentioned in Chapter 3.

We analyze the objective problem introduced in Chapter 3 by

$$\min_x f(x) = \min_x \frac{1}{2}\|ARx - b\|^2, \tag{4.13}$$

where $b \in \mathcal{R}(AR)$, and its gradient given by

$$g(x) = (AR)^* (ARx - b). \tag{4.14}$$

Again, we consider the general iterative scheme (2.3) in which the search direction is chosen to be the negative gradient $d_k = -g_k$, i.e.,

$$x_{k+1} = x_k - \gamma_k g_k. \tag{4.15}$$

In this case, the stepsize γ_k determined according to the BB method [8] is given by

$$\gamma_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} = \frac{\|s_{k-1}\|^2}{\|ARs_{k-1}\|^2}, \tag{4.16}$$

since $y_{k-1} = g_k - g_{k-1} = (AR)^* ARs_{k-1}$.

Now we present an algorithm, for computing $A^{(2)}_{T,S}$-inverse solution of the following problem.

$$Ax = b, \quad b \in \mathcal{R}(AR) \tag{4.17}$$

Algorithm 4.4.1 The BB method for computing $A^{(2)}_{T,S}$-inverse solution

Input: A matrix $A \in \mathbb{C}^{n\times n}$, a vector $b \in \mathbb{C}^n$ such that $b \in \mathcal{R}(AR)$, chosen initial point x_0 and real positive constant $0 < \varepsilon \ll 1$.

1: Set $k := 0$, compute g_0 and use $\gamma_0 = 1$.
2: If stopping conditions are satisfied then go to Step 6.
3: Compute x_{k+1} using $x_{k+1} = x_k - \gamma_k g_k$ and g_{k+1} according to (4.14).
4: Compute the stepsize γ_k using (4.16)
5: Set $k := k + 1$ and go to Step 2.
6: Return $\hat{y} = Rx_{k+1}$.

Corollary 4.4.1. *Let $A \in \mathbb{C}^{n\times n}$ and $b \in \mathbb{C}^n$ such that $b \in \mathcal{R}(AR)$. The iterative scheme, defined with Algorithm 4.4.1 converges, and the output is the vector $A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}b$.*

- **Numerical examples**

For convenience we present several illustrative examples in order to show the effectiveness of Algorithm 4.4.1. For that purpose we also use the results from Lemma 3.1.7. The code is written in the programming package MATLAB and tested on a Workstation Intel Core 2 Duo, 2 GHz. Stopping conditions are:

$$\|x_{k+1} - x_k\| \le 10^{-8} \qquad \text{and} \qquad \frac{|f(x_{k+1}) - f(x_k)|}{1 + |f(x_k)|} \le 10^{-16}.$$

As a sample matrix we use a matrix from the previous section, but now we are determining its Moore-Penrose inverse, Drazin inverse and weighted Moore-Penrose inverse by using Algorithm 4.4.1.

Example 4.4.1. *We consider the system $Ax = b$ where the matrix A is given as follows*

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 1 & -1 & 0 & 0 \\ -1 & -1 & -1 & 1 & 0 & 0 \\ -1 & -1 & -1 & 0 & 2 & -1 \\ -1 & -1 & 0 & -1 & -1 & 2 \end{bmatrix},$$

a) *Moore-Penrose invers: Since $A^\dagger = A^{(2)}_{\mathcal{R}(A^*),\mathcal{N}(A^*)}$, we choose $R = A^*$. We choose $b \in \mathcal{R}(AA^*)$. Let*

$$b = \begin{bmatrix} -2 \\ 2 \\ 5 \\ 3 \\ 9 \\ 1 \end{bmatrix},$$

By using Algorithm 4.4.1 in 37 *iteration we obtain the following solution*

$$A^{\dagger}b = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}b = \begin{bmatrix} -3 \\ -1 \\ 0 \\ -1 \\ 2 \\ -1 \end{bmatrix}.$$

b) *Drazin inverse: Since* $A^D = A^{(2)}_{\mathcal{R}(A^p),\mathcal{N}(A^p)}$*, where* $p = \mathrm{ind}(A) = 2$*, we choose* $R = A^2$*. We choose* $b \in \mathcal{R}(AR)$*. Let*

$$b = \begin{bmatrix} -14 \\ 14 \\ -22 \\ 22 \\ 81 \\ -28 \end{bmatrix},$$

By using Algorithm 4.4.1 in 59 *iteration we obtain the following solution*

$$A^{D}b = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}b = \begin{bmatrix} -7 \\ 7 \\ -11 \\ 11 \\ 41 \\ -12 \end{bmatrix}.$$

c) *Weighted Moore-Penrose inverse: Let* M *and* N *be positive definite matrices defined by*

$$M = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{bmatrix}, \qquad N = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

Since $A^{\dagger}_{M,N} = A^{(2)}_{\mathcal{R}(N^{-1}A^*M),\mathcal{N}(N^{-1}A^*M)}$ *we choose* $R = N^{-1}A^*M$*. Let*

$$b = \begin{bmatrix} -3.5 \\ 3.5 \\ 3.16667 \\ 1.83333 \\ 2.16667 \\ 2.83333 \end{bmatrix},$$

By using Algorithm 4.4.1 in 123 *iteration we obtain the following solution*

$$A^{\dagger}_{M,N}b = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}b = \begin{bmatrix} -3 \\ 0.5 \\ 0.333333 \\ -0.333333 \\ 0 \\ 0 \end{bmatrix}.$$

4.4.2 SMS method

Here we present the key points in the development of the so called Shultz method for generalized inverse computation. It is the basis for the rest of results presented in this chapter.

The classical Newton's iteration (B-IG algorithm or Shultz method)

$$X_{k+1} = \Phi(X_k) = X_k(2I - AX_k) \quad k = 0, 1, 2, \ldots \tag{4.18}$$

was proposed by Schultz in 1933 [114]. There it is shown that for a nonsingular matrix A, if the magnitudes of the eigenvalues of the matrix $I - AX_0$ are less than one, then the process converges to A^{-1}.

The method was further extended and adopted for different generalized inverses computation. The author in [9], was the first who generalized the Newton's iterative scheme in order to compute the Moore-Penrose inverse.

Proposition 4.4.1. *[9] The sequence of matrices defined by*

$$X_{k+1} = \Phi(X_k) = X_k(2P_{\mathcal{R}(A)} - AX_k) \quad k = 0, 1, 2, \ldots \tag{4.19}$$

where $X_0 \in \mathbb{C}^{n\times m}$ *satisfies*

(1) $X_0 = A^*B_0$ *for some nonsingular matrix* $B_0 \in \mathbb{C}^{m\times m}$,

(2) $X_0 = C_0A^*$ *for some nonsingular matrix* $C_0 \in \mathbb{C}^{n\times n}$,

(3) $\|AX_0 - P_{\mathcal{R}(A)}\| < 1$,

(4) $\|X_0A - P_{\mathcal{R}(A^*)}\| < 1$,

converges to the generalized inverse $A^{\dagger}$ *of* A.

One disadvantage of the previous method was the need to know $P_{\mathcal{R}(A)}$ and $P_{\mathcal{R}(A^*)}$ in order to check Condition (3) and Condition (4) from the previous proposition. This difficulty was evaded with the following proposition, given in [10].

Proposition 4.4.2. *[10] Let* A *be an arbitrary (nonzero) complex* $m \times n$ *matrix of* $\operatorname{rank}(r)$ *and let* $\lambda_1(AA^*) \geq \lambda_2(AA^*) \geq \ldots \geq \lambda_r(AA^*)$ *denote the nonzero eigenvalues of* AA^*. *If the real scalar* α *satisfies*

$$0 < \alpha < \frac{2}{\lambda_1(AA^*)}$$

then the sequence defined by:

$$X_0 = \alpha A^*$$
$$X_{k+1} = X_k(2I - AX_k), \quad k = 0, 1, \ldots$$

converges to $A^{\dagger}$ *as* $k \to \infty$.

Newton's method was later investigated in [103].

At the same time an analysis was conducted on the iterative scheme given by

$$Y_k = \alpha \sum_{i=0}^{k} A^*(I - \alpha AA^*), \quad k = 0, 1, \ldots$$

The authors in [11] showed that the iterative scheme converges to $A^\dagger$ if the condition

$$0 < \alpha < \frac{2}{\lambda_1(AA^*)}$$

is satisfied. Later the authors in [25], presented a method for the Moore-Penrose inverse computation, by showing that it is a solution of a matrix equation of the form

$$X = P_X X + Q_X.$$

Based on these consideration, they showed that the solution of the matrix equation $X = P_X X + Q_X$ can be approximated with the following iterative scheme

$$X_{k+1} = P_X X_k + Q_X, \quad X_1 = Q_X, \tag{4.20}$$

where $P_X = I - \beta A^* A$, $Q_X = \beta A^*$ and β is a relaxation parameter. Later, this method is powered by the successive matrix squaring method (SMS) method, which is based on the following ideas: Let the matrix $\overline{T}$ be defined with

$$\overline{T} = \begin{bmatrix} P_X & Q_X \\ 0 & I \end{bmatrix}. \tag{4.21}$$

and notice that the matrix $\overline{T}^k$ is given by

$$\overline{T}^k = \begin{bmatrix} P_X^k & \sum_{i=0}^{k-1} P_X^i Q_X \\ 0 & I \end{bmatrix}.$$

It is not difficult to see that the iterative scheme (4.20) produces $X_k = \sum_{i=0}^{k-1} P_X^i Q_X$. Hence, the matrix X_k is equal to the right upper block in $\overline{T}^k$. In turn, $\overline{T}^{2^k}$ can be computed by the matrix squaring repeated k times, that is

$$\begin{aligned} \overline{T}_0 &= \overline{T} \\ \overline{T}_{i+1} &= \overline{T}_i^2, \ i = 0, 1, \ldots, k-1. \end{aligned} \tag{4.22}$$

It is clear that

$$\overline{T}_k = \overline{T}^{2^k} = \begin{bmatrix} P_X^{2^k} & \sum_{i=0}^{2^k-1} P_X^i Q_X \\ 0 & I \end{bmatrix}. \tag{4.23}$$

In the same paper [25], the authors, also, showed the equivalence between Newton's method and SMS method.

Wei in [141] considered two variants of SMS algorithm which approximate the Drazin inverse and the weighted Moore-Penrose inverse of the matrix A. Finally, the authors in [126] presented

a unified algorithm for computing all types of generalized inverses which can be expressed by the $A^{(2)}_{T,S}$ inverse, for appropriately chosen subspaces $T \in \mathbb{C}^n$ and $S \in \mathbb{C}^m$ such that $AT \oplus S = \mathbb{C}^m$. The most important results related to SMS method for computing $A^{(2)}_{T,S}$ inverses [126] are given in the following.

Lemma 4.4.1. [67] *Let $H \in \mathbb{C}^{n\times n}$ and $\varepsilon > 0$ be given. There is at least one matrix norm $\|\cdot\|$ such that*

$$\rho(H) \leq \|H\| \leq \rho(H) + \varepsilon, \tag{4.24}$$

where $\rho(H)$ denotes the spectral radius of H.

Proposition 4.4.3. *[126] Let $A \in \mathbb{C}^{m\times n}_r$ and $R \in \mathbb{C}^{n\times m}_s$, $0 < s \leq r$ be given such that*

$$A\mathcal{R}(R) \oplus \mathcal{N}(R) = \mathbb{C}^m. \tag{4.25}$$

Then sequence of approximations

$$X_{2^k} = \sum_{i=0}^{2^k-1} (I - \beta RA)^i \beta R$$

determined by the SMS algorithm (4.22) *converges in the matrix norm $\|\cdot\|$ to the outer inverse $X = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}$ of A if β is a fixed real number such that*

$$\max_{1\leq i\leq t} |1 - \beta\lambda_i| < 1, \tag{4.26}$$

where $\operatorname{rank}(RA) = t$, *$\lambda_i$, $i = 1, \ldots t$ are eigenvalues of RA and $\|\cdot\|$ satisfies condition* (4.24) *from Lemma 4.4.1, for the matrix $I - \beta AR$. In the case of convergence we have the following estimate*

$$\frac{\|X - X_{2^k}\|}{\|X\|} \leq \max_{1\leq i\leq t} |1 - \beta\lambda_i|^{2^k} + O(\varepsilon), \quad k \geq 0. \tag{4.27}$$

Corollary 4.4.2. *Under the assumptions of Proposition* 4.4.3 *the following is valid:*

(*i*) *In the case $m = n$, $R = A^l$, $l \geq \operatorname{ind}(A)$, we have $A^D = \lim_{k\to\infty} X_{2^k}$;*
(*ii*) *In the case $R = A^*$, $A^\dagger = \lim_{k\to\infty} X_{2^k}$;*
(*iii*) *In the case $R = A^\# = N^{-1}A^*M$,$A^\dagger_{M,N} = \lim_{k\to\infty} X_{2^k}$;*
(*iv*) *In the case $R = A(WA)^k$, $A^{D,W} = \lim_{k\to\infty} X_{2^k}$.*

Corollary 4.4.3. *Assume that $A \in \mathbb{C}^{m\times n}_r$. Consider an arbitrary matrix $R \in \mathbb{C}^{n\times m}_s$, $0 \leq s \leq r$, and its full rank factorization $R = FG$, $F \in \mathbb{C}^{n\times s}_s$, $G \in \mathbb{C}^{s\times m}_s$, such that GAF is invertible. The sequence of approximations*

$$X_{2^k} = \sum_{i=0}^{2^k-1} (I - \beta RA)^i \beta R$$

determined by the SMS algorithm (4.22) *converges in matrix norm to the outer inverse $X = F(GAF)^{-1}G$ of A, if β is a fixed real number satisfying the equality* (4.26), *where λ_i are eigenvalues of FGA.*

4.4.3 SMS method for computing $\{2,3\}$ and $\{2,4\}$-inverses of matrices

The full-rank representations, introduced in Chapter 3, enable adaptation of well-known algorithms for computing outer inverses with prescribed range and null space into corresponding algorithms for computing $\{2,4\}$ and $\{2,3\}$-inverses [128]. In this section we derive an adaptation of the successive matrix squaring algorithm from [126]. In view of the previous general representations of $\{2,4\}$ and $\{2,3\}$-inverses, in what follows, we analyze two particular cases of the SMS algorithm in order to obtain $\{2,4\}$ and $\{2,3\}$-inverses of the matrix A. Let $A \in \mathbb{C}_r^{m\times n}$ is given and $R \in \mathbb{C}_s^{n\times m}$, $0 < s \leq r$ be an arbitrary but fixed matrix.

We consider the general iterative scheme used in [25, 126, 141, 145]

$$\begin{aligned} X_1 &= Q_X, \\ X_{k+1} &= P_X X_k + Q_X,\ k \in \mathbb{N}, \end{aligned} \tag{4.28}$$

where $P_X = I - \beta RA$, $Q_X = \beta R$ and β is a relaxation parameter, i.e., the classical SMS iterative scheme which is appropriate for computing $\{2,4\}$-inverses. We, also, find that the dual iterative scheme of the form

$$\begin{aligned} Y_1 &= Q_Y, \\ Y_{k+1} &= Y_k P_Y + Q_Y,\ k \in \mathbb{N}, \end{aligned} \tag{4.29}$$

where $P_Y = I - \beta AR$, $Q_Y = \beta R$ is more appropriate in computation of $\{2,3\}$-inverses.

Analogously, as it is done with the equations (4.21), (4.22) and (4.23), one can easily verify that the iterative scheme (4.29) can be accelerated by means of the $(n+m)\times(n+m)$ composite matrix $\overline{S}$, partitioned in the following block form

$$\overline{S} = \begin{bmatrix} P_Y & 0 \\ Q_Y & I \end{bmatrix}. \tag{4.30}$$

The improvement of (4.29) can be done by computing the matrix powers

$$\overline{S}^k = \begin{bmatrix} P_Y^k & 0 \\ \sum_{i=0}^{k-1} Q_Y P_Y^i & I \end{bmatrix}.$$

It is not difficult to see that the iterative scheme (4.29) gives $Y_k = \sum_{i=0}^{k-1} Q_Y P_Y^i$. Hence, the matrix Y_k is equal to the left lower block in $\overline{S}^k$. In turn, $\overline{S}^{2^k}$ can be computed by k repeated squaring, that is

$$\begin{aligned} \overline{S}_0 &= \overline{S} \\ \overline{S}_{i+1} &= \overline{S}_i^2,\ i = 0, 1, \ldots, k-1. \end{aligned} \tag{4.31}$$

Obviously,

$$\overline{S}_k = \overline{S}^{2^k} = \begin{bmatrix} P_Y^{2^k} & 0 \\ \sum_{i=0}^{2^k-1} Q_Y P_Y^i & I \end{bmatrix}. \tag{4.32}$$

It is not difficult to verify that $X_k \equiv Y_k$, $k \in \mathbb{N}$. This implies that the upper right block in (4.23) is equal to the lower left block in (4.32).

As it is expected all known results that hold for the iterative scheme (4.20), analogously can be proven for the iterative scheme (4.29). The following result is analogous to Proposition 4.4.3.

Theorem 4.4.1. *Let $A \in \mathbb{C}^{m\times n}_r$ and $R \in \mathbb{C}^{n\times m}_s$, $0 < s \leq r$ be given such that*

$$A\mathcal{R}(R) \oplus \mathcal{N}(R) = \mathbb{C}^m. \tag{4.33}$$

Let β be a fixed real number satisfying (4.26) where $\mathrm{rank}(AR) = t$ and $\lambda_i, i = 1, \dots t$ are eigenvalues of AR. Then, the sequence of approximations

$$Y_{2^k} = \sum_{i=0}^{2^k-1} QP_Y{}^i, \ k \geq 0$$

determined by (4.29) converges to the outer inverse $Y = A^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}$ of A. In the case of convergence we have the following estimate

$$\frac{\|Y - Y_{2^k}\|}{\|Y\|} \leq \max_{1\leq i\leq t} |1 - \beta\lambda_i|^{2^k} + O(\varepsilon), \quad k \geq 0, \tag{4.34}$$

where $\|\cdot\|$ satisfies condition (4.24) from Lemma 4.4.1 for the matrix $M = I - \beta RA$.

We now define two particular cases of the SMS algorithm which generate the classes of $\{2, 4\}$ and $\{2, 3\}$-inverses of the matrix A.

1. Let $V \in \mathbb{C}^{s\times m}_s$ is an arbitrary matrix such that $\mathrm{rank}(VA) = \mathrm{rank}(V) = s$, where $0 < s \leq r$. For the iterative scheme given by (4.20), we take $F = (VA)^*, G = V$ and $R = FG = (VA)^*V$, which implies

$$P_X = I - \beta(VA)^*VA, \ Q_X = \beta(VA)^*V, \tag{4.35}$$

   where β is a relaxation parameter.

2. Let $U \in \mathbb{C}^{n\times s}_s$, is an arbitrary matrix satisfying $\mathrm{rank}(AU) = \mathrm{rank}(U) = s$, where $0 < s \leq r$. For the iterative scheme given by (4.29), we use the particular case $F = U, G = (AU)^*$ and $R = FG = U(AU)^*$, which gives

$$P_Y = I - \beta AU(AU)^*, \ Q_Y = \beta U(AU)^*, \tag{4.36}$$

   where β is a relaxation parameter.

The next theorem presents the main result of this section and is analogous result to Proposition 4.4.3.

Theorem 4.4.2. *Let $A \in \mathbb{C}^{m\times n}_r$, $0 < s \leq r$ be chosen integer.*

1. *If $V \in \mathbb{C}^{s\times m}_s$ is chosen matrix such that $\mathrm{rank}(VA) = \mathrm{rank}(V) = s$, then the sequence of approximations*

$$X_{2^k} = \sum_{i=0}^{2^k-1} (I - \beta(VA)^*VA)^i\beta(VA)^*V, \ k \geq 0 \tag{4.37}$$

determined by the SMS iterative scheme (4.23), where P_X and Q_X are defined in (4.35), converges in the matrix norm $\|\cdot\|$ to $\{2, 4\}$-inverse of A, which is equal to $A^{(2,4)}_{\mathcal{N}(VA)^\perp,\mathcal{N}(V)}$ if β is a fixed real number such that

$$\max_{1\leq i\leq s} |1 - \beta\lambda_i| < 1, \tag{4.38}$$

where $\lambda_i,\ i = 1, \ldots, s$ *are nonzero eigenvalues of* $(VA)^*VA$. *In the case of convergence, the sequence* X_k *satisfies the error estimation*

$$\frac{\|X - X_{2^k}\|}{\|X\|} \leq \max_{1 \leq i \leq t} |1 - \beta\lambda_i|^{2^k} + O(\varepsilon), \quad k \geq 0, \tag{4.39}$$

where $\|\cdot\|$ *satisfies condition* (4.24) *for* $M = I - \beta A(VA)^*V$.

2. *If* $U \in \mathbb{C}_s^{n\times s}$ *is chosen matrix such that* $\operatorname{rank}(AU) = \operatorname{rank}(U) = s$, *then the sequence of approximations*

$$Y_{2^k} = \sum_{i=0}^{2^k-1} (I - \beta AU(AU)^*)^i \beta U(AU)^*,\ k \geq 0 \tag{4.40}$$

determined by the SMS iterative process (4.32), *where* P_Y *and* Q_Y *are defined in* (4.36), *converges in the matrix norm* $\|\cdot\|$ *to* $\{2,3\}$*-inverse of* A, *which is equal to* $A^{(2,3)}_{\mathcal{R}(U),\mathcal{R}(AU)^\perp}$ *in the case when* β *is a fixed real number satisfying*

$$\max_{1 \leq i \leq s} |1 - \beta\lambda_i| < 1, \tag{4.41}$$

where $\lambda_i,\ i = 1, \ldots, s$ *are nonzero eigenvalues of* $AU(AU)^*$.

In the case of convergence, the next error estimation holds for the sequence Y_k

$$\frac{\|Y - Y_{2^k}\|}{\|Y\|} \leq \max_{1 \leq i \leq t} |1 - \beta\lambda_i|^{2^k} + O(\varepsilon), \quad k \geq 0, \tag{4.42}$$

where $\|\cdot\|$ *satisfies condition* (4.24) *for* $M = I - \beta U(AU)^*A$.

Proof. **1.** According to main result from [25], we have

$$\begin{aligned}\lim_{k\to\infty} X_{2^k} = X &= \lim_{k\to\infty} \left(\sum_{i=0}^{2^k-1} (I - \beta(VA)^*VA)^i (VA)^* \right) \cdot V \\ &= (VA)^\dagger V\end{aligned}$$

From the general representation of $\{2,4\}$-inverses from Proposition 3.4.1, we obtain $X \in A\{2,4\}_s$.

Moreover, the conditions imposed in this case are equivalent with Proposition 4.4.3 by taking $R = (VA)^*V$. Since $\operatorname{rank}(VA) = \operatorname{rank}(V)$ it follows that the matrix $(VA)^*$ is a full-column rank matrix, from which we obtain that $\operatorname{rank}(R) = s$. Now, using $R = FG$, $F = (VA)^*, G = V$ as a full-rank factorization of R, according to Corollary 4.4.3 we conclude that

$$X^{2^k} \to X = F(GAF)^{-1}G = (VA)^*(VA(VA)^*)^{-1}V.$$

Now, the proof follows from Theorem 3.4.1, part (a) and Lemma 3.4.1, part (a).

2. Conditions imposed in this case are equivalent with Theorem 4.4.1 by taking $R = U(AU)^*$. Since $\operatorname{rank}(AU) = \operatorname{rank}(U)$ follows that the matrix $(AU)^*$ is a full-row rank matrix, from which we obtain that $\operatorname{rank}(R) = s$. Now, using $R = FG$, $F = U, G = (AU)^*$ as a full-rank factorization of R, according to Theorem 2.3 from [126] we conclude that

$$Y^{2^k} \to Y = F(GAF)^{-1}G = U((AU)^*AU)^{-1}(AU)^*.$$

Now, the proof follows from Theorem 3.4.1, part (b) and Lemma 3.4.1, part (b) □

Corollary 4.4.4. a) *In the case* $V \in \mathbb{C}_r^{r\times m}$, $\operatorname{rank}(VA) = r$, *under the conditions of Theorem* 4.4.2, *part 1, we have*

$$X = A^{(2,4)}_{\mathcal{R}(A^*),\mathcal{N}(V)} \in A\{1,2,4\}.$$

b) *In the case* $U \in \mathbb{C}_r^{n\times r}$, $\operatorname{rank}(AU) = r$, *under the conditions of Theorem* 4.4.2, *part 2, we have*

$$Y = A^{(2,3)}_{\mathcal{R}(U),\mathcal{N}(A^*)} \in A\{1,2,3\}.$$

c) *If both of the conditions* $V = A^*$ *and* $U = A^*$ *are satisfied, under the conditions of Theorem* 4.4.2, *the identities*

$$X = Y = A^\dagger$$

are satisfied

Proof. **a)** Since X is $\{2\}$-inverse of A such that

$$\operatorname{rank}(X) = \operatorname{rank}((VA)^\dagger) = \operatorname{rank}((VA)^*) = \operatorname{rank}(V) = \operatorname{rank}(A),$$

it follows that X is also $\{1\}$-inverse of A. Moreover, from $\operatorname{rank}((VA)^*) = \operatorname{rank}(A^*)$ we have that $\mathcal{R}((VA)^*) = \mathcal{R}(A^*)$ and the proof is complete.

b) Analogously.

c) In both cases, immediately follows that $X = Y = A^{(2)}_{\mathcal{R}(A^*),\mathcal{N}(A^*)} = A^\dagger$. □

- **Numerical examples**

The relaxation parameter $\beta = \beta^C_{opt}$ defined as in [126] is used in the next examples. More precisely, if we put $mRe = \min\{\text{Re } \lambda_1, \ldots, \text{Re } \lambda_t\}$, $MRe = \max\{\text{Re } \lambda_1, \ldots, \text{Re } \lambda_t\}$ and $(MIm)^2 = \max\{(\text{Im } \lambda_1)^2, \ldots, (\text{Im } \lambda_t)^2\}$, then if Re $\lambda_i > 0$, for every $i \in \{1, \ldots, t\}$,

$$\beta^C_{opt} = \frac{mRe}{(MRe)^2 + (MIm)^2}.$$

Otherwise if Re $\lambda_i < 0$, for every $i \in \{1, \ldots, t\}$, then

$$\beta^C_{opt} = \frac{MRe}{(MRe)^2 + (MIm)^2}.$$

Example 4.4.2. In this example we get an $\{2,4\}$-inverse X and $\{2,3\}$-inverse Y of A_1 in the first iteration, as it is expected. Let us consider 6×5 matrix A_1 given by

$$A_1 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & 1 & 1 \\ 1 & 0 & 0 & -2 & 0 \end{bmatrix}.$$

a) Now, let us choose

$$V = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

One can check that the nonzero eigenvalues of the matrix $R = (VA)^*VA$ are equal to 1, thus $mRe = 1, MRe = 1$ and also Im $(\lambda) = 0$ for each eigenvalue λ. Thus, in only 1 iteration we obtain the $\{2,4\}$-inverse of A_1 given by

$$X_{2^1} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

It is not difficult to verify $X_{2^1} = (VA)^\dagger V$.

b) If we take

$$U = \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{bmatrix},$$

Since all eigenvalues are real and $mRe = MRe = 4$ again only in one iteration we obtain an exact $\{2,3\}$-inverse of A_1 given by

$$Y = \begin{bmatrix} -0.25 & 0 & 0.25 & 0 & 0.25 & 0.25 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \end{bmatrix}.$$

Example 4.4.3. Consider the following 6×5 matrix of rank 4,

$$A_2 = \begin{bmatrix} -1 & 0 & 1 & 2 & 2 \\ -1 & 1 & 0 & -1 & -1 \\ 1 & -1 & 1 & 3 & 4 \\ 0 & 1 & -1 & -3 & 2 \\ 1 & -1 & 0 & 1 & 1 \\ 1 & 0 & -1 & -2 & -2 \end{bmatrix},$$

a) First, let us choose V of rank 2, given by

$$V = \begin{bmatrix} 3 & 1 & 3 & 1 & 2 & -1 \\ 0 & -1 & 0 & 0 & -2 & 1 \end{bmatrix}.$$

Iterative scheme (4.20) with P_X and $Q = Q_X$ defined as in (4.35) gives the following $\{2,4\}$-inverse of A_2:

$$X_{2^{25}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0.222092 & -0.431257 & 0.222092 & 0.0740306 & -0.862515 & 0.431257 \\ -0.0105758 & 0.0681551 & -0.0105758 & -0.00352526 & 0.13631 & -0.0681551 \\ -0.243243 & 0.567568 & -0.243243 & -0.0810811 & 1.13514 & -0.567568 \\ 0.320799 & -0.400705 & 0.320799 & 0.106933 & -0.80141 & 0.400705 \end{bmatrix}.$$

Additionally, for the same matrix A_2 if we take, for example, the matrix

$$V = \begin{bmatrix} 3 & 1 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -2 & 1 \\ 1 & 0 & 3 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 & -2 & 4 \end{bmatrix},$$

which has rank 4, we obtain the following $\{1,2,4\}$-inverse of A after 25 iterations

$$X_{2^{25}} = \begin{bmatrix} -0.238095 & -0.114286 & 1 & -0.2 & -1.31429 & 0.961905 \\ -0.119048 & 0.0761905 & 1.16667 & -0.2 & -2.12381 & 0.914286 \\ -0.285714 & 0.0761905 & 0.666667 & -0.2 & -1.12381 & 0.247619 \\ -0.452381 & 0.0761905 & 0.166667 & -0.2 & -0.12381 & -0.419048 \\ 0.52381 & -0.0285714 & 0 & 0.2 & 0.171429 & 0.32381 \end{bmatrix}.$$

b) For the same matrix A_2 we apply the iterative scheme (4.29) for the choice of matrices P_Y and $Q = Q_Y$ which are defined by (4.36), where

$$U = \begin{bmatrix} 3 & 5 \\ 1 & 7 \\ -3 & 2 \\ 1 & -2 \\ 2 & -2 \end{bmatrix}.$$

After 10 iterations, we get the $\{2,3\}$-inverse

$$Y_{2^{10}} = \begin{bmatrix} -0.213456 & -0.0607649 & 0.0827195 & 0.313031 & 0.0607649 & 0.213456 \\ -0.216572 & -0.00325779 & -0.0328612 & 0.259207 & 0.00325779 & 0.216572 \\ 0.0225921 & 0.0830737 & -0.16204 & -0.109773 & -0.0830737 & -0.0225921 \\ 0.0288244 & -0.0319405 & 0.0691218 & -0.00212465 & 0.0319405 & -0.0288244 \\ 0.00311615 & -0.0575071 & 0.115581 & 0.0538244 & 0.0575071 & -0.00311615 \end{bmatrix}.$$

After the usage

$$U = \begin{bmatrix} 1 & 3 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 0 & 1 & 0 & 2 \\ 2 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix},$$

the generated $\{1,2,3\}$-inverse of A is equal to

$$Y_{2^{10}} = \begin{bmatrix} -0.6 & 0.6 & 1 & -0.2 & -0.6 & 0.6 \\ -0.836364 & 1.56364 & 1.63636 & -0.490909 & -1.56364 & 0.836364 \\ 0.372727 & -0.327273 & -0.272727 & 0.381818 & 0.327273 & -0.372727 \\ -0.336364 & 0.563636 & 0.636364 & -0.490909 & -0.563636 & 0.336364 \\ 0.1 & -0.1 & 0 & 0.2 & 0.1 & -0.1 \end{bmatrix}.$$

4.4.4 Displacement rank and displacement operator of a Toeplitz matrix

Kailath et al. in [71] introduce the concept of displacement rank as well as displacement operator for close-to-Toeplitz matrices, which is systematically studied in the general case in [63]. It is shown that the displacement theory approach is very powerful in designing fast inversion algorithms for structured matrices, especially Toeplitz and Hankel matrices. A number of different displacement operators have been introduced and analyzed in the literature [14, 64, 71, 72, 73, 102, 146]. The authors in [14] introduce the concept of the *orthogonal displacement representation* and *ε-displacement rank* of a matrix.

In the sequel, before presentation of the new method which is based on the idea of the orthogonal displacement representation, we restate the basic concepts related to these notions.

Definition 4.4.1. *A matrix $M \in \mathbb{C}^{n\times n}$ is a Toeplitz matrix if for its elements hold $m_{i,j} = m_{i-j}$, $i, j = 1\ldots, n$, where m_k, $k \in \{-n+1, \ldots, n-1\}$ is an arbitrary sequence of complex numbers.*

The explicit form of a Toeplitz matrix is given with the following presentation

$$M = \begin{bmatrix} m_0 & m_{-1} & \ldots & m_{1-n} \\ m_1 & m_0 & \ldots & m_{2-n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n-1} & m_{n-2} & \ldots & m_0 \end{bmatrix}.$$

A Toeplitz $n \times n$ matrix is completely determined with a vector of length $2n-1$, i.e., with two vectors of length n. The first vector $c = (m_0, m_1, \ldots, m_{n-1})$ is the first column, and the other vector represents the first row $r = (m_0, m_{-1} \ldots, m_{1-n})$ of the matrix M. In the remainder of the section the notation toeplitz$[c, r]$ stands for the Toeplitz matrix which is determined by its first column c and its first row r.

The concept of displacement operator, introduced in [71], is studied in many papers [32, 60, 64, 74, 72, 104] and many different forms of it are presented. Displacement structure of the Drazin inverse, the group inverse and the $M^{(2)}_{T,S}$ inverse is considered in [19, 41, 84, 147]. Here, we pay attention on the displacement operator of Sylvester type, given by

$$\triangle_{A,B}(M) = AM - MB, \tag{4.43}$$

where A, B and M are matrices of appropriate sizes.

The next proposition gives some arithmetics regarding the defined displacement operator $\Delta_{A,B}$.

Proposition 4.4.4. *The following holds*

$$\Delta_{A,B}(MN) = \Delta_{A,B}(M)N + M\Delta_{A,B}(N) + M(B-A)N.$$

Proof. Follows immediately from the definition of the displacement operator $\Delta_{A,B}$. □

The rank of the displacement matrix $\Delta_{A,B}(M)$ is called the displacement rank of M and is denoted by drk(M). For a Toeplitz matrix M and appropriate choices of matrices A and B the following holds drk$(M) \leq 2$ [63]. Here we recall the definition of ε-displacement rank as a rank of perturbed displacement.

Definition 4.4.2. [14] *For a given $\varepsilon > 0$ the relative ε-displacement rank of a matrix M is defined as*

$$\text{drk}_\varepsilon(M) = \min_{\|E\| \leq \varepsilon \|M\|} rank(\Delta_{A,B}(M) + E),$$

where $\|\cdot\|$ denotes the Euclidian norm.

The main purpose of the displacement operator is to enable representation of the original matrix through a matrix that has low rank and from which one can also easily recover the original matrix. Besides the fact that the displacement matrix $\Delta_{A,B}(M)$ has low rank it is well-known that its singular value decomposition can be easily computed.

Remark 4.4.1. *The computation of the singular value decomposition of a matrix, in general, is a very expensive tool. In our case, what makes it cheaper, as it is also explained in* [14]*, is the fact that it can be calculated using the QR factorization of appropriate matrices.*

Let $\sigma_1 \geq \cdots \geq \sigma_l$ be the nonzero singular values of $\Delta_{A,B}(M)$ and let

$$\Delta_{A,B}(M) = U\Sigma V^T = \sum_{i=1}^{l} \sigma_i u_i v_i^T, \tag{4.44}$$

be the singular value decomposition of $\Delta_{A,B}(M)$. Suppose that the original matrix M can be recovered by the formula

$$M = c\sum_{i=1}^{l} \sigma_i f(u_i) g(v_i), \tag{4.45}$$

where f and g, generally speaking, represent a product of matrices which can be generated by the vectors u_i and v_i respectively and c is a constant. The representation (4.45) of the matrix M is called *orthogonal displacement representation* (odr) (with respect to $\Delta_{A,B}$) and the corresponding 3-tuple (U, σ, V) *orthogonal displacement generator* (odg) where $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_l)$ [14]. For example, the odr of an arbitrary Toeplitz matrix is given in [14].

Proposition 4.4.5. *[14] Suppose that M is a Toeplitz matrix. Let $\Delta_{A,B}(M) = U\Sigma V^T = \sum_{i=1}^{l} \sigma_i u_i v_i^T$ $(\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_l > 0)$ be the singular value decomposition of $\Delta_{A,B}(M)$ and let ε be such that $0 < \varepsilon < \sigma_1$. Then $\mathrm{drk}_\varepsilon(M) = r$ if and only if $\sigma_r > \varepsilon\|M\| \geq \sigma_{r+1}$.*

Proof. We prove $\sigma_{r+1} \leq \varepsilon\|M\|$ if and only if $\mathrm{drk}_\varepsilon(M) \leq r$. We can write

$$\Delta_{A,B}(M) = \sum_{i=1}^{r} \sigma_i u_i v_i^T + \sum_{i=r+1}^{l} \sigma_i u_i v_i^T.$$

If we define $E = \sum_{i=r+1}^{l} \sigma_i u_i v_i^T$, then it follows $\|E\| = \sigma_{r+1}$. Thus, if $\sigma_{r+1} \leq \varepsilon\|M\|$ we have $\mathrm{drk}_\varepsilon(M) \leq \mathrm{rank}(\Delta(M) - E) = r$. Obviously $\mathrm{drk}_\varepsilon(M) \geq r$ holds without any restriction.

Conversely, if $\mathrm{drk}_\varepsilon(M) = r$ then there exists a matrix E with $\|E\| \leq \varepsilon\|M\|$, such that $\mathrm{rank}(\Delta(M) + E) = r$. Let us define $\theta_r = \min_{\{X:\mathrm{rank}(X)=r\}} \|\Delta(M) - X\|$. Since $\theta_r = \sigma_{r+1}$ and $\sigma_{r+1} = \|\Delta(M) - \sum_{i=1}^{r} \sigma_i u_i v_i^T\|$, it follows that $\sigma_{r+1} = \theta_r \leq \|E\| \leq \varepsilon\|M\|$. □

For a given $\varepsilon > 0$, if $\mathrm{drk}_\varepsilon(M) = r$ one can get an approximate M_ε of M

$$M_\varepsilon = c\sum_{i=1}^{r} \sigma_i f(u_i) g(v_i), \quad r < l,$$

which is called an *approximate orthogonal displacement representation* (aodr) of the matrix M and the associated generator $(\hat{U}, \hat{\sigma}, \hat{V})$ an *approximate orthogonal displacement generator* (aodg), where

$$\hat{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_r), \quad \hat{U} = [u_1, u_2, \ldots, u_r], \quad \hat{V} = [v_1, v_2, \ldots, v_r].$$

We also use the definition of the operator $trunc_\varepsilon(\cdot)$ defined on the set of matrices as follows $M_\varepsilon = trunc_\varepsilon(M)$ [14].

4.4.5 Modified SMS method for computing $M^{(2)}_{T,S}$-inverses of a Toeplitz matrix M

If we look at Schultz's paper, we immediately find that he does not suggest this method for arbitrary matrices (this would be a bit too expensive), and it is interesting that he considers an example of a Toeplitz matrix, since their structure admits very cheap matrix operations. Thus, the Schultz iteration is advocated only in case of an agreeable structure in the matrices.

The initial guess X_0 is chosen such that it has low displacement rank. The limit X_∞, which represents the inverse or the generalized inverse of M, has also low displacement rank, see [63, 64]. However, the iteration matrices X_k in the Newton's method may not have low displacement rank. In the worst case, the displacement rank of X_k increases exponentially until it reaches the dimension of the matrix n. Parallel with the increase of the displacement rank, the computational cost of the Newton's process grows significantly while k increments. It follows that the growth of the displacement rank of X_k must be controlled for the purpose of developing an efficient algorithm. With the introduction of orthogonal displacement operator the authors in [14] control the growth of the displacement rank of X_k. At each step, via the truncation, the iteration matrix is approximated by an appropriate matrix with a low displacement rank. This strategy belongs to the class of compression techniques, for inversion of structured matrices which are described in [31, 32, 102, 105]. Because of the displacement structure of the iteration matrix, the matrix-vector multiplication involved in Newton's iteration can be done efficiently.

Numerical properties of the Shultz method powered by the concept of orthogonal displacement representation as well as ε-displacement rank show the effectiveness of the method applied on structured matrices. The entire computation uses much less memory and CPU time with respect to the classical method, especially for large scale matrices. The Newton's iterative scheme (4.18) has been frequently modified for the purpose of computing various generalized inverses of the structured matrices. Main results are comprised in [14, 15, 19, 133, 146]. The authors in the previously mentioned papers use various strategies in choosing the starting approximation X_0, in defining the iterations as well as in the usage of displacement operators.

In this section we present a uniform algorithm applicable in computation of outer inverses with prescribed range and null space of an arbitrary Toeplitz matrix [90]. The algorithm is based on the Successive Matrix Squaring (SMS) algorithm, which is an equivalent to the Shultz method. Besides the classical modifications of the Shultz method available in the literature, corresponding to the orthogonal displacement representation and the ε-displacement rank, we use several adaptations of the SMS algorithm which enable the uniform approach for various generalized inverses. The algorithm is an adaptation of the SMS algorithm proposed in [126] and uses the concept of the approximate orthogonal displacement representation introduced in [14, 19].

- **Algorithm construction**

As we already mentioned, we derive an algorithm for finding the outer inverse with prescribed range and null space of an arbitrary Toeplitz matrix $M \in \mathbb{C}^{n\times n}$ [90]. The entries of the matrix M are constant complex numbers along the main diagonal parallels and are defined by the general rule $m_{i,j} = m_{i-j}$. As usual, by C^+ and C^- we denote the Toeplitz matrices defined by $C^+ = Z + e_1 e_n^T$, $C^- = Z - e_1 e_n^T$, where $Z = \text{toeplitz}[(0, 1, 0, \ldots, 0), (0, \ldots, 0)]$ and e_i is the ith column of the identity matrix. The expanded form of these matrices looks as follows:

$$C^+ = \begin{pmatrix} 0 & 1 & 0 & \dots & 1 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix}, \qquad C^- = \begin{pmatrix} 0 & 1 & 0 & \dots & -1 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix}.$$

Let $C^+(x)$ and $C^-(x)$ be circulant and anti-circulant matrices which are defined by

$$\begin{aligned} C^+(x) &= \text{toeplitz } [(x_1, x_2, \dots, x_n), (x_1, x_n, \dots, x_2)], \\ C^-(x) &= \text{toeplitz } [(x_1, x_2, \dots, x_n), (x_1, -x_n, \dots, -x_2)], \end{aligned}$$

or in tabular form:

$$C^+(x) = \begin{pmatrix} x_1 & x_n & x_{n-1} & \dots & x_2 \\ x_2 & x_1 & x_n & \dots & x_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n-1} & x_{n-2} & x_{n-3} & \dots & x_n \\ x_n & x_{n-1} & x_{n-2} & \dots & x_1 \end{pmatrix}, \qquad C^-(x) = \begin{pmatrix} x_1 & -x_n & -x_{n-1} & \dots & -x_2 \\ x_2 & x_1 & -x_n & \dots & -x_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n-1} & x_{n-2} & x_{n-3} & \dots & -x_n \\ x_n & x_{n-1} & x_{n-2} & \dots & x_1 \end{pmatrix}.$$

For an arbitrary Toeplitz matrix M we make the choice $A = C^-$ and $B = C^+$ in the definition of the displacement operator (4.43). The parameters from (4.45) become [14]:

$$c = -\frac{1}{2},\ f(u_i) = C^-(u_i),\ g(v_i) = C^+(Jv_i),$$

where J is the permutation matrix having 1 on the anti-diagonal. For the sake of simplicity in the rest of the paper we use the notation Δ instead of Δ_{C^-,C^+}.

Proposition 4.4.6. *For the operator Δ the following holds*

$$\begin{aligned} &\Delta(M) = \sum_{i=1}^{l} u_i v_i^T \Longleftrightarrow M_\varepsilon = -\frac{1}{2}\sum_{i=1}^{l} C^-(u_i)C^+(Jv_i), \\ &\Delta(MN) = \Delta(M)N + M\Delta(N) + 2Me_1e_n^TN. \end{aligned}$$

Remark 4.4.2. [14] *For the approximation of the matrix $M \in \mathbb{C}^{n\times n}$ given by its aodg M_ε, the matrix–vector product can be efficiently done by $\mathcal{O}(rn\log n)$ FFT's, where $r = \text{drk}_\varepsilon(M)$.*

The following result provides an error bound of the *aodr* of M.

Proposition 4.4.7. [14] *Suppose that M is a Toeplitz matrix. Let $r = \text{drk}_\varepsilon(M) \le \text{drk}(M) = l$, M_ε be an* aodr *of M, and let $\sigma_1, \sigma_2, \dots, \sigma_l$, be the nonzero singular values of $\Delta_{A,B}(M)$. Then it holds*

$$\|M - M_\varepsilon\| \le \frac{1}{2}n\sum_{i=r+1}^{l}\sigma_i \le \frac{1}{2}n(l-r)\varepsilon. \tag{4.46}$$

Many authors built iterative methods for finding generalized inverses of an arbitrary Toeplitz matrix modifying the Newton's method, using operators of the form $\Delta_{A,B}(M)$, as well as the concept of displacement representation and ε-displacement rank of matrices [14, 15, 19, 133,

146]. It is shown that these methods are very effective and significantly decrease computational cost.

In the following we construct an algorithm for finding outer inverses with prescribed range an null space of an arbitrary square Toeplitz matrix. The based point is SMS algorithm from [126]. Although this method is already presented, for convenience we give detailed building steps of the new algorithm.

Let $M \in \mathbb{C}_{\rho}^{n\times n}$ and $R \in \mathbb{C}_{s}^{n\times n}$, $0 \leq s \leq \rho$ be arbitrary matrices. We start with the following iterative scheme (see [126])

$$\begin{aligned} &X_1 = Q, \\ &X_{k+1} = PX_k + Q, \quad k \in \mathbb{N}, \end{aligned} \tag{4.47}$$

where $P = I - \beta RM$, $Q = \beta R$ and β is relaxation parameter. The original idea of the iterative scheme (4.47), named the successive matrix squaring method, was previously proposed by Chen et al. [25] in the case $R = M^*$. The method proposed by Chen is used for computing the Moore-Penrose inverse of the matrix M. Instead of the iterative scheme (4.47) it is possible to use the $2n \times 2n$ matrix

$$\overline{T} = \begin{bmatrix} P & Q \\ 0 & I \end{bmatrix}.$$

The kth matrix power of the matrix $\overline{T}$ is given by

$$\overline{T}^k = \begin{bmatrix} P^k & \sum_{i=0}^{k-1} P^i Q \\ 0 & I \end{bmatrix}, \quad k > 1,$$

where the $n \times n$ matrix block $\sum_{i=0}^{k-1} P^i Q$ is actually the kth approximation X_k of the pseudoinverse of the matrix M from (4.47), whence $X_k = \sum_{i=0}^{k-1} P^i Q$. The matrix power $\overline{T}^k$ can be computed by means of the repeated squaring of the matrix $\overline{T}$, therefore instead of using the iterative scheme (4.47) we can observe the following iterative process

$$\begin{aligned} &\overline{T}_0 = \overline{T}, \\ &\overline{T}_{k+1} = \overline{T}_k^2, \quad k \in \mathbb{N}_0. \end{aligned} \tag{4.48}$$

Afterwards, the block $\sum_i P^i Q$ generated after k steps of the successive squaring (4.48) is equivalent with the approximation produced by 2^k steps of the iterative process (4.47). Namely, we consider the sequence $\{Y_k\}$ given by $Y_k = X_{2^k} = \sum_i P^i Q$.

As it is stated in Proposition 4.4.3, the iterative scheme given in terms of the block matrix (4.48) converges to the outer inverse of the matrix M with prescribed range and null space, determined by an appropriately chosen matrix R.

Since the usage of the partitioned $2n \times 2n$ matrix in the SMS acceleration scheme is not appropriate for application of a displacement operator, our intention is to avoid such an approach. For that purpose, we observe that the sequence of approximations given by

$$Y_k = \sum_{i=0}^{2^k-1} (I - \beta RM)^i \beta R,$$

can be rewritten by the following steps

$$\begin{aligned} &Y_{k+1} = (S_k + I)Y_k, \\ &S_{k+1} = I - Y_{k+1}M, \end{aligned} \quad k \in \mathbb{N}_0, \tag{4.49}$$

where the initial choices are $Y_0 = Q,\ S_0 = P$. With the rules given by (4.49) one can easily verify that $Y_k = X_{2^k}$ and $S_k = P^{2^k}$.

Because of ineffective manipulation with high displacement rank matrix sequence $\{Y_k\}$, whose displacement rank, in the worst case, can increases exponentially, we introduce the following modification of the method (4.49). Namely, using the concept of the displacement representation, we modify the successive matrix squaring method from [126] by approximating the matrix Y_{k+1} with a matrix with low displacement rank. We define the sequence $\{Z_k\}$ by

$$\begin{array}{l} Z_{k+1} = ((P_k + I)Z_k)_{\varepsilon_k} \\ P_{k+1} = I - Z_{k+1}M, \end{array} \quad k \in \mathbb{N}_0, \tag{4.50}$$

with the starting values $Z_0 = Q$, $P_0 = P$. For notational simplicity let us denote

$$Z_k^{'} = (P_k + I)Z_k, \tag{4.51}$$

which satisfies the inequality $drk(Z_k^{'}) \leq drk(P_k)+drk(Z_k)$. After we use the $aodr$ of $Z_k^{'}$ we have that the displacement rank of the sequence $\{Z_k\}$ is low during the iteration process. The low displacement rank of the sequence of matrices P_k is also retained, since $drk(P_k) \leq drk(Z_k) + l$, where $l = drk(M)$.

In the rest of this section we present an algorithm for computing the outer inverse with prescribed range and null space of an arbitrary Toeplitz matrix.

In our modified SMS method we compute and store the SVD of $\Delta(Z_k)$ instead of Z_k and later determine Z_k using (4.45). From (4.51) and Proposition 4.4.4 we obtain

$$\Delta(Z_k^{'}) = \Delta(P_k)Z_k + P_k\Delta(Z_k) + \Delta(Z_k) + 2P_k e_1 e_n^T Z_k. \tag{4.52}$$

If we denote the SVD of $\Delta(Z_k)$ by $U_Z\Sigma_Z V_Z^T$ and the SVD of $\Delta(P_k)$ by $U_P\Sigma_P V_P^T$ then the extended form of $\Delta(Z_k^{'})$ can be rewritten by the following matrix form

$$\Delta(Z_k^{'}) = U_{Z'}\Sigma_{Z'}V_{Z'}^T = [U_P \ \ (P_k + I)U_Z \quad P_k e_1] \begin{bmatrix} \Sigma_P & 0 & 0 \\ 0 & \Sigma_Z & 0 \\ 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} V_P^T Z_k \\ V_Z^T \\ e_n^T Z_k \end{bmatrix}. \tag{4.53}$$

On the other hand, in order to compute the SVD of $\Delta(P_{k+1})$, which we need for the next iteration, we have the following

$$\Delta(P_{k+1}) = \Delta(I) - \Delta(Z_{k+1})M - Z_{k+1}\Delta(M) - 2Z_{k+1}e_1e_n^T M.$$

Let us denote the SVD of $\Delta(Z_{k+1})$ by $U_Z\Sigma_Z V_Z^T$ and the SVD of $\Delta(M)$ by $U_M\Sigma_M V_M^T$. The extended form of the previous identity can be written by

$$\Delta(P_{k+1}) = U_P\Sigma_P V_P^T = [-U_Z \ -Z_{k+1}U_M \ -Z_{k+1}e_1 \ -e_1] \begin{bmatrix} \Sigma_Z & 0 & 0 & 0 \\ 0 & \Sigma_M & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} \begin{bmatrix} V_Z^T M \\ V_M^T \\ e_n^T M \\ e_n^T \end{bmatrix}. \tag{4.54}$$

Therefore the SVD of $\Delta(Z_{k+1})$ can be computed by Algorithm 4.4.2.

Algorithm 4.4.2 The *odg* of Z_{k+1} and the *odg* of P_{k+1}

Input: The *odg* of Z_k, the *odg* of P_k, the *odg* of M and the truncation value ε.

1: Determine the matrices $U_{Z'}$, $\Sigma_{Z'}$, $V_{Z'}^T$ according to (4.53).
2: Compute the QR factorizations $U_{Z'} = Q_1 R_1$ and $V_{Z'} = Q_2 R_2$.
3: Compute the SVD of $R_1 \Sigma_{Z'} R_2^T = U \Sigma V$.
4: Set $\hat{\Sigma} = Diag(\sigma_1, \sigma_2, \ldots, \sigma_r)$, such that r is defined in relation to the truncation value ε: $\sigma_{r+1} \leq \varepsilon \sigma_1 < \sigma_r$.
5: Set $U_{Z_{k+1}} = Q_1 \hat{U}$, $\Sigma_{Z_{k+1}} = \hat{\Sigma}$ and $V_{Z_{k+1}} = Q_2 \hat{V}$, where $\hat{U}$ and $\hat{V}$ present the first r columns of U, V respectively.
6: Determine the matrices U_P, Σ_P, V_P^T according to (4.54).
7: Compute the QR factorizations $U_P = Q_3 R_3$ and $V_P = Q_4 R_4$.
8: Compute the SVD of $R_3 \Sigma_P R_4^T = U \Sigma V$.
9: Set $U_{P_{k+1}} = Q_3 U$, $\Sigma_{P_{k+1}} = \Sigma$ and $V_{P_{k+1}} = Q_4 V$.

The computational cost of Algorithm 4.4.2 is about $\mathcal{O}(mh)$ FFT's, where $h = \max_k drk(Z_k)$. If h is small enough, or even is independent of n, the algorithm requires $\mathcal{O}(n \log n)$ operations per step.

To complete our modified SMS method we present Algorithm 4.4.3 for finding the outer inverse of a general structured matrix given in terms of *odr* for the approximate matrices with respect to the operator Δ.

Algorithm 4.4.3 Modified SMS algorithm

Input: A Toeplitz matrix M and its displacement operator Δ, a matrix R, a relaxation parameter β and a residual error bound ξ.

1: Set $k = 0$, $Z_k = \beta R$ and $P_k = I - \beta R M$
2: Compute the SVD of $\Delta(P_k) = U_{P_k} \Sigma_{P_k} V_{P_k}^T$, $\Delta(Z_k) = U_{Z_k} \Sigma_{Z_k} V_{Z_k}^T$ and $\Delta(M) = U_M \Sigma_M V_M^T$.
3: Determine the truncation value ε_k.
4: Compute the *odg* $(U_{Z_{k+1}}, \Sigma_{Z_{k+1}}, V_{Z_{k+1}})$ of Z_{k+1} and $(U_{P_{k+1}}, \Sigma_{P_{k+1}}, V_{P_{k+1}})$ of P_{k+1} according to Algorithm 4.4.2 with input $(U_{Z_k}, \Sigma_{Z_k}, V_{Z_k})$, $(U_{P_k}, \Sigma_{P_k}, V_{P_k})$, $(U_M \Sigma_M V_M^T)$ and $\varepsilon = \varepsilon_k$.
5: Determine the norm $\|res(Z_{k+1})\|$. If $\|res(Z_{k+1})\| < \xi$ goto step 7, otherwise continue
6: Set $k = k + 1$, goto step 3
7: Return the matrix Z_{k+1}.

The value of $\|res(Z_k)\|$ in Step 5 presents the norm of the residual $\|res(Z_k)\| = \|Z_k M Z_k - Z_k\|$. Since, in general, computing $\|res(Z_k)\|$ is an expensive operation, a cheaper way is to compute the norm of the vector $\|res(Z_k)e_1\|$.

- **Convergence properties**

It is well known that matrices with a AB-displacement structure possess generalized inverses with a BA-displacement structure. Investigations in this field started in the paper [64] for the Moore-Penrose inverse. In the recent paper [84] these investigations are extended to $M_{V,U}^{(2)}$ inverse.

Proposition 4.4.8. [84] *Let $M \in \mathbb{C}^{n \times n}$ be an arbitrary matrix and $M_{T,S}^{(2)}$ be its $\{2\}$-inverse. Then*

$$\operatorname{rank}(\Delta_{B,A}(M_{T,S}^{(2)})) \leq \operatorname{rank}(\Delta_{A,B}(M)) + \operatorname{rank}(\Delta_{B,A}(R)).$$

In the case when the input matrix M has a low displacement rank with respect to $\Delta_{A,B}$ and R has a low displacement rank with respect to $\Delta_{B,A}$, the generalized inverse $M^{(2)}_{T,S}$ also possesses a low displacement rank with respect to $\Delta_{B,A}$. As a consequence, it is evident that choosing the matrix R to be a matrix with low displacement rank (with respect to $\Delta_{B,A}$) is of crucial importance to keep the low displacement rank of the outer inverse with prescribed range and null space. Then, it is possible to use the following result from [60] in order to show the convergence of the algorithm.

Theorem 4.4.3. [60] *Let V be a normed space and consider a function $f : V \to V$ and $M \in V$. Assume that $B = f(M)$ can be obtained by the iteration of the form*

$$X_k = \Phi(X_{k-1}),\ k = 1, 2, \ldots, \tag{4.55}$$

where Φ is a one-step operator. Further, assume that for any initial guess X_0 sufficiently close to B, the process converges:

$$\lim_{x\to\infty} X_k = B.$$

Assume that there are constants $c_\Phi, \epsilon_\Phi > 0$ and $\alpha > 1$ such that

$$\|\Phi(X) - B\| \le c_\Phi \|X - B\|^\alpha \text{ for all } X \in V \text{ with } \|X - B\| \le \epsilon_\Phi. \tag{4.56}$$

Let $S \subset V$ be a set of structured elements and $trunc_\varepsilon : V \to S$ be a truncation operator such that

$$\|X - X_\varepsilon\| \le c_R \|X - B\| \text{ for all } X \in V \text{ with } \|X - B\| \le \epsilon_\Phi. \tag{4.57}$$

Then there exists $\delta > 0$ such that the truncated iterative process $Y_k = (\Phi(Y_{k-1}))_\varepsilon$ converges to B so that

$$\|Y_k - B\| \le c_{R\Phi} \|Y_{k-1} - B\|^\alpha \text{ with } c_{R\Phi} = (c_R + 1)c_\Phi,\ k = 1, 2, \ldots. \tag{4.58}$$

for any starting value $Y_0 = (Y_0)_\varepsilon$ satisfying $\|Y_0 - B\| < \delta$.

For our iterative process, the condition (4.56) in Theorem 4.4.3 is undoubtly satisfied for the choices $c_\Phi = \|M\|$ and $\alpha = 2$. The choice of the truncation value ε_k at each step of the iterative process (4.50) is very important, since it indicates what will be the value of the displacement rank r, which is actually the balance point between the convergence of the process and the low computational requirements of the algorithm. Instead of analyzing the conditions under which (4.57) is satisfied we give some specifics in the following alternative proof for the convergence, which characterizes the described process. In addition, the presented result also determines the distances between the values of the real process and the truncated process at each iteration.

The notion of the q-Pochhammer symbol (or q-rising factorial), defined by

$$(a, q)_k = \prod_{i=0}^{k-1} (1 - aq^i)$$

is used to derive the next results.

Theorem 4.4.4. *Let $M \in \mathbb{C}^{n\times n}_\rho$ and $R \in \mathbb{C}^{n\times n}_s, 0 \le s \le \rho$ be given such that the condition (4.33) is satisfied for the matrix M. The sequence of approximations $\{Z_k\}$ given by (4.50) converges to the outer inverse $Z = M^{(2)}_{\mathcal{R}(R),\mathcal{N}(R)}$ of the matrix M if the following conditions are satisfied:*

1. *β is a real number such that*
$$\max_{1\le i\le t}|1-\beta\lambda_i|<1,$$
where $\mathrm{rank}(RM)=t$, λ_i, $i=1,\dots t$ *are eigenvalues of* RM.

2. *The truncation values* ε_k, $k\in\mathbb{N}_0$ *are chosen such that*
$$\varepsilon_k=\frac{2}{n(h_k^{'}-h_{k+1})}\theta_k, \tag{4.59}$$
where
$$h_k^{'}=drk(Z_k^{'}),\ h_k=drk(Z_k), k\ge 0,\ \theta_0=\min\left\{\varepsilon,\frac{\delta^2-\|P\|^2}{\|M\|}\right\},$$
and
$$\theta_k=\min\left\{\varepsilon^{k+1},\frac{\delta^{k+2}(1-\delta^k)}{\|M\|}\right\},\ k\ge 1, \tag{4.60}$$
for $0<\varepsilon\ll 1$ *and* $\delta=1-\varepsilon$.

3. *We chose a matrix norm* $\|\cdot\|$ *such that the condition* (4.24) *from Lemma* 4.4.1 *is satisfied for* $H=I-\beta MR$ *and also that the property* $\|P\|\le 1-\varepsilon$ *is obeyed.*

Under these assumptions we have the quadratic convergence of the algorithm, defined by the rule
$$\frac{\|Z-Z_k\|}{\|Z\|}\le\left(\max_{1\le i\le t}|1-\beta\lambda_i|\right)^{2^k}+\mathcal{O}(\varepsilon) \tag{4.61}$$
and the norm estimation
$$\|R_k\|\le\frac{(-1,\delta)_{k+1}}{2(1+\delta)}\cdot\frac{\varepsilon\left(1-\varepsilon^k\right)}{1-\varepsilon}=\mathcal{O}(\varepsilon),\ k\ge 0, \tag{4.62}$$
where $\{Y_k\}$ *is the sequence generated by the SMS algorithm and* $R_k=Z_k-Y_k$.

Proof. Without loss of generality let us assume that $\|I\|=1$. Otherwise, in the case if $\|I\|>1$, instead of the norm $\|\cdot\|$ which satisfies Condition 3 we can use the induced matrix norm $N(M)=\sup_{\|E\|=1}\|ME\|$. According to Theorem 5.6.26 from [67] the new norm satisfies $N(M)\le\|M\|$ for all $M\in\mathbb{C}^{n\times n}$ and $N(I)=1$.

Further, since the matrix Z_{k+1} is the *aodr* of Z_k', it is possible to use $Z_{k+1}=Z_k'+E_k$. According to (4.59) as well as (4.46) it is evident that $\|E_k\|\le\theta_k$, $k\ge 0$. Let us prove the auxiliary result
$$\|P_k\|\le\delta^{k+1},\ k\ge 0.$$
In the case $k=0$ we have $\|P_0\|=\|P\|\le\delta$. The case $k=1$ follows from
$$P_1=I-Z_1M=I-(P_0+I)Z_0M-E_0M=I-Z_0M-P_0Z_0M-E_0M=P_0^2-E_0M,$$
which produces
$$\|P_1\|\le\|P_0\|^2+\theta_0\|M\|\le\|P\|^2+\frac{\delta^2-\|P\|^2}{\|M\|}\|M\|=\delta^2.$$

Assuming that $\|P_k\| \le \delta^{k+1}$ and taking into account $P_{k+1} = P_k^2 - E_k M$ as well as $\|E_k\| \le \theta_k$ we verify the inductive step

$$\|P_{k+1}\| \le \|P_k\|^2 + \theta_k\|M\| \le \delta^{2k+2} + \frac{\delta^{k+2}(1-\delta^k)}{\|M\|}\|M\| = \delta^{k+2}.$$

Now we estimate the norm $\|R_k\|$ as in (4.62). In the case $k = 0$ we have $\|R_0\| = 0$. To prove the statement in the case $k \ge 1$ we again use the mathematical induction. The base of the induction (case $k = 1$) can be verified from

$$Z_1 = (Z_0{}')^-_{\varepsilon_0} = (Y_1)^-_{\varepsilon_0} = Y_1 + E_0,$$

which implies

$$\|R_1\| = \|E_0\| \le \theta_0 \le \varepsilon = \frac{(-1,\delta)_2}{2(1+\delta)}\varepsilon.$$

Let us suppose the approximation of the matrix norm $\|R_k\|$ as in (4.62). Then the inductive step follows from

$$\begin{aligned} Z_{k+1} &= ((P_k + I)Z_k)^-_{\varepsilon_k} = (P_k + I)(Y_k + R_k) + E_k \\ &= Y_{k+1} + (P_k + I)R_k + E_k, \end{aligned}$$

which together with (4.60) gives

$$\begin{aligned} \|R_{k+1}\| &\le (\delta^{k+1}+1)\|R_k\| + \theta_k \le (\delta^{k+1}+1)\frac{(-1,\delta)_{k+1}}{2(1+\delta)} \cdot \frac{\varepsilon\left(1-\varepsilon^k\right)}{1-\varepsilon} + \theta_k \\ &\le (\delta^{k+1}+1)\sum_{s=1}^{k}\prod_{i=2}^{k}(\delta^i+1)\varepsilon^s + \varepsilon^{k+1} \le \sum_{s=1}^{k}\prod_{i=2}^{k+1}(\delta^i+1)\varepsilon^s + \prod_{i=2}^{k+1}(\delta^i+1)\varepsilon^{k+1} \\ &= \sum_{s=1}^{k+1}\prod_{i=2}^{k+1}(\delta^i+1)\varepsilon^s. \end{aligned}$$

So continuing the previous inequalities we obtain

$$\|R_{k+1}\| \le \frac{(-1,\delta)_{k+2}}{2(1+\delta)}\sum_{s=1}^{k+1}\varepsilon^s = \frac{(-1,\delta)_{k+2}}{2(1+\delta)} \cdot \frac{\varepsilon\left(1-\varepsilon^{k+1}\right)}{1-\varepsilon} = \mathcal{O}(\varepsilon).$$

In this way, we verify the induction and prove $\|R_k\| = \mathcal{O}(\varepsilon)$.

Denote by $Y = \lim Y_k$, $Z = \lim Z_k$. Since $\sum_{i=0}^{\infty}\delta^i < \infty$, it follows that $(-1,\delta)_k$ converges to $(-1,\delta)_\infty$, and

$$\|Z - Y\| \le \frac{(-1,\delta)_\infty}{2(2-\varepsilon)} \cdot \frac{\varepsilon}{1-\varepsilon} = \mathcal{O}(\varepsilon).$$

Therefore, the estimation (4.62) is verified and it is possible to use $Z = Y$. Now, following the assumptions 1 and 3 we have

$$\frac{\|Y - Y_k\|}{\|Y\|} \le \left(\max_{1\le i\le t}|1-\beta\lambda_i|\right)^{2^k} + \mathcal{O}(\varepsilon),$$

(see also [126], Theorem 2.1). On the other hand we got the estimate for $\|R_k\|$, which produces

$$\|Z - Z_k\| = \|Z - Y_k - R_k\| = \|Y - Y_k - R_k\| \le \|Y - Y_k\| + \|R_k\|.$$

Finally, we get

$$\frac{\|Z - Z_k\|}{\|Z\|} \leq \left(\max_{1\leq i\leq t} |1 - \beta\lambda_i|\right)^{2^k} + \mathcal{O}(\varepsilon) + \frac{(-1,\delta)_{k+1}}{2(1+\delta)\|Z\|} \cdot \frac{\varepsilon\left(1-\varepsilon^k\right)}{1-\varepsilon},$$

which immediately confirms (4.61) and the quadratic convergence of the sequence (4.50).

It is known that all norms on a finite dimensional vector space are equivalent and we found a norm for which the process converges. Therefore, the general convergence with respect to any norm follows. □

However, based on some strategies, one can choose ε_k to be constant during the process or to be dynamic (to depend on some changeable parameters). A realistic strategy in choosing the value for ε is according to some specific heuristics. Some heuristics for the choice of the parameter ε_k are presented in the next section.

Remark 4.4.3. *For the purpose of choosing appropriate truncation value we remark that the restrictive condition $\|P\| \leq 1-\varepsilon$ in Theorem* 4.4.4 *is always satisfied in the case of the ordinary inverse, the Moore-Penrose inverse, the group and the Drazin inverse. In the first two cases we have $R = M^*$, which yields $P = P^*$ i.e. $\rho(P) = \|P\|$ and in the case of the Drazin inverse $R = M^l$, $l \geq \mathrm{ind}(M)$ from which follows $P = H$.*

- **Application and numerical examples**

The {2}-inverses have application in the iterative methods for solving nonlinear equations [12, 99] as well as in statistics [52, 69]. In particular, outer inverses play an important role in stable approximations of ill-posed problems and in linear and nonlinear problems involving rank-deficient generalized inverses [97, 152].

On the other hand, the theory of Toeplitz matrices arises in scientific computing and engineering, for example, image processing, numerical differential equations and integral equations, time series analysis and control theory (see, for example [23, 72]). Toeplitz matrices play an important role in the study of discrete time random processes. Covariance matrices of weakly stationary processes are Toeplitz matrices and triangular Toeplitz matrices provide a matrix representation of causal linear time invariant filters [57]. The Moore-Penrose inverse of a Toeplitz matrix is used in the image restoration process [15].

In the sequel we state some results which are relevant for computing outer inverses of an arbitrary Toeplitz matrix $M \in \mathbb{C}^{n\times n}$. In order to get an explicit representation of the outer inverse of M, which is a limit value of the sequence of approximations (4.50) for given M and R, we observe the full rank factorization of the matrix R. In that sense we have the following result.

Corollary 4.4.5. *For given $M \in \mathbb{C}_\rho^{n\times n}$ let us choose an arbitrary matrix $R \in \mathbb{C}_s^{n\times n}$, $0 \leq s \leq \rho$, and its full rank factorization $R = FG$, $F \in \mathbb{C}_s^{n\times s}$, $G \in \mathbb{C}_s^{s\times n}$, such that GMF is invertible. Suppose that β is a selected real number satisfying $\max\limits_{1\leq i\leq s} |1-\beta\lambda_i| < 1$, where λ_i are eigenvalues of FGM. Under the assumption* 2 *from Theorem* 4.4.4 *the sequence of approximations given by* (4.50) *converges to the outer inverse of M,*

$$Z = F(GMF)^{-1}G = M^{(2)}_{\mathcal{R}(F),\mathcal{N}(G)} \in M\{2\}_s.$$

Proof. Follows immediately from $Z = Y$, Corollary 4.4.3 and Theorem 4.4.4. □

Corollary 4.4.6. *Under the assumptions of Theorem* 4.4.4 *the following is valid:*
(*i*) $Z = M^D$ *in the case* $R = M^l$, $l \geq \text{ind}(M)$;
(*ii*) $Z = M^\dagger$ *in the case* $R = M^*$;
(*iii*) $Z = M^\#$ *in the case* $R = M$;
(*iv*) $Z \in M\{1,2\}$ *for* $s = \rho$.

Proof. It suffices to use the following representations:

$$M^D = M^{(2)}_{\mathcal{R}(M^l),\mathcal{N}(M^l)},\ l \geq \text{ind}(M),\quad M^\dagger = M^{(2)}_{\mathcal{R}(M^*),\mathcal{N}(M^*)},\quad M^\# = M^{(2)}_{\mathcal{R}(M),\mathcal{N}(M)},$$
$$M\{1,2\} = M\{2\}_\rho$$

in conjunction with Corollary 4.4.5. □

Next, we report some numerical results obtained by testing our method, named modified SMS method, for computing outer inverses of a Toeplitz matrix. We have selected 5 different types of Toeplitz matrices. During the execution of the algorithm we have observed and later reported three different indicators: number of iterations (No of iter), maximum displacement rank (Mdrk), as well as the sum of the displacement ranks (Sdrk). The code is written in the programming package MATLAB on a Workstation Intel Celeron 2.0 GHz. The stopping condition for Algorithm 4.4.3 is $\|res(Z_k)\| \leq \xi = 10^{-6}$. Also, the value of the starting parameter is $\beta = 1/\|RM\|$, where the matrix R is chosen according to Theorem 4.4.4 and Corollary 4.4.6.

In order to find the unified approach in choosing the parameter ε for all test matrices we have used the static choice $\varepsilon = 10^{-8}$, which is actually the square root of the double precision of the floating point arithmetic. In the case of computing the inverse as well as the group inverse of the matrix we present the strategy of choosing the ε according to some specific heuristic. This heuristic gives improvements of the algorithm such as lower Mdrk as well as lower Sdrk which decrease the computational cost of the algorithm and the number of operations per step.

Remark 4.4.4. *Based on Theorem* 4.4.4*, similarly as in* [126] *we can also determine a priori the number of steps required for achieving predefined accuracy. It is obvious that the number of steps for SMS and modified SMS is equivalent.*

By $M[p|q]$ we denote the submatrix of M obtained by extracting the rows indexed by p and the columns indexed by q. When $q = \{1, 2, \ldots, n\}$, we denote $M[p|q]$ by $M[p|:]$, and in case of $p = \{1, 2, \ldots, n\}$, we denote $M[p|q]$ by $M[:|q]$. The sequence $\{1, \ldots, s\}$ is simply denoted by $1 : s$.

Example 4.4.4. *In this example we consider the Toeplitz matrices* $M \in \mathbb{R}^{n\times n}_\rho$ *and* $R \in \mathbb{R}^{n\times n}_s$ (n *is an odd number*) *given by*

$$M = \text{toeplitz}[(1, 0, \ldots, 0, 1)],\quad R = \text{toeplitz}[(1, 0, \ldots, 0, 1, 0, \ldots, 0, 1)],$$

where $s = (n-1)/2 < \rho = n - 1$. *The vector which represents* R *has* 1 *in the middle.*

One of the possible full rank factorizations of the matrix $R = FG$ *is* $F = R[:|1 : s]$, $G = R[1 : s|:]$. *In the case* $n = 7$ *we have*

$$R = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

Based on Corollary 4.4.5*, the explicit representation of the outer inverse is*

$$M^{(2)}_{\mathcal{R}(F),\mathcal{N}(G)} = F(GMF)^{-1}G = \begin{bmatrix} 0.2 & 0 & 0 & 0.2 & 0 & 0 & 0.2 \\ 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0.5 & 0 \\ 0.2 & 0 & 0 & 0.2 & 0 & 0 & 0.2 \\ 0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0 & 0 & 0.5 & 0 \\ 0.2 & 0 & 0 & 0.2 & 0 & 0 & 0.2 \end{bmatrix}$$

The sequence of approximations $\{Z_k\}$ *given by Algorithm* 4.4.3 *also converges to the outer inverse given by this explicit representation. In Table* 4.4.9 *we present numerical results obtained by computing the outer inverse of* M *by Algorithm* 4.4.3 *considering several different dimensions of* M*. Also, the number of iterations of the original SMS method is presented as comparison.*

Table 4.4.9. Numerical results generated by computing the outer inverse of M

	Modified SMS			SMS
Dim	No of iter	Mdrk	Sdrk	No of iter
17	5	3	15	5
33	5	3	15	5
65	5	3	15	5
129	5	3	15	5
257	5	3	15	5
513	5	3	15	5
1025	5	3	15	5
2049	5	3	15	5

As one can see from Table 4.4.9 *the number of iterations of the original SMS method is the same as the number of iterative steps obtained by our method. Also, the* Mdrk *is very low for all dimensions of the matrix* M*, i.e.* Mdrk *=3.*

In the remaining of the section, for each test matrix, we have considered 8 different numerical experiments with the dimensions as follows: 16, 32, 64, 128, 256, 512, 1024, 2048.

Example 4.4.5. *The inverse of two matrices*

$$M_1 = \text{toeplitz } [(1, 1/2, \ldots, 1/n)], \quad M_2 = \text{toeplitz } [(4, 1, 0, \ldots, 0)],$$

chosen from [14]*, is computed in the case* $R = M^*$*. Numerical results obtained during the computation of the inverses of matrices* M_1 *and* M_2 *are given in Tables* 4.4.10 *and* 4.4.11*. Besides the static choice of the truncation value* $\varepsilon = 10^{-8}$*, we observe the heuristic* $\varepsilon = \max(res/\|M\|, 10^{-8})$.

Table 4.4.10. Numerical results generated by computing the inverse of M_1

	Modified SMS						SMS
	$\varepsilon = \max(res/\|M\|, 10^{-8})$			$\varepsilon = 10^{-8}$			
Dim	No of iter	Mdrk	Sdrk	No of iter	Mdrk	Sdrk	No of iter
16	11	2	15	11	13	105	11
32	11	2	17	12	16	129	12
64	12	2	20	13	16	148	13
128	13	2	25	13	16	158	13
256	13	3	27	14	16	171	14
512	14	4	35	14	16	176	14
1024	14	4	38	14	16	180	14
2048	14	4	42	15	16	189	15

Table 4.4.11. Numerical result generated by computing the inverse of matrix M_2

	Modified SMS						SMS
	$\varepsilon = \max(res/\|M\|, 10^{-8})$			$\varepsilon = 10^{-8}$			
Dim	No of iter	Mdrk	Sdrk	No of iter	Mdrk	Sdrk	No of iter
16	7	2	11	7	13	65	7
32	7	2	11	7	13	69	7
64	7	3	16	7	13	69	7
128	7	3	18	7	13	69	7
256	7	3	18	7	13	69	7
512	7	4	20	7	13	69	7
1024	7	4	21	7	13	69	7
2048	7	4	21	7	13	69	7

We see that the number of iterations is almost identical for the two different heuristics. On the other hand, it is evident that the great improvement with respect to all other indicators is attained by putting $\varepsilon = \max(res/\|M\|, 10^{-8})$ *instead of* $\varepsilon = 10^{-8}$*, observing the results for both matrices* M_1 *and* M_2*.*

Example 4.4.6. *The test matrix*

$$M = \text{toeplitz}[(1, 1/2, \ldots, 1/(n-1), 1), (1, 1/(n-1), \ldots, 1/2, 1)]$$

is taken from [19]*. Tables* 4.4.12 *and* 4.4.13 *report the numerical results obtained during the computation of the group inverse* (*generated in the case* $R = M$)*, as well as the Moore-Penrose inverse* ($R = M^*$) *of a Toeplitz matrix* $M \in \mathbb{R}^{n\times n}$*.*

Table 4.4.12. Numerical results for computing the group inverse

	Modified SMS						SMS
	$\varepsilon = \min((1 + 10^{-6} - \|P_k\|^2)/\|M\|, 10^{-6})$			$\varepsilon = 10^{-8}$			
Dim	No of iter	Mdrk	Sdrk	No of iter	Mdrk	Sdrk	No of iter
16	9	9	60	9	9	66	9
32	10	10	66	10	10	76	10
64	10	10	68	10	11	81	10
128	10	9	72	10	12	87	10
256	11	9	77	11	11	92	11
512	11	9	80	11	11	91	11
1024	11	9	82	11	11	93	11
2048	11	9	82	11	11	95	11

Let us mention that another dynamic heuristic is chosen for computing the group inverse with respect to the heuristic used in Example 4.4.5. *It is obvious that the presented heuristic, which makes* ε *at least as small as* 10^{-6}*, provides significant improvements of the observed indicators with respect to the static choice* $\varepsilon = 10^{-8}$.

Table 4.4.13. Numerical results generated by computing the Moore-Penrose inverse

	Modified SMS			SMS
Dim	No of iter	Mdrk	Sdrk	No of iter
16	9	13	84	9
32	10	14	98	10
64	10	15	107	10
128	10	15	112	10
256	11	15	119	11
512	11	15	122	11
1024	11	15	124	11
2048	11	16	128	11

It is interesting to point out that in the tables 4.4.12 and 4.4.13 the results are obtained for the same starting matrix M as well as that the group and the Moore-Penrose inverse, are identical. As one can see from the tables, although the number of iterations is the same, the Mdrk as well as Sdrk required during the computation of the group inverse is essentially smaller than in the case of computing the MP inverse, for the same choice of the parameter $\varepsilon = 10^{-8}$.

Example 4.4.7. *Let us consider the matrix* $M \in \mathbb{R}^{n\times n}$ *represented by*

$$M = \text{toeplitz}[(0,\dots,0,1,1,0,\dots,0),(0,0,\dots,0,1,1)].$$

Since the index of the matrix M *is* $\text{ind}(M) = 2$*, the Drazin inverse of* M *is computed in the case* $R = M^l$, $l \geq 2$. *For the chosen matrix* R *we have that* $\text{rank}(R) = 6$. *According to Corollary* 4.4.6, *the iterative process of the modified SMS method converges to the Drazin inverse. After the use of the static choice* $\varepsilon = 10^{-8}$ *we get the following results.*

Table 4.4.14. Numerical results generated by computing the Drazin inverse of M

	Modified SMS			SMS
Dim	No of iter	Mdrk	Sdrk	No of iter
16	8	6	48	8
32	8	6	48	8
64	8	6	48	8
128	8	6	48	8
256	8	6	48	8
512	8	6	48	8
1024	8	6	48	8
2048	8	6	48	8

It is evident from Table 4.4.14 *that* Mdrk $= 6$, *which is a good result having in mind that* $\text{drk}(Z_0) = \text{drk}(\beta R) = 4$.

Chapter 5

Application in image restoration

5.1 Preliminaries

Images are produced to memorize useful information, but unfortunately the presence of blur is unavoidable. Motion blur is the effect of the relative motion between the camera and the scene during image exposure time. Restoration of motion-blurred images has been a fundamental problem in digital imaging for a long time. The recovery of an original image from degraded observations is of crucial importance and finds application in several scientific areas including medical imaging and diagnosis, military surveillance, satellite and astronomical imaging, remote sensing etc. Expectably, the field of image restoration has been of great interest in the scientific literature [7, 24, 27, 28, 66, 113].

We consider the problem of removing uniform and non–uniform blur in images, which corresponds to an integral number of pixels. The recorded image, degraded with linear motion (denoted by an image array G), is usually modeled as a linear convolution of the original image (denoted by an image array F) with a point spread function (PSF), also known as the blurring kernel (represented by a matrix H).

As we already discussed, the Moore-Penrose inverse is a useful tool for solving linear systems and matrix equations. It appears that the properties of the Moore-Penrose inverse imply its extensive usage in the image restoration process [17, 27, 28, 29]. The approach based on the usage of the matrix pseudo-inverse in the image reconstruction, presents one of the most common approaches [17]. The appearance of blur, caused by a linear motion, is modeled by the matrix equations $FH^T = G$ and $HF = G$ with respect to the unknown matrix F, where H, F and G are given matrices of appropriate dimensions. The Moore-Penrose inverse $H^\dagger$ of the matrix H which causes blurring to the original image F, resulting with degraded image G, has been used to solve these equations [27, 28].

Given in other words, the main problem we are faced to, is choosing an efficient algorithm for computing the Moore-Penrose inverse $H^\dagger$. The algorithm used in [27, 28] for computing $H^\dagger$ is based on the fast computational method for finding the Moore-Penrose inverse of full rank matrix, introduced in [79, 80]. Approximations obtained in [27] are reliable and very accurate. A lot of direct methods were proposed to compute the Moore-Penrose generalized inverse of a matrix (see for instance [12, 118]). According to [118], they can be classified as: methods based on matrix decomposition; methods based on the formula $A^\dagger = (A^*A)^{(1,3)}A^*$ and Pyle's gradient projection methods. The method based on Singular-Value Decomposition possesses very high computation load (approximately $\mathcal{O}(n^3)$ operations). P. Courrieu in [36] proposed an

algorithm for fast computation of the Moore-Penrose inverse based on a known reverse order law and on a full-rank Cholesky factorization of possibly singular symmetric positive matrices. A fast method for computing the Moore-Penrose inverse of full rank $m \times n$ matrices and of square matrices with at least one zero row or column is introduced in [79, 80]. This method exploits a special type of tensor product of two vectors, that is usually used in infinite dimensional Hilbert spaces. Greville in [58] proposed a recurrent rule for determining the Moore-Penrose inverse. Udwadia and Kalaba gave an alternative and a simple constructive proof of Grevilles formula [134]. Due to its ability to undertake sequential computing, the Grevile's partitioning method has been extensively applied in statistical inference, filtering theory, linear estimation theory, system identification, optimization as well as in analytical dynamics [56, 75, 76, 108, 159]. Recursive computation of the Moore-Penrose inverse of a matrix, when a block is added to it, was presented by Bhimsankaram [13]. However, Bhimsankaram used a proof which simply verified that the output of his algorithm satisfies the four conditions for the Moore-Penrose inverse. Udwadia and Kalaba in [135] provided a constructive proof for the recursive determination of the Moore-Penrose inverse of a matrix to which a block of columns is added. In [135] these results are also extended for other types of generalized inverses.

Our intention in this chapter is, to present one direct method for recovering blurred images by a uniform blur, and to present an application of the recursive block partitioning method from [135] as well as the partitioning method from [58] in the process of removing non–uniform blur from the images. More precisely, both the block partitioning method and the Grevile's single-column partitioning method are appropriately modified and applied in computing the Moore-Penrose inverse solution of the matrix equations $H_c F H_r^T = G$ with respect to unknown matrix F. The motivation for the usage of these methods is the specific structure of the convolution matrix H. The characteristic structure of the matrix H reduces the computational complexity of the partitioning method in the pseudoinverse $H^\dagger$ calculation.

5.1.1 Uniform linear blur

In this paragraph we are going to shortly explain how the phenomenon of uniform linear blurring is modeled mathematically.

Let $\mathbb{R}$ be the set of real numbers, $\mathbb{R}^{m\times n}$ be the set of $m \times n$ real matrices. Suppose that the matrix $F \in \mathbb{R}^{r\times m}$ corresponds to the original image with picture elements $f_{i,j}$, $i = 1, \ldots, r$, $j = 1, \ldots, m$ and $G \in \mathbb{R}^{r\times m}$ with pixels $g_{i,j}$, $i = 1, \ldots, r$, $j = 1, \ldots, m$, is the matrix corresponding to the degraded image. Let l be an integer indicating the length of the linear motion blur in pixels and $n = m+l-1$. In practice the degradation (index l) is rarely known exactly, so that it must be identified from the blurred image itself. To estimate the index l, two different cepstral methods can be used: one dimensional or two dimensional cepstral method [81]. To avoid the problem when the information from the exact image spills over the edges of the recorded image, we supplement the original image with boundary pixels that best reflect the original scene. Without any confusion we are using the same symbol F for the enlarged original image (matrix) with remark that F now becomes a matrix of dimensions $r \times n$. First, we suppose that the blurring is a horizontal phenomenon. Let us denote the degradation matrix by $H \in \mathbb{R}^{m\times n}$. For each row f_i of the matrix F and the respective row g_i of the matrix G we consider an equation of the form

$$g_i^T = H f_i^T, \quad g_i^T \in \mathbb{R}^m, \ f_i^T \in \mathbb{R}^n, \ H \in \mathbb{R}^{m\times n}. \tag{5.1}$$

The objective is to estimate the original image F, row per row, using the corresponding rows of the known blurred image G and a priori knowledge of the degradation phenomenon H. Equation (5.1) can be written in the matrix form as

$$G = \left(HF^T\right)^T = FH^T, \;\; G \in \mathbb{R}^{r\times m}, \; H \in \mathbb{R}^{m\times n}, \; F \in \mathbb{R}^{r\times n}. \tag{5.2}$$

There is an infinite number of exact solutions for f that satisfy the equation (5.1). But, only the Moore-Penrose solution solves the next minimization problem (see, for example [12]):

$$\min \|f\|_2, \text{ subject to } \min \|Hf - g\|_2. \tag{5.3}$$

The unique vector $\tilde{f}$ satisfying (5.3) can be taken as the row of the restored image [17, 27, 28, 29], defined by

$$\tilde{f} = H^\dagger g. \tag{5.4}$$

The matrix form of the equation (5.4), i.e., the restored image $\tilde{F}$ is given by

$$\tilde{F} = G(H^\dagger)^T. \tag{5.5}$$

The matrix $\tilde{F}$ defined in (5.5) is the minimum-norm least-squares solution of the matrix equation (5.2).

The matrix equation which characterizes the vertical motion blurring process is given by

$$G = HF, \;\; G \in \mathbb{R}^{r\times m}, \; H \in \mathbb{R}^{r\times n}, \; F \in \mathbb{R}^{n\times m}, \; n = r + l - 1. \tag{5.6}$$

The corresponding restored image can be computed using the Moore-Penrose inverse by the following formula

$$\tilde{F} = H^\dagger G. \tag{5.7}$$

If additionally, we suppose that the blurring is a local phenomenon, it is spatially invariant, the imaging process captures all light and no additional noise is included; then the point spread function initiated by all these conditions is the vector $(1/l, \ldots, 1/l) \in \mathbb{R}^l$. Consequently, when the blur is caused by a horizontal motion, the matrix $H = \text{toeplitz}(h^1, h_1)$ is a non-symmetric Toeplitz matrix consisting of m rows and $n = m + l - 1$ columns, determined by its first column $h^1 = (h_{i,1})_{i=1}^m$ and its first row $h_1 = (h_{1,j})_{j=1}^n$ as follows:

$$h_{i,1} = \begin{cases} 1/l, & i = 1, \\ 0, & i = 2, \ldots, m, \end{cases} \quad \text{and} \quad h_{1,j} = \begin{cases} 1/l, & j = 1, \ldots, l, \\ 0, & j = l+1, \ldots, n. \end{cases} \tag{5.8}$$

An arbitrary ith row of the blurred image can be expressed using the ith row of the original image extended with the boundary pixels as

$$\begin{bmatrix} g_{i,1} \\ g_{i,2} \\ g_{i,3} \\ \vdots \\ g_{i,m} \end{bmatrix} = \begin{bmatrix} \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} & 0 & 0 & 0 \ldots & 0 \\ 0 & \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} & 0 & 0 \ldots & 0 \\ 0 & 0 & \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} & 0 \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{l} & \frac{1}{l} & \cdots & \frac{1}{l} \end{bmatrix} \begin{bmatrix} f_{i,1} \\ f_{i,2} \\ f_{i,3} \\ \vdots \\ f_{i,n} \end{bmatrix}, \tag{5.9}$$

where $l - 1$ elements of the vector f_i, are not actually the pixels from the original scene; rather they are *boundary pixels*. How many boundary pixels will be added above the vector f depends of the nature and direction of the movement. However, the rest of them, i.e., $l - 1$ minus the number of pixels added above the vector f, would present the boundary pixels right of the horizontal line, and are added below the vector f [62].

5.1.2 Non-uniform linear blur

The presented model, of uniform linear motion, is further generalized to a model, where the blur of an image is caused by a non-uniform linear motion. By using the notation introduced previously, in the sequel it is exposed the mathematical model for this phenomenon.

The process of non–uniform blurring assumes that the blurring of the columns in the image is independent with respect to the blurring of the rows. Again, to avoid the problem when the information from the exact image spills over the edges of the recorded image, we supplement the original image with boundary pixels that best reflect the original scene. Without any confusion, for the enlarged image we are using the same symbol F, with remark that, now, F becomes a matrix of dimensions $n \times s$, where $n = r + l_c - 1$, $s = m + l_r - 1$, l_c is length of the vertical blurring and l_r is length of the horizontal blurring given in pixels. The relation between the original and blurred image can be displayed with the relation

$$G = H_c F H_r^T, \;\; G \in \mathbb{R}^{r\times m}, \; H_c \in \mathbb{R}^{r\times n}, \; F \in \mathbb{R}^{n\times s}, \; H_r \in \mathbb{R}^{m\times s}. \tag{5.10}$$

In order to restore the blurred image given by the model (5.10) we use the Moore–Penrose inverse approach which leads to the solution

$$\tilde{F} = H_c^{\dagger} G (H_r^{\dagger})^T.$$

An arbitrary non-uniform linear blurring process can be represented by (5.10) where the matrices H_c and H_r are characteristic Toeplitz matrices of the following general form

$$H = \left[\begin{array}{cccccccc|cccc}
h_1 & h_2 & h_3 & \dots & h_l & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & h_1 & h_2 & h_3 & \dots & h_l & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \ddots & \ddots & \ddots & \dots & \ddots & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & h_1 & h_2 & h_3 & \dots & h_l & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & h_1 & h_2 & h_3 & \dots & h_l & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & h_1 & h_2 & h_3 & \dots & h_l & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \ddots & \ddots & \ddots & \dots & \ddots & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & h_1 & h_2 & h_3 & \dots & h_l
\end{array}\right], \tag{5.11}$$

where the sum of the elements of each row is equal to 1.

In order to see how boundary conditions can be incorporated in the model, for the sake of simplicity, let us retain on the horizontal blurring model ($H_c = I$). Similarly, as in (5.9), an arbitrary ith row of the blurred image can be expressed using the ith row of the original image extended with the pixels from the boundary conditions as follows

$$\begin{bmatrix} g_{i,1} \\ \vdots \\ g_{i,m} \end{bmatrix} = H \begin{bmatrix} f_{i,1} \\ \vdots \\ f_{i,n} \end{bmatrix}, \qquad i = 1, 2, \dots, r, \tag{5.12}$$

where $l - 1$ elements of the vector f_i, are not actually the pixels from the original scene; rather they are *boundary pixels*.

5.2 Removal of uniform blur in X-ray images

The most simple case, of uniform linear blur, usually appears in the process of recording X-ray images. The methods based on the usage of the Moore-Penrose solution, in order to solve the matrix equations presented in Section 5.1.1, has been exploited in the image restoration process in recent papers [27, 28, 29]. Several methods for computing the Moore-Penrose inverse have been introduced (see, for example [12]). One of the most commonly used methods is the Singular Value Decomposition (SVD) method. This method is very accurate but also time-intensive since it requires a large amount of computational resources, especially in the case of large matrices. An algorithm for fast computation of the Moore-Penrose inverse is presented in the recent work of P. Courrieu [36]. Courrieu's algorithm is based on a known reverse order law for the matrix pseudoinverse (eq. 3.2 from [109]), and on a full-rank Cholesky factorization of possibly singular symmetric positive matrices. Another very fast and reliable method to estimate the Moore-Penrose inverse matrix of a rank-n tensor-product matrix is given by V. Katsikis and D. Pappas [80].

All known methods for computing the Moore-Penrose inverse are either iterative or use some matrix factorization. Our method explore the structure of the degradation matrix in a specific case and generates the Moore-Penrose inverse analytically, by means of a set of rules. Our motivation to introduce a new algorithm is the very proper structure of the matrix which participates as a degradation system in the image formation process. The introduced method is very fast, which is its main advantage. The main disadvantage of the method is its limitation to uniform linear motion blur degradations. Since our method is straightforward and does not use any iterations or factorizations, the presented numerical results only claim the expected saving of the CPU time. We firstly define the method in the case when the the number of the columns of the image enlarged for boundary pixels, can be divided by the number of blurring pixels. Later we extend the method in the general case.

As we can see from (5.9) the matrix H is a Toeplitz matrix with very suitable structure. We will show later in this section that its Moore-Penrose inverse in the case when $n = l \cdot p$ can be generated immediately, without any computation. Later we will benefit from this specific case to restore an image in general case, when the dimension n is arbitrary.

As we said, firstly we pay especial attention on the Moore-Penrose inverse of the matrix H in the case

$$n = m + l - 1 = l \cdot p,$$

where the number of blurring pixels p is a positive integer and divisor of the number of pixel of the enlarged image F. Our aim in the present paper is to show that in this case the blurring matrix is enough structured so its Moore-Penrose inverse can be calculated analytically. In the rest we will construct a matrix $\widetilde{H} = [\widetilde{h}_{ij}], i = 1, \ldots, n,\ j = 1, \ldots, m$, and show that it is actually the Moore-Penrose inverse of the degradation matrix H. We move systematically from the general layout of the matrix $\widetilde{H}$ in order to get its exact form.

All elements of the matrix $\widetilde{H}$, excluding zero elements, can be represented by the following two sequences:

$$x_k = -\frac{l}{n}(m - l(k-1) - 1) = -\frac{m - l(k-1) - 1}{p}, \quad k = 1, 2 \ldots, p-1,$$
$$y_k = \frac{l}{n}(m - l(k-1)) = \frac{m - l(k-1)}{p}, \qquad k = 1, 2 \ldots, p.$$

Additionally, we put $z = y_p = \frac{1}{p}$.

The following representation gives the general layout of the matrix $\widetilde{H} \in \mathbb{C}^{n\times n}$.

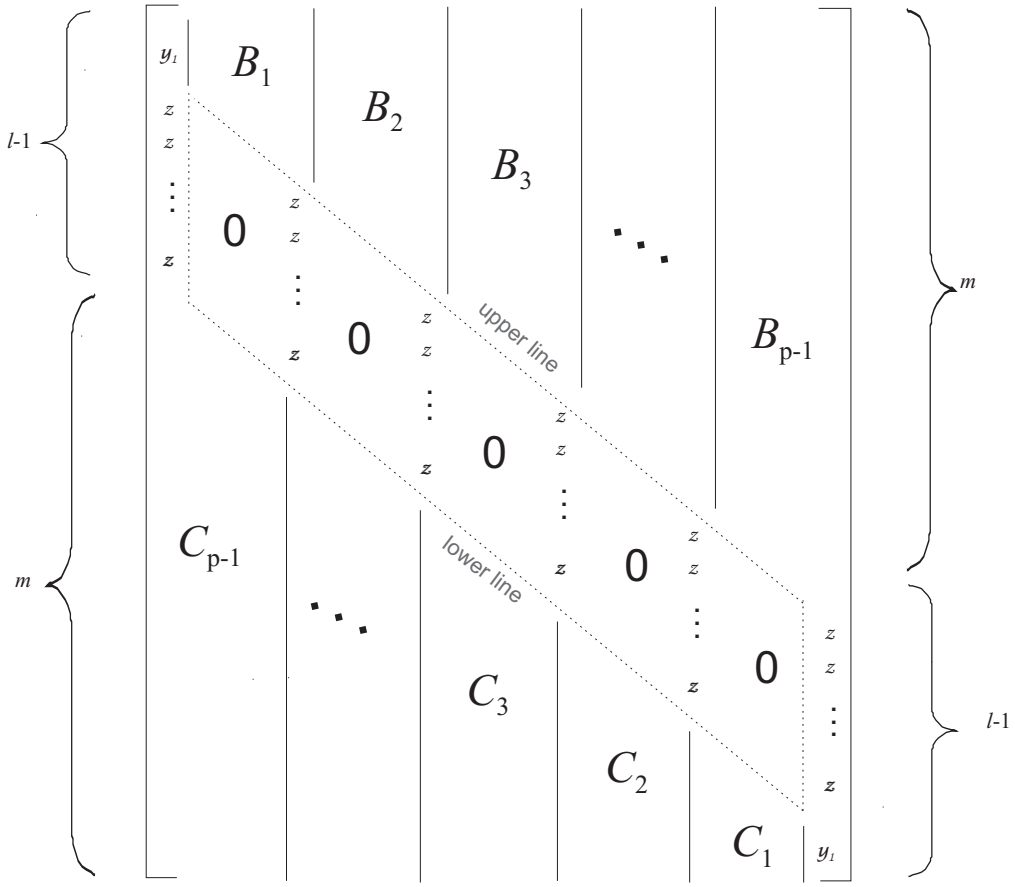

Figure 5.2.1. General layout of the matrix $H^{\dagger}$.

The parallelogram between the blocks B_k and C_k is called *the zero layer*. With *upper line* we refer to the elements above the zero layer of the matrix $\widetilde{H}$ which actually constitute the diagonal of the square $m \times m$ matrix formed from the first m rows of the matrix $\widetilde{H}$. Similarly, *lower line* refers to the elements below the zero layer of the matrix $\widetilde{H}$ which constitute the diagonal of the square $m \times m$ matrix formed from the last m rows of the matrix $\widetilde{H}$. The upper line, the lower line, the first l elements of the first row of $\widetilde{H}$ and the last l elements of the last column of $\widetilde{H}$ will be denominated as *sides of the zero layer*.

Further we preview into the structure of the blocks B_k, $k = 1, \ldots, p-1$. Each block B_k can be represented via appropriate block P_k of the following form:

$$P_k = \left|\begin{array}{ccccccc}
0 & & & & & & \\
0 & 0 & & & & & \\
0 & 0 & 0 & & & & \\
\vdots & \ddots & \ddots & \ddots & & & \\
0 & 0 & \dots & \dots & 0 & & \\
x_k & 0 & 0 & \dots & \dots & 0 & \\
-x_k & x_k & 0 & 0 & \dots & 0 & z \\
 & -x_k & x_k & 0 & 0 & \vdots & z \\
 & & \ddots & \ddots & \ddots & & \vdots \\
 & & & \ddots & \ddots & 0 & \vdots \\
 & & & & -x_k & x_k & z \\
 & & & & & -x_k & z + x_k \\
 & & & & & & y_{k+1}
\end{array}\right|$$

Zero parallelogram blocks are of dimensions $(l-2) \times (l-1)$. The block B_k is obtained by pasting up k times the block P_k, from the bottom upwards, and then from the resulting block we cut by horizontal line the most upper triangle which has a vertical side containing the upper $l-2$ elements of the block P_k. The missing element of the resulting block B_k is filled by the value y_{k+1}. The three steps in formation process of the blocks B_k are presented on Figure 5.2.2.

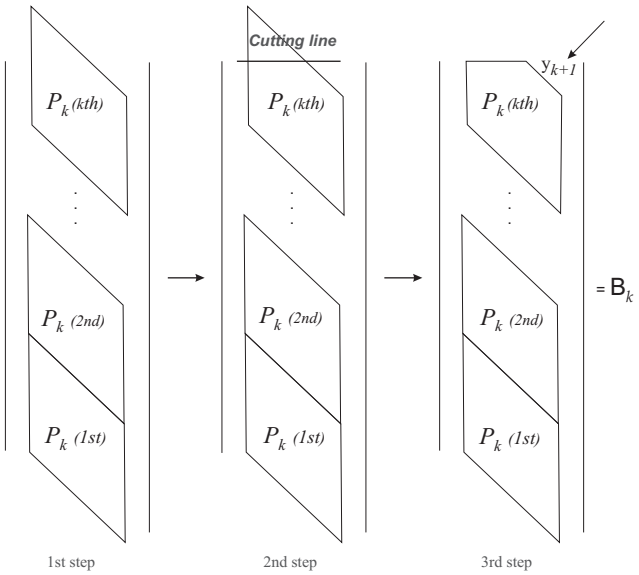

Figure 5.2.2. Formation of the block B_i from the block P_i.

In order to write the analytical form of the matrix $\widetilde{H}$ we will use the following notation. Let q_i, q_j, r_i and r_j be integers such that for a given ith row and jth column holds

$$i = q_i l + r_i \text{ and } j = q_j l + r_j.$$

From the previous considerations it is clear that:

$$i \leq j \quad \text{if and only if} \quad \widetilde{h}_{ij} \in B_k,\ k = 1, 2, \dots, p-1.$$

Similarly

$$i \geq l \text{ and } i > j \quad \text{if and only if} \quad \widetilde{h}_{ij} \in C_k,\ k = 1, 2, \ldots, p-1.$$

Taking this into account we present the analytical form of the matrix $\widetilde{H}$.

$$\widetilde{h}_{ij} = \begin{cases} y_{q_j+1}, & i \leq j,\ r_j = 1,\ r_i = 1, \\ z + x_{q_j}, & i \leq j,\ r_j = 1,\ r_i = 0, \\ (-1)^{d+1} x_{q_{j-1}+1}, & i \leq j,\ r_j \neq 1,\ q_j \geq q_i,\ (r_{i-j} = 0 \text{ or } r_{i-j} = l-1) \\ \\ z + x_{p-q_j-1}, & i \geq l,\ i > j,\ r_j = 1,\ r_i = 1, \\ y_{p-q_j}, & i \geq l,\ i > j,\ r_j = 1,\ r_i = 0, \\ (-1)^{d} x_{p-q_{j-1}-1}, & i \geq l,\ i > j,\ r_j \neq 1,\ q_j \leq q_i,\ (r_{i-j} = 0 \text{ or } r_{i-j} = l-1) \\ \\ z, & r_j = 1,\ r_i \neq 0,\ r_i \neq 1 \\ 0, & \text{otherwise} \end{cases} \tag{5.13}$$

where d is 0 if $r_{i-j} = 0$ and $d = 1$ otherwise. The first case in (5.13) gives the elements that are not equal to zero or z of the blocks B_k, $k = 0, \ldots, p-1$ plus y_1 from the first column. The second case produces the elements that are not equal to zero or z of the blocks C_k, $k = 0, \ldots, p-1$ plus y_1 from the last column. In order to store the whole matrix we need only to store $2p-1$ elements which is lower than n in the case $l > 2$.

To illustrate our description we give the full form of the matrix $\widetilde{H}$ observing the case $p = 4$.

$$\left[\begin{array}{cccccc|cccccc|cccccc|cccccc}
y_1 & x_1 & & & & y_2 & x_2 & & & & y_3 & x_3 & & & & y_4 \\
z & -x_1 & x_1 & & & z & -x_2 & x_2 & & & z & -x_3 & x_3 & & & z \\
z & & -x_1 & x_1 & & z & & -x_2 & x_2 & & z & & -x_3 & x_3 & & z \\
\vdots & & & \ddots & \ddots & \vdots & & & \ddots & \ddots & \vdots & & & \ddots & \ddots & \vdots \\
z & & & & -x_1 & z & & & & -x_2 & z & & & & -x_3 & z \\
y_4 & & & & -x_1 & z+x_1 & & & & -x_2 & z+x_2 & & & & -x_3 & z+x_3 \\
\hline
z+x_3 & -x_3 & & & & y_2 & x_2 & & & & y_3 & x_3 & & & & y_4 \\
z & x_3 & -x_3 & & & z & -x_2 & x_2 & & & z & -x_3 & x_3 & & & z \\
\vdots & & \ddots & \ddots & & \vdots & & -x_2 & x_2 & & z & & -x_3 & x_3 & & z \\
z & & & x_3 & -x_3 & z & & & \ddots & \ddots & \vdots & & & \ddots & \ddots & \vdots \\
z & & & & x_3 & z & & & & -x_2 & z & & & & -x_3 & z \\
y_4 & & & & x_3 & y_3 & & & & -x_2 & z+x_2 & & & & -x_3 & z+x_3 \\
\hline
z+x_3 & -x_3 & & & & z+x_2 & -x_2 & & & & y_3 & x_3 & & & & y_4 \\
z & x_3 & -x_3 & & & z & x_2 & -x_2 & & & z & -x_3 & x_3 & & & z \\
\vdots & & \ddots & \ddots & & \vdots & & \ddots & \ddots & & \vdots & & -x_3 & x_3 & & z \\
z & & & x_3 & -x_3 & z & & & x_2 & -x_2 & z & & & \ddots & \ddots & \vdots \\
z & & & & x_3 & z & & & & x_2 & z & & & & -x_3 & z \\
y_4 & & & & x_3 & y_3 & & & & x_2 & y_2 & & & & -x_3 & z+x_3 \\
\hline
z+x_3 & -x_3 & & & & z+x_2 & -x_2 & & & & z+x_1 & -x_1 & & & & y_4 \\
z & x_3 & -x_3 & & & z & x_2 & -x_2 & & & z & x_1 & -x_1 & & & z \\
\vdots & & \ddots & \ddots & & \vdots & & \ddots & \ddots & & \vdots & & \ddots & \ddots & & \vdots \\
z & & & x_3 & -x_3 & z & & & x_2 & -x_2 & z & & & x_1 & -x_1 & z \\
z & & & & x_3 & z & & & & x_2 & z & & & & x_1 & z \\
y_4 & & & & x_3 & y_3 & & & & x_2 & y_2 & & & & x_1 & y_1
\end{array}\right].$$

In order to verify that the matrix $\widetilde{H}$ is actually the Moore-Penrose inverse of the matrix H, i.e. that $\widetilde{H} = H^{\dagger}$ we will use the following two lemmas:

Lemma 5.2.1. *The equality $H\widetilde{H} = I$ holds for the matrix $\widetilde{H}$ given by* (5.13).

Proof. Each row of the matrix H contains l non zero constant elements equal to $1/l$, so that it is obvious that the elements of the matrix $H\widetilde{H}$ are

$$(H\widetilde{H})_{ij} = \frac{1}{l}\sum_{s=i}^{i+l-1} \widetilde{h}_{sj}, \quad i = 1, \ldots, m \quad \text{and} \quad j = 1, \ldots, m.$$

Therefore, we need to explore the properties of jth column of the matrix $\widetilde{H}$, $j = 1, \ldots, m$, i.e. the sums of its l consecutive elements. For this reason and from the representation of the matrix $\widetilde{H}$ given by (5.13) we distinguish two different cases:

1 case : $r_j = 1$.

Each set of l consecutive elements contains only two elements different from z. If both of them are above the zero layer their sum is $y_{q_j+1} + (l-1)z + x_{q_j} = 0$. If both of them are below the zero layer their sum is $x_{p-q_j-1} + (l-1)z + y_{p-q_j} = 0$. Otherwise, if one of them is above the zero layer and the other one is below the zero layer, then their sum is $y_{q_j+1} + y_{p-q_j} = l$. Unfortunately the last, as we mentioned before, is the case only when $i = j$.

2 case : $r_j \neq 1$.

Each set of l consecutive elements contains only two non zero elements. If those two elements are above the zero layer or both of them are below the zero layer then it is obvious that their sum is 0. If one of them is above the zero layer and the other one is below the zero layer, thus $i = j$, their sum is $-x_{q_{j-1}+1} - x_{p-q_{j-1}-1}$, which by easy calculations can be shown that equals l.

So finally for the both cases we have

$$(H\widetilde{H})_{ij} = \frac{1}{l}\sum_{s=i}^{i+l-1} \widetilde{h}_{sj} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}, \quad i = 1, \ldots m, \; j = 1, \ldots m.$$

i.e. $H\widetilde{H} = I$. □

Lemma 5.2.2. *The equality $(\widetilde{H}H)^T = \widetilde{H}H$ holds for the matrix $\widetilde{H}$ given by* (5.13).

Proof. For a given $i = 1, \ldots, n$ and $j = 1, \ldots, n$, we should show that $(\widetilde{H}H)_{ij} = (\widetilde{H}H)_{ji}$. The elements of the matrix $\widetilde{H}H$ can be presented as

$$(\widetilde{H}H)_{ij} = \frac{1}{l}\sum_{s=\max\{1, j-l+1\}}^{\min\{j,m\}} \widetilde{h}_{is}, \quad i = 1, \ldots, n \quad \text{and} \quad j = 1, \ldots, n. \tag{5.14}$$

Let us denote by $s = \min\{j, m\} - \max\{1, j-l+1\}$.

Consequently, we should show that the sum of s consecutive elements in the ith row of the matrix $\widetilde{H}H$ where the last element is in the jth column; equals the sum s consecutive elements in the jth row of the matrix $\widetilde{H}H$, where the last element is in the ith column. So the case when $i = j$ is clear, actually, for a given i these elements actually present the sum of the s

consecutive elements that belong to the zero layer as well as his sides. We continue with the opposite case when $i \neq j$.

Let us explore the properties of each row i of the matrix $\widetilde{H}$, $i = 1, \ldots, n$. First we recall that if $i \leq j$ that means that $\widetilde{h}_{ij}$ is either y_1 or belongs in a block B_k, $k = 1, \ldots, p-1$. And, if $i > l$ and $i > j$ that means that $\widetilde{h}_{ij}$ is either y_1 or belongs in a block C_k.

1 case : $i < j,\ r_i = 1$ and $r_j \neq 1$ $(i = 1, \ldots, n-1$ and $j = 1, \ldots, n)$

From (5.14) and definition of the matrix $\widetilde{H}$ we have

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(y_{q_j+1} + x_{q_j+1}) = \frac{z}{l}.$$

That means that the sum of each s consecutive elements equals z, if the elements are in the $(kl+1)st$ row, $k = 0, \ldots, p-1$, and the last element is not in the $(kl+1)st$ column, $k = 0, \ldots, p-1$. Also since $i < j$ the last element is above the diagonal of the square matrix constituted of the first m rows of $\widetilde{H}$.

We need to compare these values with the values of $(\widetilde{H}H)_{ji}$. For this situation we analyze the opposite case i.e. we interchange the conditions for i and j. Suppose,

$$i > j,\ r_i \neq 1 \text{ and } r_j = 1 \quad (i = 1, \ldots, n \text{ and } j = 1, \ldots, n-1).$$

From here we continue with two different possibilities:

a1: $r_i = 0$ then

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(-x_{q_j} + z + x_{q_j}) = \frac{z}{l}.$$

a2: $r_i \neq 0$ then

$$(\widetilde{H}H)_{ij} = \frac{1}{l}\begin{cases} z, & j = 1 \\ z + x_{p-q_j} - x_{p-q_j} = z, & j > 1 \end{cases}$$

Thus the first case is completed i.e. if $i < j,\ r_i = 1$ and $r_j \neq 1$ $(i = 1, \ldots, n-1$ and $j = 1, \ldots, n)$ then $(\widetilde{H}H)_{ij} = (\widetilde{H}H)_{ji}$.

2 case : $i < j,\ r_i = 1$ and $r_j = 1$ $(i = 1, \ldots, m-1$ and $j = 1, \ldots, m)$ then

$$(\widetilde{H}H)_{ij} = y_{q_j+1} + x_{q_j} = z(1-l)$$

Note: Since $m = l(p-1)+1$ and $n = m + l - 1$, if $j > m$ follows that $r_j \neq 1$.
And for the opposite case: i.e. $i > j,\ r_i = 1$ and $r_j = 1$ $(i = 1, \ldots, m$ and $j = 1, \ldots, m-1)$

$$(\widetilde{H}H)_{ij} = \begin{cases} z + x_{p-1} = z(1-l), & j = 1 \\ z + x_{p-q_j-1} - x_{p-q_j} = z(1-l), & j \neq 1 \end{cases},$$

which completes the case.

3 case : $i < j,\ r_i \neq 1$ and $r_j = 1$ $(i = 1, \ldots, m-1$ and $j = 1, \ldots, m)$ then

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(-x_{q_j} + z + x_{q_j}) = \frac{z}{l}.$$

And for the opposite case: i.e. $i > j,\ r_i = 1$ and $r_j \neq 1$ $(i = 1, \ldots, m$ and $j = 1, \ldots, m-1)$

$$(\widetilde{H}H)_{ij} = \frac{1}{l}(z + x_{p-q_j-1} - x_{p-q_j-1}) = \frac{z}{l}.$$

and the case is completed.

4 case : $i < j,\ r_i \neq 1$ and $r_j \neq 1\ \ (i = 1, \ldots, n-1$ and $j = 1, \ldots, n)$ then similar as previous after considering several cases one can compute the following

$$(\widetilde{H}H)_{ij} = \frac{1}{l} \begin{cases} z(l-1), & r_i = r_j,\ i \leq m \\ z, & \text{otherwise} \end{cases},$$

And for the opposite case: i.e. $i > j,\ r_i \neq 1$ and $r_j \neq 1\ \ (i = 1, \ldots, n$ and $j = 1, \ldots, n-1)$

$$(\widetilde{H}H)_{ij} = \frac{1}{l} \begin{cases} z(l-1), & r_i = r_j,\ j \leq m \\ z, & \text{otherwise} \end{cases},$$

and the case is completed. □

Theorem 5.2.1. *The matrix $\widetilde{H}$ given by* (5.13) *is the Moore-Penrose inverse of the matrix H.*

Proof. Since the matrix H is full row rank matrix its Moore-Penrose inverse is its right inverse. From this fact and from the previous two lemmas follows the proof of the theorem. □

Algorithm 5.2.1 Direct method for image deblurring.

Input: The blurred image G of dimensions $r \times m$ defined in the blurring process (5.9).

1: If $m + l - 1 \mod l \neq 0$ then add $l * quotient(m + l - 1, l) + l - m$ boundary pixels, else add $l - 1$ boundary pixels.
2: Compute the matrix $H^{\dagger}$ according to the formula (5.13)
3: Apply formula (5.5)
4: Return $\widetilde{F}$.

5.2.1 Experimental results

In this section we have tested the proposed method on X-ray images and present numerical results. In order to confirm the efficiency, we compared it with three recently announced methods for computing the Moore-Penrose inverse of the matrix H. Therefore, the following methods are compared:

1. The method proposed in Algorithm 5.3.1,
2. Pappas1 method, defined by the MATLAB function `ginv.m` from [80],
3. Pappas2 method, defined by the MATLAB function `qrginv.m` from [29],
4. Courrieu method from [36].

The experiments are done using *Matlab* programming language [159] on an Intel(R) CPU T2130 @ 1.86 GHz 1.87 GHz 32-bit system with 2 GB of RAM memory. Tests are made for several images of dimensions $r \times m$. The index l that takes values between 10 and 100 is the varying parameter for a given image.

Also we compare the efficiency of four different methods for the image restoration: the Moore-Penrose inverse, Wiener filter, Constrained least-squares (LS) filter, and Lucy-Richardson algorithm, respectively. For the implementation of the Wiener filter, Constrained least-squares (LS) filter, and Lucy-Richardson algorithm we used incorporated built-in functions from the *Matlab* package [54].

In image restoration the quality enhancement of the restored image over the recorded blurred one is evaluated by the signal-to-noise ratio improvement (*ISNR*). The *ISNR* of the recorded (blurred) image is defined as follows in decibels:

$$ISNR = 10 \log_{10} \left(\frac{\sum_{n1,n2}(G(n_1, n_2) - F(n_1, n_2))^2}{\sum_{n1,n2}(\tilde{F}(n_1, n_2) - F(n_1, n_2))^2} \right). \tag{5.15}$$

The decrease of disagreement with the ideal image in the case of comparison between the distorted and restored image is measured by the improvement in *SNR*. It is only possible to compute all of the above signal-to-noise measures when the ideal image is available, i.e., in an experimental system or in a design phase of the restoration algorithm.

The problem of restoring an X-ray images that has been blurred by a uniform linear motion, usually results of camera panning or fast object motion. The X-ray image making provides a crucial method of diagnostic by using the image analysis. Figure 5.2.3 presents one practical example for restoring blurred X-ray image. The image is taken from the results obtained from Google Image search with the keyword "X-ray image".

Original image from Figure 5.2.3 shows the original X-ray image. The image is divided into $r = 948$ rows and $m = 1450$ columns. To prevent loosing information from the boundaries of the image, we assumed zero boundary conditions, which implies that values of the pixels of the original image F outside the domain of consideration are zero. This choice is natural for X-ray images since the background of these images is black. *Degraded image* presents the degraded X-ray image for $l = 90$. Finally, from Figure 5.2.3 (Moore-Penrose Inverse, Wiener Restored Image, Constrained LS Restored Image and Lucy-Richardson Restored Image), it is clearly seen that the details of the original image are recovered in all cases.

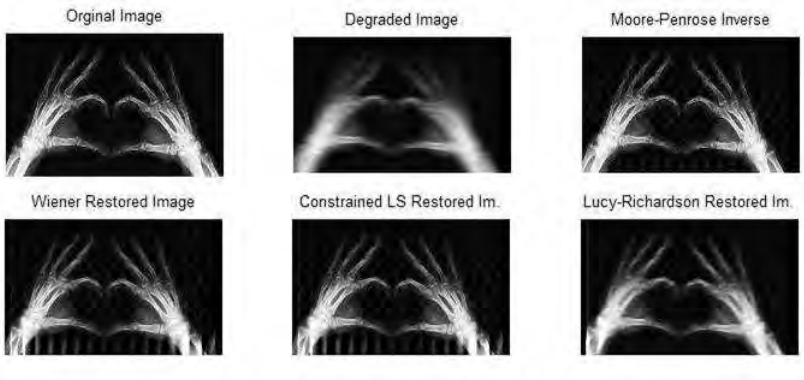

Figure 5.2.3. Removal of blur, caused by a uniform horizontal motion, in a simulated degraded X-ray image, with dimensions $r = 948, n = 1450$ and $l = 90$.

The difference in quality of restored images between the three methods is insignificant, and can hardly be seen by human eye. For this reason, the *ISNR* has been chosen in order to compare the restored images obtained by the Moore-Penrose inverse method, the Wiener filter methods, the Constrained least-squares filter method and the Lucy-Richardson algorithm.

Figure 5.2.4 (left) shows the corresponding *ISNR* value for restored images as a function of l for the Moore-Penrose inverse method and the mentioned classical methods. The figures illustrate that the quality of the restoration is as satisfactory as the classical methods or better from them. This means that the Moore-Penrose inverse method for image restoration have the better quality of the restored image from the other methods.

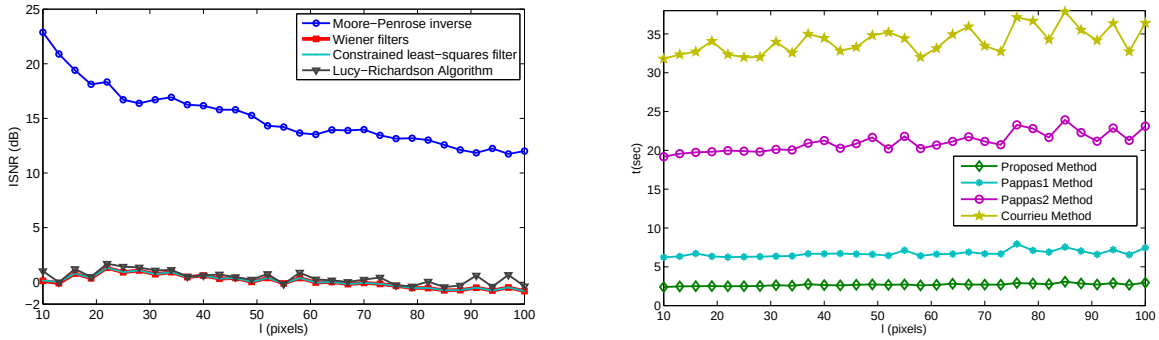

Figure 5.2.4. (left) Improvement in signal-to-noise-ratio vs. length of the blurring process

(right) Computational time vs. length of the blurring process.

The main advantage of the proposed method for computing the Moore-Penrose inverse is the time required to obtain the restored image compared to other methods for computing the Moore-Penrose inverse. Figure 5.2.4 (right) shows the corresponding computational time *t(sec)* value for restored images as a function of $l < 100$ pixels for the proposed method and the mentioned other methods. Realistically speaking, large motions do not occur frequently in radiography.

5.2.2 Restoring uniform blur and noise

The next example refers to case where the image noise is firstly included into the image degradation and later the noise is followed by the uniform linear blur. The corresponding mathematical model which generalizes the horizontal blurring process presented by (5.2) becomes

$$G_N = (F + N)H^T = F_N H^T, \tag{5.16}$$

where G_N is blurred noisy image and N is an additive noise. To obtain approximation of the original image, we apply two steps:

1. Calculate the restored matrix of F_N with (5.5), which gives $\tilde{F}_N = G_N(H^\dagger)^T$;
2. Obtain the restored image $\tilde{F}$ by applying filtering process on the image $\tilde{F}_N$ obtained in Step 1.

Similarly, we can formulate a process when we have the two ways degradation with noise and vertical blur of the original image. When we use original image from Figure 5.2.3, the results presented on the Figure 5.2.5 and Figure 5.2.6 are refereed to the noise added to the image from type "salt and pepper" (white and black) noise with noise density of 0.04. For filtering we use a 2-D median filtering.

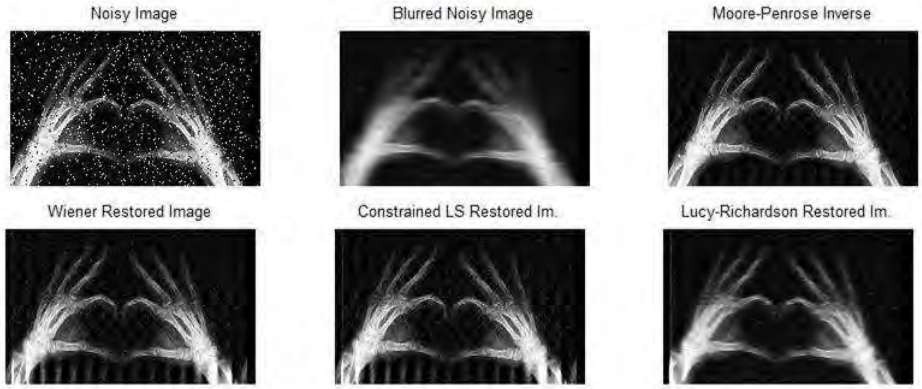

Figure 5.2.5. Removal of blur, caused by noise and uniform horizontal motion, in a simulated degraded X-ray image, with dimension $r = 948, n = 1450$ and $l = 90$.

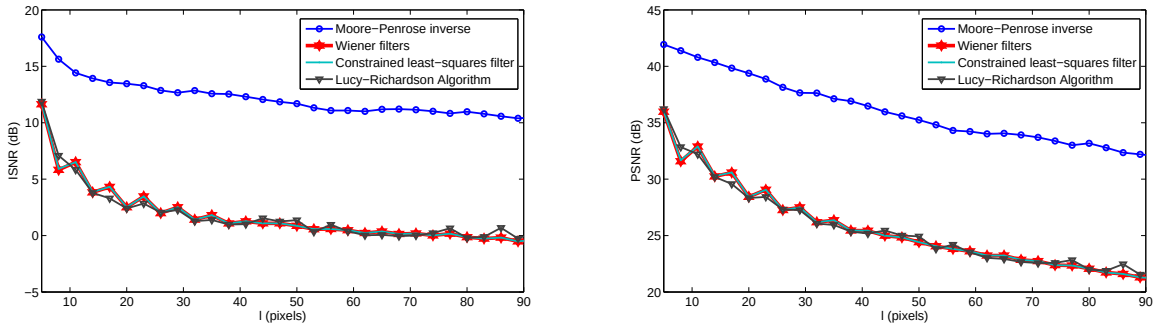

Figure 5.2.6. (left) Improvement in signal-to-noise-ratio vs. length of the blurring process (right) Peak signal-to-noise ratio vs. length of the blurring process.

From the results for image restoration of blurred and noisy images, we can conclude that in this case, as well, the proposed method is better compared to other methods.

5.3 Partitioning method for removing non–uniform blur in images

Courrieu in the paper [36] compared the introduced method *geninv* for computing the Moore-Penrose inverse of a given matrix, with four usual algorithms (the Greville's partitioning method, the SVD method, full rank QR and an iterative method of optimized order [9]). The best results are achieved by the *geninv* method, while the worst results are generated by the partitioning method. In the present section we propose an adaptation of the partitioning method for Toeplitz matrices of the form (5.11).

Our goal in this section is to present appropriate modification of the well-known partitioning method, in order to adjust it, for restoring images, blurred by a non–uniform linear motion. In the sequel the basic explanation of the partitioning method is presented. The notation $A_i,\ i \in \{1, \ldots, n\}$ stands for the submatrix of A which consists of the first i columns of a given arbitrary matrix $A \in \mathbb{R}^{m\times n}$. Particularly, the ith row of A is denoted by a_i and a^i means the ith column of A. By ${}_iA_k$, $i \in \{1, \ldots, n-1\}$, $k \in \{1, \ldots, n-i\}$ we denote the submatrix of A which consists of the columns $a_{i+1}, \ldots, a_{i+k}$. The $m \times m$ identity matrix is denoted by I_m and

O_m is the quadratic $m \times m$ zero matrix. The notation $\mathbf{0}$ stands for the zero column vector of an appropriate dimension. For the purpose of introducing our algorithm, we restate the block recursive algorithm for computing Moore-Penrose inverse of matrix $B = [A|C]$, which denotes a matrix A augmented by an appropriate matrix C, in terms of the matrix $A^\dagger$.

Lemma 5.3.1. [135] *Let $B = [A|\ C]$ be an $m \times (r + p)$ complex matrix whose last p columns are denoted by C. Let*

$$R = I - AA^\dagger, \qquad Q = (RC)^T RC, \qquad F = I - Q^\dagger Q, \qquad Z = A^\dagger CF. \tag{5.17}$$

Then

$$B^\dagger = \begin{bmatrix} A^\dagger(I - CV) \\ V \end{bmatrix} \tag{5.18}$$

where

$$V = Q^\dagger C^T R + (I + Z^T Z)^{-1} Z^T A^\dagger (I - CQ^\dagger C^T R). \tag{5.19}$$

We also restate the Grevile's single-column finite recursive algorithm from [58].

Lemma 5.3.2. [58] *Let A be an $m \times n$ complex matrix and a be an $m \times 1$ constant vector. By $[A|a]$ we denote the matrix A augmented by an appropriate vector a. Then*

$$[A|a]^\dagger = \begin{bmatrix} A^\dagger - db^* \\ b^* \end{bmatrix}$$

where

$$\begin{aligned} d &= A^\dagger a, \\ b &= \begin{cases} \frac{1}{c^* c} c, & c \neq \mathbf{0} \\ \frac{1}{1+d^* d}(A^\dagger)^* d, & c = \mathbf{0}, \end{cases} \end{aligned}$$

and

$$c = (I - AA^\dagger)a.$$

In order to invert the left quadratic block H_m of the matrix H given with (5.11), first we consider the matrix equation $H_m H_m^{-1} = I$. Since the matrix H_m is upper triangular Toeplitz matrix it is well-known that its inverse is also upper triangular Toeplitz matrix. Therefore, the whole matrix H_m^{-1} is determined by its last column which will be denoted by x. To generate the vector x we consider the following equation

$$H_m \cdot x = e_m, \tag{5.20}$$

where e_m denotes the m-th column of the identity matrix I_m.

The following particular case of the Lemma 5.3.1 is useful in calculating the Moore-Penrose inverse of the degradation matrix H.

Lemma 5.3.3. *Assume that the matrix $H \in \mathbb{R}^{m \times n}$, $n = m + l - 1$ causes the blurring process in (5.9). The Moore-Penrose inverse of its first $p + k$ columns, partitioned in the block form $H_{p+k} = [H_p|\ {}_pH_k]$, $p \in \{1, \ldots, n-1\}$, $k \in \{1, \ldots, n-p\}$, is defined by*

$$H_{p+k}^\dagger = \begin{bmatrix} H_p^\dagger \left(I - {}_pH_k \cdot B^T\right) \\ B^T \end{bmatrix} = \begin{bmatrix} H_p^\dagger - DB^T \\ B^T \end{bmatrix} \tag{5.21}$$

where

$$D = H_p^\dagger \cdot\ {}_pH_k \quad \text{and} \quad B = (H_p^\dagger)^T D(I + D^T D)^{-1}. \tag{5.22}$$

Proof. Follows from Lemma 5.3.1, taking into account that the degradation matrix is of full row rank and the fact that equalities in (5.17) reduce to

$$R = Q = O_m,\ F = I_m,\ V = B^T,\ Z = D = H_p^\dagger \cdot\ {}_pH_k$$

observing this particular case. □

Also, the Grevile's Partitioning method reduces to the following computational procedure.

Lemma 5.3.4. *The Moore-Penrose inverse of the matrix H_i is equal to*

$$H_i^\dagger = \begin{bmatrix} H_{i-1}^\dagger - d_i b_i^T \\ b_i^T \end{bmatrix} \tag{5.23}$$

where

$$d_i = H_{i-1}^\dagger \cdot h_i \quad \textit{and} \quad b_i = \left(1 + d_i^T d_i\right)^{-1} (H_{i-1}^\dagger)^T d_i. \tag{5.24}$$

Since we know the inverse H_m^{-1} which is completely determined by vector x from (5.20), any pragmatical implementation of the new method uses only partitions of the form $H_{p+k} = [H_p|\ {}_pH_k]$, $p \geq m$, $k \in \{1, \ldots, n-m\}$.

According to Lemma 5.3.3 we propose the following algorithm for computing the Moore-Penrose inverse of the matrix H.

Algorithm 5.3.1 Computing the Moore-Penrose inverse of the matrix H.

Input: The matrix H of dimensions $m \times (m+l-1)$ given by (5.11).

1: Separate the block H_m of the matrix H.
2: Generate $H_m^\dagger = H_m^{-1}$ using the vector x from (5.20).
3: Take $p = m$ and choose k such that $1 \leq k \leq l-1$ as well as $\frac{l-1}{k} \in \mathbb{N}$.
4: Compute $H_{p+k}^\dagger = [H_p|\ {}_pH_k]^\dagger$ according to Lemma 5.3.3.
5: Set $p = p + k$.
6: If $p \neq n$ then go to Step 4; otherwise, go to the next step.
7: Return $H_n^\dagger$

It is not difficult to verify that the choice $k = 1$ in all recursive steps of Algorithm 5.3.1 produces the particular case of Grevile's recursive method corresponding to Lemma 5.3.4. Also, in the case $p = m, k = l-1$ Algorithm 5.3.1 reduces to the following algorithm.

Algorithm 5.3.2 Computing the Moore-Penrose inverse of the matrix H.

Input: The matrix H of dimensions $m \times (m+l-1)$ defined in the blurring process (5.9).

1: Separate matrix H into two blocks H_m and ${}_mH_{l-1}$, that is $H = [H_m|\ {}_mH_{l-1}]$.
2: Generate $H_m^\dagger = H_m^{-1}$ using the vector x from (5.20).
3: Compute $H^\dagger = [H_m|\ {}_mH_{l-1}]^\dagger$ according to Lemma 5.3.3.

Choosing the most efficient case with respect to the computational time, we derive an efficient method for computing the Moore-Penrose inverse of the degradation matrix H, and respectively an efficient method for image restoring processes based on the equation (5.9). For this purpose, let us denote by $\mathrm{I}(n)$ the complexity of the algorithm for inverting a given $n \times n$ matrix (as in [35]). Also by $\mathcal{A}(n)$ we denote the complexity of the addition/subtraction of two

$n \times n$ matrices and by $\mathcal{M}(m,n,k)$ the complexity of multiplying $m \times n$ matrix with $n \times k$ matrix. The simpler notation $\mathcal{M}(n)$ (taken from [35]) is used instead of $\mathcal{M}(n,n,n)$.

In the remaining of this section we consider the computational complexity of the two opposite cases: $p = m, k = 1$ (used in Algorithm 5.3.1) and $p = m, k = l-1$ (used in Algorithm 5.3.2). The complexity of Algorithm 5.3.2 is of the order

$$E_{3,2} = 3\,\mathcal{M}(m,m,l-1) + 3\,\mathcal{M}(m,l-1,l-1) + \mathrm{I}(l-1) + A(l-1). \tag{5.25}$$

Scanning Algorithm 5.3.1 in a similar way, it is not difficult to verify that in ith recursive step it requires complexity of the order

$$\mathcal{M}(m+i-1,m,1) + \mathcal{M}(1,m+i-1,1) + \mathcal{M}(m,m+i-1,1),$$

for each $i = 1,\ldots,l-1$. Therefore, the complexity of complete algorithm is

$$E_{3,1} = \sum_{i=1}^{l-1} \left(\mathcal{M}(m+i-1,m,1) + \mathcal{M}(1,m+i-1,1) + \mathcal{M}(m,m+i-1,1)\right). \tag{5.26}$$

It is well-known that matrix inversion is equivalent to matrix multiplication. More precisely, the ordinary inverse of any real nonsingular $n \times n$ matrix can be computed in time $\mathrm{I}(n) = \mathcal{O}(\mathcal{M}(n))$ [35]. The notation $\mathcal{O}(f(n))$ is described, for example, in [35]. We conclude that the computational complexity of computing $(I + D^T D)^{-1}$ is substantially smaller than the complexity of calculating $H^\dagger$. Also, the measure of complexity $E_{3,1}$ does not include the computational effort of the matrix inversion. The upper bounds for the complexity of Algorithm 5.3.2 and Algorithm 5.3.1 are given by

$$E_{3,2} \leq \mathcal{O}(\mathcal{M}(m,m,l-1)), \quad E_{3,1} \leq l \cdot \mathcal{O}(\mathcal{M}(m,m+l,1)).$$

These upper bounds are incomparable in the general case. Therefore, the choice of the best version of the partitioning algorithm is determined only on the basis of performed numerical experiments. CPU times obtained in these experiments depend upon two parameters: computational complexity and implementation details incorporated into the programming language MATLAB.

5.3.1 Experimental results

In order to confirm the efficiency of our algorithm, we compared it with two recently announced methods for computing the Moore-Penrose inverse in [27] and [36]. These methods are called *Ginv* method, *Qrginv* method and Courrieu method respectively. The following algorithms for computing the Moore-Penrose inverse of the matrix H are compared:

1. Block partitioning method presented in Algorithm 5.3.2,
2. Ginv method, defined by the MATLAB function `ginv.m` from [80],
3. Qrginv method, defined by the MATLAB function `qrginv.m` from [29],
4. Courrieu method from [36].

The experiments are done using MATLAB programming package [159] on an Intel(R) Core(TM) i5 CPU M430 @ 2.27 GHz 64/32-bit system with 4 GB RAM memory.

In Figure 5.3.7 we present the results which refer to the computational time t(sec) needed to compute the Moore-Penrose inverse, as a function of the length of the blurring process

$l \leq 90$ (pixels). Tests are made for randomly generated matrix of dimensions 1000×1200, which corresponds to a randomly generated image of the same dimensions, which is blurred by Gaussian function. The confirmation that the proposed method for computing the Moore-Penrose inverse (restoring blurred image) is faster than the other methods is illustrated on the next figure.

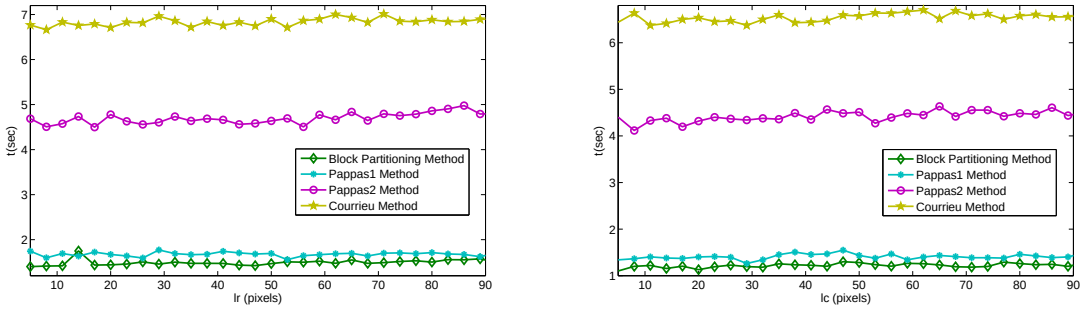

Figure 5.3.7. (left) CPU time versus l_r of the random matrix deblurring process ($l_c = 30$)

(right) CPU time versus l_c of random matrix deblurring process ($l_r = 25$).

It is easy to observe that the block partitioning method overcomes Ginv and QrGinv methods. On the other hand, the authors in [80] concluded that the method Ginv possesses is faster with respect to Courrieu method. Therefore, after the modifications described before, partitioning method becomes the fastest with respect to all methods described in [80] and [36].

5.3.2 Restoring non–uniform blur and noise

In this section we devote attention to the serial degraded images. First, the noise is imposed to the image and after that the noisy image is blurred by the non-uniform Gaussian function. In this case the mathematical model of the non-uniform blurring process presented by (5.10) becomes

$$G_N = H_c(F + N)H_r^T = H_c F_N H_r^T, \tag{5.27}$$

where G_N is blurred noisy image and N is an additive noise. Two steps are used to restore the original image:

1. Calculate the restored matrix $\tilde{F}_N = H_c^\dagger G_N (H_r^\dagger)^T$ of F_N;
2. Restore image $\tilde{F}$ by applying the filtering process on the image $\tilde{F}_N$.

In Figure 5.3.8 we present the X-ray image, introduced in the previous section, in the case when it is blurred by a Gaussian model with $l_c = 25$ and $l_r = 45$. The four methods for the image restoration: the Moore-Penrose inverse, Wiener filter, Constrained least-squares (LS) filter, and Lucy-Richardson algorithm, are as well used, respectively.

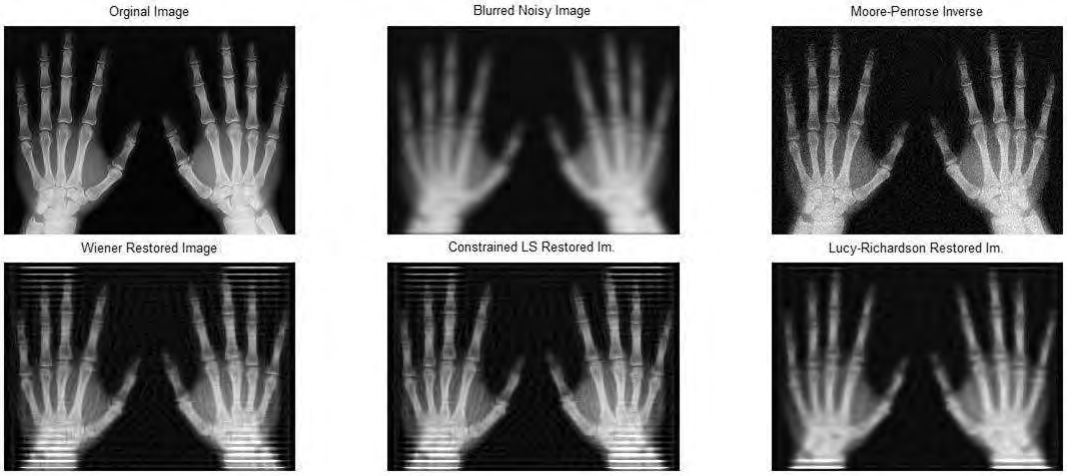

Figure 5.3.8. Removal of blur, caused by Gaussian model with $l_c = 25$ and $l_r = 45$, on a X-ray image.

On the figure we illustrate the original, the blurred noisy and the restored images obtained by different methods. *Original image* from Figure 5.3.8 shows the original X-ray image. The image is divided into $r = 750$ rows and $m = 1050$ columns. To prevent loosing information from the boundaries of the image, we assumed zero boundary conditions, which implies that values of the pixels of the original image F outside the domain of consideration are zero. This choice is natural for X-ray images since the background of these images is black. The pixels of the original image are degraded by the Gaussian white noise of mean 0 and variance $0,01$ and later blurred by non-uniform Gaussian function according to model (5.27). For filtering we use a rotationally symmetric Gaussian low pass filter of size 3 with standard deviation 45.

The results for the parameters $ISNR$ [16], presented on Figure 5.3.9, show that the restoration of the serial degraded images with the Moore-Penrose inverse is more reliable and accurate than restoration with other mentioned methods.

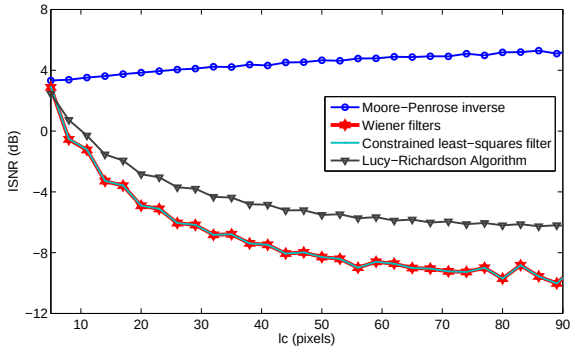

Figure 5.3.9. ISNR versus l_c for the removal of blur given by model (5.27) ($l_r = 35$).

In the sequel we compare the CPU time for the restoration process of our method with the CPU time of three other, previously mentioned, methods. The computational time needed to restore the degraded X-ray image by means of these methods which use the Moore-Penrose inverse approach, is shown in the Figure 5.3.10. For a given image the varying parameter is the parameter l_r (l_c) that takes values between 5 and 90.

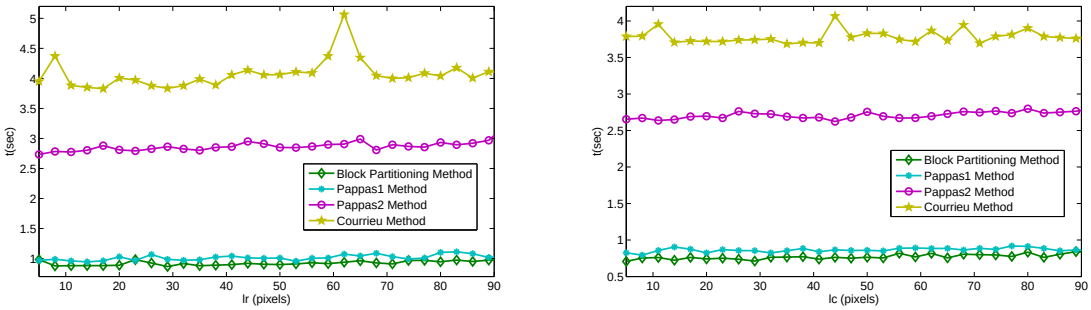

Figure 5.3.10.

(left) CPU time versus l_r in the removing of blur caused by Gaussian function and noise ($l_c = 25$) (right) CPU time versus l_c in the removing of blur caused by Gaussian function and noise ($l_r = 35$).

As it is expected the proposed method, again, shows better performances with respect to the other tested methods.

Obviously, the proposed method is not only restricted to restoration of blurred X-ray images, but also can be used for other practical implementations such as deblurring images arising in Automatic Number Plate Recognition (ANPR) systems. We assume that the blur that appears in images from ANPR systems is caused by "salt and paper" noise and non-uniform Gaussian model, given by (5.27). To prevent loosing information from the boundaries of the image, we assumed periodic boundary conditions for ANPR images. Figure 5.3.11 presents the results obtained by restoring an image from the ANPR system with dimensions 1023×1250. ANPR image is taken from implemented system of automatic recognition of license plates of the Customs Administration in Serbia.

Figure 5.3.11. Removal of blur, caused by "salt and paper" noise and Gaussian function ($l_c = 25$, $l_r = 40$).

Figure 5.3.12 decidedly approves that the time required to obtain a restored ANPR image using the proposed method is again the smallest with respect to other considered methods.

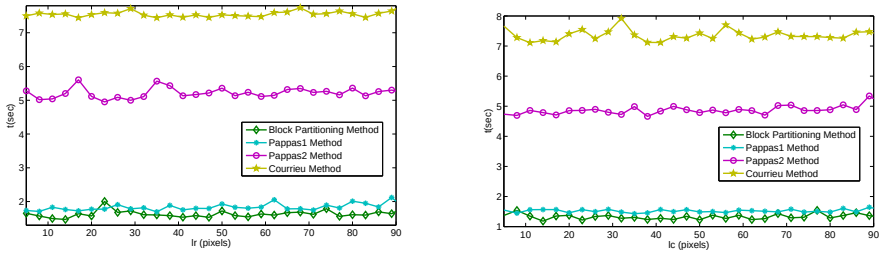

Figure 5.3.12.

(left) time versus l_r in removal of blur given by model (5.27) and "salt and paper" noise ($l_c = 35$). (right) time versus l_c in removal of blur given by model (5.27) and "salt and paper" noise ($l_r = 40$).

Chapter 6

Conclusion

In the Ph.D. dissertation, we analyzed and presented new results related to two main disciplines: *unconstrained optimization* and *generalized inverses*. Usually the textbooks we encounter in the literature are either devoted to optimization theory, or to generalized inverses theory. In the Ph.D. dissertation we stressed the interaction between these two disciplines. Namely, we observed the theory and methods of unconstrained optimization, as a useful tool for finding generalized inverses of matrices, and in some instances, for finding generalized inverses of linear bounded operators on Hilbert spaces. In the beginning, we presented analysis and new results related to unconstrained optimization and generalized inverses, separately. After, we used the presented results for constructing new efficient algorithms for solving specific optimization problems, in order to determine different types of generalized inverses.

In the following we present a short summary of all obtained results and propose possible directions for further investigations.

The second chapter, which is devoted to optimization theory, contains new results related to the minimization of a given objective function on finite dimensional real space, as well as finding a least-squares solution of an operator on Hilbert space. After presenting the summary of nonlinear unconstrained optimization in Section 2.1., the main results are presented in Section 2.2 and Section 2.3.

As a motivation for the results presented in Section 2.1. we used the idea of nonmonotone line search strategy proposed by Grippo et al. [59], If we combine the nonmonotone line search strategy with the Barzilai and Borwein method in [111], we get a global convergence of the algorithm. The resulting algorithm, known as global Barzilai and Borwein method (or shortly the GBB method), allows significant reduction in the number of line searches and also in the number of gradient evaluations, so that it has been used as a benchmark in the construction of unconstrained optimization algorithms.

Based on these ideas, we proposed a new two-point stepsize gradient descent method, which is given by Algorithm 2.2.3. The underlying ideas are the usage of a two-point approximation to the secant equation required in quasi-Newton methods as well as the approximation of the inverse Hessian matrix by a scalar matrix and its updating by means of an appropriately selected scalar. Our idea was to ease the choice of the initial trial steplength and maximize its values as much as possible. After analyzing few choices of possible initial stepsizes, as it is presented in (2.32) and (2.33), we choose those with largest values observing different cases. The technique of the nonmonotone line search proposed in [59] is accompanied by our algorithm to make it comparable with the BB method. The resulting algorithm is named the GSC algorithm.

Due to the characteristics of the GSC algorithm, such as the simplicity, efficiency and very low memory requirements (similar to the GBB method), it tends to be very attractive and applicable to large-scale test problems. The numerical results presented in the Ph.D. dissertation favor the GSC method in comparison to the GBB method.

The results in Section 2.3. are related to the usage of quasi-Newton methods for finding an optimal solution of a given function. The presented results represent a natural continuation of the idea that can be described as an application of the nonlinear optimization in computing least-squares solutions of the operator equation $Ax = b$. This strategy adapted for functionals on Hilbert spaces can be also successfully used for computing $\{1, 3\}$-inverses, including the Moore-Penrose inverse, of a given complex matrix.

In Section 2.3. we presented the previously mentioned adaptation, i.e., we did an extension of the secant equation to Hilbert spaces. Furthermore, using the idea of the scalar correction method (SC method), we established a new gradient method for computing least-squares solutions of the equation $Ax = b$ on Hilbert spaces, which in addition to the good properties of the secant equation, also preserves monotonicity. Despite the independency of the results related to Hilbert spaces, their greatest advantage is the illustration how generalization of some practical problems can then be used to solve other practical problems.

Further investigation related to the results in Chapter 2 will be, first of all, targeted to the improvements of the GSC method. The possibility of existing other trial stepsizes which can contribute to enhance the performances of the algorithm is not excluded. Also, the constrained optimization, although it is not captured in the Ph.D. dissertation will be subject of our investigation, and we will research the interpretation of the results in the field of constrained optimization.

The main results from Chapter 3 are presented in Section 3.2, 3.3 and 3.4. Here, we paid a special attention to the minimal properties of the Drazin inverse and $A^{(2)}_{T,S}$-inverses. The new presented results related to the minimal properties of the Drazin inverse and $A^{(2)}_{T,S}$-inverse, are very useful in order to utilize the optimization methods for the purpose of determining the Drazin inverse solution and $A^{(2)}_{T,S}$-inverse solution of a given matrix equation or its special case, a system of linear equations.

In the papers [20, 140, 142], the authors present some minimal properties of the Drazin-inverse solution. It can be argued that, in some way, these properties correspond to the properties of the Moore-Penrose inverse solution. Namely, in [20] it is shown that if $b \in \mathcal{R}(A^p)$, where $p = \mathrm{ind}(A)$, then the Drazin-inverse solution is the unique solution of the system $Ax = b$ which belongs to $\mathcal{R}(A^p)$. Also, Wei et al. in [140, 142] show that the Drazin-inverse solution of the system $Ax = b$ is a solution of minimum P-norm, where P is the Jordan matrix obtained with the Jordan decomposition of the matrix A.

In Section 3.2, we explore further minimal properties of the Drazin-inverse solution of the system $Ax = b$, where $b \in \mathcal{R}(A^p)$. The presented results, actually, reveal the connection between the Drazin-inverse solution and $\{1, 3\}$-inverses of the matrix A^{p+1}.

As it is well known, the Drazin inverse always exists for a square matrix, although it provides a solution of the system $Ax = b$ only in the case when $b \in \mathcal{R}(A^p)$. In addition to the previous results, in Section 3.3, we develop a methodology for finding the vector of the form $A^D b$, in case of arbitrary square matrix A and an arbitrary vector b of appropriate dimensions. The

goal is achieved by presenting relationships between the vector $A^D b$ and P-norm least-squares solutions of the system $A^p x = b$.

The obtained results related to the Drazin-inverse solution of a given system of linear equations, inspired us to research whether they can be used in order to calculate the Drazin inverse of a given matrix, i.e., to find the Drazin-inverse solution of the matrix equation $AXB = D$, in general. With appropriate modification we found the matrix of the form $A^D G B^D$, which is not always a solution of the matrix equation, but however it can be always used in order to calculate the Drazin inverse of arbitrary matrix. Two formulas for the Drazin inverse calculation are presented in Theorem 3.2.9.

Finally, we round off these ideas with their generalization to $A^{(2)}_{T,S}$ inverses, in Section 3.3. However, what is presented in the Ph.D. dissertation is only finding $A^{(2)}_{T,S}$-inverse solution of the system $Ax = b$ in the special case when $b \in \mathcal{R}(AR)$, where R is appropriately chosen matrix.

Additionally, related to $A^{(2)}_{T,S}$ inverses, in Section 3.4, we interpreted full rank representation of $\{2,4\}$ and $\{2,3\}$-inverses through the full rank representation of outer inverses. In this way, the methods developed for computing outer inverses with prescribed range and null space can be applied in computation of $\{2,4\}$ and $\{2,3\}$-inverses with prescribed range and null space.

Finding $A^{(2)}_{T,S}$-inverse solutions of the system $Ax = b$ in general case and $A^{(2)}_{T,S}$-solution of the matrix equation $AXB = G$ given with $A^{(2)}_{T_1,S_1} G B^{(2)}_{T_2,S_2}$ will be subject of our further investigation. The practical implementation, i.e., investigating the fields where the minimal properties of the Drazin inverse can be used, also will be subject of our future research.

In Chapter 4, we presented the obtained algorithms for generalized inverses computation, we showed their convergence properties, and we presented numerical results which are noting else, but an illustrative confirmation of the efficiency of the algorithms.

In the beginning, in Section 4.2 we showed how, the generalization of the idea of scalar correction method in order to find least-squares solutions on Hilbert space, which is presented in Section 2.3, can be used for computing the Moore-Penrose inverse of a given matrix.

The set of least-squares solutions of the operator equation $Ax = b$ is identified with the set of limits of the iterative process (2.48). Any limit value of the iterative process given by (2.48) is completely described by the initial approximation, and it is a least-squares solution. Conversely, for a given least-squares solution we found the set of all initial approximations for the iterative process which lead to that least-squares solution. As should be expected, respective results are presented regarding the computation of $\{1,3\}$-inverses as well as the Moore-Penrose inverse of a given complex matrix $A \in \mathbb{C}^{m\times n}$.

The presented numerical results confirm the expectation relative to the bad convergence properties of the steepest descent method for ill-conditioned problems. And thus make favorable two point stepsize methods with respect to the steepest descent method. Additionally, regarding the numerical results we conclude that the scalar correction method is competitive with the favorable Barzilai Borwein method, not only in the number of iterations but also in the accuracy. Also, the complexity of the SC algorithm is very similar to the complexity of the BB method, which is known as a method easy for the implementation.

The revealed connection between the Drazin-inverse solution and $\{1,3\}$-inverses of the matrix A^{p+1}, which are presented in Chapter 2, impose all methods for minimizing the norms

$\|A^{2p}x - b\|_P$, i.e., $\|A^{2p_1}XB^{2p_2} - G\|^2_{PQ}$ to be a useful tool for computing the Drazin-inverse solution of the system $Ax = b$, i.e., $AXB = G$. In Section 4.3 we presented algorithms for iterative computing of the Drazin-inverse solution and the Drazin inverse of a matrix. As an illustration, by using the two-point stepsize gradient method, we showed how gradient iterative schemes can be used for computing the Drazin-inverse solution. With the presented results, the idea of using the optimization theory in order to efficiently calculate the Moore-Perose inverse solution, is naturally continued for the Drazin-inverses solution.

Section 4.4 is completely devoted to the iterative methods for computing $A^{(2)}_{T,S}$-inverses. The first such algorithm is a gradient algorithm for computing $A^{(2)}_{T,S}$-inverses of the system $Ax = b$, where $b \in \mathcal{R}(AR)$, R is appropriately chosen matrix. It is very similar to the previously mentioned algorithm for finding the Drazin-inverse solution, and is based on the results obtained in Section 3.4.

Next, by modifying and applying the SMS algorithm from [25, 126], we presented two algorithms for computing generalized inverses. The first one is intended for computing $\{2, 4\}$-inverses and the second one for computing $\{2, 3\}$-inverses. These algorithms actually show the utility of the successive matrix squaring algorithm (SMS algorithm) for computing $A^{(2)}_{T,S}$-inverses. The first application is straight whereas for the second one, we did some modifications and showed the respective convergence results.

Additionally, we indicate some necessary conditions for obtaining $\{1, 2, 3\}$, $\{1, 2, 4\}$-inverses and finally the Moore-Penrose inverse. In this way we filled the gap of finding S-inverses, where $S \subset \{1, 2, 3, 4\}$ using SMS method for computing $\{2\}$-inverses with prescribed range and null space from [126].

However this is not the only part where we have used the idea of the SMS method for computing $A^{(2)}_{T,S}$-inverses. In the next algorithm which is presented, we used the idea introduced in [126], accompanied by the concept of displacement operator and displacement rank of iteration matrices. Namely, we present the modification of the SMS algorithm for computing the outer inverse with prescribed range and null space of an arbitrary Toeplitz matrix. Our method is actually a generalization of the ideas from [14, 19] and represent a universal algorithm for computing generalized inverses containing these results as partial cases. Furthermore, by using the circulant and anti-circulant matrices for obtaining the *odr* of the sequence of approximations Z_k, we propose a different approach for computing the Moore-Penrose inverse of a square Toeplitz matrix, with respect to the method from [15, 146]. Also, by giving the unified approach we resolve the previously mentioned diversities in strategies for choosing the starting approximations as well as the diversities in defining the iterative rules and in the usage of various displacement operators.

The presented quadratic convergence and numerical results show that our algorithm is as good as the SMS algorithm for computing outer generalized inverses, when it comes to the number of iterations. Furthermore, the achieved low values of Mdrk and Sdrk evidently decrease the computational cost per step having in mind that the strategy of FFT's and convolutions is used in matrix-vector multiplications. Additionally, three different heuristics are used for the choice of the truncation value ε_k. For given test matrices the numerical results presented in the paper favor dynamic strategy in comparison to the static strategy.

In our future investigations we will try to extend the algorithm for calculation $A^{(2)}_{T,S}$-inverse solutions to more general cases, where system $AXB = G$ is considered. On this way we will

circle the theory of minimal properties of the generalized inverses representable via $A^{(2)}_{T,S}$ inverses. Also we will try to improve the displacement SMS method for calculating $A^{(2)}_{T,S}$ inverses of a given Toeplitz matrix, by using some iterative scheme for which Algorithm 4.4.5 will search for the orthogonal displacement generators of only one matrix, instead of two matricess. This will surely reduce the CPU time of the algorithm. The idea of orthogonal displacement operator can be used as well for determination of $A^{(2)}_{T,S}$ inverses of other structured matrices.

In Chapter 5, motivated by the problem of restoring blurred images via well developed mathematical methods, based on the Moore-Penrose inverse computation, we introduced a computational method to restore images that has been blurred by uniform and non–uniform motion.

The first method which is presented is based on the provided formula for calculating the Moore-Penrose inverse of the blurring matrix. The other presented method is based on appropriate adaptations of well-known computational methods introduced in [135] and [58]. Using the specific structure of the matrix H as well as the fact that we know $H^{\dagger}_{m}$ explicitly, we use the partitioning methods in order to obtained efficient method and make advantage with regard to existing methods.

Presented numerical results firstly show the superiority of the Moore-Penrose inverse techniques over the other methods such as Wiener filter, Constrained least-squares filter and Lucy-Richardson algorithm. Later, we compare our method with respect to the method for fast computing the Moore-Penrose matrix inverse introduced in [79] and used in [28, 27] as well as with the Courrieu method [36].

The main advantage of the proposed method is substantially decreased time required to obtain the restored image compared to other methods based on the usage of the Moore-Penrose inverse.

Introduced method can be used to restore a noisy X-ray image that has been blurred by the uniform or non–uniform motion. Our method also, except in radiography, can be used in different practical realizations, like restoration of images from ANPR systems.

We believe that the results presented in the Ph.D. dissertation besides their contribution in the filed of unconstrained optimization theory and generalized inverses theory will serve as a motivation for future investigations in these fields, and implementation of the results in some practical disciplines.

References

[1] H. Akaike, *On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method*, Ann. Inst. Statist. Math. Tokyo **11** (1959), 1–17.

[2] N. Andrei, *An Unconstrained Optimization Test Functions Collection*, http://camo.ici.ro/neculai/t1.pdf (2005)

[3] I.K. Argyros, *Weak sufficient covergence conditions and applications for Newton methods*, J. Appl. Math. Comput. **16**, 1–17 (2004)

[4] L. Armijo, *Minimization of functions having Lipschitz first partial derivatives*, Pac. J. Math. **16**, 1–3 (1966)

[5] W.E. Arnoldi, *The principle of minimized iterations in the solution of the matrix eigenvalue problem*, Quart. Appl. Math. **9** (1951), 17–29.

[6] S. Axler, *Linear Algebra Done Right*, Second Ed., Springer, 1997.

[7] M.R. Banham, A.K. Katsaggelos, *Digital image restoration*, IEEE Signal Processing Magazine **14** (1997), 24–41.

[8] J. Barzilai, J.M. Borwein, *Two point step size gradient method*, IMA J. Numer. Anal. **8** (1988) 141–148.

[9] A. Ben-Israel, *An iterative method for computing the generalized inverse of an arbitrary matrix*, Math, Comp. **19** (1965), 452–455.

[10] A. Ben-Israel, *A note on an iterative method for generalized inversion of matrices*, Math. Comp. **20** (1966), 439440.

[11] A. Ben-Israel, A. Chanes, *Contributions to the theory of generalized inverses*, SIAM J. **11** (1963), 667669.

[12] A. Ben-Israel, T.N.E. Greville, *Generalized inverses: theory and applications*, Second Ed., Springer, 2003.

[13] P. Bhimasankaram, *On Generalized Inverses of Partitioned Matrices*, Sankhya **33** (1971), 331–314.

[14] D. Bini, B. Meini, *Approximate displacement rank and applications*, in Structured Matrices in Mathematics, Computer Science and Engineering II, V. Olshevsky Editor, Contemporary Mathematics, **281** (2001), 215–232, American Mathematical Society, Rhode Island.

[15] D. Bini, G. Codevico, M.V. Barel, *Solving Toeplitz least squares problems by means of Newton's iteration*, Numer. Algorithms **33** (2003), 93–103.

[16] A. Bovik, *The essential guide to the image processing*, Academic Press, 2009.

[17] A. Bovik, *Handbook of image and video processing*, Academic Press, San Diego, San Francisko, New York, Boston, London, Sydney, Tokyo, 2000.

[18] C. Brezinski, *A classification of quasi-Newton methods*, Numer. Algorithms **33**, 123-135 (2003)

[19] J.F. Cai, M.K. Ng, Y. Wei, *Modified Newton's Algorithm for Computing the Group Inverses of Singular Toeplitz Matrices*, J. Comput. Math. **24** (2006), 647–656.

[20] S.L. Campbell, C.D. Meyer, *Generalized Inverse of Linear Transformation*, Pitman, London (1979).

[21] C.G. Cao, X. Zhang, *The generalized inverse $A^{(2)}_{T,*}$ and its applications*, J. Appl. Math. Comput. **11** (2003), 155 –164.

[22] A. Cauchy, *Metode generale pour la resolution des systems d'equations simultanees*, Comp. Rend. Sci. **25**, 46–89 (1847)

[23] R. Chan, M.K. Ng, *Conjugate gradient methods for Toeplitz systems*, SIAM Rev., **38** (1996), 427-482.

[24] G.K. Chantas, N.P. Galatsanos, N.A. Woods, *Super-resolution based on fast registration and maximum a posteriori reconstruction*, IEEE Trans. Image Process. **16** (2007), 1821–1830.

[25] L. Chen, E.V. Krishnamurthy, I. Macleod, *Generalized matrix inversion and rank compuation by successive matrix powering*, Parallel Comput. **20** (1994), 297–311.

[26] Y. Chen, *The generalized Bott–Duffin inverse and its application*, Linear Algebra Appl. **134** (1990), 71–91.

[27] S. Chountasis, V.N. Katsikis, D. Pappas, *Applications of the Moore-Penrose inverse in digital image restoration*, Math. Probl. Eng., Volume 2009, Article ID 170724, 12 pages doi:10.1155/2009/170724.

[28] S. Chountasis, V.N. Katsikis, D. Pappas, *Digital Image Reconstruction in the Spectral Domain Utilizing the Moore-Penrose Inverse*, Math. Probl. Eng., Volume 2010, Article ID 750352, 14 pages doi:10.1155/2010/750352.

[29] S. Chountasis, V.N. Katsikis, D. Pappas, *Image restoration via fast computing of the Moore-Penrose inverse matrix*, Systems, Signals and Image Processing, 2009, IWSSIP 2009.

[30] J.J. Climent, M. Neumann, A. Sidi, *A semi-iterative method for real spectrum singular linear systems with an arbitrary index*, J. Comput. Appl. Math. **87** (1997), 21–38.

[31] G. Codevico, V.Y. Pan, M.V. Barel, X. Wang, A. Zheng, *The least squares compression policy for Newton-like iteration of structured matrices* (P. Mitic, J. Carne, Eds), IMS2004 Proceedings, 2004, 1–22.

[32] G. Codevico, V.Y. Pan, M.V. Barel, *Newton-like iteration based on a cubic polynomial for structured matrices*, Numer. Algorithms, **36(4)** (2004), 365–380.

[33] A.I. Cohen, *Stepsize analysis for descent methods*, J. Optim. Theory Appl., **33** (1981), 187–205.

[34] L. Collatz, *Functional analysis and numerical mathematics*, Academic Press, New York, 1966.

[35] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, *Introduction to Algorithms, Second Edition*, The MIT Press, Cambridge, Massachusetts London, McGraw-Hill Book Company, Boston, New York, San Francisco, St. Louis, Montreal, Toronto, 2001.

[36] P. Courrieu, *Fast Computation of Moore-Penrose Inverse Matrices*, Neural Information Processing - Letters and Reviews **8** (2005), 25–29.

[37] Y.H. Dai, L.Z. Liao, *R-linear convergence of the Barzilai and Borwein gradient method*, IMA J. Numer. Anal. **22**, 1–10 (2002)

[38] Y.H. Dai, H. Zhang, *Adaptive two-point stepsize gradient algorithm*, Numer. Algorithms **27** (2001), 377–385.

[39] Y.H. Dai, *On the Nonmonotone Line Search*, J. Optim. Theory Appl. **112** (2002), 315–330.

[40] E.D. Dolan, J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Program. **91** (2002), 201–213.

[41] H. Diao, Y. Wei, S. Qiao, *Displacement rank of the Drazin inverse*, J. Comput. Appl. Math. **167** (2004), 147–161.

[42] U. Diwekar, *Introduction to applied optimization*, Springer- Verlag New York, 2008.

[43] D.S. Djordjevic, P.S. Stanimirović, *Iterative methods for computing generalized inverses related with optimization methods*, J. Aust. Math. Soc. 78 (2005), 257–272.

[44] D.S. Djordjevic, V. Rakocevic, *Lectures on generalized inverses*, Faculty of Sciences and Mathematics, University of Niš, 2008.

[45] D.S. Djordjević, P.S. Stanimirović, *General representations of pseudoinverses*, Mat. Vesnik **51** (1999), 69-76.

[46] S.C. Eisenstat, H.C. Elman, M.H. Schultz, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal. **20** (1983), 345–357.

[47] B. Fischer, M. Hanke, M. Hochbruck, *A note on conjugate gradient type methods for indefinite and/or inconsistent linear systems*, Numer. Algorithms **11** (1996), 181–187.

[48] R. Fletcher, *On the Barzilai-Borwein method*, Dundee Numerical Analysis Report NA/207 (2001)

[49] G.E. Forsythe, *On the asimptotic directions of the s-dimensional optimum gradient method*, Numer. Math. **11** (1968), 57–76.

[50] I. Fredholm, *Sur une classe d'équations fonctionnelles*, Acta Math. **27** (1903), 365–390.

[51] R. Freund, M. Hochbruck, *On the use of two QMR algorithms for solving singular systems and applications in Markov chain modeling*, Numer. Linear Algebra Appl. **1** (1994), 403–420.

[52] A.J. Getson, F.C. Hsuan, {2}-*Inverses and their Statistical Applications*, Lecture Notes in Statistics **47**, Springer, Berlin, 1988.

[53] R.C. Gonzalez, R.E. Woods, *Digital Image Processing*, 2nd Edition, Prentice-Hall, 2002.

[54] R.C. Gonzalez, R.E. Woods, S.L. Eddins, *Digital Image Processing Using MATLAB*, Prentice-Hall, 2003.

[55] L. Guo, X. Du, *Representations for the Drazin inverses of* 2×2 *block matrices*, Appl. Math. Comput. **217** (2010), 2833–2842.

[56] F. Graybill, *Matrices and Applications to Statistics*, second ed., Wadsworth, Belmont, California, 1983.

[57] U. Grenander, M. Rosenblatt, *Statistical Analysis of Stationary Time Series*, Wiley and Sons, NY, 1966, Chapter 1.

[58] T.N.E. Grevile, *Some applications of the pseudo-inverse of matrix*, SIAM Rev. **3** (1960), 15–22.

[59] L. Grippo, F. Lampariello, S. Lucidi, *A nonmonotone line search technique for Newton's method*, SIAM J. Numer. Anal. **23**, 707–716 (1986)

[60] W. Hackbush, B.N. Khoromskij, E.E. Tyrtyshnikov, *Approximate iterations for structured matrices*, Numer. Math. **109** (2008), 365-383.

[61] M. Hanke, M. Hochbruck, *A Chebyshev-like semiiteration for inconsistent linear systems*, Electro. Trans. Numer. Anal. **1** (1993), 89–103.

[62] P.C. Hansen, J.G. Nagy, D.P. O'Leary, *Deblurring images: matrices, spectra, and filtering*, SIAM, Philadelphia, 2006.

[63] G. Heinig, K. Rost, *Algebraic method for Toeplitz-like matrices and operators*, Akademie-Verlag, Berlin and Birkhauser, 1984.

[64] G. Heinig, F. Hellinger, *Displacement structure of pseudoinverses*, Linear Algebra Appl. **197/198** (1994), 623–649.

[65] M.R.Hestenes, *Pseudoinverses and conjugate gradients*, Commun. ACM **18** (1975), 40–43.

[66] M. Hillebrand, C. H. Müller, *Outlier robust corner-preserving methods for reconstructing noisy images*, Ann. Statist. **35** (2007), 132–165.

[67] R.A. Horn, C.R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, New York, New Rochelle, Melbourne, Sydney, 1986.

[68] R.E. Hufnagel, N.R. Stanley, *Modulation Transfer Function Associated with Image Transmission through Turbulence Media*, J. Opt. Soc. Am. **54** (1964), 52–60.

[69] F. Husen, P. Langenberg, A. Getson, *The {2}-inverse with applications to satistics*, Linear Algebra Appl. **70** (1985), 241-248.

[70] E.F. Kaaschieter, *Preconditioned conjugate gradients for solving singular systems*, Comput. Appl. Math. **24** (1988), 265–275.

[71] T. Kailath, S.Y. Kung, M. Morf, *Displacement rank of matrices and linear equations*, J. Math. Anal. Appl. **68** (1979), 395–407.

[72] T. Kailath, A.H. Sayed, *Displacement structure: theory and applications*, SIAM Rev. **37** (1995), 297–386.

[73] T. Kailath, S.Y. Kung, M. Morf, *Displacement rank of a matrix*, Bull. Amer. Math. Soc. **1** (1979), 769–773.

[74] T. Kailath, A.H. Sayed, *Fast algorithms for generalized displacement structures, in Recent advances in mathematical theory of systems, control, networks and signal processing II*, Proc. of the MTNS-91 (H.Kimura, S.Kodama, Eds) Mita Press, Japan, 1992, 27–32.

[75] R.E. Kalaba, F.E. Udwadia, *Analytical Dynamics: A New Approach*, Cambridge University Press, Cambridge 1996.

[76] R.E. Kalaba, F.E. Udwadia, *Associative memory approach to the identification of structural and mechanical systems*, J. Optim. Theory Appl. **76** (1993), 207–223.

[77] W.J. Kammerer, M.Z. Nashed, *On the convergence of the conjugate gradient method for singular linear operators equations*, SIAM J. Numer. Anal. **9**7 (1972), 165–181.

[78] L.V. Kantorovich, G. P. Akilov, *Functional analysis* (in Russian), Moscow, 1977.

[79] S. Karanasios, D. Pappas, *Generalized inverses and special type operator algebras*, Facta Universitatis, Series: Mathematics and Informatics **21** (2006), 41–48.

[80] V. Katsikis, D. Pappas, *Fast computing of the Moore- Penrose Inverse matrix*, Electron. J. Linear Algebra **17** (2008), 637-650.

[81] F. Krahmer, Y. Lin, B. Mcadoo, K. Ott, J. Wang, D. Widemann, B.Wohlberg, *Blind image deconvolution: Motion blur estimation*, In IMA Preprints Series, 2133-5, 2006.

[82] C. Lanczos, *Solution of systems of linear equations by minimized iterations*, J.Res. N.B.S. **49**, 1952, 33–53.

[83] J. Levine, R.E. Hartwig, *Applications of Drazin inverse to the Hill cryptographic systems*, Cryptologia, **4** (1980), 71–85.

[84] S. Li, *Displacement structure of the generalized inverse* $A^{(2)}_{T,S}$, Appl. Math. Comput. **156** (2004), 33-40.

[85] D.G. Luenberg, Y. Ye, *Linear and nonlinear programming*, Springer Science+Business Media, LLC, New York, 2008.

[86] J. E. Marsden, T. Radiu, *Manifolds, Tensor Analysis, and Application*, Springer-Verlag New York, Inc, 2001.

[87] C.D.Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, Philadelphia, 2000.

[88] S.G. Mikhlin, *The Problem of the Minimum of a Quadratic Functional*, Holden-Day, San Francisco, 1965.

[89] M. Miladinović, P.S. Stanimirović, S. Miljković, *Scalar correction method for computing large scale unconstrained minimization problems*, J. Optim. Theory Appl. **151** (2011), 304–320.

[90] M. Miladinović, S. Miljković, P.S. Stanimirović, *Modified SMS method for computing outer inverses of Toeplitz matrices*, Appl. Math. Comput. **218** (2011), 3131-3143.

[91] S. Miljković, M. Miladinović, P.S. Stanimirović, D. Djordjević, *Scalar correction method for finding least-squares solutions on Hilbert spaces and its application*, Comput. Appl. Math., submitted.

[92] S.Miljković, M.Miladinović, P.S. Stanimirović, Y. Wei, *Iterative methods for computing the Drazin-inverse solution*, J. Comput. Appl. Math., submitted.

[93] S.Miljković, M.Miladinović, P.Stanimirović, I. Stojanović, *Removal of blur in X-ray images based on direct pseudoinverse computation*, Filomat, accepted.

[94] S. Miljković, M. Miladinović, P.S. Stanimirović, *Drazin-inverse solution of a matrix equation*, in preparation.

[95] A. Mohsen, J. Stoer, *A Variable Metric Method for Approximating Generalized Inverses of Matrices*, ZAMM - Z. Angew. Math. Mech. **81** (2001), 435–446.

[96] E.H. Moore, *On the reciprocal of the general algebraic matrix*, Bull. Amer. Math. Soc. 26 (1920) 394–395.

[97] M.Z. Nashed, *Steepest descent for singular linear operators equations*, SIAM J. Numer. Anal. **7** (1970), 358–362.

[98] M.Z. Nashed, *Generalized Inverses and Applications, Edited by M. Zuhair Nashed*, Proceedings of an Advanced Seminar, Academic Press, New York, San Francisko, London, 1976.

[99] M.Z. Nashed, X. Chen, *Convergence of Newton-like methods for singular operator equations using outer inverses*, Numer. Math. **66** (1993), 235-257.

[100] J. Nocedal, S. J. Wright, *Numerical Optimization*, Springer-Verlag New York, Inc, 1999.

[101] V. Olshevsky, *Structured Matrices in Mathematics, Computer Science, and Engineering I*, Proceedings of an AMS-IMS-SIAM Joint Summer Research Conference, University of Colorado, Boulder, June 27July 1, 1999, in: Contemporary Mathematics, vol. 280, AMS, Providence, RI, 2001.

[102] V.Y. Pan, *Structured matrices and polynomials. Unifed superfast algorithms*, Birkhauser Springer, 2001.

[103] V.Y. Pan, R. Schreiber, *An improved Newton iteration for generalized inverse of a matrix with applications*, SIAM J. Sci. Stat. Comput. **12** (1991), 1109–1131.

[104] V.Y. Pan, Y. Rami, X. Wang, *Structured matrices and Newton's iteration: unified approach*, Linear Algebra Appl. **343-344** (2002), 233-265.

[105] V.Y. Pan, M.V. Barel, X.M. Wang, G. Codevico, *Iterative inversion of structured matrices*, Theoret. Comput. Sci., **315** (2004), 581–592.

[106] R. Penrose, *A generalized inverse for matrices*, Proc. Cambridge Philos. Soc. **51** (1955), 406–413

[107] W.H. Pierce, *A Self-Correcting Matrix Iteration for the Moore-Penrose Generalized Inverse*, Linear Algebra Appl. **244** (1996), 357–363.

[108] C.R. Rao, S.K. Mitra, *Generalized Inverse of Matrices and its Applications*, John Wiley and Sons, Inc, New York, London, Sydney, Toronto, 1971.

[109] M. A. Rakha. *On the Moore-Penrose generalized inverse matrix*, Appl. Math. Comput. **158** (2004), 185–200.

[110] M. Raydan, *On the Barzilai and Borwein choice of steplength for the gradient method*, IMA J. Numer. Anal. **13**, 321–326 (1993)

[111] M. Raydan, *The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem*, SIAM J. Optim., **7** (1997) 26–33.

[112] M. Raydan, B.F. Svaiter, *Relaxed Steepest Descent and Cauchy-Barzilai-Borwein Method*, Comput. Optim. Appl. **21** (2001), 155–167.

[113] R.W. Schafer, R.M. Mersereau, M.A. Richards, *Constrained iterative restoration algorithms*, Proceedings of the IEEE **69** (1981), 432–450.

[114] G. Schulz, *Iterative Berechnung der reziproken Matrix*, ZAMM Z. Angew. Math. Mech. **13** (1933), 57–59.

[115] Z.J. Shi, *Convergence of line search methods for unconstrained optimization*, Appl. Math. Comput. **157** (2004), 393–405.

[116] Z.J. Shi, J. Shen, *Convergence of descent method without line search*, Appl. Math. Comput. **167** (2005), 94–107.

[117] X. Sheng, G. Chen, *Full-rank representation of generalized inverse $A^{(2)}_{T,S}$ and its applications*, Comput. Math. Appl. **54** (2007), 1422–1430.

[118] N. Shinozaki, M. Sibuya, K. Tanabe, *Numerical algorithms for the Moore-Penrose inverse of a matrix: Direct methods*, Annals of the Institute od Statistical Mathematics **24** (1972), 193–203.

[119] A. Sidi, *A unified approach to Krylov subspace methods for the Drazin-inverse solution of singular nonsymmetric linear systems*, Linear Algebra Appl. **6** (2000), 72–94.

[120] A. Sidi, *DGMRES: A GMRES-type algorithm for Drazin-inverse solution of singular nonsymmetric linear systems*, Linear Algebra Appl. **335** (2001), 189–204.

[121] A. Sidi, *A BI-CG type iterative method for Drazin-inverse solution of singular inconsistent nonsymetric linear systems of arbitrary index*, Electron. J. Linear Algebra **335** (2000), 72–94.

[122] P.S. Stanimirović, *Block representations of {2}, {1,2} inverses and the Drazin inverse*, Indian J. Pure Appl. Math. **29** (1998), 1159–1176.

[123] P.S. Stanimirović, *Applications of hyper-power method for computing matrix products*, Univ. Beograd. Publ. Elektrotehn. Fak. Ser. Mat. **15** (2004), 13–25.

[124] P.S. Stanimirović, D.S. Djordjević, *Full-rank and determinantal representation of the Drazin inverse*, Linear Algebra Appl. **311** (2000), 31–51.

[125] P.S. Stanimirović, S. Bogdanović, M. Ćirić, *Adjoint mappings and inverses of matrices*, Algebra Colloq. **13(3)** (2006), 421–432.

[126] P.S. Stanimirović, D. Cvetkovic-Ilić, *Successive matrix squaring algorithm for computing outer inverses*, Appl. Math. Comput. **203** (2008), 19-29.

[127] P.S. Stanimirović, M. Miladinović, *Accelerated gradient descent methods with line search*, Numer. Algor. **54** (2010), 503-520.

[128] P.S. Stanimirović, D.S. Cvetković-Ilić, S. Miljković, M. Miladinović, *Full-rank representations of* $\{2,4\}$, $\{2,3\}$*-inverses and successive matrix squaring algorithm*, Appl. Math. Comput., **217** (2011), 9358–9367.

[129] P.S. Stanimirović, M. Miladinović, I. Stojanović, S. Miljković, *Application of partitioning method in removal of blur in images*, in preparation.

[130] W. Sun, Y.X. Yuan, *Optimization theory and methods: nonlinear programming*, Springer, 2006.

[131] A.E. Taylor, *Introduction to Functional Analysis*, John Wiley, New York, 1958.

[132] K. Tanabe, *Conjugate-gradient method for computing the Moore-Penrose inverse and rank of a matrix*, JOTA **22** (1977), 1–23.

[133] W.R. Trench, *An algorithm for the inversion of finite Toeplitz matrices*, SIAM J. Appl. Math. **12** (1964), 515–522.

[134] F.E. Udwadia and R.E. Kalaba, *An Alternative Proof for Greville's Formula*, J. Optim. Theory Appl. **94** (1997), 23–28.

[135] F.E. Udwadia, R.E. Kalaba, *General Forms for the Recursive determination of generalized inverses: Unified approach*, J. Optim. Theory Appl. **101** (1999), 509–521.

[136] G. Wang, Y. Wei, S. Qiao, *Generalized inverses: theory and computations*, Science Press, 2003.

[137] G. Wang, Y. Wei, S. Qiao, *Generalized Inverses: Theory and Computations*, Science Press (2003).

[138] G. Wang, Z. Xu, *Solving a kind of restricted matrix equations and Cramer rule*, Appl. Math. Comput. **162** (2005), 329–338.

[139] G. Wang, *A Cramer rule for finding the solution of a class of singular equations*, Linear Algebra Appl. **116** (1989), 27–34.

[140] Y. Wei, *Index splitting for the Drazin inverse and the singular linear system*, Appl. Math. Comput. **95** (1998), 115–124.

[141] Y. Wei, *Successive matrix squaring algorithm for computing Drazin inverse*, Appl. Math. Comput. **108** (2000), 67–75.

[142] Y. Wei, H. Wu, *Additional results on index splitting for Drazin inverse of singular linear system*, Electron. J. Linear Algebra **95** (1998), 115–124.

[143] Y. Wei, H. Wu, *The representation and approximation for Drazin inverse*, J. Comput. Appl. Math. **126** (2000), 417–432.

[144] Y. Wei, H. Wu, *Convergence properties of Krylov subspace methods for singular linear systems with arbitrary index*, J. Comput. Appl. Math. **114** (2000), 305–318.

[145] Y. Wei, H. Wu, J. Wei, *Successive matrix squaring algorithm for parallel computing the weighted generalized inverse* $A^{\dagger}_{MN}$, Appl. Math. Comput. **116** (2000), 289–296.

[146] Y. Wei, J. Cai, M.K. Ng, *Computing Moore-Penrose inverses of Toeplitz matrices by Newton's iteration*, Math. Comput. Modelling **40** (2004), 181–191.

[147] Y. Wei, M.K. Ng, *Displacement structure of group inverses*, Numer. Linear Algebra Appl. **12** (2005), 103–110.

[148] P. Wolfe, *Convergence Conditions for Ascent Methods*, SIAM Review **11** (1969), 226–235.

[149] H. Yang, D. Liu, *The representations of the generalized inverses* $A^{(2,3)}_{T,S}$ *and its applications*, J. Comput. Appl. Math. **224** (2009), 204–209.

[150] H. Yang, D. Liu, J. Xu, *Matrix left symmetry factor and its applications in generalized inverses* $A^{(2,4)}_{T,S}$, Appl. Math. Comput. **197** (2008), 836–843.

[151] B. Zheng, L. Ye, D.S. Cvetković-Ilić, *Generalized inverses of a normal matrix*, Appl. Math. Comput. **206** (2008), 788–795.

[152] B. Zheng, R.B. Bapat, *Generalized inverse* $A^{(2)}_{T,S}$ *and a rank equation*, Appl. Math. Comput. **155** (2004), 407-415.

[153] N. Zhang, Y. Wei, *On the convergence of general stationary iterative methods for range-Hermitian singular linear systems*, Numer. Linear Algebra Appl. **17** (2010), 139–154.

[154] J. Zhou, Y. Wei, *Stagnation analysis of DGMRES*, Appl. Math. Comput. **151** (2004), 27–39.

[155] J. Zhou, Y. Wei, *The analysis of restart DGMRES for solving singular linear systems*, Appl. Math. Comput. **176** (2006), 293–301.

[156] J. Zhou, Y. Wei, *DFOM algorithm and error analysis for projection methods for solving singular linear system*, Appl. Math. Comput. **157** (2004), 313–329.

[157] J. Zhou, Y. Wei, *A two-step algorithm for solving singular linear systems with index one*, Appl. Math. Comput. **8** (2001), 83–93.

[158] G. Zielke, *Report on test matrices for generalized inverses*, Computing **36** (1986), 105–162.

[159] *Image Processing Toolbox User's Guide*, The Math Works, Inc., Natick, MA, 2009.

[160] *MATLAB 7 Mathematics*, The Math Works, Inc., Natick, MA, 2010.

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

KEY WORDS DOCUMENTATION

Accession number, **ANO**:		
Identification number, **INO**:		
Document type, **DT**:		**monograph**
Type of record, **TR**:		**textual / graphic**
Contents code, **CC**:		**doctoral dissertation**
Author, **AU**:		**Sladjana Lj. Miljković**
Mentor, **MN**:		**Predrag S. Stanimirović**
Title, **TI**:		ITERATIVE METHODS FOR COMPUTING GENERALIZED INVERSES OF MATRICES
Language of text, **LT**:		**English**
Language of abstract, **LA**:		**English**
Country of publication, **CP**:		**Serbia**
Locality of publication, **LP**:		**Serbia**
Publication year, **PY**:		**2012**
Publisher, **PB**:		**author's reprint**
Publication place, **PP**:		**Niš, Višegradska 33.**
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)		**140 p. ; graphic representations**
Scientific field, **SF**:		**mathematics**
Scientific discipline, **SD**:		**nonlinear optimization, generalized inverses**
Subject/Key words, **S/KW**:		**nonlinear optimization, generalized inverses**
UC		512.643 : 519.613 512.643 : 519.654 + 519.863
Holding data, **HD**:		**library**
Note, **N**:		
Abstract, **AB**:		The subject of investigation of the doctoral dissertation is the calculation of generalized inverses of matrices, as well as the connection between the generalized inverses of matrices with the optimization theory concepts. There are defined new iterative methods for solving optimization problems. A special attention is devoted on defining new iterative methods for generalized inverses calculation. The defined methods enables efficient calculation of generalized inverses, as well an analysis of their properties. The contribution of the doctoral dissertation is in the field of generalized inverses, as well, in the field of unconstrained optimization theory. This is claimed by the proposal of the new effective algorithms which can be compared to the most favorable ones in their disciplines.
Accepted by the Scientific Board on, **ASB**:		26.12.2011
Defended on, **DE**:		
Defended Board, **DB**:	President:	
	Member:	
	Member, Mentor:	

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:	
Идентификациони број, **ИБР**:	
Тип документације, **ТД**:	монографска
Тип записа, **ТЗ**:	текстуални / графички
Врста рада, **ВР**:	докторска дисертација
Аутор, **АУ**:	Слађана Љ. Миљковић
Ментор, **МН**:	Предраг С. Станимировић
Наслов рада, **НР**:	ИТЕРАТИВНИ МЕТОДИ ЗА ИЗРАЧУНАВАЊЕ УОПШТЕНИХ ИНВЕРЗА МАТРИЦА
Језик публикације, **ЈП**:	енглески
Језик извода, **ЈИ**:	енглески
Земља публиковања, **ЗП**:	Србија
Уже географско подручје, **УГП**:	Србија
Година, **ГО**:	2012.
Издавач, **ИЗ**:	ауторски репринт
Место и адреса, **МА**:	Ниш, Вишеградска 33.
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)	140 стр., граф. прикази
Научна област, **НО**:	математика
Научна дисциплина, **НД**:	нелинеарна оптимизација, уопштени инверзи
Предметна одредница/Кључне речи, **ПО**:	нелинеарна оптимизација, уопштени инверзи
УДК	512.643 : 519.613 512.643 : 519.654 + 519.863
Чува се, **ЧУ**:	библиотека
Важна напомена, **ВН**:	
Извод, **ИЗ**:	Предмет истраживања докторске дисертације припада областима нелинеарне оптимизације и уопштених инверза матрица уз посебан осврт на њихова повезаности. Циљ истраживања је израчунавње уопштених инверза матрица као и изучавање повезаности уопштених инверза са концептима теорије оптимизације. Дефинисани су итеративни методи за решавање оптимизационих модела за израчунавање уопштених инверза. Неки од итеративних метода представљају аналогије методима који су дефинисани у решавању оптимизационих модела, док су неки методи настали генерализацијом познатих метода. Добијених резултата би допринели, како у теоријском тако и у практичном погледу, за унапређењу постојећих метода за успешна и ефикасна израчунавања на генералисаним инверзима и њиховој даљој примени.
Датум прихватања теме, **ДП**:	26.12.2011
Датум одбране, **ДО**:	
Чланови комисије, **КО**: Председник:	
Члан:	
Члан, ментор:	