UNIVERSITAS STUDIORUM · INSUBRIAE

# ALGORITHMIC variations on the theme of

$$\begin{pmatrix} s & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & t & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & u & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & c & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & t & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & u & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & r & 0 & 0 \\ 0 & 0 & m & a & t & r & i & c & e & s \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & d \end{pmatrix}$$

WITH APPLICATIONS TO GRAPHS AND IMAGING

Università degli Studi dell'Insubria
Dottorato in Scienze Fisiche e Matematiche
Matematica del Calcolo
XXV ciclo

# Algorithmic variations on the theme of structured matrices, with applications to graphs and imaging

Candidate
Pietro Dell'Acqua

Supervisors
Prof. Marco Donatelli
Prof. Claudio Estatico

# Contents

# The cue

As suggested by the cover, the title of this thesis is a pun that involves math and music, while its content concerns the fruits of research activity carried out during the PhD period. We have gathered here all the work done under the aegis of *structured matrices*, which is a wide and interesting field that includes the two main threads of our investigation. In the first one we have dealt with the minimum cost flow problem and the matrices have structure of graph, while in the second one we have dealt with the image restoration problem, in which space invariance leads to strong algebraic structures in the system matrix; depending on the boundary conditions enforced in the discretization, we find circulant, Toeplitz or even more complicated structures. Since we do not intend to bore the rare and precious reader, we have devoted only a few pages to introduce or recall known results or techniques available in literature and we have concentrated on describing our original contribution, based on the papers cited at the end of this introduction.

More in detail, in Part I we consider multigrid type techniques for the numerical solution of large linear systems whose coefficient matrices show the structure of (weighted) graph Laplacian operators. We combine ad hoc coarser-grid operators with iterative techniques used as smoothers. We show that the most effective smoothers have to be of Krylov type with spanning tree preconditioners, while the projectors, which define the coarser-grid operators, have to be designed for maintaining as much as possible the graph structure of the projected matrix at the inner levels. Some necessary and sufficient conditions are proved: in this framework it is possible to explain why the classical projectors inherited from differential equations are good in the differential context and why they behave unsatisfactorily for unstructured graphs. We report the outcome of several numerical experiments showing that our approach is effective even in very difficult cases where the known approaches are rather slow. However the main advantage of the proposed approach is the robustness, since our multigrid type technique behaves uniformly well in all cases, without requiring either the setting or the knowledge of critical parameters, as it happens when using the best known precondi-

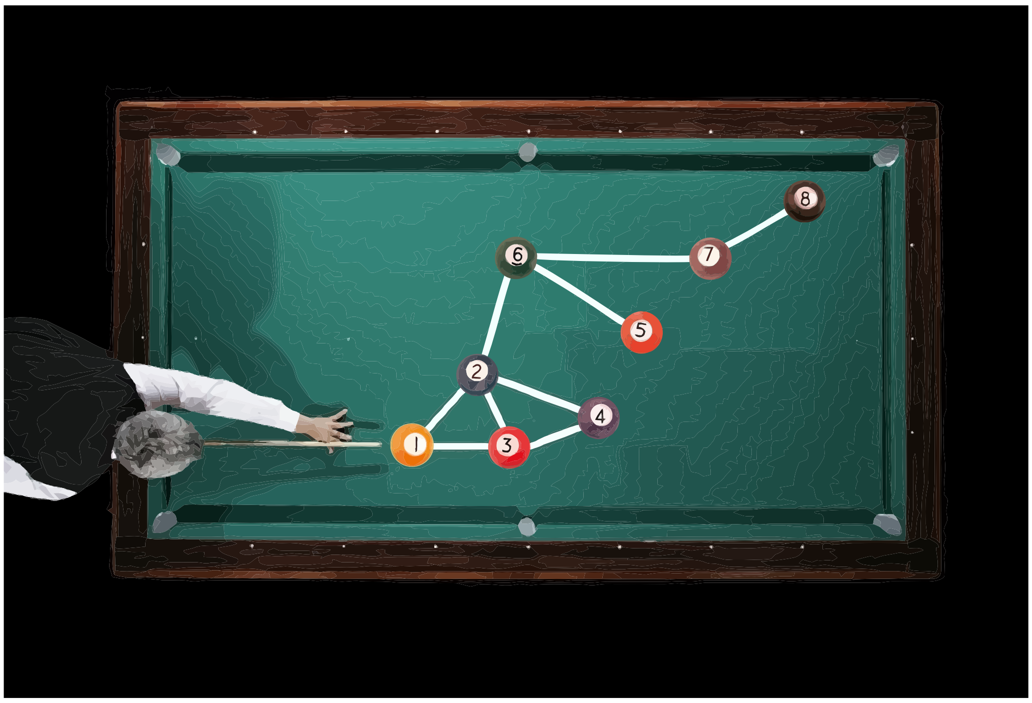


Figure 1: Graph of the thesis on a billiard table.

tioned Krylov methods. This first Part is organized as follows. In Chapter 1 we introduce the minimum cost flow problem and we describe in general terms the multigrid procedure. Chapter 2 is devoted to the theoretical characterization of the family of projectors that preserve the graph structure at all steps of the multigrid chain, thereby allowing combined use of multigrid and combinatorial preconditioning. Chapter 3 is devoted to the presentation of several classes of projectors with the right properties, either newly proposed or drawn from the relevant literature. Finally Chapter 4 is devoted to the presentation of a large set of numerical experiments comparing preconditioned Krylov methods and accelerated multigrid methods.

As we said, Part II is devoted to image deblurring problem, which is an important task with many applications. The blurring of images may be caused by object motion, calibration errors of imaging devices, random fluctuations of the medium, for instance atmosphere. We are interested in restoring images that have been contaminated by blur and noise. To do this, we have to solve a linear system, usually of very large size and very sensitive to data error, due to the ill-posed nature of the continuous problem. Thus in Chapter 5 we introduce the image restoration problem, highlighting the importance of boundary conditions, and then, inspired by the theoretical results on optimal preconditioning stated by Ng, R. Chan, and Tang in the framework of Reflective boundary conditions, we present analogous results for Anti-Reflective ones. In both cases, the optimal preconditioner is the blurring matrix associated to the symmetrized point spread function. Computational results show that the preconditioning strategy is effective and it is able to give rise to a meaningful acceleration. In Chapter 6, once recalled that classical iterative deblurring algorithms that use the conjugate transpose $A^H$ of the coefficient matrix $A$ show usually a slow convergence, we propose a variant which replaces $A^H$ with a new matrix $Z$. This approach, which is linked with preconditioning theory and reblurring processes, can be applied to a wide set of iterative methods. Computational tests show that this strategy leads to a significant improvement of the convergence speed of the methods. Moreover it can be naturally combined with other widely used acceleration techniques. And precisely to that topic is devoted Chapter 7, in which we present a particular acceleration technique, that is nothing but the application of the so-called $\nu$-method, conceived for speed up Landweber method, in the new framework of statistical methods. Computational results show the effectiveness of this strategy, which gives rise to remarkable acceleration factors, evaluated by comparing accelerated methods with classical ones. In Chapter 8 we describe a map acceleration for statistical methods inspired by theory on Landweber method in Banach spaces and we show the link between it and Meinel acceleration, which consists in the introduction of an exponent in the iterative formulas. Furthermore we propose a generalization of this Banach technique in order to overcome the difficulties that this strategy displays when it is compared with classical (Hilbert) Landweber method. Numerical results highlight the goodness of our proposals, both in terms of stability and velocity.

In summary, both Part I and Part II concern various techniques to improve speed and stability of iterative methods, employed to gain an approximated solution for linear systems that have large size. Another links between the two parts are the full weighting operator and more in general multigrid theory, which are mainly used in the first one, but are also utilized in the second one, in particular in Chapter 6, to illustrate a coarsening technique for building $Z$. This is the starting-cue; the overall structure of the thesis is drawn in Figure 1; finally the pocket is devoted to conclusions, open issues and further ideas for future research.

## Papers

P. Dell'Acqua, A. Frangioni, S. Serra Capizzano
*Accelerated multigrid for graph Laplacian operators*
Numerical Algorithms, submitted.

P. Dell'Acqua, A. Frangioni, S. Serra Capizzano
*Computational evaluation of multi-iterative approaches for solving graph-structured large linear systems*
Applied Numerical Mathematics, submitted.

P. Dell'Acqua, C. Tablino Possio, S. Serra Capizzano.
*Optimal preconditioning for image deblurring with Anti-Reflective boundary conditions*
BIT Numerical Mathematics, submitted.

P. Dell'Acqua, M. Donatelli, C. Estatico
*A unifying variant for deblurring algorithms*
Journal of Computational and Applied Mathematics, submitted.

P. Dell'Acqua, M. Donatelli, C. Estatico
*A general Z variant for iterative and direct regularization methods*
In preparation.

P. Dell'Acqua
*Automatic acceleration for EM method is not the best*
In preparation.

P. Dell'Acqua, C. Estatico.
*A connection between Banach spaces and Meinel acceleration*
In preparation.

# Part I

# The minimum cost flow problem

# Chapter 1

# The multi-iterative idea

Large linear systems with (weighted) graph-structured matrices can be found in several applications where a network structure is present (e.g. [35] and references therein). Even restricting to the symmetric case, with matrices arising from graph *Laplacian* operators, these linear systems can be found, among the others, in Web searching engines [86], general Markov chains [39], consensus algorithms [96] and optimization problems in networks [2, 10]. All these different problems exhibit three main issues: a) the large size of the considered linear systems; b) the sparsity and the graph structure of the involved matrices; c) the potential ill-conditioning, as a function of the matrix dimension and/or of other critical parameters. These features should immediately refrain from recommending direct solvers. For instance, the methods based on factorizations [62], such as Gaussian $LR$, $QR$ or real Cholesky $LL^T$, do not exploit the structure and in particular the sparsity of the matrices, so that the cost will become unacceptably high both in space and time [29], and the precision unacceptably poor in case of ill-conditioning. On the other hand the use of iterative solvers has the immediate advantage that the matrix vector product can be done with linear work in case of sparsity.

The specific application motivating our research is the solution of linear systems arising at all iterations of Interior Point (IP) techniques for the Minimum Cost Flow (MCF) problem. We start by recalling the definition of *node-arc* incidence matrix of a directed graph.

**Definition 1.1** *Let $\mathcal{G} \equiv \mathcal{G}_n = (\mathcal{U}_n, \mathcal{V}_n)$ be a* directed graph *with $n$ nodes $\mathcal{U}_n = \{u_1, \dots, u_n\}$ and $m$ arcs $\mathcal{V}_n = \{v_1, \dots, v_m\}$; its* node-arc incidence matrix *$E \equiv E_n = E(\mathcal{G}_n)$ is the $n \times m$ matrix such that $E_{ij} = 1$ if $v_j$ emanates from $u_i$, $E_{ij} = -1$ if $v_j$ terminates at $u_i$ and $E_{ij} = 0$ otherwise. Hence $E$ has exactly two non-zero elements (a $1$ and a $-1$) in every column.*

Given a directed graph $\mathcal{G}$, the linear MCF problem [10] is the Linear Program (LP)

$$\min \left\{ \, c^T x \; : \; Ex = d \; , \; 0 \leq x \leq u \, \right\} \tag{1.1}$$

where $E$ is the node-arc incidence matrix of $\mathcal{G}$, $c$ is the vector of arc costs, $u$ is the vector of arc upper capacities, $d$ is the vector of node deficits and $x$ is the vector of flows. The *flow conservation constraints* $Ex = d$ express the fact that the flow has to travel in the graph from *sources* (nodes with $d_i > 0$) to *destinations* (nodes with $d_i < 0$), while for the remaining *transhipment nodes* (with $d_i = 0$) the total inbound flow must equal the total outbound flow. This problem has a huge set of applications, either in itself or, more often, as a submodel of more complex and demanding problems [2]. Without loss of generality we can restrict our analysis to connected graphs, as the general case of more than one connected component can be traced back to this case (basically there is a separate LP for each one).

We study graph matrices coming from the application of IP methods, which have grown a well-established reputation as efficient algorithms for large-scale problems. In these methods, at each step we have to solve linear systems of the form

$$E\Theta E^T x = b \;\; , \tag{1.2}$$

where $E$ is fixed, while $b \in \mathbb{R}^n$ and the $m \times m$ diagonal positive definite matrix $\Theta$ depend on the IP iteration; as we will not consider (possible) strategies for re-use of information between two different IP iterations, we will disregard this dependence, thus focusing our attention to the solution of (1.2) at any one fixed iteration. Since each diagonal element of $\Theta$ is associated to a specific arc of $\mathcal{G}$, we can consider $\mathcal{G}$ as a *weighted* graph, with $\Theta$ specifying the arc weights. In the following we will often use the shorthand $L = E\Theta E^T$, also disregarding the fact that $L$ actually depends on $\Theta$. As the following remark shows, $L$ is closely tied to other well-known graph matrices.

**Remark 1.2** *Let $\mathcal{G}' = (\mathcal{U}, \mathcal{V}')$ be the* undirected *graph obtained from $\mathcal{G}$ by ignoring the orientation of the arcs. $\mathcal{G}'$ also is a weighted graph, the weight $w_{uv}$ of each edge $\{u, v\} \in \mathcal{V}'$ being the sum of the weights of all arcs of $\mathcal{G}$ which "collapse" in $\{u, v\}$ (note that multiple parallel arcs are allowed in (1.1), as they can have different cost and capacity). Then let $A(\mathcal{G}')$ be the symmetric (weighted) adjacency matrix of $\mathcal{G}'$, such that $A_{uv}$ is the weight of the edge $\{u, v\}$ if it belongs to $\mathcal{V}'$ and* 0 *otherwise. Further let $D(\mathcal{G}')$ be the $n \times n$ the diagonal matrix with $D_{uu} = \sum_{v \in \mathcal{U}} A_{uv}$ ($d_u$ is the node degree of $u$ in the unweighted case). The* Laplacian *of $\mathcal{G}'$ is*

$$L(\mathcal{G}') = D(\mathcal{G}') - A(\mathcal{G}') \;\; ,$$

*and it is easy to show that*

$$L(\mathcal{G}') = E(\mathcal{G})\Theta(\mathcal{G})E(\mathcal{G})^T \ .$$

*Hence, topological information about the original directed graph $\mathcal{G}$ is contained in $E(\mathcal{G})$ as well as in $L(\mathcal{G}')$. The Laplacian of a graph has very many applications in such diverse fields as graph theory, statistics and combinatorial optimization [31, 78, 89].*

In most general-purpose LP solvers, the linear systems (1.2) are solved by means of direct methods, typically the Cholesky decomposition preceded by a heuristic reordering of the columns of $E$ aimed at minimizing the fill-in. For very large, sparse networks this approach is inefficient, as disastrous fill-in (e.g. [29]) may occur which renders the iteration cost unbearable. One thus has to revert to iterative methods instead, but these approaches can be competitive only if the rate of convergence is sufficiently high. This motivates studies of the extreme singular values of $E$ and of the spectral behaviour of $L$, since the convergence rate of iterative methods largely depends on the conditioning $\mu(\cdot)$ of the matrix [61]. In the first IP iterations, the matrix $\Theta$ is close to the identity and the spectral difficulties are mild: $\mu(L) \leq cn^2$, with $c$ absolute constant and the bound attained up to lower order terms in the case of linear graphs [61]. However in the last IP iterations the matrix $\Theta$ becomes highly unbalanced and the conditioning of $L$ is essentially described by the wild conditioning of $\Theta$. This phenomenon is analysed in [94] for the case of linear graphs and this analysis is particularly relevant for the numerical solution of (1.2) through a preconditioned conjugate gradient (PCG) method. Most PCG-based IP algorithms employ *support-graph preconditioners* [119, 12, 18, 19] of the form

$$L_{\mathcal{S}} = E_{\mathcal{S}}\Theta_{\mathcal{S}}E_{\mathcal{S}}^T \ ,$$

where $E_{\mathcal{S}}$ and $\Theta_{\mathcal{S}}$ denote the restriction of $E$ and $\Theta$ respectively, on the arcs of a “simple” subgraph $\mathcal{S}$ of $\mathcal{G}$. Basically there are two possible choices for $\mathcal{S}$. The first is to aim at minimizing the computational burden of inversion (factorization) of $L_{\mathcal{S}}$, which boils down to choosing it as a spanning tree or some other chordal-type graph [90, 58, 59], so that the corresponding node-arc incidence matrix of is in triangular or block triangular form and $L_{\mathcal{S}}$ can be factorized without fill-in. Computationally spanning trees are the most effective choice in all but the most difficult cases, due to the fact that $\mathcal{S}$ can be chosen as an (approximate) Minimum Spanning Tree in $O(m)$, e.g. with the Prim algorithm. The other choice is instead to aim at selecting $\mathcal{S}$ in such a way to obtain the best possible provable improvement of the conditioning of the preconditioned system. This usually reduces to appropriately (recursively) partitioning the graph into “weakly interacting” clusters [112, 81].

Alternatively (or in addition) Steiner preconditioners [63, 80] allow to simplify the task of choosing the right subgraph at the cost of introducing "fake" nodes in the graph ("do not care equations") and therefore solving a slightly larger problem.

While theoretically sound and supported by a sophisticated analysis, the status of "complex" support-graph preconditioners from the computational standpoint is not yet very clear, as some of the hidden constants in the complexity results may be large. This is even more relevant insomuch actual implementations of the corresponding approaches are complex and not widely available, while "simple" PCG approaches using tree or tree-like subgraphs can be implemented with minor modifications of standard solution methods. Yet, while these preconditioners often work quite well, there are some cases where the convergence rate is slow. Along the lines of [82, 42, 43], our objective is therefore to evaluate whether "simple" tree-based PCG approaches can be complemented with ideas from the algebraic multigrid (AMG) field [102, 115] — and in particular from multi-iterative techniques [107] — to yield methods that combine a relatively simple implementation and robustness, in the sense of uniformly delivering good performances without the need of complex parameter tuning.

## 1.1 Multigrid methods

Let $L = L^T \in \mathbb{R}^{n\times n}$ be a positive definite matrix, $b \in \mathbb{R}^n$ be the right-hand-side and, $l \in (0, n)$ (most often $l \approx \log n$) be the *number of levels.* Fix integers $n_0 = n > n_1 > n_2 > \ldots > n_l > 0$, take $R^i_{i+1} \in \mathbb{R}^{n_{i+1}\times n_i}$ full-rank matrices and consider a class $\mathcal{S}_i$ of iterative methods for $n_i$-dimensional linear systems. The related *V-cycle method* [117] produces the sequence $\{x^{(k)}\}_{k\in\mathbb{N}}$ according to the rule $x^{(k+1)} = \mathcal{MGM}(0, x^{(k)}, b)$, with $\mathcal{MGM}$ recursively defined as follows:

```
            $x_i^{(\text{out})} := \mathcal{MGM}(i, x_i^{(\text{in})}, b_i)$
if( $i = l$ ) then  Solve($L_l x_l^{(\text{out})} = b_l$)
            else  1 | $r_i := L_i x_i^{(\text{in})} - b_i$
                  2 | $b_{i+1} := R^i_{i+1} r_i$
                  3 | $L_{i+1} := R^i_{i+1} L_i (R^i_{i+1})^T$
                  4 | $y_{i+1} := \mathcal{MGM}(i+1, 0_{n_{i+1}}, b_{i+1})$
                  5 | $x_i^{(\text{int})} := x_i^{(\text{in})} - (R^i_{i+1})^T y_{i+1}$
                  6 | $x_i^{(\text{out})} := \mathcal{S}_i^{\nu}\big(x_i^{(\text{int})}\big)$
```

Step 1 calculates the residual of the proposed solution. Steps 2, 3, 4, 5 define the *recursive coarse grid correction* by projection (step 2) of the residual, subgrid correction (steps 3, 4) and interpolation (step 5), while step 6 performs some ($\nu$) iterations of a “post-smoother”.

By using the MGM as an iterative technique [102], at the $k$-th iteration, we obtain the linear systems $L_i x_i^{(k)} = b_i^{(k)}$, $i = 0, \dots, l$, where the matrices $L_i = L_i^T \in \mathbb{R}^{n_i \times n_i}$ are all positive definite. Only the last one is solved exactly, while all the others are recursively managed by reduction to low-level system and smoothing. The procedures $\mathcal{S}_i$ are most often standard stationary iterative methods [122], such as Richardson, (damped) Jacobi, Gauss-Seidel etc., with prescribed iteration matrix $S_i \in \mathbb{R}^{n_i \times n_i}$, i.e.

$$\mathcal{S}_i(x_i^{(\text{int})}) = S_i x_i^{(\text{int})} + (I_{n_i} - S_i) L_i^{-1} b_i^{(k)} \quad , \quad x_i \in \mathbb{R}^{n_i} \quad , \quad i = 0, \dots, l-1 \ .$$

If we recursively define the multigrid iteration matrix of level $i = l-1, \dots, 0$ as

$$\begin{cases} MGM_l = 0_{n_l \times n_l} \\ MGM_i = S_i^{\nu_i} \Big[ I_{n_i} - \big(R_{i+1}^i\big)^T \big(I_{n_{i+1}} - MGM_{i+1}\big) L_{i+1}^{-1} R_{i+1}^i L_i \Big] \end{cases} , \qquad (1.3)$$

then $x_i^{(\text{out})} = MGM_i x_i^{(\text{in})} + (I_{n_i} - MGM_i) L_i^{-1} b_i$, so in the finer grid we have

$$x^{(k+1)} = MGM_0 x^{(k)} + (I_{n_0} - MGM_0) L_0^{-1} b, \qquad x^{(r)} = x_0^{(r)} \ \forall r,$$

and $MGM_i$ depends on $i$ but not on any of the $x_i^{(k)}$ and $b_i^{(k)}$. For each $i = 0, \dots, l-1$, the algorithm has essentially two degrees of indetermination: the choice of the projectors $R_{i+1}^i$ and the choice of the smoothers $\mathcal{S}_i$. The former, as well as the calculation of the $L_i$ matrices, are performed before the beginning of the V-cycle procedure (pre-computing phase). While this is not necessary in the general multi-iterative approach, we will only consider convergent smoother iterations. Moreover, if the smoother is convergent in the $L$-norm ($\|x\|_L^2 = x^T L x$, $L$ symmetric and positive definite), then the multigrid iteration matrix has $L$-norm smaller than that of the smoother. In other words, the multigrid iteration is never worse than the smoother alone [107, 64].

Two further modifications can also be applied: using a *pre-smoother*, i.e. adding a step 0 similar to step 6 where a further stationary iterative method is employed; allowing, in steps 6 (and 0), the number of smoothing iterations to depend on the level $i$. This corresponds to the matrix in (1.3) being multiplied on the right by a further iteration matrix, i.e.

$$MGM_i = S_{i,\text{post}}^{\nu_{i,\text{post}}} \Big[ I_{n_i} - \big(R_{i+1}^i\big)^T \big(I_{n_{i+1}} - MGM_{i+1}\big) L_{i+1}^{-1} R_{i+1}^i L_i \Big] S_{i,\text{pre}}^{\nu_{i,\text{pre}}} \ ,$$

where the number of smoothing steps $\nu_{i,\text{pre}}$ and $\nu_{i,\text{post}}$ depend on the level $i$. In practice (cf. [5, 108] and the references therein) the application of the pre-smoother accelerates the global convergence substantially, but the explanation of this phenomenon falls outside the convergence theory of the algebraic multigrid and indeed pertains to multi-iterative methods [107]. For instance, looking just at the two-grid method, in the case of the $d$-dimensional discrete Laplacian, it is easy to prove that the post-smoothing given by the Richardson iteration with $\omega = \omega_1 \equiv 1/4d$ is strongly converging in the subspace of the high frequencies and that the coarse grid correction strongly reduces the error in the low frequencies subspace. Therefore, the combination of the two complementary iterations, which separately are slowly convergent on the global space $\mathbb{R}^n$, leads to a fast convergent two-grid method. However a finer analysis tells us that the global error is now essentially localized in the middle frequencies: the iteration again given by Richardson but with with $\omega = \omega_2 \equiv 1/2d$ is not smoother, but it is fast convergent just in the middle frequencies subspace. Therefore its further use in step 6 (or equivalently in step 0) increases very much the "spectral complementarity", so that we obtain a real multi-iterative method whose spectral radius is really small. We call an iteration having a spectral behaviour complementary to both the coarse grid correction and the smoother an "intermediate iteration"; an example is the Richardson iteration with $\omega_2 = 2\omega_1$ in the case of the discrete Laplacian. As for the level-dependent number of smoothing iterations [108], it can be easily shown that a polynomial growth with $i$ does not affect the global cost, that remains linear for banded or sparse structures, only changing the constants involved in the big $O$. This strategy, together with sophisticated preconditioners in the smoothing phases, was the key for developing very effective multigrid solvers for very ill-conditioned Sinc-Galerkin matrices [93], where, like in the case of weighted Laplacians, there is the presence of a positive diagonal matrix whose conditioning is extremely high (in fact exponential as the size of the matrix in the Sinc-Galerkin setting). However, while in the standard differential setting there is a gain in using an increasing $\nu_i = \nu_{i,\text{pre}} + \nu_{i,\text{post}}$, for problems with a smaller ill-conditioning (regularized Laplacian in [46]) the method that achieves the smallest theoretical cost and that minimizes the actual CPU times for reaching the solution with a preassigned accuracy is the simplest V-cycle with only one step of post-smoothing given by a classical damped Jacobi.

Since several possible choices exist in the implementation of the MG approach, in the remainder of this section we will briefly discuss some initial computational tests, performed in the context of MCF problems, that provide a guidance about how to choose at least the two main components: projectors and smoothers.

## 1.2 Ghost node

Let $e = e_{(n)}$ be the all-ones vector of length $n$. It is immediate to realize that $e^T E = 0^T$ and therefore $e^T L = 0^T$, i.e. $\mathrm{rank}(L) = n - 1$. However $e^T d = 0$ as well (for otherwise (1.1) cannot have any feasible solution [2]), and this property is transmitted to the right-hand sides $b$. Hence by the Rouché-Capelli Theorem the linear system (1.2) has $\infty^1$ solutions of the form $x(\alpha) = \hat{x} + \alpha e$ for $\alpha \in \mathbb{R}$. Thus, let $E'$ and $b'$ be obtained by $E$ and $b$ respectively by erasing any one row and $L'$ obtained by $E'$ as usual. A solution to (1.2) can be obtained by solving the *cut system*

$$L'x' = b' \tag{1.4}$$

and setting $\hat{x} = (x', 0)$. Intuitively this amounts at creating a "ghost node", the one corresponding to the deleted row, in the graph, as the columns of $E$ corresponding to arcs entering (leaving) the ghost node will only have a 1 ($-1$) without the corresponding $-1$ (1).

It is well-known that, since $L$ is a semidefinite positive matrix, CG or PCG approaches (started from the zero vector) solve (1.2) in the least-squares sense, i.e. find

$$\bar{x} = \operatorname*{argmin}_{x \in \mathbb{R}^n} \|Lx - b\|_2 \qquad \text{or equivalently} \qquad \|\bar{x}\|_2 = \min_{\alpha \in \mathbb{R}} \|\hat{x} + \alpha e\|_2 \ . \tag{1.5}$$

This is, in general, not true for other methods. Hence using CG or PCG allows to work on the original graph at all levels but the last one, where eliminating one row (i.e. passing from (1.2) to (1.4)) to recover $L'_l$ is necessary since a non-singular matrix is required by direct methods. Thus [P]CG can maintain our graph "sheltered from ghosts" which, as we will see, can have a positive impact on performances.

## 1.3 Smoothers

In accordance with the most successful strategies in the differential case, we tried as smoothers a combination of classical iterations. We recall that the discrete Laplacian can be viewed as a special instance of a graph matrix. In addition, by the analysis in [61], all the (unweighted) graph matrices with $\Theta = I$ share the property that the small eigenvalues are related to smooth eigenvectors. In other words, as in the differential setting, the degenerating subspace is located essentially in the low frequencies. Therefore we tested several combinations of pre- and post-smoothers to find the one with better spectral complementarity and therefore performances. The preliminary numerical results were not encouraging especially in the last iterations of the IP process where the $\Theta$ becomes very unbalanced, with very large entries ($\approx$ `1e+6`) and very small entries ($\approx$ `1e-10`).

A similar occurrence was already experienced in [93], where the distribution of the nodes in the Sinc-Galerkin method induced very unbalanced diagonal entries. In that application, the only successful strategy was the combination of PCG smoothers with sophisticated and powerful preconditioners, in connection with the classical projection used in the differential context. Taking inspiration from this setting and from the similarity induced by the wild behaviour of the diagonal entries, we focused our attention on more powerful smoothers, that is the PCG family with a large set of specialized preconditioners: a) diagonal; b) incomplete Cholesky factorization with tolerance parameter droptol (non-zero fill-in); c) tree-based preconditioners. However the problem with general graph matrices is more complicated than the one in [93], since the graph structure is not nearly as regular as those arising in the context of differential operators.

Regarding the computational burden, we assume that the cost at every level of the MG iteration is linear with respect to the dimension (let us say bounded by $O(n_i) = \gamma n_i$) plus the recursion part and we assume that the size is halved at every recursion step. Concerning the number of smoothing steps, we consider the growth function $\nu_{i,\text{pre}} = i^k$ that is we make more smoothing steps at the inner levels and we assume that the cost of the smoothing iteration is responsible of the linear cost $O(n_i) = \gamma n_i$. The total work $W_i$ at level $i$ required to perform a MG iteration is then given by the recursive law

$$W_i \leq \gamma n_i (l+1-i)^k + W_{i-1}, \quad n_{i-1} = \frac{n_i}{2},$$

with $W_0$ constant. Therefore, setting $n = n_l$, at the finest level $l \approx \log(n)$ we have

$$W_l \leq \sum_{i=1}^{l} \gamma \frac{n}{2^{i-1}} i^k + W_0 < \gamma n \sum_{i=1}^{\infty} \frac{i^k}{2^{i-1}} + W_0 \ . \tag{1.6}$$

The rightmost series in (1.6) is convergent; its initial values for $k = 0, 1, 2, 3, 4, 5$ are respectively 2, 4, 12, 52, 300 and 2164. This estimate has to be multiplied by 2 when employing both pre- and post-smoother, as in our case. Preliminary numerical experiments with different values of $k$ ($\in \{0, 1, 2\}$) clearly showed that $k = 0$ results in the best performances, which was therefore the setting used in all the subsequent test.

## 1.4 Projectors

Our initial interest was to verify whether standard operators with good performances in the context of differential equations, such as the very classical Full Weighting Operator (FWO)

$$R_{\mathrm{FWO}} = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & & & \\ & & 1 & 2 & 1 & & & \\ & & & & & \ddots & & \\ & & & & & 1 & 2 & 1 \end{pmatrix}, \tag{1.7}$$

would perform equally well in the context of general graph matrices. The preliminary results quickly made it clear that this was not the case: in fact, the corresponding MGM is not efficient even for these easy graphs and $\Theta = I$, which is generally the easiest case because it shows a moderate conditioning [61]. The results are also confirmed by the fact that using more sophisticated choices, such as quadratic approximation [45] with stencil

$$\frac{1}{16}[1, 4, 6, 4, 1]$$

and even cubic interpolation [117] with stencil

$$\frac{1}{32}[-1, 0, 9, 16, 9, 0, -1]$$

does not lead to better results (actually even to worse ones). Thus the differential setting is of no help in our graph problem. This is due to the fact that the matrix at lower levels is far away from a graph matrix and this destroys the structure which the MGM relies upon.

All this pushed us towards the development of the different operators described in the following. In particular we quickly focused our attention on Aggregation Operators (AGO) of the generic form

$$R_{\mathrm{AGO}} = \begin{pmatrix} 1 & 1 & 1 & & & & & & \\ & & & 1 & 1 & & & & \\ & & & & & \ddots & & & \\ & & & & & 1 & 1 & 1 & 1 \end{pmatrix}, \tag{1.8}$$

which *preserve the graph structure* in the recursion, i.e.

$$R_{\mathrm{AGO}} E \Theta E^T R_{\mathrm{AGO}}^T \; = \; R_{\mathrm{AGO}} L(\mathcal{G}) R_{\mathrm{AGO}}^T \; = \; L(\mathcal{G}')$$

is the Laplacian of a new graph $\mathcal{G}'$ corresponding to the *aggregation of nodes* of the original graph (see Definition 2.1). Similar operators have been used with good results in different contexts, for example when designing multigrid solvers for Markov Chains [39]. Numerical testing clearly showed that MG approach with a simple structure-preserving projector appeared to be competitive. Furthermore the tests showed that, for general graphs and arc weights (such as these of MCF matrices at final IP iterations), the only effective smoothers are preconditioned Krylov methods. Therefore the only ingredient that may possibly make the V-cycle competitive is the right choice of the projectors. Since most of the effective preconditioners for this class of systems are support-graph, it is interesting to investigate the class of projection operators that allow preserving the graph structure — and therefore using support-graph preconditioners — at all levels of the MG approach.

# Chapter 2

# Graph operators theory

In this chapter we study the conditions under which projection operators preserve the graph structure of the matrix. We start with some preliminary definitions.

**Definition 2.1** *An operator $R$ is called a* graph *operator if, given any incidence matrix $E$,*

$$RE = E'\Theta' \ , \tag{2.1}$$

*where $E'$ is an incidence matrix and $\Theta'$ is an invertible diagonal matrix.*

According to (1.2), one has

$$RE\Theta E^T R^T \ = \ E'\Theta'\Theta\Theta' E'^T \ = \ E'\tilde{\Theta}E' \ ,$$

where $\tilde{\Theta} = \Theta'\Theta\Theta'$ is a positive diagonal matrix if $\Theta$ is (we do not need the diagonal entries of $\Theta'$ to be positive). Consequently it is evident that if the restriction $R^i_{i+1}$ in the multigrid method is a graph operator, then it preserves the graph structure at the lower levels, in the sense that the projected problem is still a (smaller) weighted Laplacian, thus allowing an efficient recursive strategy in the MG solver. A weaker notion is:

**Definition 2.2** *An operator $R$ is called a* graph operator for a given matrix *$E$ if (2.1) holds for $E$, although it may not hold for all possible incidence matrices.*

**Definition 2.3** *An operator $R$ is called* admissible *if it does not have any column with all zero elements and any row with all equal elements.*

The rationale for this definition is that an all-zero column means that a node is "ignored", so that the projection ($r_i = (R^i_{i+1})^T r_{i+1}$) cannot produce any correction of the error. Symmetrically, a row with all equal elements in $R$ means that $RE$ has an all-zero row, i.e. $E'$ in (2.1) has an isolated node. Furthermore note that AMG conditions impose that $R$ has to be of full rank, so that in any case at most only a unique row could have all equal elements. We remark that Definition 2.3 implies that $\mathrm{nnz}(R) \geq n$, where $\mathrm{nnz}(\cdot)$ denotes the number of non-zero elements. This leads to the following refinement of the concept.

**Definition 2.4** *An admissible graph operator $R$ such that* $\mathrm{nnz}(R) = n$ *is called* minimum.

Of course, any admissible operator with $\mathrm{nnz}(R) > n$ is *non-minimum*, and the set of graph operators is partitioned between the two subsets. Similar definition can be given for graph operators w.r.t. a specific matrix $E$.

## 2.1 Theoretical results

We now work our way towards a characterization of the set of graph operators. In this section we will stick to the following notation: $A^{[k]}$ is the $k$-th column of a generic matrix $A$, $E$ is the $n \times m$ node-arc incidence matrix of the underlying *connected* graph $\mathcal{G}$ (see Chapter 1), for each $k = 1, \ldots, m$ the tail and head nodes of the corresponding arc (the row indices corresponding to 1 and $-1$ in $E^{[k]}$) are $i_k$ and $j_k$ respectively, $R$ is a $n' \times n$ operator with $n' < n$ (usually $n' \approx n/2$) so that $F = RE$ is a $n' \times m$ matrix, $e' = e_{(n')}$ is the all-ones vector of length $n'$.

**Definition 2.5** *$R$ is a* constant column sums *(CCS) operator if the sum of elements of every column is constant, i.e. there exists $\rho \in \mathbb{R}$ such that $(e')^T R = \rho e^T$.*

**Definition 2.6** *$R$ is a* zero column sum *(ZCS) operator for $E$ if $(e')^T F = (e')^T RE = 0^T$ (i.e. $R$ conserves the property of $E$ that all columns have zero sum). $R$ is a* ZCS operator *if it is a ZCS operator for all graph matrices $E$.*

**Theorem 2.7** *Let $E$ be an incidence matrix. $R$ is a ZCS operator for $E$ if and only if $R$ is a CCS operator.*

**Proof.** [$\Rightarrow$] We want to prove that $r^T = (e')^T R = \rho e^T$ for some $\rho \in \mathbb{R}$. Suppose otherwise and let $i$, $j$ be two indices such that $r_i \neq r_j$. Because $\mathcal{G}$ is connected, there exists at least one path in $\mathcal{G}$ having the nodes $i$ and $j$ as endpoints. It cannot be that $r_p = r_q$ for every arc $k = (p, q)$ belonging to the path, as this would imply $r_i = r_j$. Hence there must be $k = (p, q)$ such that $r_p \neq r_q$, which leads to

$$0 = (e')^T F^{[k]} = (e')^T (RE)^{[k]} = r^T E^{[k]} = r_p - r_q \neq 0$$

that is a contradiction.

[$\Leftarrow$] For every CCS operator $R$ and for every incidence matrix $E$ we have $(e')^T F = (e')^T RE = \rho e^T E = 0^T$. ■

**Corollary 2.8** *If $R$ is a ZCS operator for $E$ then it is a ZCS operator.*

**Proof.** Since $R$ is a ZCS operator for $E$, by Theorem 2.7 $R$ is also a CCS operator. Now, by the second point of the previous proof, it follows that $R$ is a ZCS operator. ■

**Remark 2.9** *Corollary 2.8 implies that the two forms in Definition 2.6 are equivalent.*

**Corollary 2.10** *Let $E$ be an incidence matrix. Any graph operator for $E$ is a CCS operator.*

**Proof.** Since $R$ is a graph operator for $E$, we have (cf. §1.2)

$$(e')^T F \;=\; (e')^T RE \;=\; (e')^T E' \Theta'^T \;\;,$$

i.e. $R$ is a ZCS operator for $E$. Hence by Theorem 2.7 it is a CCS operator. ■

**Corollary 2.11** *Let $R$ be a minimum operator. Then, up to a scalar factor $\rho$, $R$ is a binary matrix, i.e. $R \in \{0, 1\}^{n' \times n}$.*

**Proof.** By Definition 2.4 every column of the graph operator $R$ only has one non-zero element and by Corollary 2.10 $(e')^T R = \rho e^T$, hence every non-zero entry must be equal to $\rho$. ■

**Definition 2.12** *$R$ is a* difference operator *for a set $\mathcal{I} = \{(i_1, j_1), (i_2, j_2), \ldots\}$ of pairs of node indices if for every $(i_k, j_k) \in \mathcal{I}$ the vector $R^{[i_k]} - R^{[j_k]}$ has either* 0 *or* 2 *non-zero elements.*

**Lemma 2.13** *For each $k = 1, \ldots, m$, $F^{[k]} = (RE)^{[k]} = R^{[i_k]} - R^{[j_k]}$.*

**Proof.** Immediate by the definition of $E$. ■

**Theorem 2.14** *Let $R$ be a CCS operator. $R$ is a graph operator for $E$ if and only if $R$ is a difference operator for the set $\mathcal{I} = \mathcal{V}$ of the arcs of $\mathcal{G}$.*

**Proof.** [$\Rightarrow$] Assume by contradiction that $R$ is not a difference operator for $\mathcal{I} = \mathcal{V}$. There exists an arc $k = (i_k, j_k) \in \mathcal{V}$ such that $R^{[i_k]} - R^{[j_k]}$ has neither 0 nor to 2 non-zero elements. This by Lemma 2.13 implies that the same holds for $F^{[k]} = (RE)^{[k]}$, therefore $R$ is not a graph operator, since (2.1) cannot hold.
[$\Leftarrow$] If $R$ is a difference operator for $\mathcal{V}$, then for each $k = (i_k, j_k) \in \mathcal{V}$ the column $F^{[k]} = R^{[i_k]} - R^{[j_k]}$ has either 0 or 2 non-zeroes. Assume the latter and let $p$ and $q$ be the row indices of the non-zeroes in $F^{[k]}$. Since $R$ is a CCS operator, by Theorem 2.7 we have

$$0 = (e')^T F^{[k]} = F_{pk} + F_{qk} \Rightarrow F_{pk} = -F_{qk} \ .$$

We can then scale $F^{[k]}$ by using the $k$-th diagonal entry of $\Theta'$, finally yielding $F = E'\Theta'$ where $E'$ is an incidence matrix (possibly with "empty arcs"). Therefore $R$ is a graph operator. ■

**Remark 2.15** *It is easy to verify that minimum operators are CCS operators and difference operators for any set $\mathcal{I}$.*

Finally we prove a somewhat surprising result about admissible graph operators.

**Definition 2.16** *$C \in \mathbb{R}^{n' \times n}$ is called a* row matrix *if each of its rows is a scalar multiple of $e^T$, i.e. $C = ce^T$ for some $c \in \mathbb{R}^{n'}$.*

**Lemma 2.17** *Let $A \in \mathbb{R}^{n' \times n}$, $s \in \mathbb{R}^{n'}$ and $Q \in \mathbb{R}^{n' \times n}$ defined as*

$$Q_{ij} = \begin{cases} 0 & \text{if } A_{ij} = s_i \\ 1 & \text{otherwise} \end{cases} .$$

*Then for any pair $(i, j)$ the number of non-zero elements of $A^{[i]} - A^{[j]}$ is equal to the number of non-zero elements of $Q^{[i]} - Q^{[j]}$ plus the number of indices $k$ such that $A_{ki} \neq A_{kj}$ and $Q_{ki} = Q_{kj} = 1$.*

**Proof.** For any $k$, denote $\alpha_k = A_{ki} - s_k$ and $\beta_k = A_{kj} - s_k$. If $Q_{ki} \neq Q_{kj}$, it means that $Q_{ki} = 1$ and $Q_{kj} = 0$ or vice-versa; hence $\alpha_k \neq 0$ and $\beta_k = 0$ or vice versa. Therefore $A_{ki} - A_{kj} = (A_{ki} - s_k) - (A_{kj} - s_k) = \alpha_k - \beta_k \neq 0$. When $Q_{ki} = Q_{kj} = 0$, one has $\alpha_k = \beta_k = 0$, and therefore $A_{ki} - A_{kj} = 0$. In the only remaining case $Q_{ki} = Q_{kj} = 1$ one directly counts whether $A_{ki} \neq A_{kj}$ or not, hence the proof is finished. ■

**Theorem 2.18** *If $R$ is an admissible graph operator and $n' > 2$ then*

$$R = M + C \ ,$$

*where $C$ is a row matrix and $M$ is a minimum operator.*

**Proof.** Since $R$ is admissible, after a proper reordering of rows and columns of $R$, we have $R = [R_1 \mid R_2]$ where $R_1$ is a $n' \times n'$ matrix such that $R_{ii} \neq 0$ for all $i$. For minimum operators, without loss of generality, $R_1 = \rho \cdot I$, where $\rho$ is a scalar factor and $R_2$ contains only copies of some columns of $R_1$.

We arbitrarily select one column of $R$ — say $R^{[t]}$ — as $s$ and we apply Lemma 2.17 to $R$. Since $R$ is admissible, there cannot be an all-zeroes row in $Q$, because $R$ does not have any row with all equal elements. Since $R$ is also a graph operator, (2.1) holds for *any graph* $\mathcal{G}$. By Theorem 2.14 $R$ is therefore a difference operator for *any set* $\mathcal{I}$, hence in particular for $\mathcal{I} = \{(1,t),(2,t),\ldots,(n,t)\}$ (the star tree rooted at $t$). By the definition of difference operator, in each column of $Q$ there are either 0 or 2 ones. Moreover, for any pair $Q^{[i]}$ and $Q^{[j]}$ of non-zero columns at least one of the two non-zeroes must be in the same row. In fact, assume by contradiction that $Q^{[i]}$ and $Q^{[j]}$ have all their non-zeroes in different positions: this means that there exists for $Q$ a difference vector $Q^{[i]} - Q^{[j]}$ with 4 non-zero elements and by Lemma 2.17 the same holds for $R$, since there is no index $k$ such that $Q_{ki} = Q_{kj} = 1$.

These two facts imply that we can reorder the rows and columns of $Q$ (which corresponds to reordering $R$) by putting the column $t$ as the first one and all the others appropriately in such a way that $Q = [Q_1 \mid Q_2]$, where $Q_1$ is the following $n' \times n'$ matrix

$$Q_1 = \begin{pmatrix} 0 & 1 & 1 & \cdots & 1 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

and $Q_2$ contains only copies of some columns of $Q_1$. Now assume that $R$ has in fact been reordered (if necessary) to match $Q$, consider the first row in (the reordered) $R$ and pick any two column indices $h \neq j$ such that $Q^{[h]}$ and $Q^{[j]}$ are not all-zero; hence $Q_{1h} = Q_{1j} = 1$ (remind that the columns in $Q_2$ are copies of these in $Q_1$, hence all the non-zero ones have a 1 in the first row). We claim that $R_{1h} = R_{1j}$. We start with the case where the other two non-zeroes are in different rows, i.e. $Q_{ph} = 1$ and $Q_{qj} = 1$ for $1 < p \neq q > 1$: by Lemma 2.17 we have a difference vector for $R$ with 2 or

3 non-zero elements, depending on whether $R_{1h} = R_{1j}$ or not, which proves our claim since $R$ is a difference operator. The case when $p = q$ easily follows by transitivity considering one further column $h$ (which must exists) that has its other non-zero element in a different row: $R_{1i} = R_{1h}$ and $R_{1j} = R_{1h}$, so $R_{1i} = R_{1j}$. Note that the hypothesis $n' > 2$ is crucial in this part of the proof.

Consider now any row $i > 1$. Needless to say, all entries $R_{ij}$ such that $Q_{ij} = 0$ have the same value (that of $R_{ik}$). Therefore we know that $R$ has an arrangement as $R = [\ R_1 \mid R_2\ ]$, where

$$R_1 = \begin{pmatrix} r_1 & c_1 & c_1 & \cdots & c_1 \\ c_2 & r_2 & c_2 & \cdots & c_2 \\ c_3 & c_3 & r_3 & \cdots & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n'} & c_{n'} & c_{n'} & \cdots & r_{n'} \end{pmatrix}$$

and each column $R^{[j]}$ of $R_2$ has the form $[c_1, c_2, c_3, \ldots, c_{n'}]^T$ except (possibly) for one unique element, $R_{ij} = r_j \neq c_i$, at the unique row $i > 1$ (if any) such that $Q_{ij} = 1$. Consider any two column indices $h \neq j$ such that $Q_{ih} = Q_{ij} = 1$: we claim that, again, $R_{ih} = R_{ij}$, i.e. that all columns of $R_2$ are copies of some column of $R_1$ (the one having the non-zeroes in $Q$ in the same position). This comes from the fact that, being a graph operator, $R$ is also a CCS operator: $(e')^T R^{[j]} = (e')^T R^{[h]} = \rho$. Then one has

$$\rho = (e')^T R^{[j]} = \sum_{p \neq i} c_p + r_j \qquad \text{and} \qquad \rho = (e')^T R^{[h]} = \sum_{p \neq i} c_p + r_h$$

whence $R_{ih} = r_h = \rho - \sum_{p \neq i} c_p = r_j = R_{ij}$. Again from the fact that $R$ is a CCS operator, we have

$$\rho = (e')^T R^{[1]} = r_1 + c_2 + \sum_{p>2} c_p = (e')^T R^{[2]} = c_1 + r_2 + \sum_{p>2} c_p$$

and similarly for all pairs $(i, i+1)$ with $i < n'$, whence $r_1 - c_1 = r_2 - c_2 = \ldots = r_{n'} - c_{n'}$. Let us call the common value $r_i - c_i = \alpha$: it must be $\alpha \neq 0$, for otherwise one would have $r_i = c_i$ for all $i$, i.e. $R = C = ce^T$, contradicting the hypothesis that $R$ is admissible (*each* row would have all equal elements). Hence $M = R - C \neq 0$. In particular $M$ has exactly one non-zero per column, no all-zero rows, and all its non-zeroes have the same value $\alpha$. Thus $M$ clearly is a minimum operator (cf. Corollary 2.11). ■

**Remark 2.19** *The action on an incidence matrix $E$ of any graph operator $R$ and of the related minimum operator $M$ (see Theorem 2.18) is the same. In fact, $F = RE = ME + CE = ME$.*

**Remark 2.20** *The hypothesis $n' > 2$ in Theorem 2.18 is necessary, as for $n' = 2$ the thesis does not hold. In fact, the conditions which have to be satisfied are:*

*i.* $R_{1j} + R_{2j} = \rho$ *for all* $j$,

*ii.* $R_{1j} = R_{1k}$ *if and only if* $R_{2j} = R_{2k}$ *for all* $j$ *and* $k$,

*and these do not prevent from choosing all different* $R_{ij}$, *so in general* $R \neq M + C$. *An easy counterexample with* $m = 5$ *is for instance*

$$R = \begin{pmatrix} 0.1 & 0.8 & -1 & 1 & 0.25 \\ 0.9 & 0.2 & 2 & 0 & 0.75 \end{pmatrix} .$$

**Corollary 2.21** *Let $R$ be an admissible graph operator. If $R$ is a binary matrix, then either $R = M$ or $R = e'e^T - M$ where $M$ is a minimum operator.*

**Proof.** By Theorem 2.18 $R = M + C$. By Corollary 2.11 $M$ is a binary matrix, up to a scalar factor $\alpha$. Hence either $C = 0$ and $R = M$ or $C$ is the matrix with all entries equal to 1 and $R = C - M$ is the complement of the minimum operator $M$ (the one having 0 where $M$ has 1 and vice-versa). ■

We finish this section by noting that if the original graph $\mathcal{G}$ is connected (as is the case in our applications), then so is the "restricted" graph $\mathcal{G}'$.

**Theorem 2.22** *Let $R$ be an admissible graph operator for $E(\mathcal{G})$. Then $\mathcal{G}'$ is a connected graph.*

**Proof.** By Remark 2.19 $F = RE = ME$, hence $R$ acts on $E$ as a minimum operator; these aggregate nodes, which can be adjacent (cf. Definition 3.1) or not. Clearly contracting nodes in an already connected graph produces a connected graph: each path in $\mathcal{G}$ corresponds to a (possibly non-simple) path in $\mathcal{G}'$. In other words, connectivity in $\mathcal{G}'$ is the same as connectivity in the graph obtained by adding to $\mathcal{G}$ arcs forming cliques within the aggregated nodes. ■

### 2.1.1 A note on the FWO operator

The above results help heuristically in understanding why the full weighting operator is a good choice for Poisson problems: basically it is a graph operator for the corresponding very special graph matrix. Recalling its algebraic expression in (1.7), omitting the constant 1/4, and leaving out the first and the last column, one immediately notes that the FWO is a CCS operator with constant 2. The fundamental observation now is that the matrix associated with the Poisson problem has the $EE^T$ form with the elimination of the first and the last row and the first and the last column:

$$(L_{\text{Poisson}})_n = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}_{n\times n} = P_n E_n E_n^T P_n^T \ ,$$

$$\text{where} \qquad E_n = \begin{pmatrix} 1 & & & & \\ -1 & 1 & & & \\ & -1 & \ddots & & \\ & & \ddots & 1 & \\ & & & -1 & 1 \\ & & & & -1 \end{pmatrix}_{(n+2)\times(n+1)} ,$$

$$P_n = \left(\ 0 \mid I_n \mid 0\ \right)_{n\times(n+2)} \ .$$

The incidence matrix $E$ is associated to the linear graph with $n+2$ nodes, in which every node points to the next one, as depicted in Figure 2.1.

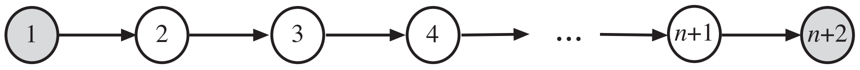


Figure 2.1: Linear graph with $n+2$ nodes. Nodes colored in grey are related to the Dirichlet boundary conditions.

Indeed, $R_{\text{FWO}}$ — which is clearly an admissible operator — is a difference operator for the set $\mathcal{I} = \{(2,3),\ (3,4),\ \ldots,\ (n-2,n-1)\}$, as the difference vector is

$$\begin{pmatrix} 2 \\ \\ \\ \end{pmatrix} - \begin{pmatrix} \\ 1 \\ 1 \\ \\ \end{pmatrix} = \begin{pmatrix} \\ 1 \\ -1 \\ \\ \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} \\ 1 \\ 1 \\ \\ \end{pmatrix} - \begin{pmatrix} \\ \\ \\ 2 \end{pmatrix} = \begin{pmatrix} \\ 1 \\ -1 \\ \\ \end{pmatrix} .$$

The exceptions represented by the first and the last column, in which the difference vector has one non-zero element, do not contradict our results, as they can be explained by the elimination of rows and columns related to the Dirichlet boundary conditions. By Theorem 2.14 we conclude that $R_{\text{FWO}}$ is a graph operator for the matrix $E$ of Poisson problems (cf. Definition 2.2), although not a graph operator in general. Moreover, by Theorem 2.22 also the graph associated to $E' = E_{n'}$ is connected; better yet, it is still a linear graph. In fact,

$$R_{\text{FWO}}(L_{\text{Poisson}})_n P_{\text{FWO}} \;=\; \frac{1}{4}(L_{\text{Poisson}})_{n'} \;=\; \frac{1}{4}P_{n'}E_{n'}E_{n'}^T P_{n'}^T$$

where $P_{\text{FWO}} = 2(R_{\text{FWO}})^T$ is the operator of linear interpolation, $E_{n'}$ is the incidence matrix associated to the linear graph with $n' + 2$ nodes, and $P_{n'}$ is defined as above. Thus the FWO projection preserves the Poisson structure and so it is not surprising that it does work well in the Poisson case. However for general graphs, the structure of weighted Laplacian is not conserved at all by the FWO projection and in fact the related multigrid performances are not good (see the discussion in §1.4). Our heuristic indication is to consider only projections that preserve the weighted Laplacian structure.

## 2.2 Minimum operators

Summarizing what we have done, in the previous section we have provided a characterization of the set of minimum graph operators along the following lines:

a) any minimum operator is a binary matrix up to a scalar factor;

b) for any admissible graph operator $R$, $R = M + C$ where $C$ is a row matrix (each of its rows is composed by all identical numbers) and $M$ is a minimum operator;

c) for any admissible graph operator $R$ that is also a binary matrix, then either $R = M$ or $R = U - M$ where $M$ is a minimum operator and $U$ is the all-ones matrix.

Actually the last two results only hold if $R$ has more than two rows, but this is clearly not an issue in practice. What these results say is that, basically, any admissible graph operator that is a binary matrix is either a minimum operator or the "complement" of a minimum operator. In turn all minimum operators are binary matrices up to a(n irrelevant) scalar factor, and therefore restricting to Aggregation Operators (1.8) appears to do very little harm. Even more so when one considers that, by the above results, the action on an incidence matrix $E$ of any graph operator $R$ and of the related minimum operator $M$ is the same.

As discussed in this chapter, the set of non-minimum operators, as a subset of admissible graph operators, is not empty, although it just contains row matrix "perturbations" of minimum operators. One may then wonder whether non-minimum operators could be preferable to minimum ones. In practice, we did not identify any promising way to construct non-minimum operators. All our attempts with non-minimum operators invariably gave either equivalent or worse results than these with minimum ones. Therefore in the following we will concentrate on MG approaches which use minimum operators only. These operators pick some subset of nodes and "shrink" them together into a super-node, which inherits all the incident arcs of the original ones, while arcs between any of the nodes shrank together disappear. Obviously a fundamental question has now to be answered, i.e. how to select the subset of nodes to be shrank.

There are many possible ways to approach this question. For instance, one possibly meaningful observation is that if $\mathcal{G}$ were a tree, then (1.2) could be solved in $O(n)$ [2]. More in general, if $\mathcal{G}$ were a chordal-type graph then (1.2) could be solved in $O(m)$ [58, 59] (this is the idea subgraph-based preconditioners are based upon). A promising approach may be that of choosing the aggregation in such a way that after a few levels the "shrank" graph is chordal, so as to pass as quickly as possible to the direct solution step. On the other hand, it is not clear how effective these aggregations may be in terms of reducing the overall number of MG iterations. In the following we will describe some classes of minimum operators that we have devised and tested to start shedding some light on this intricate issue. For each of them we describe the restriction operator $R \in \{0,1\}^{n' \times n}$, where $n'$ is a fraction of $n$ ($n$ actually is $n_k$, i.e. the matrix dimension at the $k$-th level). The value $d = n - n'$ is called *descent parameter* and usually $d \approx n' \approx n/2$.

We now present a first list of simple aggregation projectors. We call them *oblivious* operators, as their form is independent of the matrix to which they are applied. Oblivious operators are attractive from the computational viewpoint because they can be computed a-priori, therefore they are as cheap as they possibly can.

- The First Operator aggregates the first $d + 1$ nodes into one:

$$R_{\text{First}} = \begin{pmatrix} 1 & 1 & \cdots & 1 & 1 & & & \\ & & & & & 1 & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{pmatrix} .$$

- The Last Operator aggregates the last $d+1$ nodes into one:

$$R_{\text{Last}} = \begin{pmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & & & & \\ & & & 1 & 1 & \cdots & 1 & 1 \end{pmatrix} .$$

- The Extreme Operator aggregates the first and the last $(d+1)/2$ nodes into one:

$$R_{\text{Extreme}} = \begin{pmatrix} 1 & 1 & 1 & & & & 1 & 1 \\ & & & 1 & & & & \\ & & & & \ddots & & & \\ & & & & & 1 & & \end{pmatrix} .$$

- The Medium Operator aggregates the $d+1$ "central" nodes into one:

$$R_{\text{Medium}} = \begin{pmatrix} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & 1 & \cdots & 1 & & \\ & & & & & & \ddots & \\ & & & & & & & 1 \end{pmatrix} .$$

- The Random Operator is iteratively obtained by randomly selecting a set $\mathcal{R}$ of pairs of nodes (with $|\mathcal{R}| = d$) and aggregating them; formally, $R_{\text{Random}} = \prod_{(i,j)\in\mathcal{R}} R(i,j)$ where

$$R(i,j) = \begin{pmatrix} 1 & & i & & & j & & \\ & 1 & \downarrow & & & \downarrow & & \\ & & 1 & & & 1 & & \\ & & & \ddots & & & & \\ & & & & 1 & & & \\ & & & & & & 1 & \\ & & & & & & & 1 \end{pmatrix}$$

is the Pair Operator w.r.t. the pair $(i,j)$.

- The Couple Operator aggregates every node with the following one:

$$R_{\text{Couple}} = \begin{pmatrix} 1 & 1 & & & & & & \\ & & 1 & 1 & & & & \\ & & & & \ddots & \ddots & & \\ & & & & & & 1 & 1 \end{pmatrix} ;$$

in other words, it is the product of $\approx n/2$ pair operators corresponding to pairs $(1,2)$, $(3,4)$, $\ldots$.

Furthermore it is possible to alternate two or more of these operators along MGM levels. Clearly the effectiveness of these operators is strongly influenced by the structure of the underlying graph $\mathcal{G}$ and the relative performances of different oblivious projectors vary wildly as the underlying problem changes. So it is difficult to choose any fixed oblivious projector whose performances are predictable and stable enough on a large class of instances. This suggests that it may be necessary to adapt the projector to the topological structure of the graph and maybe to the weights of the arcs too. Therefore we have to investigate non-oblivious operators, despite the fact that they can more costly to determine. This is precisely what the next chapter is devoted to.

# Chapter 3

# Projection operators

As we have just seen, many different minimal operators can be constructed by iterating the simplest possible form of aggregation, that is the one given by the *Pair Operator*

$$R(i,j) = \begin{pmatrix} 1 & & i & & j & & \\ & 1 & \downarrow & & \downarrow & & \\ & & 1 & & 1 & & \\ & & & \ddots & & & \\ & & & & 1 & & \\ & & & & & 1 & \\ & & & & & & 1 \end{pmatrix}$$

w.r.t. the pair $(i,j)$; in other words, by selecting a set $\mathcal{R}$ of pairs of nodes (with $|\mathcal{R}| = d$) and defining $R_{\mathcal{R}} = \prod_{(i,j)\in\mathcal{R}} R(i,j)$. For instance one can define oblivious operators that are independent of the graph they are applied to, such as the Couple Operator aggregating successive nodes ($\mathcal{R} = \{(1,2),(3,4),\ldots\}$) or the Random Operator where $\mathcal{R}$ is randomly chosen. Oblivious operators are attractive from the computational viewpoint because they can be computed a-priori, therefore they do not require any setup (precomputing) phase . However their effectiveness is strongly influenced by the structure of the underlying graph $\mathcal{G}$: in order to construct "robust" operators, *adaptive* choices must be considered which depend on the values of the matrix at hand (topological structure of the graph and weights of the arcs too) despite the fact that they can be more costly to determine, and therefore that a non-trivial trade-off between increase of the performances and computational cost have to be struck. In choosing the rules to determine the elements of $\mathcal{R}$, it is likely a wise choice to preserve as much as possible the topological structure of the graph.

**Definition 3.1** *A graph operator $R$ is a* contraction operator *for $E$ if it aggregates exclusively adjacent nodes in $\mathcal{G}$.*

**Remark 3.2** *$R_{\mathrm{FWO}}$ is a contraction operator for the matrix $E$ of Poisson problem.*

Contractor operators are intuitively attractive, since they do not "mess up" with the structure of the graph. Indeed we tested a simple but effective scheme for choosing the pairs $(i, j)$ to aggregate:

- select $i$ either at random or as the diagonal entry of $L$ with the least (or the greatest) value;
- select $j$ either at random or as the index of the non-zero off-diagonal entry in the $i$-th column with the least (or the greatest) *absolute* value (since $L_{ij} = -\Theta_{ij} < 0$).

Note that, when the process is repeated, the (iteratively) aggregated matrix $R(i, j)LR(i, j)^T$ is looked at instead of the original $L$. This gives rise to nine "$\{x, y\}$" projectors with $x$ and $y$ chosen between "rand", "min" and "max". Our preliminary results showed that $\{x, \max\}$ projectors substantially outperform the corresponding variants where $j$ is selected either at random or by looking at arcs with small weight. Thus in our experience using contractor operators turns out to be better then aggregating non-adjacent nodes. Furthermore, as with subgraph-based preconditioners, choosing arcs with large weight for $\mathcal{R}$ appears to be the best option.

It is clear that different, and possibly more sophisticated, choices of $R$ may exist. For instance, every possible pair $(i, j)$ actually defines the $2 \times 2$ minor

$$L^{ij} = \begin{bmatrix} L_{ii} & L_{ij} \\ L_{ji} & L_{jj} \end{bmatrix} \quad ,$$

where $L_{ii} > 0$, $L_{jj} > 0$, $L_{ij} = L_{ji} < 0$ and $\det(L^{ij})$ is positive. Because weak dominance for rows holds, that is $\sum_{j \neq i} |L_{ij}| \leq L_{ii}$ with strict inequality at least for one index (if $R$ is a graph operator, then this is true at every MGM level), it follows that $|L_{ij}| \leq \min(L_{ii}, L_{jj})$. The choice of $j$ of $\{x, \max\}$ operators implies that, if we suppose that $\min(L_{ii}, L_{jj}) = L_{jj}$, then $L_{ij} \approx L_{jj}$ and

$$\det(L^{ij}) = L_{ii}L_{jj} - L_{ij}^2 \approx (L_{ii} - L_{jj})L_{jj} \quad ,$$

i.e. if $L_{ii} \approx L_{jj}$ then one aggregates a badly conditioned part, whereas if $L_{ii} \gg L_{jj}$ then one aggregates a nicely conditioned part. Hence one may use quantities related to $\det(L^{ij})$, such as "normalized" versions like

$\det(L^{ij})/(L_{ii}^2 + L_{jj}^2)$ or $\det(L^{ij})/(L_{ii}L_{jj})$, which measures how well or badly conditioned the $2 \times 2$ minor is, to gauge how promising a $(i, j)$ pair is. This gives rise to max-minor or min-minor operators, depending on whether one chooses to preferentially aggregate well-conditioned or ill-conditioned minors.

Our experience with these two variants is that aggregating badly conditioned minors is by far the most effective variant. The choice of the $(i, j)$ pair is done as in the previous case: first $i$ is selected with either one of the three above strategies, then $j$ is selected in $O(n)$ ($O(1)$ for sparse graphs) as the one giving the most ill-conditioned minor. Note that it would be possible to determine the "overall best" minor operator by looking at all arcs in $\mathcal{G}$, but that would have a $O(m)$ cost. The $\{x, \text{min-minor}\}$ operator seems to somewhat improve upon the previous $\{x, \max\}$ ones. In particular its performances seems to be even less dependent by the choice of $i$, so the average behaviour is somewhat better. This is especially true when treating extremely ill-conditioned problems, which therefore makes it our iterative pairwise minimal projector of choice. In the following we will refer to these projection techniques respectively as *Max* and *Minor* aggregation.

## 3.1 Relationships with preconditioning

As anticipated in §1.3, PCG with subgraph-based preconditioners is the most promising class of approaches to serve as effective pre- and post-smoothers in MG methods. Thus, at least by restricting to *contraction operators* $R$, one actually have to choose *two subsets of arcs* at each level of the MGM: $\mathcal{S}$ for the preconditioner, $\mathcal{R}$ for the projector.

Clearly the two choices are not entirely independent. In particular $\mathcal{R}$ at one level influences the graph and therefore possibly $\mathcal{S}$, at the next. We therefore aim at exploring more in detail the relationships between the two choices. In order to do this, we now present and analyse two possible preconditioning techniques which takes into account the effect of projection. In the following, we denote by $S = E_{\mathcal{S}}\Theta_{\mathcal{S}}E_{\mathcal{S}}^T$ the subgraph-based preconditioner. We assume $S$ positive definite and in fact "easy" to invert, which requires specific care in the choice of $\mathcal{S}$ [58, 59].

### 3.1.1 Inverse projection

The preconditioned matrix of PCG at the first level of the MG approach is $P = S^{-1}L$. At the $k$-th level, *inverse projection* uses the preconditioned matrix

$$P_k = (R_k R_k^T)^{-1} R_k S^{-1} R_k^T (R_k R_k^T)^{-1} R_k L R_k^T \quad , \tag{3.1}$$

where $R_k$ is the cumulative restriction operator after $k$ levels. The analysis of inverse projection requires the following two lemmas, whose proofs are based on continuity and density arguments of invertible matrices in the space of all matrices (see e.g. [14]).

**Lemma 3.3** *For all $A, B \in \mathbb{C}^{n\times n}$ the characteristic polynomials of $AB$ and $BA$ coincide and therefore $\sigma(AB) = \sigma(BA)$.*

**Lemma 3.4** *Let $A \in \mathbb{C}^{m\times n}$ and $B \in \mathbb{C}^{n\times m}$ with $m > n$, then for the characteristic polynomials of $AB$ and $BA$ one has $p_{AB}(\lambda) = p_{BA}(\lambda)\cdot\lambda^{m-n}$ and therefore $\sigma(AB) = \sigma(BA) \cup \{0, \ldots, 0\}$ (repeated $m - n$ times).*

**Theorem 3.5** *If $\sigma(P) \subset [a, b]$ then $\sigma(P_k) \subset [a_k, b_k]$ with $a_k > a$.*

**Proof.** We denote by $(a \leq)\ \alpha_1 \leq \ldots \leq \alpha_n\ (\leq b)$ the eigenvalues of $P$ and with $\beta_1 \leq \ldots \leq \beta_{n_k}$ those of $P_k$. We single out $Q_k = R_k^T(R_kR_k^T)^{-1}R_k$ from (3.1). Thanks to Lemma 3.4, $Q_k$ has only 0 and 1 eigenvalues (as the non-zero ones are the same of $R_kR_k^T(R_kR_k^T)^{-1} = I$). Again by Lemma 3.4, $P_k$ has essentially the same spectrum as $S^{-1}Q_kLQ_k$, which in turn is similar to $S^{-1/2}Q_kLQ_kS^{-1/2}$. In other words, its eigenvalues are

$$0 = \lambda_1 = \lambda_2 = \ldots = \lambda_{n-n_k} < \lambda_{n-n_k+1} \leq \ldots \leq \lambda_n$$

where $\lambda_{n-i} = \beta_{n_k-i}$ for $i = 0, \ldots, n_k - 1$. We can now apply the MinMax Theorem to "discover" the first non-zero eigenvalue ($\lambda_{n-n_k+1} = \beta_1$):

$$\begin{aligned}\lambda_{n-n_k+1} &= \min_{\dim(U)=n-n_k+1}\ \max_{u\in U}\frac{u^TS^{-1/2}Q_kLQ_kS^{-1/2}u}{u^Tu}\\ &= \min_{\dim(V)=n-n_k+1}\ \max_{v\in V}\frac{v^TQ_kLQ_kv}{v^TSv} \qquad \text{, where} \quad v = S^{-1/2}u\ .\end{aligned}$$

Now let $X$ be the $n_k$-dimensional space generated by columns of $R_k^T$. For every $x \in X$ we have $Q_kx = [R_k^T(R_kR_k^T)^{-1}R_k]R_k^T\tilde{x} = R_k^T\tilde{x} = x$. On the other hand, for every $x \in X^\perp$ we have $Q_kx = 0$. Since $V$ is a $(n - n_k + 1)$-dimensional space and $X$ is a $n_k$-dimensional one, their intersection $Z_V = V \cap X$ must have dimension at least 1. Therefore, we conclude

$$\lambda_{n-n_k+1} \geq \min_{\dim(V)=n-n_k+1}\max_{z\in Z_V}\frac{z^TQ_kLQ_kz}{z^TSz} = \min_{\dim(V)=n-n_k+1}\max_{z\in Z_V}\frac{z^TLz}{z^TSz} \geq \alpha_1 \geq a\ .$$

■

Had the result been $[a_k, b_k] \subset [a, b]$, one would have reached the very significant conclusion that the spectral properties of the preconditioned matrix get better and better as the level increases, so have good hopes that the same holds for the practical behaviour of the PCG. Theorem 3.5 instead only gives indications about the lower extreme of the interval. However this is the most important one for PCG convergence, as well emphasized in [8]. Therefore it should be expected that a good practical behaviour of PCG be observed at all levels of the MG approach, provided that $\mathcal{S}$ has been properly chosen at the first level.

### 3.1.2 Dense projection

A different and somewhat simpler approach is the *dense projection*, that entails the use of

$$S_k = R_k S R_k^T$$

as preconditioner at the $k$-th level. The advantage is that $S_k$ is the subgraph-based preconditioner on the graph at the $k$-th level simply obtained by applying to $\mathcal{S}$ the same aggregations applied to $\mathcal{G}$, so it is very inexpensive to compute. Yet this is dubbed dense since number of non-zero elements can significantly increase with respect to that of the original subgraph $\mathcal{S}$ (chordal subgraphs are typically fairly sparse), thus causing a relevant growth in the cost for inverting $S_k$. It is possible to choose $\mathcal{R}$ appropriately so as to avoid this issue.

**Theorem 3.6** *Let $\mathcal{S}$ be a chordal graph, $(i, j) \in \mathcal{S}$ and $\mathcal{S}'$ obtained by $\mathcal{S}$ by aggregating $i$ and $j$. Then $\mathcal{S}'$ is a chordal graph.*

**Proof.** Basically it all depends on the fact that aggregating an existing arc cannot create any new cycle. Assume by contradiction that a cycle $\mathcal{C}'$ of length $k \geq 4$ exists in $\mathcal{S}'$ which has no *chord* (an arc joining two non-consecutive vertices in $\mathcal{C}'$), thus negating triangularity of $\mathcal{S}'$. Hence $\mathcal{S}'$ and $\mathcal{S}$ are as depicted in Figure 3.1, where continuous lines indicate arcs which are surely present, whereas *at least* one arc of pairs indicated with dotted and dashed lines is present. In plain words, $p$ and $q$ are not adjacent in $\mathcal{S}$, but there exists a path joining them (passing through $i$ or $j$). Let $\mathcal{Y}$ be the subgraph $\mathcal{C}' \setminus \{i'\}$, which is a linear graph of order $k - 1$ belonging both to $\mathcal{S}'$ and $\mathcal{S}$: neither $i$ nor $j$ are adjacent to nodes of $\mathcal{Y} \setminus \{p, q\}$. Therefore there exists in $\mathcal{S}$ a cordless cycle $\mathcal{C} = \mathcal{Y} \cup \{i\}$ (or $\mathcal{Y} \cup \{j\}$) of length $k$. This concludes the proof.

■

Thus, choosing $\mathcal{R} \subset \mathcal{S}$, the triangular structure is preserved and we avoid any fill-in effect. It is clear that such a choice severely restricts the set of available projection operators, possibly unnecessarily so. For instance, it

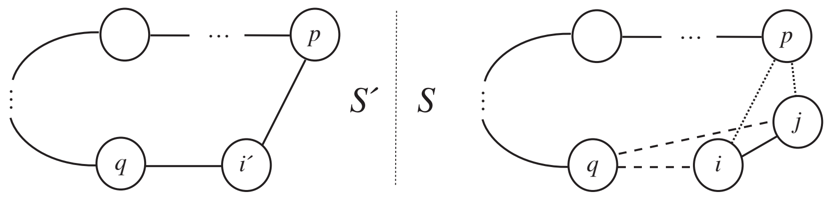


Figure 3.1: Aggregation in a chordal graph preserves the property.

is easy to realize that if $\mathcal{S}$ is a tree, then triangularity is preserved by $R_{ij}$ not only if $(i, j) \in \mathcal{S}$ (joining "a father and a son"), but also if $i$ and $j$ are "brothers", i.e. they have a common adjacent node in $\mathcal{S}$. Allowing to aggregate along arcs in $\mathcal{V} \setminus \mathcal{S}$ may be possible without incurring in fill-in and any improvement in the flexibility of the approach is in principle desirable.

Exploiting arcs "joining brothers" is precisely the idea behind *Brother-Connected Trees*, one of the most effective more-than-tree subgraph-based preconditioners [58, 59]. Roughly speaking, these are obtained by adding this kind of arcs to an existing (or being constructed) spanning tree. In other words, if $\mathcal{S}$ were a tree and $(i, j) \notin \mathcal{S}$ were an arc joining brothers which may be deemed useful to be part of $\mathcal{R}$, then $(i, j)$ could have been added to $\mathcal{S}$ in the first place. It appears that more-than-tree chordal preconditioners may be even more attractive in the context of MG methods (at least when using minimum operators and dense projection) than they are in the context of direct application of PCG, since adding arcs to $\mathcal{S}$ not only improves its preconditioning capabilities, but also improves the set of available choices for the projector. Then again, finding the appropriate balance between the increase in computational cost due to finding and factoring a larger preconditioner and the corresponding decrease in iterations count is already rather delicate in the PCG context, even more so in the MGM one. In our research we have limited ourselves to simpler tree-based preconditioners.

Both inverse projection and dense projection approaches seem to hold promises for finding the right balance between the selection of the preconditioner and that of the projector. Choosing $\mathcal{R}$ and $\mathcal{S}$ so as to attain good convergence results at all levels with an acceptable computational cost remains a significant challenge, which will require further investigation.

## 3.2 Strength-based aggregation operators

Different projection operators have been defined for different but related problems. For instance, a series of papers [39, 40, 41, 42, 43] by De Sterck and his collaborators examine *strength-based operators* in the context of the solution of problems of the form

$$Bx = x \quad , \quad x \geq 0 \quad , \quad ||x||_1 = 1 \quad ,$$

where $B$ a irriducible Markov matrix, which can be rewritten as $Lx = 0$, where $L = I - B$. The techniques proposed in these papers are all based on the concept of *strong dependence*; namely, node $i$ strongly depends on node $j$ if

$$-L_{ij} \geq \theta \max_{k \neq i} -L_{ik}$$

where $\theta \in [0, 1]$ is *strength threshold parameter*. This says that the coefficient $L_{ij}$ is comparable in magnitude to the largest off-diagonal coefficient in the $i$-th equation; in other context, the same concept is described "in reverse" by saying that the node $j$ *strongly influences* the node $i$. Let us immediately remark that this notion is strongly influenced by the choice of the parameter $\theta$ which is far from being trivial, as discussed in the computational section. Taking into consideration the latter, different aggregation strategies can be developed based on this notion.

**Strength aggregation**. The algorithm, presented in [39], is based on the notion that "important states", i.e. nodes $i$ that have a large value in the current iterate $x$, are good candidate "seed points" for new aggregates and that states that are strongly influenced by the seed point of an aggregate are good candidates to join that aggregate. The idea is then simply to choose, among the unassigned nodes, the one which has the largest value in current iterate $x$; a new aggregate is formed, and add all unassigned nodes that are strongly influenced by that seed point are added to the aggregate. This process is repeated until all nodes are assigned to one aggregate. In the original implementation, the algorithm recompute aggregates at every level of every V-cycle. In our experience this did not lead to good results, while a similar approach using $b$ instead of $x$ (and therefore run only once before applying MGM) proved to be competitive.

**Neighborhood aggregation**. This algorithm, proposed in [121] and used in [42], is based on the undirected version of strong influence: the (undirected) pair of nodes $\{i, j\}$ is considered to be *strongly connected* if at least one of the nodes is strongly dependent on the other. This gives rise to the *strong neighborhood* $\mathcal{N}_i$ of node $i$ — the set of all points that are strongly connected to $i$ — and to the two-phase neighborhood-based aggregation algorithm. In the first phase, the algorithm assigns entire neighborhoods to aggregates: for each node $i$, if none of the nodes of $\mathcal{N}_i$ have been assigned yet, then a new aggregate is formed and all the nodes in $\mathcal{N}_i$ are assigned to the aggregate. This phase terminates with at least one aggregate formed, and possibly some non-assigned nodes: then in the second phase each of the remaining nodes is assigned to the aggregate it is “most connected” to, i.e. the one having the largest number of nodes in $\mathcal{N}_i$.

**(Double) Pairwise aggregation**. This pairwise aggregation algorithm, proposed in [95] and used in [43], is based on the “directed version” of strong neighborhoods, i.e. on the sets $\mathcal{D}_i$ of all the nodes $j$ strongly influenced by $i$ (upon which $i$ strongly depends). The algorithm chooses the unassigned node $i$ with minimal cardinality of $\mathcal{D}_i$ and forms a new aggregate containing $i$. Then it looks for the unassigned node $j$ with the strongest negative connection with $i$, i.e. with the minimal value of $L_{ij}$: if $j \in \mathcal{D}_i$, then $j$ is also added to the new aggregate together with $i$, otherwise $i$ is left alone (in a single-node aggregate). The process is repeated, but the assigned nodes ($i$, and possibly $j$) are removed from the sets $\mathcal{D}_h$ they belong to, so that at each step the selected node is the one with the smallest number of *unassigned* strongly influenced nodes. This approach is therefore similar to these proposed by us, except that the number of aggregations is not fixed a-priori but depends on $\theta$ and that all aggregations are performed “in parallel” (on the original matrix). Yet, because the number of aggregations can be too small, the *double* pairwise aggregation is also proposed, whereby the pairwise aggregation algorithm is run once providing a projector $R_1$, then the projected matrix $L_1 = R_1 L R_1^T$ is formed, the pairwise aggregation algorithm is run again on $L_1$ providing a second projector $R_2$, and the final $R$ is just the combination of the two $R = R_2 R_1$. This approach is even more similar to these proposed by us and it is the one we tested in our computational experiments.

It has to be remarked that these techniques have been proposed for, and applied to, methods that are significantly different from the ones we consider. This is made apparent by the fact that the theoretical results are based on the estimate of the *multiplicative error* $e$ of the current iterate $x$, defined as the one which solves $A\text{diag}(x)e = 0$, as well as from the fact the matrix at the lower level is given by

$$RA\text{diag}(x)R^T \quad ,$$

i.e. a scaling of the standard restricted system using the current iterate $x$. This is motivated by the interpretation of the scaled matrix in Markov context: “for a link in the Markov chain from state $i$ to state $j$, state $i$ contributes to the probability of residing in state $j$ in the steady state not just by the size of the transition probability from $i$ to $j$ but by the product of that transition probability and the probability of residing in state $i$” [39]. These modifications to the standard MG approach are not applicable in our setting (at the very least they require that $b = 0$, which is not the case), nor they are any likely to be effective. Indeed [42] reports that "a standard Krylov acceleration technique cannot be applied, because the spaces involved are not related by a fixed preconditioner applied to residual vectors", while according to [43] "in the case of CG or GMRES acceleration of stationary multigrid cycles (our cycles are non-stationary), excellent convergence properties are often obtained because the spaces are nested". In order to obtain an acceleration, ad-hoc techniques are needed such as the minimization of a functional [42] or the solution of a two-dimensional quadratic programming problem and repeated use of W cycles [43]. This is true also for the *smoothed* version of aggregate operators proposed in [40] on the basis of [23], which necessitate several ad-hoc interventions (a "lumping procedure" to keep the M-matrix nature of the coarse-level operators) and may exhibit high memory and execution time complexity. Our preliminary tests confirmed that smoothing applied to projectors almost always results in a substantially slower method. Thus, most of the theoretical results of [39, 40, 41, 42, 43] are largely inapplicable in our context, although the aggregation techniques can be mirrored.

## 3.3 Strength-based AMG operators

We remark that the projectors in the previous §3.2 are basically the "aggregation form" of non-aggregation operators, based on the notion of strong dependence, proposed for the Algebraic Multigrid Method (AMG) in [23, 24, 41, 42, 28]. In the AMG parlance, the projection process is denoted as finding the "$C/F$ splitting", i.e. deciding which equations will remain in the coarsened (restricted) system ($C$) and which ones only belong to the finer (original) system ($F$). Once this is is done, the projector $R$ is chosen so as to satisfy

$$(c^T R)_i = \begin{cases} c_i & \text{if } i \in C \\ \sum_{j \in C \cap \mathcal{S}_i} w_{ij} c_j & \text{if } i \in F \end{cases}$$

where $\mathcal{S}_i$ denote the set of points that strongly influences $i$ (i.e. the "opposite" of the sets $\mathcal{D}_i$ above), $c$ is the coarse-level error approximation and the $w_{ij}$s are the *interpolation weights* given by [28]

$$w_{ij} = -\frac{L_{ij} + \sum_{m \in F \cap \mathcal{S}_i} \left( \frac{L_{im} L_{mj}}{\sum_{k \in C \cap \mathcal{S}_i} L_{mk}} \right)}{L_{ii} + \sum_{r \notin \mathcal{S}_i} L_{ir}} \quad .$$

While $C$ and $F$ are therefore used in rather different ways than constructing an aggregation operator, they are found in similar ways as the aggregations in §3.2. In particular, two heuristic criteria are defined to guide the search for the $C/F$ splitting:

- for each $F$-point $i$, every point $j \in \mathcal{S}_i$ should either be in the coarse interpolatory set $C \cap \mathcal{S}_i$ or should strongly depend on at least one point in $C \cap \mathcal{S}_i$;
- the set of coarse points $C$ should be a maximal subset of all points with the property that no $C$-point strongly depends an another $C$-point.

Satisfying both criteria is not always possible and typically the first is given priority over the second if that needs be. The standard heuristics used for finding the $C/F$ splitting are two-phase greedy algorithms that closely resemble the neighborhood aggregation approach of §3.2, except for looking at the "directed" sets $\mathcal{S}_i$ and $\mathcal{D}_i$ rather than to the "undirected" $\mathcal{N}_i$ (as the pairwise aggregation does).

It is worth remarking that the results of [23, 24] were aimed at problematic cases in which "errors missed by standard relaxation processes can vary substantially along strong matrix connections". This suggested a generalization of the classical AMG so that a number of vectors (*prototypes*), which are related to the error components that relaxation cannot break down, are in the range of interpolation, instead of having only the vector 1 as prototype as usual in the AMG development. The matrices in our application are very close to those for which the original AMG was designed. Furthermore the aggregations operators based on strong connection of §3.2 worked quite well already. So in our numerical tests we employed the standard AMG approach [28] rather than the variants of [23, 24, 41, 42], which do not seem to fit our application. In our setting, these are interesting only as yet another example of a different possible definition of strength-based operators. Nevertheless there does not appear to be reason to believe that aggregation operators based on these principles should dramatically outperform the quite similar ones of §3.2.

## 3.4 Combinatorial operators

While all the previous aggregation operators are inherently heuristic, the combinatorial operators of the recent [82] are based on a sound theoretical analysis. Interestingly this is grounded on the theoretical results developed for the study of an apparently unrelated subject, that of *Steiner tree* and *Steiner support graph* preconditioners. The former were introduced in [63], while [79] and [80] extended the results to the latter.

"Sophisticated" support-graph preconditioners (with guarantees on the conditioning number of the resulting preconditioned system) can be constructed with a process that is based on identifying a partitioning of the graph into “weakly interacting” clusters, selecting the (few) arcs that join them (which have to be “important” arcs, since their removal disconnects the graph), and possibly recursively repeating the process within each cluster [112, 81]. Since the clustering process is non-trivial to attain with low computational cost, the idea of Steiner support graph has been proposed to simplify it. Basically, once a set of promising vertex-disjoint clusters $\mathcal{V}_i$ is identified (with non-trivial algorithms which provide some appropriate guarantees on their quality), a new *Steiner node* $r_i$ is created for each cluster, and the star trees rooted at $r_i$ with leaves corresponding to the vertices in $\mathcal{V}_i$ are added to the preconditioner (which, of course, is defined for the new Steiner graph comprising all the original nodes plus the new roots). The process can then be repeated on the *quotient graph* corresponding to all the roots $r_i$; with appropriate care, this produces provably good preconditioners [80]. However in [82] it was realized that the very same process can be used for different purposes: namely the clusters $\mathcal{V}_i$ then become these of an aggregation operator, so that the quotient graph becomes the coarsened graph at the next level of a MGM. With appropriate choices, the guarantees on the conditioning obtained for the Steiner graph preconditioner can be “extended” to the coarsened system. This suggests the implementation of multigrid-like solvers. In particular, in [82] a solver running quite sophisticated cycles (more complex than the standard V and W cycles) and employing Jacobi as smoother obtains promising results for image processing applications. According to the authors, for the same underlying clustering approach, multigrid-like methods should almost always be preferable to the direct application of Steiner support graph preconditioners.

Our computational experience indicated that direct use of the approach of [82] to our instances did not seem to provide compelling results. This may be due to the use of “poor” smoothers like Jacobi, that are typically ineffective in our applications. However nothing prevents to re-use the “core” of the approach, i.e. the sophisticated choice of the aggregation operator, into a more standard MGM (accelerated or not) with more powerful Krylov-based smoothers.

# Chapter 4

# Computational results

We report a wide set of numerical tests aimed at comparing the computational behaviour of PCG and MG approaches on instances of (1.2) coming from real applications, in particular MCF problems. The stopping criterion used for all the methods was the slightly non-standard

$$|b_i - L_i x| \leq \varepsilon \max(|b_i|, 1) \qquad \forall i \tag{4.1}$$

(with $\varepsilon =$ `1e-5`), because this is the form typically required by the specific application [58, 59]. Note that (4.1) is basically a (scaled) $\infty$-norm stopping criterion, and therefore more difficult to attain (for the same $\varepsilon$) than more standard 2-norm stopping criteria.

The tests have been performed on matrices $L$ coming from the solution of randomly-generated MCF instances. Three different well-known random problem generators have been used: `Net`, `Grid`, `Goto`. These have been used in several cases to produce (both single and multicommodity) flow test instances [29, 30, 56, 58, 59, 60]. Each generator produces matrices with different topological properties, as shown in Figure 4.1. Furthermore the solution of the MCF instances via an IP methods produces weight matrices $\Theta$ with a different behaviour.

As the pictures show, the graph in `Net` problems has a random topological structure; these are the easiest instances to solve with the IP algorithm. Both `Grid` and `Goto` (Grid On TOrus) problems have a grid structure, but the latter are considerably more difficult to solve than the former, both in terms of IP algorithm and as the corresponding linear system. The difficulty of `Goto` is likely to be related by the structure of the $L$ matrix, which is far from the block and the banded case. We recall that the latter is the classical pattern related to standard grid graphs. Under the same conditions (problem size, IP iteration and preconditioning), generally a `Goto` system requires an order of magnitude more PCG iterations than `Net` or `Grid` ones.

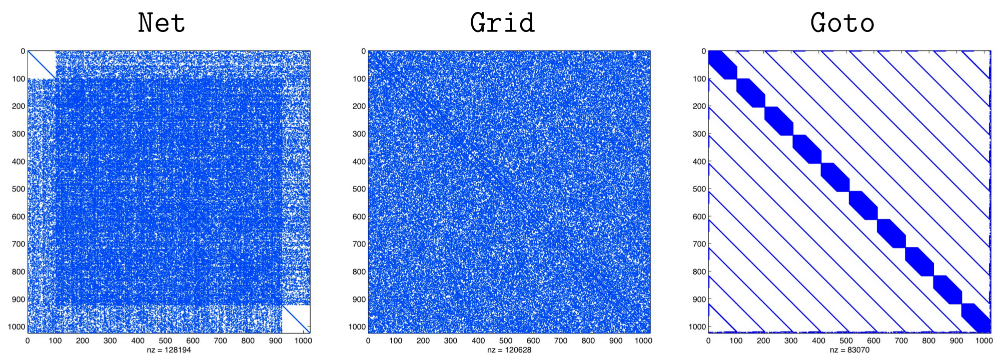


Figure 4.1: Structure of $L$ for different problem classes.

## 4.1 List of methods

For our experiments, we have compared a large number of "pure" PCG approaches and accelerated multigrid approaches, i.e. PCG algorithm where a single V-cycle is used as preconditioner. This is because on one hand PCG approaches have been the method of choice for the solution of this kind of systems in the context of MCF problems [29, 100, 30, 90, 58, 59] and on the other hand the most recent developments suggest that accelerated multigrid is both more effective and efficient than stand-alone multigrid [82, 42, 43].

The preconditioners employed are the diagonal one (D), the incomplete Cholesky one (IC) with tolerance parameter droptol (non-zero fill-in), the maximum spanning tree one (T), the "maximum spanning tree + diagonal" (T+D) one, obtained by summing to the T preconditioner the diagonal of the non-selected part the matrix. In particular, previous results [58, 59] have shown that the T+D preconditioner is always preferable to the T preconditioner for `Net` and `Grid` matrices, and therefore it is the only one employed in these cases. Conversely no dominance between T and T+D exist for the more difficult `Goto` matrices (with T+D being preferable in the first IP iterations and T in the last ones) and therefore both are tested in these cases.

All methods showed convergence, which however in some cases was exceedingly slow; thus we had to resort to setting an a priori upper-bound for the number of iterations. This was (somewhat arbitrarily) chosen as 1000 iterations for "pure" PCG and at 500 iterations for accelerated multigrid methods, since the latter have an higher cost per iteration.

We have to remark that providing consistent running times for all the methods proved to be very challenging, because of the different levels of optimizations of the available routines. For instance, while all aggregation operators have similar complexity, only for some of them efficient C implementations (and the corresponding MEX interface routines) were available, while others were implemented as ordinary (interpreted) m-files. This alone can produce orders-of-magnitude differences in running times. While highly sophisticated C++ implementations of support graph preconditioners are available [58, 59], for uniformity the built-in `pcg` function of MatLab has always been used, which, besides being an m-file, does not allow to exploit some of the relevant structure. Analogously the multigrid approach has been implemented as an m-file and therefore is far less efficient than what a compiled version could be. Furthermore the maximum spanning tree has been computed with the `graphminspantree` function of MatLab; this uses an exact ordering rather than an approximated one as in [58, 59] and therefore is significantly less efficient, despite being a compiled C routine. Moreover it does not automatically produce the right column ordering which avoids fill-in in the Cholesky factorization of the preconditioner, thereby making this step less efficient, too. So the running times in the following tables are not necessarily indicative of these that a fully optimized C or Fortran version may obtain. In particular, the times for forming and applying the projectors has been excluded from the figures, due to the above mentioned issues. Yet all methods have been implemented with the maximum possible uniformity, therefore we believe that the figures should be reasonably indicative of the relative performances of the approaches.

## 4.2 Comments and comparisons

In a first phase of our experiments we compared the different aggregation operators among them. Some data is shown in Table 4.1 for `Net` and `Grid` matrices and in Table 4.2 for `Goto` matrices. Only the case with $n = 2^{12}$ and density 64 ($m \approx 64n$) is shown, but the results are fairly indicative of the general trends. In the Tables, iteration numbers and total running times are reported for accelerated multigrid methods employing the different preconditioners. "lev" indicates that the aggregation operator failed to produce a reasonable number of levels (see below for more details) or that there were memory issues with the matrices. For CMG Aggregation operators, for instance, this happens occasionally due to the fact that the available implementation still lacks an appropriate sparsification routine. "**" instead indicates that the approach could not converge to the required accuracy within the allotted iteration limit.

The Tables show a rather complex picture. There is no single aggregation operator that dominates all the others for all matrices, preconditioners and IP iterations. However, especially when paired with "powerful" preconditioners (IC, T+D), all aggregation operators are quite effective, oftentimes (although not always) performing similarly. However it should be remarked that the different operators require different amounts of tuning. In particular all strength-based aggregation operators strongly depend on the parameter $\theta$. Finding an appropriate value of that parameter is not trivial and bad choices can lead to extremely poor performances. A few examples of this are shown in Table 4.3, where (a subset of) the levels produced on the same matrix by different settings of $\theta$ for the Strength Aggregation are reported. In the matrix, three stacked cells of the form $(k, \dots, h)^T$ mean that $k - h + 1$ levels have been generated, the size of each one being just one less than that of the previous. Thus inappropriate setting of $\theta$ can lead to very many levels of very similar size, quite the opposite of the expected exponential reduction. Although the number of levels tends to decrease as $\theta$ does, the behaviour is not monotone and can be highly erratic. Note that the AMG operators, although not aggregation ones, are still strength-based and suffer from the same drawbacks.

These issues happen much less frequently to the CMG operator, although sometimes the "descent" in the set of levels is not as smooth as one could imagine. For instance, for the `Grid` 16-8 matrix at IP 22 the obtained levels have size respectively 65535, 16907, 4518, 2461, 2198, 2142, 2122, 2114, 2110, 2104, 2097, 2092, 2087, 2070, 2063, 2061, 2060, 2056, 2011, 1938, 1765, 926, and 1. By contrast, Max and Minor operators always precisely halve the number of nodes at each level, therefore are preferable in this respect.

The aggregation operators are not — we underline this — the only part of the approach which may require tuning. This is also true, in particular, for the Incomplete Cholesky preconditioner, which requires setting of the droptol parameter. Finding the appropriate setting is crucial for the overall performances, but this setting is by far not uniform across all our instances. This is shown e.g. in Table 4.4, where the performances of the IC PCG approach are reported for varying droptol for the same basic instance (`Goto` 14-64) and different IP iterations. In the Table, "it" is the number of PCG iterations, "time" the total time, and "p.t." the time spent in solving the preconditioned system at each iteration. As the Table shows, too large values of droptol result in "poor" preconditioners, which are very easy to invert but provide poor convergence and therefore high overall running time. On the contrary, too small values result in very dense preconditioners, which are effective but too costly. The right trade-off crucially depends on the IP iterations and can vary of several orders of magnitude.

We then proceeded at comparing the best PCG variants and the best aggregation-based multigrid variants, throwing the standard AMG (with the same set of possible preconditioners) in the mix. The results are shown in

Table 4.5, Table 4.6, Table 4.7 for `Net`, `Grid`, `Goto` matrices respectively.

The Tables show that the aggregation based multigrid variants perform in general better than standard AMG techniques, especially in combination with $D$ and $T + D$ preconditioners. The comparison with the most competitive PCG is more involved: in fact, while the comparison concerning the number of iterations (taking into account the cost per iteration) can be done in fair way, the comparison in terms of CPU time is very complicate since different parts of the code show different levels of optimization. At any rate, some conclusions can be extracted. Indeed, while the aggregation based multigrid variants are always reasonably effective with $D$ (or $T + D$) preconditioners, the PCG is definitely less robust: in reality, if the paramenters are not chosen properly (and this is sometimes not trivial), the PCG algorithm could become unacceptably slow.

In conclusion numerical experiments show that our approach is effective in a uniform way, independently from the conditioning of the considered matrices, from the type of the problems, from the IP step in which the considered matrices arise. We think that this robustness is the most relevant result obtained in our work.

| | D | | IC | | T+D | | D | | IC | | T+D | | D | | IC | | T+D | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IP | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| `Net` | CMG Aggregation | | | | | | Max Aggregation | | | | | | Minor Aggregation | | | | | |
| 1 | lev | lev | lev | lev | lev | lev | 4 | 0.09 | 4 | 0.12 | 4 | 0.82 | 4 | 0.15 | 4 | 0.24 | 4 | 1.59 |
| 2 | lev | lev | lev | lev | lev | lev | 5 | 0.12 | 4 | 0.12 | 5 | 0.90 | 5 | 0.93 | 5 | 1.07 | 5 | 2.88 |
| 8 | 10 | 0.57 | 8 | 0.63 | 6 | 2.55 | 17 | 0.40 | 9 | 0.32 | 6 | 1.13 | 15 | 0.49 | 8 | 0.39 | 6 | 1.60 |
| 17 | 19 | 2.13 | 11 | 1.68 | 6 | 2.19 | 85 | 2.16 | 19 | 0.99 | 10 | 1.45 | 74 | 2.15 | 13 | 0.96 | 6 | 1.45 |
| 25 | 19 | 2.09 | 7 | 1.57 | 6 | 2.04 | ** | ** | 11 | 1.43 | 14 | 1.65 | 494 | 13.60 | 10 | 1.49 | 14 | 1.77 |
| 32 | 60 | 6.25 | 11 | 2.05 | 5 | 1.91 | ** | ** | 13 | 1.27 | 6 | 1.29 | ** | ** | 11 | 1.27 | 5 | 1.37 |
| 33 | 58 | 6.08 | 10 | 1.90 | 5 | 1.98 | ** | ** | 10 | 1.17 | 5 | 1.24 | ** | ** | 10 | 1.24 | 5 | 1.35 |
| `Net` | Strength Aggregation | | | | | | Neighborhood Aggregation | | | | | | Double Pairwise Aggregation | | | | | |
| 1 | 4 | 0.46 | 4 | 0.42 | 4 | 1.26 | 4 | 0.09 | 4 | 0.12 | 4 | 0.70 | 4 | 0.51 | 4 | 0.62 | 4 | 1.93 |
| 2 | 5 | 0.15 | 5 | 0.21 | 5 | 1.27 | 5 | 0.12 | 5 | 0.18 | 5 | 0.99 | 5 | 0.63 | 5 | 0.73 | 5 | 2.04 |
| 8 | 13 | 0.43 | 8 | 0.39 | 6 | 1.56 | 11 | 0.40 | 8 | 0.43 | 6 | 1.65 | 13 | 1.66 | 8 | 1.17 | 6 | 2.37 |
| 17 | 19 | 3.07 | 9 | 2.21 | 6 | 3.21 | 59 | 9.25 | 23 | 4.58 | 11 | 4.21 | 90 | 11.34 | 17 | 2.63 | 9 | 3.18 |
| 25 | ** | ** | 7 | 1.91 | 6 | 2.48 | 222 | 34.38 | 13 | 3.44 | 13 | 4.46 | ** | ** | 11 | 2.35 | 9 | 2.85 |
| 32 | ** | ** | 11 | 2.43 | 6 | 2.46 | ** | ** | 172 | 31.48 | 12 | 4.38 | ** | ** | 12 | 2.41 | 5 | 2.19 |
| 33 | ** | ** | 11 | 2.37 | 5 | 2.32 | ** | ** | 128 | 23.52 | 10 | 3.77 | 457 | 57.42 | 17 | 2.99 | 5 | 2.23 |
| `Grid` | CMG Aggregation | | | | | | Max Aggregation | | | | | | Minor Aggregation | | | | | |
| 1 | lev | lev | lev | lev | lev | lev | 4 | 0.11 | 4 | 0.12 | 4 | 0.87 | 4 | 0.71 | 4 | 0.76 | 5 | 2.65 |
| 2 | lev | lev | lev | lev | lev | lev | 7 | 0.15 | 7 | 0.20 | 7 | 0.99 | 5 | 0.17 | 5 | 0.21 | 5 | 1.32 |
| 11 | lev | lev | lev | lev | lev | lev | 5 | 0.12 | 6 | 0.18 | 5 | 0.93 | 5 | 0.84 | 5 | 0.98 | 5 | 2.91 |
| 23 | 14 | 1.57 | 6 | 1.37 | 7 | 2.26 | 73 | 1.82 | 12 | 1.11 | 19 | 1.85 | 30 | 0.93 | 7 | 1.12 | 6 | 1.51 |
| 34 | 18 | 1.95 | 12 | 1.65 | 7 | 2.27 | 145 | 3.65 | 21 | 1.04 | 17 | 1.74 | 86 | 4.43 | 13 | 0.87 | 8 | 1.49 |
| 44 | 18 | 1.90 | 42 | 6.27 | 5 | 1.88 | ** | ** | 146 | 6.59 | 14 | 1.67 | ** | ** | 166 | 7.86 | 4 | 1.31 |
| 45 | 18 | 1.95 | 41 | 6.09 | 5 | 1.91 | ** | ** | 55 | 3.34 | 11 | 1.54 | ** | ** | ** | ** | 6 | 1.40 |
| `Grid` | Strength Aggregation | | | | | | Neighborhood Aggregation | | | | | | Double Pairwise Aggregation | | | | | |
| 1 | 5 | 0.15 | 5 | 0.22 | 5 | 1.23 | 5 | 0.49 | 5 | 0.54 | 5 | 1.54 | 5 | 0.59 | 5 | 0.73 | 5 | 1.96 |
| 2 | 7 | 0.21 | 7 | 0.28 | 6 | 1.38 | 7 | 0.23 | 7 | 0.28 | 7 | 1.29 | 7 | 0.87 | 7 | 0.99 | 7 | 2.34 |
| 11 | 6 | 0.18 | 6 | 0.32 | 5 | 1.23 | 5 | 0.15 | 5 | 0.21 | 6 | 1.21 | 5 | 0.63 | 5 | 0.74 | 5 | 2.06 |
| 23 | 18 | 3.35 | 10 | 2.99 | 12 | 5.03 | 33 | 5.01 | 11 | 2.93 | 12 | 4.32 | 41 | 5.03 | 10 | 2.09 | 9 | 3.07 |
| 34 | 63 | 10.59 | 21 | 4.47 | 12 | 4.63 | 114 | 16.81 | 24 | 4.46 | 14 | 4.83 | 173 | 20.99 | 28 | 4.07 | 13 | 3.86 |
| 44 | ** | ** | ** | ** | 13 | 4.68 | ** | ** | ** | ** | 15 | 4.92 | ** | ** | ** | ** | 16 | 3.99 |
| 45 | ** | ** | 238 | 47.79 | 9 | 4.02 | ** | ** | 426 | 77.23 | 10 | 3.94 | ** | ** | ** | ** | 14 | 3.93 |

Table 4.1: Comparison between aggregation operators for `Net` and `Grid` 12-64.

| | D | | IC | | T+D | | T | | D | | IC | | T+D | | T | | D | | IC | | T+D | | T | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IP | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| | CMG Aggregation | | | | | | | | Max Aggregation | | | | | | | | Minor Aggregation | | | | | | | |
| 1 | 8 | 0.35 | 7 | 0.39 | 8 | 2.10 | ** | ** | 20 | 2.01 | 16 | 1.59 | 15 | 3.96 | ** | ** | 13 | 0.32 | 11 | 0.32 | 11 | 1.23 | ** | ** |
| 2 | 21 | 0.40 | 7 | 0.18 | 6 | 0.78 | 91 | 3.93 | 46 | 1.07 | 9 | 0.31 | 7 | 1.01 | 93 | 6.13 | 24 | 2.76 | 6 | 0.95 | 5 | 1.84 | 81 | 17.73 |
| 24 | 23 | 1.99 | 16 | 1.80 | 15 | 2.82 | 49 | 7.67 | 481 | 10.53 | 26 | 1.35 | 45 | 2.68 | 42 | 3.07 | 493 | 11.38 | 22 | 1.21 | 35 | 2.35 | 41 | 2.99 |
| 47 | 27 | 2.21 | 31 | 2.97 | 15 | 2.94 | 33 | 5.42 | 265 | 6.11 | 26 | 1.06 | 22 | 1.80 | 30 | 2.87 | ** | ** | 21 | 0.88 | 63 | 3.63 | 31 | 2.52 |
| 71 | 25 | 2.18 | 16 | 1.63 | 17 | 3.10 | 24 | 4.92 | ** | ** | 12 | 0.70 | 23 | 1.82 | 23 | 2.41 | ** | ** | 12 | 0.67 | 28 | 2.09 | 23 | 2.29 |
| 93 | 21 | 1.73 | 10 | 1.04 | 9 | 2.07 | 16 | 3.66 | ** | ** | 7 | 0.49 | 11 | 1.34 | 16 | 2.09 | ** | ** | 9 | 0.49 | 14 | 1.51 | 16 | 1.79 |
| 94 | 20 | 1.70 | 14 | 1.40 | 10 | 2.18 | 13 | 3.21 | ** | ** | 7 | 0.48 | 12 | 1.35 | 14 | 1.91 | ** | ** | 10 | 0.49 | 13 | 1.46 | 14 | 1.79 |
| | Strength Aggregation | | | | | | | | Neighborhood Aggregation | | | | | | | | Double Pairwise Aggregation | | | | | | | |
| 1 | 14 | 1.46 | 13 | 1.54 | 14 | 3.55 | ** | ** | 14 | 0.87 | 12 | 0.82 | 12 | 1.81 | ** | ** | lev | lev | lev | lev | lev | lev | lev | lev |
| 2 | 17 | 1.74 | 6 | 0.82 | 5 | 1.59 | 80 | 14.49 | 21 | 1.48 | 7 | 0.65 | 6 | 1.31 | 76 | 9.53 | 19 | 0.29 | 7 | 0.18 | 6 | 0.60 | 77 | 2.41 |
| 24 | 23 | 3.21 | 14 | 2.79 | 15 | 4.92 | 46 | 16.45 | 26 | 2.83 | 15 | 2.23 | 13 | 3.18 | 45 | 9.48 | 427 | 38.15 | 18 | 2.09 | 14 | 2.94 | 48 | 10.73 |
| 47 | 23 | 3.15 | 22 | 3.60 | 14 | 4.52 | 33 | 8.73 | 31 | 3.29 | 25 | 3.18 | 15 | 3.66 | 32 | 8.01 | ** | ** | 29 | 3.10 | 15 | 3.04 | 36 | 6.11 |
| 71 | 22 | 3.10 | 11 | 1.99 | 13 | 4.61 | 23 | 8.25 | 48 | 5.11 | 12 | 1.73 | 15 | 3.79 | 24 | 7.23 | ** | ** | 18 | 2.10 | 17 | 3.35 | 24 | 5.83 |
| 93 | 19 | 2.79 | 11 | 2.16 | 10 | 3.99 | 16 | 6.81 | 80 | 9.03 | 10 | 1.48 | 18 | 4.50 | 16 | 4.66 | ** | ** | 12 | 1.41 | 10 | 2.40 | 16 | 4.21 |
| 94 | 17 | 2.48 | 12 | 2.40 | 10 | 3.61 | 13 | 5.47 | 28 | 3.10 | 11 | 1.70 | 8 | 2.57 | 13 | 4.07 | ** | ** | 12 | 1.37 | 10 | 2.32 | 13 | 2.83 |

Table 4.2: Comparison between aggregation operators for `Goto` 12-64.

| | `Grid` 16-8, IP 11 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $\theta$ | 0.9 | 0.8 | 0.7 | 0.6 | 0.5 | 0.4 | 0.3 | 0.2 | 0.1 |
| | 65535 | 65535 | 65535 | 65535 | 65535 | 65535 | 65535 | 65535 | 65535 |
| | 35959 | 34380 | 32660 | 30729 | 28578 | 26235 | 23610 | 20696 | 17113 |
| | 20919 | 18393 | 15959 | 13712 | 11562 | 9583 | 7687 | 5851 | 4085 |
| | 13133 | 10343 | 7984 | 6122 | 4777 | 3903 | 3292 | 2740 | 2296 |
| | 8755 | 6190 | 4333 | 3229 | 2627 | 2688 | 2434 | 882 | 1 |
| | 6336 | 4182 | 2637 | 2433 | 658 | 6 | 2 | 1 | |
| | 4818 | 3068 | 1456 | 11 | 1 | | | | |
| | 4210 | 2964 | 362 | | | | | | |
| | 4059 | 975 | 361 | | | | | | |
| | 4022 | 974 | 360 | | | | | | |
| | ⋮ | ⋮ | ⋮ | | | | | | |
| | 1 | 1 | 1 | | | | | | |
| | `Net` 12-64, IP 1 | | | | | | | | |
| | 4095 | 4095 | 4095 | 4095 | 4095 | 4095 | 4095 | 4095 | 4095 |
| | 1567 | 1567 | 1567 | 1567 | 523 | 523 | 523 | 523 | 523 |
| | 1322 | 1268 | 1123 | 796 | 450 | 404 | 332 | 152 | 63 |
| | 1290 | 1257 | 1111 | 689 | 449 | 403 | 331 | | |
| | 1284 | 1255 | 1110 | 646 | 448 | 402 | 330 | | |
| | 1283 | 1254 | 1109 | 645 | 447 | 401 | 329 | | |
| | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | | |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | |
| | `Goto` 14-64, IP 1 | | | | | | | | |
| | 16383 | 16383 | 16383 | 16383 | 16383 | 16383 | 16383 | 16383 | 16383 |
| | 7417 | 7418 | 7418 | 7404 | 1409 | 1394 | 1368 | 1288 | 1089 |
| | 3686 | 2914 | 2158 | 2080 | 912 | 864 | 757 | 674 | 579 |
| | 2283 | 2056 | 1621 | 1490 | 685 | 657 | 597 | 562 | 410 |
| | 1713 | 1752 | 1455 | 1363 | 642 | 581 | 586 | 551 | 405 |
| | 921 | 1576 | 1325 | 1344 | 285 | 242 | 559 | 43 | 397 |
| | 422 | 1292 | 1320 | 1320 | 284 | 241 | 558 | | 396 |
| | 388 | 995 | 804 | 1319 | 283 | 240 | 557 | | 395 |
| | 387 | 411 | 1 | 1318 | 282 | 239 | 556 | | 394 |
| | ⋮ | ⋮ | | ⋮ | ⋮ | ⋮ | ⋮ | | ⋮ |
| | 378 | 408 | | 1306 | 267 | 220 | 520 | | 384 |
| | 1 | 1 | | 2 | 1 | 1 | 1 | | 1 |

Table 4.3: Multigrid levels as a function of $\theta$ for the Strength Aggregation.

|  | IP 2 |  |  | IP 56 |  |  | IP 84 |  |  | IP 111 |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| droptol | it | p.t. | time | it | p.t. | time | it | p.t. | time | it | p.t. | time |
| 1e-02 | 1581 | 0.23 | 18.42 | ** | ** | ** | ** | ** | ** | ** | ** | ** |
| 1e-03 | 415 | 0.24 | 5.14 | 1916 | 0.45 | 25.95 | ** | ** | ** | ** | ** | ** |
| 1e-04 | 189 | 0.39 | 2.88 | 760 | 1.06 | 10.88 | 4198 | 0.50 | 57.12 | ** | ** | ** |
| 1e-05 | 99 | 24.85 | 26.34 | 187 | 12.64 | 14.49 | 375 | 0.82 | 5.78 | 3875 | 0.45 | 48.60 |
| 1e-06 | 48 | 56.99 | 57.95 | 80 | 38.61 | 39.99 | 105 | 5.38 | 6.95 | 574 | 0.50 | 7.69 |
| 1e-07 | 28 | 0.90 | 108.54 | 46 | 54.67 | 55.64 | 51 | 27.25 | 28.09 | 95 | 0.72 | 1.98 |
| 1e-08 | ** | ** | ** | 21 | 118.94 | 119.62 | 28 | 48.26 | 48.80 | 35 | 1.82 | 2.30 |
| 1e-09 | ** | ** | ** | ** | ** | ** | 17 | 67.39 | 67.82 | 18 | 13.65 | 13.94 |
| 1e-10 | ** | ** | ** | ** | ** | ** | ** | ** | ** | 12 | 30.52 | 30.74 |

Table 4.4: Performances of IC PCG as a function of droptol.

|  | PCG |  |  |  |  |  | AGO |  |  |  |  |  | AMG |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IP | D |  | IC |  | T+D |  | D |  | IC |  | T+D |  | D |  | IC |  | T+D |  |
| 12-64 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 13 | 0.05 | 13 | 0.06 | 13 | 0.64 | 4 | 0.09 | 4 | 0.12 | 4 | 0.70 | 12 | 0.82 | 12 | 0.87 | 12 | 3.44 |
| 2 | 15 | 0.03 | 15 | 0.08 | 15 | 0.67 | 5 | 0.12 | 4 | 0.12 | 5 | 0.90 | 17 | 2.13 | 16 | 2.19 | 15 | 7.20 |
| 8 | 59 | 0.15 | 24 | 0.12 | 17 | 0.68 | 11 | 0.40 | 9 | 0.32 | 6 | 1.13 | 19 | 3.47 | 17 | 3.33 | 11 | 4.66 |
| 17 | 468 | 1.21 | 39 | 0.29 | 25 | 0.73 | 19 | 2.13 | 13 | 0.96 | 6 | 1.45 | 11 | 1.62 | 10 | 1.99 | 6 | 2.87 |
| 25 | ** | ** | 37 | 0.67 | 19 | 0.68 | 19 | 2.09 | 11 | 1.43 | 14 | 1.65 | 13 | 1.19 | 6 | 1.88 | 5 | 2.55 |
| 32 | ** | ** | 191 | 1.10 | 11 | 0.63 | 60 | 6.25 | 13 | 1.27 | 6 | 1.29 | 12 | 1.76 | 9 | 2.38 | 5 | 2.63 |
| 33 | ** | ** | 148 | 0.97 | 9 | 0.62 | 58 | 6.08 | 10 | 1.17 | 5 | 1.24 | 14 | 2.15 | 8 | 2.21 | 5 | 2.48 |
| 14-64 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 14 | 0.21 | 14 | 0.34 | 14 | 5.21 | 5 | 1.67 | 5 | 1.79 | 5 | 7.41 | 12 | 3.69 | 11 | 4.11 | 17 | 26.83 |
| 2 | 18 | 0.26 | 18 | 0.42 | 16 | 5.38 | 5 | 0.87 | 6 | 1.34 | 5 | 9.51 | 19 | 13.44 | 21 | 16.83 | 18 | 40.48 |
| 11 | 73 | 0.99 | 21 | 0.59 | 18 | 5.49 | 11 | 2.10 | 6 | 1.87 | 6 | 11.84 | 26 | 15.81 | 14 | 10.06 | 21 | 30.40 |
| 23 | 473 | 5.95 | 207 | 3.09 | 23 | 5.57 | 21 | 3.07 | 40 | 6.42 | 10 | 10.18 | 12 | 6.42 | 40 | 22.80 | 7 | 17.08 |
| 34 | ** | ** | 372 | 6.16 | 22 | 5.50 | 20 | 8.45 | 38 | 19.53 | 8 | 11.98 | 15 | 5.47 | 36 | 18.57 | 7 | 15.47 |
| 44 | ** | ** | ** | ** | 13 | 5.47 | 26 | 11.79 | ** | ** | 6 | 10.79 | 18 | 8.48 | 40 | 38.18 | 5 | 15.28 |
| 45 | ** | ** | ** | ** | 13 | 5.55 | 26 | 11.76 | ** | ** | 7 | 8.54 | 16 | 8.25 | 48 | 40.99 | 5 | 14.92 |
| 16-8 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 26 | 0.40 | 25 | 0.53 | 25 | 47.29 | 9 | 0.95 | 8 | 1.10 | 8 | 49.10 | lev | lev | lev | lev | lev | lev |
| 2 | 39 | 0.57 | 23 | 0.73 | 24 | 49.46 | 9 | 1.45 | 6 | 1.66 | 7 | 57.93 | 17 | 24.13 | 17 | 24.99 | 15 | 128.15 |
| 11 | 99 | 1.43 | 41 | 0.98 | 24 | 47.72 | 16 | 3.29 | 9 | 2.83 | 6 | 64.05 | 22 | 8.50 | 18 | 8.86 | 14 | 85.25 |
| 21 | 305 | 4.46 | 126 | 2.37 | 21 | 51.68 | 38 | 6.83 | 28 | 6.39 | 9 | 64.69 | 38 | 11.57 | 35 | 12.60 | 12 | 81.66 |
| 31 | ** | ** | ** | ** | 23 | 52.04 | 36 | 5.67 | 208 | 50.27 | 8 | 59.48 | 14 | 8.79 | 141 | 103.71 | 6 | 77.47 |
| 40 | ** | ** | ** | ** | 38 | 58.67 | 28 | 9.92 | ** | ** | 21 | 87.26 | 15 | 12.30 | ** | ** | 8 | 89.49 |
| 41 | ** | ** | ** | ** | ** | ** | 27 | 4.32 | ** | ** | ** | ** | 15 | 10.15 | ** | ** | ** | ** |

Table 4.5: Comparison between all approaches for `Net` matrices.

| | PCG | | | | | | AGO | | | | | | AMG | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IP | D | | IC | | T+D | | D | | IC | | T+D | | D | | IC | | T+D | |
| 12-64 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 13 | 0.03 | 13 | 0.06 | 13 | 0.61 | 4 | 0.11 | 4 | 0.12 | 4 | 0.87 | 12 | 1.32 | 12 | 1.62 | 12 | 6.45 |
| 2 | 12 | 0.03 | 12 | 0.06 | 11 | 0.62 | 7 | 0.15 | 7 | 0.20 | 7 | 0.99 | 6 | 1.59 | 6 | 1.90 | 6 | 4.71 |
| 11 | 13 | 0.05 | 13 | 0.06 | 12 | 0.62 | 5 | 0.12 | 6 | 0.18 | 5 | 0.93 | 7 | 2.96 | 7 | 2.71 | 7 | 7.73 |
| 23 | 120 | 0.32 | 15 | 0.49 | 19 | 0.67 | 14 | 1.57 | 12 | 1.11 | 6 | 1.51 | 10 | 1.54 | 6 | 1.88 | 6 | 2.90 |
| 34 | 607 | 1.48 | 51 | 0.35 | 21 | 0.68 | 18 | 1.95 | 13 | 0.87 | 8 | 1.49 | 13 | 1.98 | 10 | 1.93 | 6 | 2.94 |
| 44 | ** | ** | 281 | 1.70 | 15 | 0.63 | 18 | 1.90 | 42 | 6.27 | 4 | 1.31 | 14 | 2.11 | 28 | 5.99 | 5 | 2.67 |
| 45 | ** | ** | 257 | 1.63 | 14 | 0.64 | 18 | 1.95 | 55 | 3.34 | 6 | 1.40 | 14 | 2.10 | 32 | 6.59 | 5 | 2.66 |
| 14-64 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 15 | 0.20 | 15 | 0.34 | 16 | 3.82 | 6 | 1.04 | 5 | 1.21 | 5 | 8.26 | 16 | 15.03 | 16 | 18.17 | 16 | 54.16 |
| 2 | 13 | 0.18 | 13 | 0.31 | 12 | 5.07 | 7 | 1.13 | 7 | 1.45 | 7 | 9.70 | 7 | 5.76 | 7 | 6.75 | 7 | 31.59 |
| 8 | 29 | 0.43 | 17 | 0.57 | 16 | 5.17 | 8 | 1.51 | 6 | 1.84 | 5 | 10.99 | 16 | 9.93 | 12 | 9.64 | 15 | 27.23 |
| 16 | 567 | 7.11 | 205 | 3.22 | 28 | 5.53 | 20 | 3.93 | 27 | 6.27 | 6 | 12.12 | 12 | 6.24 | 20 | 9.41 | 6 | 15.66 |
| 24 | ** | ** | 665 | 10.92 | 18 | 5.36 | 23 | 10.20 | 118 | 18.98 | 8 | 8.97 | 14 | 5.78 | 52 | 29.67 | 6 | 15.08 |
| 31 | ** | ** | ** | ** | 20 | 5.27 | 90 | 38.67 | 72 | 61.13 | 12 | 11.31 | 36 | 13.74 | 34 | 48.67 | 9 | 17.17 |
| 32 | ** | ** | ** | ** | 20 | 5.33 | 26 | 11.66 | 22 | 35.20 | 7 | 11.46 | 16 | 7.28 | 12 | 35.84 | 39 | 32.79 |
| 16-8 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 28 | 0.39 | 28 | 0.46 | 23 | 17.26 | 11 | 1.13 | 11 | 1.27 | 9 | 21.07 | lev | lev | lev | lev | lev | lev |
| 2 | 23 | 0.24 | 16 | 4.57 | 17 | 34.22 | 11 | 1.24 | 9 | 5.54 | 9 | 40.52 | 9 | 4.68 | 7 | 8.78 | 7 | 51.48 |
| 11 | 34 | 0.39 | 18 | 7.19 | 20 | 41.12 | 9 | 1.40 | 6 | 8.26 | 7 | 56.45 | 11 | 7.41 | 9 | 14.78 | 9 | 73.69 |
| 22 | 436 | 5.11 | 104 | 1.77 | 24 | 42.54 | 17 | 2.76 | 19 | 4.07 | 6 | 57.76 | 26 | 13.97 | 20 | 12.96 | 8 | 67.14 |
| 33 | ** | ** | ** | ** | 26 | 43.68 | 24 | 9.23 | 197 | 168.66 | 6 | 51.46 | 13 | 6.56 | 144 | 162.81 | 6 | 64.61 |
| 43 | ** | ** | ** | ** | 20 | 43.50 | 27 | 11.06 | ** | ** | 7 | 53.14 | 14 | 4.33 | ** | ** | 5 | 65.52 |
| 44 | ** | ** | ** | ** | ** | ** | 30 | 11.37 | ** | ** | ** | ** | 14 | 7.59 | ** | ** | ** | ** |

Table 4.6: Comparison between all approaches for `Grid` matrices.

| | PCG | | | | | | | | AGO | | | | | | | | AMG | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | D | | IC | | T+D | | T | | D | | IC | | T+D | | T | | D | | IC | | T+D | | T | |
| 12-64 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 30 | 0.07 | 27 | 0.07 | 26 | 0.51 | 807 | 3.86 | 13 | 0.32 | 11 | 0.32 | 11 | 1.23 | ** | ** | 17 | 1.38 | 15 | 1.40 | 14 | 2.63 | ** | ** |
| 2 | 475 | 1.03 | 68 | 0.18 | 43 | 0.53 | 94 | 0.82 | 19 | 0.29 | 7 | 0.18 | 6 | 0.60 | 77 | 2.41 | 7 | 1.06 | 5 | 0.96 | 3 | 1.95 | 59 | 14.67 |
| 24 | ** | ** | 417 | 1.21 | 472 | 2.73 | 69 | 0.82 | 23 | 1.99 | 22 | 1.21 | 35 | 2.35 | 41 | 2.99 | 15 | 1.81 | 12 | 2.10 | 10 | 2.96 | 43 | 12.51 |
| 47 | ** | ** | 829 | 1.96 | 898 | 4.80 | 54 | 0.74 | 27 | 2.21 | 21 | 0.88 | 22 | 1.80 | 31 | 2.52 | 18 | 2.02 | 22 | 2.91 | 12 | 3.21 | 31 | 6.77 |
| 71 | ** | ** | 381 | 0.95 | ** | ** | 41 | 0.70 | 25 | 2.18 | 12 | 0.67 | 23 | 1.82 | 23 | 2.29 | 16 | 1.88 | 14 | 1.96 | 12 | 3.36 | 25 | 7.28 |
| 93 | ** | ** | 367 | 0.90 | ** | ** | 26 | 0.67 | 21 | 1.73 | 7 | 0.49 | 11 | 1.34 | 16 | 1.79 | 16 | 1.82 | 12 | 1.65 | 8 | 2.46 | 16 | 4.58 |
| 94 | ** | ** | 398 | 0.96 | ** | ** | 23 | 0.63 | 20 | 1.70 | 7 | 0.48 | 12 | 1.35 | 14 | 1.79 | 15 | 1.71 | 11 | 1.57 | 9 | 2.74 | 13 | 4.35 |
| 14-64 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 54 | 0.62 | 54 | 0.71 | 51 | 3.79 | ** | ** | 15 | 3.99 | 16 | 4.74 | 11 | 15.70 | ** | ** | 55 | 5.92 | 56 | 6.24 | 63 | 15.75 | ** | ** |
| 2 | ** | ** | 189 | 2.88 | 279 | 7.97 | 37 | 3.08 | 46 | 4.61 | 14 | 3.18 | 27 | 7.64 | 21 | 7.90 | 22 | 2.83 | 8 | 2.82 | 10 | 6.50 | 20 | 10.01 |
| 28 | ** | ** | 384 | 22.94 | ** | ** | 137 | 6.25 | 36 | 3.69 | 14 | 19.37 | 21 | 7.78 | 112 | 22.43 | 20 | 2.96 | 12 | 20.93 | 14 | 9.95 | 99 | 30.17 |
| 56 | ** | ** | 760 | 10.88 | ** | ** | 102 | 6.27 | 34 | 3.29 | 17 | 3.57 | 20 | 8.41 | 78 | 18.03 | 19 | 2.69 | 13 | 4.13 | 16 | 10.62 | 58 | 22.86 |
| 84 | ** | ** | 375 | 5.78 | ** | ** | 73 | 5.63 | 35 | 3.35 | 10 | 2.40 | 18 | 7.76 | 50 | 14.43 | 22 | 2.99 | 8 | 2.93 | 14 | 10.20 | 47 | 20.43 |
| 111 | ** | ** | 574 | 7.69 | ** | ** | 35 | 4.63 | 39 | 5.11 | 9 | 2.69 | 27 | 9.32 | 19 | 8.34 | 18 | 2.35 | 10 | 2.49 | 11 | 8.89 | 20 | 10.88 |
| 112 | ** | ** | 91 | 1.85 | ** | ** | 34 | 4.44 | 39 | 3.93 | 8 | 1.98 | 19 | 8.31 | 19 | 8.89 | 19 | 2.49 | 5 | 2.10 | 11 | 8.86 | 19 | 10.37 |
| 16-8 | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time | it | time |
| 1 | 76 | 1.04 | 64 | 0.95 | 65 | 22.35 | ** | ** | 12 | 1.06 | 11 | 1.27 | 11 | 25.64 | ** | ** | 17 | 2.32 | 14 | 2.76 | 13 | 37.83 | ** | ** |
| 2 | ** | ** | 644 | 8.06 | 753 | 50.73 | 66 | 17.50 | 116 | 11.09 | 31 | 4.33 | 43 | 26.75 | 182 | 74.24 | 48 | 6.98 | 17 | 5.13 | 22 | 24.18 | 263 | 255.49 |
| 10 | ** | ** | 928 | 12.77 | ** | ** | 38 | 14.86 | 124 | 11.60 | 38 | 6.22 | 48 | 30.26 | 102 | 47.11 | 39 | 4.96 | 19 | 4.10 | 26 | 21.40 | 76 | 50.24 |
| 19 | ** | ** | 694 | 11.18 | ** | ** | 76 | 24.58 | 44 | 3.82 | 14 | 3.61 | 23 | 30.56 | 94 | 83.72 | 28 | 3.35 | 13 | 3.04 | 20 | 27.93 | 281 | 149.01 |
| 28 | ** | ** | 601 | 8.70 | ** | ** | 30 | 24.91 | 37 | 3.40 | 8 | 1.70 | 13 | 31.44 | 20 | 31.55 | 20 | 2.51 | 8 | 1.99 | 32 | 35.03 | 17 | 39.48 |
| 36 | ** | ** | 68 | 1.62 | ** | ** | 20 | 30.15 | 32 | 2.76 | 4 | 1.79 | 9 | 29.68 | 12 | 35.22 | 24 | 2.73 | 4 | 1.34 | 74 | 46.64 | 9 | 43.41 |
| 37 | ** | ** | 68 | 1.51 | ** | ** | 17 | 34.99 | 32 | 2.73 | 4 | 1.51 | 9 | 29.76 | 9 | 39.17 | ** | ** | 4 | 1.46 | 32 | 37.65 | 8 | 48.59 |

Table 4.7: Comparison between all approaches for `Goto` matrices.

# Part II

# The image restoration problem

# Chapter 5

# Optimal preconditioning

In the last years important progress were made both in the theory and in the applications of inverse problems. Several methods for overcoming the difficulties due to the ill-posedness have been successfully developed. We remind that the solution of a well-posed problem satisfies the requirements of existence, uniqueness and continuous dependence on the data. So an ill-posed problem is a problem whose solution does not exists or is not unique or does not depend continuously on the data. Image deblurring problem [13, 53, 70, 71] represents an important and deeply studied exponent into the inverse problems field. It consists in finding the true image of an unknown object, having only the detected image, which is affected by blur and noise. Astronomy and civil or biomedical tomography are only few of the many scientific areas in which this problem needs to be solved.

We deal with the classical image restoration problem of blurred and noisy images in the case of a space invariant blurring, which means that the same blur arises all over the image domain. Under such assumption the image formation process is modelled according to the following integral equation with space invariant kernel

$$b(x) = \int h(x-\tilde{x})f(\tilde{x})d\tilde{x} + \eta(x),\ x \in \mathbb{R}^2, \tag{5.1}$$

where $f$ denotes the true physical object to be restored, $b$ is the recorded blurred and noisy image, $\eta$ takes into account unknown errors in the collected data, e.g. measurement errors and noise. As customary, we consider the discretization of (5.1) by means of a standard $2D$ generalization of the rectangle quadrature formula on an equispaced grid, ordered row-wise from the top-left corner to the bottom-right one. Hence, we obtain the relations

$$b_i = \sum_{j\in\mathbb{Z}^2} h_{i-j}f_j + \eta_i, \quad i \in \mathbb{Z}^2, \tag{5.2}$$

in which an infinite and a shift-invariant matrix $\widetilde{A}_\infty = [h_{i-j}]_{(i,j)=((i_1,i_2),(j_1,j_2))}$, i.e. a two-level Toeplitz matrix, is involved.

In principle, (5.2) presents an infinite summation since the true image scene does not have a finite boundary. Nevertheless, the data $b_i$ are clearly collected only at a finite number of values, so representing only a finite region of such an infinite scene. In addition, the blurring operator typically shows a finite support, so that it is completely described by a Point Spread Function (PSF) mask such as

$$h_{PSF} = [h_{i_1,i_2}]_{i_1=-q_1,\ldots,q_1, i_2=-q_2,\ldots,q_2} \tag{5.3}$$

where $h_{i_1,i_2} \geq 0$ for any $i_1, i_2$ and $\sum_{i=-q}^{q} h_i = 1$, $i = (i_1, i_2)$, $q = (q_1, q_2)$ (normalization according to a suitable conservation law). Therefore, relations (5.2) imply

$$b_i = \sum_{s=-q}^{q} h_s f_{i-s} + \eta_i, \quad i_1 = 1, \ldots, n_1, i_2 = 1, \ldots, n_2, \tag{5.4}$$

where the range of collected data defines the so called Field of View (FOV).

Once again, we are assuming that all the involved data in (5.5), similarly to (5.2), are reshaped in a row-wise ordering. In such a way we obtain the linear system

$$\widetilde{A}\tilde{f} = b - \eta \tag{5.5}$$

where $\widetilde{A} \in \mathbb{R}^{N(n)\times N(n+2q)}$ is a finite principal sub-matrix of $\widetilde{A}_\infty$, with main diagonal containing $h_{0,0}$, $\tilde{f} \in \mathbb{R}^{N(n+2q)}$, $b, \eta \in \mathbb{R}^{N(n)}$ and with $N(m) = m_1 m_2$, for any two-index $m = (m_1, m_2)$.

As evident from (5.4), the problem is undetermined since the number of unknowns involved in the convolution exceeds the number of recorded data. Boundary conditions (BCs) are introduced to artificially describe the scene outside the FOV: the values of unknowns outside the FOV are fixed or are defined as linear combinations of the unknowns inside the FOV, the target being to reduce (5.5) into a square linear system

$$A_n f = b - \eta \tag{5.6}$$

with $A_n \in \mathbb{R}^{N(n)\times N(n)}$, $n = (n_1, n_2)$, $N(n) = n_1 n_2$ and $f, b, \eta \in \mathbb{R}^{N(n)}$.

The choice of the BCs does not affect the global spectral behaviour of the matrix. However, it may have a valuable impact both with respect to the accuracy of the restored image (especially close to the boundaries where ringing effects can appear) and to the computational costs for recovering $f$ from the blurred datum, with or without noise. Notice also that, typically, the matrix $A$ is very ill-conditioned and there is a significant intersection between the subspace related to small eigen/singular values and the high frequency subspace.

## 5.1 The role of boundary conditions in the restoration problem

Hereafter we summarize the relevant properties of two recently proposed type of BCs, i.e. the Reflective [92] and Anti-Reflective BCs [106], with respect both to structural and spectral properties of the arising matrices. Indeed, the use of classical periodic BCs enforces a circulant structure and the spectral decomposition can be computed efficiently with the Fast Fourier Transform (FFT) [37], but these computational facilities are coupled with significant ringing effects whenever a significant discontinuity is introduced into the image, since an image usually is not periodic. Thus, the target is to obtain the best possible approximation properties, keeping unaltered the fact that the arising matrix shows an exploitable structure. Reflective and Anti-Reflective BCs more carefully describe the scene outside the FOV and give rise to exploitable structures. Clearly, several other methods deal with this topic in the image processing literature, e.g. local mean value [110] or extrapolation techniques (see [83] and references therein). Nevertheless, the penalty of their good approximation properties could lie in a linear algebra problem more difficult to cope with. Hereafter, as more natural in the applications, we will use a two-index notation: we denote by $[f_{i_1,i_2}]_{i_1=1,\ldots,n_1,i_2=1,\ldots,n_2}$ the pixels of the true image and by $[b_{i_1,i_2}]_{i_1=1,\ldots,n_1,i_2=1,\ldots,n_2}$ the pixels of the recorded image.

### 5.1.1 Reflective boundary conditions

In [92] Ng *et al.* analyse the use of Reflective BCs, both from model and linear algebra point of view. The improvement with respect to Periodic BCs is due to the preservation of the continuity of the image. In fact, the scene outside the FOV is assumed to be a reflection of the scene inside the FOV. For example, with a boundary at $x_1 = 0$ and $x_2 = 0$ the reflective condition is given by $f(\pm x_1, \pm x_2) = f(x_1, x_2)$. More precisely, along the borders, the BCs impose

$$\begin{array}{rclrcl} f_{i_1,1-i_2} &=& f_{i_1,i_2}, & f_{i_1,n_2+i_2} &=& f_{i_1,n_2+1-i_2}, \text{for any } i_1 = 1,\ldots,n_1,\ i_2 = 1,\ldots,q_2 \\ f_{1-i_1,i_2} &=& f_{i_1,i_2}, & f_{n_1+i_1,i_2} &=& f_{n_1+1-i_1,i_2}, \text{for any } i_1 = 1,\ldots,q_1,\ i_2 = 1,\ldots,n_2, \end{array}$$

and, at the corners, the BCs impose for any $i_1 = 1,\ldots,q_1$, $i_2 = 1,\ldots,q_2$

$$\begin{array}{rclrcl} f_{1-i_1,1-i_2} &=& f_{i_1,i_2}, & f_{n_1+i_1,n_2+i_2} &=& f_{n_1+1-i_1,n_2+1-i_2}, \\ f_{1-i_1,n_2+i_2} &=& f_{i_1,n_2+1-i_2}, & f_{n_1+i_1,1-i_2} &=& f_{n_1+1-i_1,i_2}, \end{array}$$

i.e. a double reflection, first with respect to one axis and after with respect to the other, no matter about the order.

As a consequence the rectangular matrix $\widetilde{A}$ is reduced to a square Toeplitz-plus-Hankel block matrix with Toeplitz-plus-Hankel blocks, i.e. $A_n$ shows the two-level Toeplitz-plus-Hankel structure. Moreover, if the blurring operator satisfies the strong symmetry condition, i.e. it is symmetric with respect to each direction, formally

$$h_{|i|} = h_i \quad \text{for any } i = -q, \ldots, q. \tag{5.7}$$

then the matrix $A_n$ belongs to DCT-III matrix algebra. Therefore, its spectral decomposition can be computed very efficiently using the fast discrete cosine transform (DCT-III) [114]. More in detail, let $\mathcal{C}_n = \{A_n \in \mathbb{R}^{N(n)\times N(n)}, n = (n_1, n_2), N(n) = n_1 n_2 \mid A_n = R_n \Lambda_n R_n^T\}$ be the two-level DCT-III matrix algebra, i.e. the algebra of matrices that are simultaneously diagonalized by the orthogonal transform

$$R_n = R_{n_1} \otimes R_{n_2}, \quad R_m = \left[ \sqrt{\frac{2-\delta_{t,1}}{m}} \cos\left\{ \frac{(s-1)(t-1/2)\pi}{m} \right\} \right]_{s,t=1}^{m}, \tag{5.8}$$

with $\delta_{s,t}$ denoting the Kronecker symbol. Thus, the explicit structure of the matrix is $A_n = \text{Toeplitz}(V) + \text{Hankel}(\sigma(V), J\sigma(V))$, with $V = [V_0\ V_1\ \ldots\ V_{q_1}\ 0 \ldots 0]$ and where each $V_{i_1}$, $i_1 = 1, \ldots, q_1$ is the unilevel DCT-III matrix associated to the $i_1^{th}$ row of the PSF mask, i.e. $V_{i_1} = \text{Toeplitz}(v_{i_1}) + \text{Hankel}(\sigma(v_{i_1}), J\sigma(v_{i_1}))$, with $v_{i_1} = [h_{i_1,0}, \ldots, h_{i_1,q_2}, 0, \ldots, 0]$. Here, we denote by $\sigma$ the shift operator such that $\sigma(v_{i_1}) = [h_{i_1,1}, \ldots, h_{i_1,q_2}, 0, \ldots, 0]$ and by $J$ the usual flip matrix; at the block level the same operations are intended in block-wise sense.

Beside this structural characterization, the spectral description is completely known. In fact, let $f$ be the bivariate generating function associated to the PSF mask (5.3), that is

$$\begin{aligned} f(x_1, x_2) \;=\; & h_{0,0} + 2\sum_{s_1=1}^{q_1} h_{s_1,0} \cos(s_1 x_1) + 2\sum_{s_2=1}^{q_2} h_{0,s_2} \cos(s_2 x_2) \\ & +4 \sum_{s_1=1}^{q_1} \sum_{s_2=1}^{q_2} h_{s_1,s_2} \cos(s_1 x_1) \cos(s_2 x_2), \end{aligned} \tag{5.9}$$

then the eigenvalues of the corresponding matrix $A_n \in \mathcal{C}_n$ are given by

$$\lambda_s(A_n) = f\left(x_{s_1}^{[n_1]}, x_{s_2}^{[n_2]}\right), \; s = (s_1, s_2), \quad x_r^{[m]} = \frac{(r-1)\pi}{m},$$

where $s_1 = 1, \ldots, n_1$, $s_2 = 1, \ldots, n_2$, and where the two-index notation highlights the tensorial structure of the corresponding eigenvectors. Finally, notice that standard operations like matrix-vector products, resolution of linear systems and eigenvalues evaluations can be performed by means of FCT-III [92] within $O(n_1 n_2 \log(n_1 n_2))$ arithmetic operations (ops).

### 5.1.2 Anti-reflective boundary conditions

More recently, Anti-reflective boundary conditions (AR-BCs) have been proposed in [106] and studied [6, 7, 4, 51, 47, 50, 98, 116]. The improvement is due to the fact that not only the continuity of the image, but also of the normal derivative, are guaranteed at the boundary. This regularity, which is not shared with Dirichlet or periodic BCs, and only partially shared with reflective BCs, significantly reduces typical ringing artifacts.

The key idea is simply to assume that the scene outside the FOV is the anti-reflection of the scene inside the FOV. For example, with a boundary at $x_1 = 0$ the anti-reflexive condition impose $f(-x_1, x_2) - f(x_1^*, x_2) = -(f(x_1, x_2) - f(x_1^*, x_2))$, for any $x_2$, where $x_1^*$ is the center of the one-dimensional anti-reflection, i.e.

$$f(-x_1, x_2) = 2f(x_1^*, x_2) - f(x_1, x_2), \text{ for any } x_2.$$

In order to preserve a tensorial structure, at the corners, a double anti-reflection, first with respect to one axis and after with respect to the other, is considered, so that the BCs impose

$$f(-x_1, -x_2) = 4f(x_1^*, x_2^*) - 2f(x_1^*, x_2) - 2f(x_1, x_2^*) + f(x_1, x_2),$$

where $(x_1^*, x_2^*)$ is the center of the two-dimensional anti-reflection.

More precisely, by choosing as center of the anti-reflection the first available data, along the borders, the BCs impose

$$\begin{array}{l} f_{1-i_1,i_2}=2f_{1,i_2}-f_{i_1+1,i_2}, \quad f_{n_1+i_1,i_2}=2f_{n_1,i_2}-f_{n_1-i_1,i_2}, \; i_1 = 1,\ldots,q_1, \; i_2 = 1,\ldots,n_2, \\ f_{i_1,1-i_2}=2f_{i_1,1}-f_{i_1,i_2+1}, \quad f_{i_1,n_2+i_2}=2f_{i_1,n_2}-f_{i_1,n_2-i_2}, \; i_1 = 1,\ldots,n_1, \; i_2 = 1,\ldots,q_2. \end{array}$$

At the corners, the BCs impose for any $i_1 = 1, \ldots, q_1$ and $i_2 = 1, \ldots, q_2$,

$$\begin{aligned} f_{1-i_1,1-i_2} &= 4f_{1,1} - 2f_{1,i_2+1} - 2f_{i_1+1,1} + f_{i_1+1,i_2+1}, \\ f_{1-i_1,n_2+i_2} &= 4f_{1,n_2} - 2f_{1,n_2-i_2} - 2f_{i_1+1,n_2} + f_{i_1+1,n_2-i_2}, \\ f_{n_1+i_1,1-i_2} &= 4f_{n_1,1} - 2f_{n_1,i_2+1} - 2f_{n_1-i_1,1} + f_{n_1-i_1,i_2+1}, \\ f_{n_1+i_1,n_2+i_2} &= 4f_{n_1,n_2} - 2f_{n_1,n_1-i_2} - 2f_{n_1-i_1,n_2} + f_{n_1-i_1,n_2-i_2}. \end{aligned}$$

As a consequence the rectangular matrix $\widetilde{A}$ is reduced to a square Toeplitz-plus-Hankel block matrix with Toeplitz-plus-Hankel blocks, plus an additional structured low rank matrix. More details on this structure in the general case are reported in §5.2. Hereafter, we observe that again under the assumption of strong symmetry of the PSF and of a mild finite support condition (more precisely $h_i = 0$ if $|i_j| \geq n-2$, for some $j \in \{1,2\}$), the resulting linear system $A_n f = b$ is such that $A_n$ belongs to the $\mathcal{AR}_n^{2D}$ commutative matrix algebra [7]. This new algebra shares some properties with the $\tau$ (or DST-I) algebra [17].

Going inside the definition, a matrix $A_n \in \mathcal{AR}_n^{2D}$ has the following block structure

$$A_n = \left[\begin{array}{c|c|c} \tilde{H}_0 + Z_1 & 0^T & 0 \\ \hline \tilde{H}_1 + Z_2 & & 0 \\ \vdots & & \vdots \\ \tilde{H}_{q_1-1} + Z_{q_1} & & 0 \\ \tilde{H}_{q_1} & \tau(\tilde{H}_0, \dots, \tilde{H}_{q_1}) & \tilde{H}_{q_1} \\ 0 & & \tilde{H}_{q_1-1} + Z_{q_1} \\ \vdots & & \vdots \\ 0 & & \tilde{H}_1 + Z_2 \\ \hline 0 & 0^T & \tilde{H}_0 + Z_1 \end{array}\right], \qquad (5.10)$$

where $\tau(\tilde{H}_0, \dots, \tilde{H}_{q_1})$ is a block $\tau$ matrix with respect to the $\mathcal{AR}^{1D}$ blocks $\tilde{H}_{i_1}$, $i_1 = 1, \dots, q_1$ and $Z_k = 2\sum_{t=k}^{q_1} \tilde{H}_t$ for $k = 1, \dots, q_1$. In particular, the $\mathcal{AR}^{1D}$ block $\tilde{H}_{i_1}$ is associated to $i_1^{th}$ row of the PSF, i.e. $h_{i_1}^{[1D]} = [h_{i_1,i_2}]_{i_2=-q_2,\dots,q_2}$ and it is defined as

$$\tilde{H}_{i_1} = \left[\begin{array}{c|c|c} h_{i_1,0} + z_{i_1,1} & 0^T & 0 \\ \hline h_{i_1,1} + z_{i_1,2} & & 0 \\ \vdots & & \vdots \\ h_{i_1,q_2-1} + z_{i_1,q_2} & & 0 \\ h_{i_1,q_2} & \tau(h_{i_1,0}, \dots, h_{i_1,q_2}) & h_{i_1,q_2} \\ 0 & & h_{i_1,q_2-1} + z_{i_1,q_2} \\ \vdots & & \vdots \\ 0 & & h_{i_1,1} + z_{i_1,2} \\ \hline 0 & 0^T & h_{i_1,0} + z_{i_1,1} \end{array}\right], \qquad (5.11)$$

where $z_{i_1,k} = 2\sum_{t=k}^{q_2} h_{i_1,t}$ for $k = 1, \dots, q_2$ and $\tau(h_{i_1,0}, \dots, h_{i_1,q_2})$ is the unilevel $\tau$ matrix associated to the one-dimensional PSF $h_{i_1}^{[1D]}$ previously defined. Notice that the rank-1 correction given by the elements $z_{i_1,k}$ pertains to the contribution of the anti-reflection centers with respect to the vertical borders, while the low rank correction given by the matrices $Z_k$ pertains to the contribution of the anti-reflection centers with respect to the horizontal borders.

It is evident from the above matrix structure that favorable computational properties are guaranteed also by virtue of the $\tau$ structure in the central block. Therefore, firstly we recall the relevant properties of the two-level $\tau$ algebra [17]. Let $\mathcal{T}_n = \{A_n \in \mathbb{R}^{N(n)\times N(n)}, n = (n_1, n_2), N(n) = n_1 n_2 \mid A_n = Q_n \Lambda_n Q_n\}$ be the two-level $\tau$ matrix algebra, i.e. the algebra of matrices that are simultaneously diagonalized by the symmetric orthogonal transform

$$Q_n = Q_{n_1} \otimes Q_{n_2}, \quad Q_m = \left[\sqrt{\frac{2}{m+1}} \sin\left\{\frac{st\pi}{m+1}\right\}\right]_{s,t=1}^{m}. \tag{5.12}$$

With the same notation as the DCT-III algebra case, the explicit structure of the matrix is two level Toeplitz-plus-Hankel. More precisely,

$$A_n = \text{Toeplitz}(V) - \text{Hankel}(\sigma^2(V), J\sigma^2(V))$$

with $V = [V_0\ V_1\ \ldots\ V_{q_1}\ 0\ldots0]$, where each $V_{i_1}$, $i_1 = 1,\ldots,q_1$ is a the unilevel $\tau$ matrix associated to the $i_1^{th}$ row of the PSF mask, i.e. $V_{i_1} = \text{Toeplitz}(v_{i_1}) - \text{Hankel}(\sigma^2(v_{i_1}), J\sigma^2(v_{i_1}))$ with $v_{i_1} = [h_{i_1,0},\ldots,h_{i_1,q_2},0,\ldots,0]$. Here, we denote by $\sigma^2$ the double shift operator such that $\sigma^2(v_{i_1}) = [h_{i_1,2},\ldots,h_{i_1,q_2},0,\ldots,0]$; at the block level the same operations are intended in block-wise sense. Once more, the spectral characterization is completely known since for any $A_n \in \mathcal{T}_n$ the related eigenvalues are given by

$$\lambda_s(A_n) = f\left(x_{s_1}^{[n_1]}, x_{s_2}^{[n_2]}\right), s = (s_1, s_2), \quad x_r^{[m]} = \frac{r\pi}{m+1},$$

where $s_1 = 1,\ldots,n_1$, $s_2 = 1,\ldots,n_2$, and $f$ is the bivariate generating function associated to the PSF defined in (5.9).

As in the DCT-III case, standard operations like matrix-vector products, resolution of linear systems and eigenvalues evaluations can be performed by means of FST-I within $O(n_1 n_2 \log(n_1 n_2))$ (ops). Now, with respect to the $\mathcal{AR}_n^{2D}$ matrix algebra, a complete spectral characterization is given in [7, 4]. A really useful fact is the existence of a transform $T_n$ that simultaneously diagonalizes all the matrices belonging to $\mathcal{AR}_n^{2D}$, although the orthogonality property is partially lost.

**Theorem 5.1** *[4] Any matrix $A_n \in \mathcal{AR}_n^{2D}$, $n = (n_1, n_2)$, can be diagonalized by $T_n$, i.e.*

$$A_n = T_n \Lambda_n T_n^{-1},$$

*where* $T_n = T_{n_1} \otimes T_{n_2}$, $T_n^{-1} = T_{n_1}^{-1} \otimes T_{n_2}^{-1}$, *with*

$$T_m = \begin{bmatrix} \alpha_m^{-1} & 0^T & 0 \\ \alpha_m^{-1} p & Q_{m-2} & \alpha_m^{-1} Jp \\ 0 & 0^T & \alpha_m^{-1} \end{bmatrix} \text{ and } T_m^{-1} = \begin{bmatrix} \alpha_m & 0^T & 0 \\ -Q_{m-2}p & Q_{m-2} & -Q_{m-2}Jp \\ 0 & 0^T & \alpha_m \end{bmatrix}$$

*The entries of the vector* $p \in \mathbb{R}^{m-2}$ *are defined as* $p_j = 1 - j/(m-1)$, $j = 1, \ldots, m-2$, $J \in \mathbb{R}^{m-2 \times m-2}$ *is the flip matrix, and* $\alpha_m$ *is a normalizing factor chosen such that the Euclidean norm of the first and last column of* $T_m$ *will be equal to* 1*.*

**Theorem 5.2** *[7] Let* $A_n \in \mathcal{AR}_n^{2D}$, $n = (n_1, n_2)$, *the matrix related to the PSF* $h_{PSF} = [h_{i_1,i_2}]_{i_1=-q_1,\ldots,q_1, i_2=-q_2,\ldots,q_2}$. *Then, the eigenvalues of* $A_n$ *are given by*

- 1 *with algebraic multiplicity* 4*;*
- *the* $n_2 - 2$ *eigenvalues of the unilevel* $\tau$ *matrix related to the one-dimensional PSF* $h^{\{r\}} = [\sum_{i_1=-q_1}^{q_1} h_{i_1,-q_2}, \ldots, \sum_{i_1=-q_1}^{q_1} h_{i_1,q_2}]$, *each one with algebraic multiplicity* 2*;*
- *the* $n_1 - 2$ *eigenvalues of the unilevel* $\tau$ *matrix related to the one-dimensional PSF* $h^{\{c\}} = [\sum_{i_2=-q_2}^{q_2} h_{-q_1,i_2}, \ldots, \sum_{i_2=-q_2}^{q_2} h_{q_1,i_2}]$, *each one with algebraic multiplicity* 2*;*
- *the* $(n_1-2)(n_2-2)$ *eigenvalues of the two-level* $\tau$ *matrix related to the two-dimensional PSF* $h_{PSF}$*.*

Notice that the three sets of multiple eigenvalues are exactly related to the type of low rank correction imposed by the BCs through the centers of the anti-reflections. More in detail, the eigenvalues of $\tau_{n_2-2}(h^{\{r\}})$ and of $\tau_{n_1-2}(h^{\{c\}})$ take into account the condensed PSF information considered along the horizontal and vertical borders respectively, while the eigenvalue equal to 1 takes into account the condensed information of the whole PSF at the four corners. In addition, it is worth noticing that the spectral characterization can be completely described in terms of the generating function associated to the PSF defined in (5.9), simply by extending to 0 the standard $\tau$ evaluation grid, i.e. it holds

$$\lambda_s(A_n) = f\left(x_{s_1}^{[n_1]}, x_{s_2}^{[n_2]}\right), s = (s_1, s_2), s_j = 0, \ldots, n_j, \quad x_r^{[m]} = \frac{r\pi}{m+1},$$

where the 0−index refers to the first/last columns of the matrix $T_m$ [7]. See [6, 4] for some algorithms related to standard operations like matrix-vector products, resolution of linear systems and eigenvalues evaluations with a computational cost of $O(n_1 n_2 \log(n_1 n_2))$ ops. It is worthwhile stressing that the computational cost of the inverse transform is comparable with that of the direct transform and, at least at first sight, the very true penalty is the loss of orthogonality due to the first/last column of the matrix $T_m$.

## 5.2 Theoretical results on optimal preconditioning

In this section we consider in more detail the matrices arising when Anti-Reflective BCs are applied to the case of a non-symmetric PSF, the aim being to define the corresponding optimal preconditioner in the $\mathcal{AR}_n^{2D}$ algebra. More precisely, let $A = A(h)$ be the Anti-Reflective matrix generated by the generic PSF $h_{PSF} = [h_{i_1,i_2}]_{i_1=-q_1,\dots,q_1, i_2=-q_2,\dots,q_2}$ and let $P = P(s) \in \mathcal{AR}_n^{2D}$ be the Anti-Reflective matrix generated by the symmetrized PSF $s_{PSF} = [s_{i_1,i_2}]_{i_1=-q_1,\dots,q_1, i_2=-q_2,\dots,q_2}$. We are looking for the optimal preconditioner $P^* = P^*(s^*)$ in the sense that

$$P^* = \arg\min_{P\in\mathcal{AR}_n^{2D}} \|A-P\|_{\mathcal{F}}^2 \ , \quad s^* = \arg\min_{s} \|A(h)-P(s)\|_{\mathcal{F}}^2 \ , \tag{5.13}$$

where $\|\cdot\|_{\mathcal{F}}$ is the Frobenius norm, defined as $\|A\|_{\mathcal{F}} = \sqrt{\sum_{i,j} |a_{i,j}|^2}$. Indeed, an analogous result is know in [92] with respect to Reflective BCs: given a generic PSF $h_{PSF} = [h_{i_1,i_2}]$, the optimal preconditioner in the DCT-III matrix algebra is generated by the strongly symmetric PSF $s_{PSF} = [s_{i_1,i_2}]$, given by

$$s_{\pm i_1,\pm i_2} = \frac{h_{-i_1,-i_2} + h_{-i_1,i_2} + h_{i_1,-i_2} + h_{i_1,i_2}}{4}, \tag{5.14}$$

which is a symmetrization of the original PSF. Our interest is clearly motivated by the computational facilities proper of $\mathcal{AR}_n^{2D}$ algebra, coupled with its better approximation properties. We preliminary consider the one-dimensional case in order to introduce the key idea in the proof with a simpler notation. Moreover, the proof argument of the two-dimensional case is also strongly connected to the one-dimensional one.

### 5.2.1 One-dimensional case

Let us consider a generic PSF $h_{PSF} = [h_i]_{i=-q,\dots,q}$. As introduced in §5.1.2, the idea is to apply an anti-reflection with respect to the border points $f_1$ and $f_n$. Thus, we impose

$$f_{1-i} = 2f_1 - f_{1+i}, \quad f_{n+i} = 2f_n - f_{n-i}, \quad i = 1, \dots, q.$$

The resulting matrix shows a more involved structure with respect to the Reflective BCs, i.e. it is Toeplitz + Hankel plus a structured low rank correction matrix, as follows

$$A = \left[\begin{array}{c|c|c} v_0 & u^T & 0 \\ \hline v_1 & & \\ \vdots & & \\ v_q & B & w_q \\ & & \vdots \\ & & w_1 \\ \hline 0 & -(Ju)^T & w_0 \end{array}\right] \tag{5.15}$$

with

$$\begin{aligned} u^T &= [h_{-1} - h_1, \dots, h_{-q} - h_q, 0, \dots, 0], \\ -(Ju)^T &= [0, \dots, 0, h_q - h_{-q}, \dots, h_1 - h_{-1}], \\ v_k &= h_k + 2\sum_{j=k+1}^{q} h_j, \quad w_k = h_{-k} + 2\sum_{j=k+1}^{q} h_{-j}, \\ B &= T([h_{-q}, \dots, h_q]) - H_{\mathrm{TL}}([h_2, \dots, h_q]) - H_{\mathrm{BR}}([h_{-2}, \dots, h_{-q}]), \end{aligned}$$

where $T([h_{-q}, \dots, h_q])$ is the Toeplitz matrix associated to the PSF $h_{PSF}$,

$$T([h_{-q}, \dots, h_q]) = \begin{pmatrix} h_0 & h_{-1} & \cdots & h_{-q} & 0 & \cdots & 0 \\ h_1 & \ddots & \ddots & & h_{-q} & \ddots & \vdots \\ \vdots & \ddots & & & & h_{-q} & 0 \\ h_q & & & \ddots & & & h_{-q} \\ 0 & h_q & & & & \ddots & \vdots \\ \vdots & \ddots & h_q & & \ddots & \ddots & h_{-1} \\ 0 & \cdots & 0 & h_q & \cdots & h_1 & h_0 \end{pmatrix},$$

while $H_{\mathrm{TL}}([h_2,\dots,h_q])$ and $H_{\mathrm{BR}}([h_{-2},\dots,h_{-q}])$ are respectively the top-left Hankel and the bottom-right Hankel matrices

$$H_{\mathrm{TL}}([h_2,\dots,h_q]) = \begin{bmatrix} h_2 & h_3 & \cdots & h_q & 0 & \cdots & 0 \\ h_3 & & h_q & 0 & & & \vdots \\ \vdots & h_q & 0 & & & & \\ h_q & 0 & & & & & \\ 0 & & & & & & \\ \vdots & & & & & & \vdots \\ 0 & \cdots & & & & \cdots & 0 \end{bmatrix},$$

$$H_{\mathrm{BR}}([h_{-2},\dots,h_{-q}] = \begin{bmatrix} 0 & \cdots & & & & \cdots & 0 \\ \vdots & & & & & & \vdots \\ & & & & & & 0 \\ & & & & & 0 & h_{-q} \\ & & & & 0 & h_{-q} & \vdots \\ \vdots & & & 0 & h_{-q} & & h_{-3} \\ 0 & \cdots & 0 & h_{-q} & \cdots & h_{-3} & h_{-2} \end{bmatrix}.$$

On the other hand, the Anti-Reflective matrix $P \in \mathcal{AR}^{1D}$ generated by a strongly symmetric PSF $s_{PSF} = [s_q,\dots,s_1,s_0,s_1,\dots,s_q]$, among which the minimizer $P^*$ in (5.13) will be searched, is clearly given by

$$P = \left[\begin{array}{c|c|c} r_0 & 0^T & 0 \\ \hline r_1 & & \\ \vdots & & \\ r_q & \tau(s) & r_q \\ & & \vdots \\ & & r_1 \\ \hline 0 & 0^T & r_0 \end{array}\right]$$

where $r_k = s_k + 2\sum_{j=k+1}^{q} s_j$ and $\tau(s)$ is the $\tau$ (or DST-I) matrix generated by the PSF $s_{PSF}$.

The optimality of the Anti-Reflective matrix generated by the symmetrized PSF defined as

$$s_{\pm i} = \frac{h_{-i} + h_i}{2}. \tag{5.16}$$

can be proved analogously as in [92] with respect to the internal part $C_I = B - \tau(s)$ and by invoking a non-overlapping splitting argument in order to deal with the external border $C_B$. In fact, we have

$$\|C\|_{\mathcal{F}}^2 = \|C_I\|_{\mathcal{F}}^2 + \|C_B\|_{\mathcal{F}}^2$$

and it easy to show that the minimizer found for the first term is the same than for the second one.

Notice that $\|u^T\|^2$ and $\|-(Ju)^T\|^2$ are constant terms in the minimization process. So, as naturally expected, the obtained minimum value will be greater, the greater is the loss of symmetry in the PSF. Moreover, with the choice (5.16), the first and last column in $C_B$ share the same norm, i.e. again the most favourable situation. It is worth stressing that the minimization process of the second term $C_B$ allows to highlight as the tuning of each minimization parameter can be performed just by considering two proper corresponding entries in the matrix, i.e.

$$(r_p - v_p) + (r_p - w_p) = 0, \quad p = 0, \dots, q$$

where $v_p$ and $w_p$ are linear combination of the same coefficients with positive and negative indices, respectively. Taking this fact in mind, we can now consider a more geometrical approach to the proof, that allows to greatly simplify also the proof with respect to the minimization of the internal part. Moreover this proof can be applied to any type of BCs based on the fact that the values of unknowns outside the FOV are fixed or are defined as linear combinations of the unknowns inside the FOV.

**Theorem 5.3** *Let $A = A(h)$ be the Anti-Reflective matrix generated by the generic PSF $h_{PSF} = [h_i]_{i=-q,\dots,q}$. The optimal preconditioner in the $\mathcal{AR}_n^{1D}$ algebra is the matrix associated with the symmetrized PSF $s_{PSF} = [s_q, \dots, s_1, s_0, s_1, \dots, s_q]$, with*

$$s_i = \frac{h_{-i} + h_i}{2}. \tag{5.17}$$

**Proof.** Preliminarly, as shown in Figure 5.1, we simply observe that if we consider in the Cartesian plane a point $R = (R_x, R_y)$, its optimal approximation $Q^*$, among the points $Q = (Q_x, Q_y)$ such that $Q_x = Q_y$, is obtained as the intersection between the line $y = x$, with the perpendicular line that pass through $R$, that is

$$\begin{cases} y - R_y = -(x - R_x) \\ y = x \end{cases}$$

hence $Q_x^* = Q_y^* = (R_x + R_y)/2$. The same holds true if we consider the swapped point $R^S = (R_y, R_x)$, since they share the same distance, i.e. $d(R, Q^*) = d(R^S, Q^*)$.

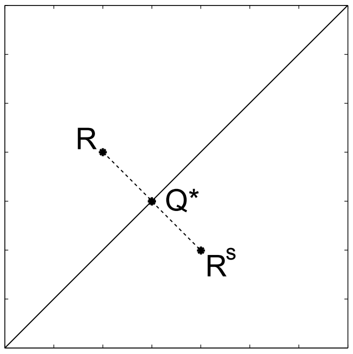


Figure 5.1: A point $R$, its swapped point $R^S$, the optimal approximation of both $Q^*$.

Clearly, due to linearity of obtained expression, this result can be extended also to the case of any linear combination of coordinates. Thus, by explicitly exploiting the structure of $A$ and $P$, we define as x-coordinate of a point the entry with negative index and as y-coordinate of the same point the corresponding entry with positive index. For the sake of simplicity we report an example for $q = 3$, in which we put in evidence the x or y coordinate definition,

$$\begin{aligned}
C &= A - P = \\
&= \begin{bmatrix}
\omega_0^y & \nu_1^x & \nu_2^x & \nu_3^x & & & \\
\omega_1^y & \zeta_0^y & \zeta_2^x & \theta_2^x & \theta_3^x & & \\
\omega_2^y & \zeta_1^y & \theta_0 & \theta_1^x & \theta_2^x & \theta_3^x & \\
\omega_3^y & \theta_2^y & \theta_1^y & \theta_0 & \theta_1^x & \theta_2^x & \omega_3^x \\
 & \theta_3^y & \theta_2^y & \theta_1^y & \theta_0 & \zeta_1^x & \omega_2^x \\
 & & \theta_3^y & \theta_2^y & \zeta_2^y & \zeta_0^x & \omega_1^x \\
 & & & \nu_3^y & \nu_2^y & \nu_1^y & \omega_0^x
\end{bmatrix} - \begin{bmatrix}
\hat{\omega}_0^y & 0 & 0 & 0 & & & \\
\hat{\omega}_1^y & \hat{\zeta}_0^y & \hat{\zeta}_2^x & \hat{\theta}_2^x & \hat{\theta}_3^x & & \\
\hat{\omega}_2^y & \hat{\zeta}_1^y & \hat{\theta}_0 & \hat{\theta}_1^x & \hat{\theta}_2^x & \hat{\theta}_3^x & \\
\hat{\omega}_3^y & \hat{\theta}_2^y & \hat{\theta}_1^y & \hat{\theta}_0 & \hat{\theta}_1^x & \hat{\theta}_2^x & \hat{\omega}_3^x \\
 & \hat{\theta}_3^y & \hat{\theta}_2^y & \hat{\theta}_1^y & \hat{\theta}_0 & \hat{\zeta}_1^x & \hat{\omega}_2^x \\
 & & \hat{\theta}_3^y & \hat{\theta}_2^y & \hat{\zeta}_2^y & \hat{\zeta}_0^x & \hat{\omega}_1^x \\
 & & & 0 & 0 & 0 & \hat{\omega}_0^x
\end{bmatrix},
\end{aligned}$$

Here, we set the points

$$\begin{aligned}
\Theta_i &= (\theta_i^x, \theta_i^y) = (h_{-i}, h_i) \\
\Omega_i &= (\omega_i^x, \omega_i^y) = (h_{-i} + 2\sum_{j=i+1}^{q} h_{-j}, h_i + 2\sum_{j=i+1}^{q} h_j) = \\
&= (\theta_i^x + 2\sum_{j=i+1}^{q} \theta_j^x, \theta_i^y + 2\sum_{j=i+1}^{q} \theta_j^y) \\
N_i &= (\nu_i^x, \nu_i^y) = (h_{-i} - h_i, h_i - h_{-i}) = (\theta_i^x - \theta_i^{Sx}, \theta_i^y - \theta_i^{Sy})
\end{aligned}$$

and

$$\begin{aligned}
Z_0 &= (\zeta_0^x, \zeta_0^y) = (h_0 - h_{-2}, h_0 - h_2) = (\theta_0^x - \theta_2^x, \theta_0^y - \theta_2^y)\\
Z_1 &= (\zeta_1^x, \zeta_1^y) = (h_{-1} - h_{-3}, h_1 - h_3) = (\theta_1^x - \theta_3^x, \theta_1^y - \theta_3^y)\\
Z_2 &= (\zeta_2^x, \zeta_2^y) = (h_{-1} - h_3, h_1 - h_{-3}) = (\theta_1^x - \theta_3^{Sx}, \theta_1^y - \theta_3^{Sy})
\end{aligned}$$

related to the Hankel corrections. The points $\hat{\Theta}_i$, $\hat{\Omega}_i$, $\hat{Z}_i$ related to the matrix $P$ are defined in a similar manner, taking into account the strong symmetry property, i.e. they have the same x and y coordinates. More in general, the key idea is to transform the original minimization problem in the equivalent problem of minimizing the quantity

$$\begin{aligned}
c_0\mathbf{d}(\Theta_0, \hat{\Theta}_0)^2 + \ldots + c_q\mathbf{d}(\Theta_q, \hat{\Theta}_q)^2 + \mathbf{d}(Z_0, \hat{Z}_0)^2 + \ldots + \mathbf{d}(Z_m, \hat{Z}_m)^2\\
+\mathbf{d}(\Omega_0, \hat{\Omega}_0)^2 + \ldots + \mathbf{d}(\Omega_q, \hat{\Omega}_q)^2 + \mathbf{d}(N_1, 0)^2 + \ldots + \mathbf{d}(N_q, 0)^2,
\end{aligned} \tag{5.18}$$

where $c_j$ are some constants taking into account the number of constant Toeplitz entries. Now, by referring to the initial geometrical observation, we start from points pertaining to the Toeplitz part, that can be minimized separately, and we obtain the minimizer (5.17). It is also an easy check to prove the same claim with respect to any other terms, by invoking the quoted linearity argument. ■

### 5.2.2 Two-dimensional case

Let us consider a generic PSF $h_{PSF} = [h_{i_1,i_2}]_{i_1=-q_1,\ldots,q_1, i_2=-q_2,\ldots,q_2}$. As introduced in §5.1.2, the idea is to apply an anti-reflection with respect to the border points $f_{1,i_2}$, $f_{i_1,1}$ and $f_{n_1,i_2}$, $f_{i_1,n_2}$, $i_1 = 1, \ldots, n_1$, $i_2 = 1, \ldots, n_2$, and a double anti-reflection at the corners in order to preserve the tensorial structure. The resulting matrix shows a more involved structure, i.e. it is block Toeplitz + Hankel with Toeplitz + Hankel blocks plus a structured low rank correction matrix, as follows

$$A = \left[\begin{array}{c|c|c}
V_0 & U & 0\\ \hline
V_1 & & \\
\vdots & & \\
V_{q_1} & B & W_{q_1}\\
 & & \vdots\\
 & & W_1\\ \hline
0 & -JU & W_0
\end{array}\right], \tag{5.19}$$

with

$$\begin{aligned}
U &= [H_{-1}-H_1,\ldots,H_{-q_1}-H_{q_1},0,\ldots,0]\,,\\
-JU &= [0,\ldots,0,H_{q_1}-H_{-q_1},\ldots,H_1-H_{-1}]\,,\\
V_j &= H_j+2\sum_{i=j+1}^{q_1}H_i,\quad W_j=H_{-j}+2\sum_{i=j+1}^{q_1}H_{-i},\\
B &= T(H_{-q_1},\ldots,H_{q_1})-H_{\mathrm{TL}}(H_2,\ldots,H_{q_1})-H_{\mathrm{BR}}(H_{-2},\ldots,H_{-q_1}).
\end{aligned}$$

where $T$ indicates the block Toeplitz matrix, while $H_{\mathrm{TL}}$ and $H_{\mathrm{BR}}$ are respectively the top-left block Hankel matrix and the bottom-right block Hankel matrix as just previously depicted in the unilevel setting

$$T(H_{-q_1},\ldots,H_{q_1}) = \begin{pmatrix}
H_0 & H_{-1} & \cdots & H_{-q_1} & 0 & \cdots & 0\\
H_1 & \ddots & \ddots & & H_{-q_1} & \ddots & \vdots\\
\vdots & \ddots & & & & H_{-q_1} & 0\\
H_{q_1} & & & \ddots & & & H_{-q_1}\\
0 & H_{q_1} & & & & \ddots & \vdots\\
\vdots & \ddots & H_{q_1} & & \ddots & \ddots & H_{-1}\\
0 & \cdots & 0 & H_{q_1} & \cdots & H_1 & H_0
\end{pmatrix},$$

$$H_{\mathrm{TL}}(H_2,\ldots,H_{q_1}) = \begin{pmatrix}
H_2 & H_3 & \cdots & H_{q_1} & 0 & \cdots & 0\\
H_3 & & H_{q_1} & 0 & & & \vdots\\
\vdots & H_{q_1} & 0 & & & & \\
H_{q_1} & 0 & & & & & \\
0 & & & & & & \\
\vdots & & & & & & \vdots\\
0 & \cdots & & & & \cdots & 0
\end{pmatrix},$$

$$H_{\mathrm{BR}}(H_{-2},\ldots,H_{-q_1}) = \begin{pmatrix}
0 & \cdots & & & & \cdots & 0\\
\vdots & & & & & & \vdots\\
 & & & & & & 0\\
 & & & & & 0 & H_{-q_1}\\
 & & & & 0 & H_{-q_1} & \vdots\\
\vdots & & & 0 & H_{-q_1} & & H_{-3}\\
0 & \cdots & 0 & H_{-q_1} & \cdots & H_{-3} & H_{-2}
\end{pmatrix}$$

and where the block $H_j$ is defined, according to (5.15), as

$$H_j = \left[\begin{array}{c|c|c} v_{j,0} & u_j^T & 0 \\ \hline v_{j,1} & & \\ \vdots & & \\ v_{j,q_2} & B_j & w_{j,q_2} \\ & & \vdots \\ & & w_{j,1} \\ \hline 0 & -(Ju_j)^T & w_{j,0} \end{array}\right], \tag{5.20}$$

with $B_j = T(h_{j,-q_2}, \ldots, h_{j,q_2}) - H_{\mathrm{TL}}(h_{j,2}, \ldots, h_{j,q_2}) - H_{\mathrm{BR}}(h_{j,-2}, \ldots, h_{j,-q_2})$.

Refer to (5.10) and (5.11) for the structure of the matrix $P$ related to a strongly symmetric PSF in which the minimizer $P^*$, see (5.13), will be searched.

**Theorem 5.4** *Let $A = A(h)$ be the Anti-Reflective matrix generated by the generic PSF $h_{PSF} = [h_{i_1,i_2}]_{i_1=-q_1,\ldots,q_1, i_2=-q_2,\ldots,q_2}$. The optimal preconditioner in the $\mathcal{AR}_n^{2D}$ algebra is the matrix associated with the symmetrized PSF $s_{PSF} = [s_{i_1,i_2}]_{i_1=-q_1,\ldots,q_1, i_2=-q_2,\ldots,q_2}$, with*

$$s_{\pm i_1, \pm i_2} = \frac{h_{-i_1,-i_2} + h_{-i_1,i_2} + h_{i_1,-i_2} + h_{i_1,i_2}}{4}. \tag{5.21}$$

**Proof.** The proof can be done by extending the geometrical approach just considered in the one-dimensional case: we simply observe that if we consider in the 4-dimensional space a point $R = (R_x, R_y, R_z, R_w)$, its optimal approximation $Q^*$ among the points $Q = (Q_x, Q_y, Q_z, Q_w)$ belonging to the line $\mathcal{L}$

$$\begin{cases} x = t \\ y = t \\ z = t \\ w = t \end{cases}$$

is obtained by minimizing the distance

$$\begin{aligned} \mathbf{d}^2(\mathcal{L}, R) &= (t - R_x)^2 + (t - R_y)^2 + (t - R_z)^2 + (t - R_w)^2 \\ &= 4t^2 - 2t(R_x + R_y + R_z + R_w) + R_x^2 + R_y^2 + R_z^2 + R_w^2. \end{aligned}$$

This is a trinomial of the form $\alpha t^2 + \beta t + \gamma$, with $\alpha > 0$ and we find the minimum by using the formula for computing the abscissa of the vertex of a parabola

$$t^* = -\frac{\beta}{2\alpha} = \frac{R_x + R_y + R_z + R_w}{4}.$$

Hence the point $Q^*$ is defined as $Q^*_x = Q^*_y = Q^*_z = Q^*_w = t^*$. The same holds true if we consider any swapped point $R^S$, not unique but depending on the permutation at hand, since they share the same distance, i.e. $d(R, Q^*) = d(R^S, Q^*)$. Again, thanks to the linearity of obtained expression, this result can be extended also in the case of any linear combination of coordinates.

Thus, by exploiting the structure of $A$ and $P$, we define a point by referring to the entry with positive and negative two-index. For instance, points pertaining to the Toeplitz part are defined as

$$\begin{aligned} \Theta_{i_1,i_2} &= (\theta^x_{i_1,i_2}, \theta^y_{i_1,i_2}, \theta^z_{i_1,i_2}, \theta^w_{i_1,i_2}) = (h_{-i_1,-i_2}, h_{-i_1,i_2}, h_{i_1,-i_2}, h_{i_1,i_2}), \\ \hat{\Theta}_{i_1,i_2} &= (\hat{\theta}^x_{i_1,i_2}, \hat{\theta}^y_{i_1,i_2}, \hat{\theta}^z_{i_1,i_2}, \hat{\theta}^w_{i_1,i_2}) = (s_{-i_1,-i_2}, s_{-i_1,i_2}, s_{i_1,-i_2}, s_{i_1,i_2}), \end{aligned}$$

respectively.

As in the unilevel setting, the original minimization problem is transformed in the equivalent problem of minimizing the sum of squared distances analogously as in (5.18). We start again from points pertaining to the Toeplitz part, that can be minimized separately, and we obtain the minimizer (5.21). It is also an easy check to prove the same claim with respect to any other couple of points pertaining to Hankel or low rank corrections, by invoking the quoted linearity argument. ■

It is worth stressing that this proof idea is very powerful in its generality. It can be applied to any type of BCs based on the fact that the values of unknowns outside the FOV are defined as linear combinations of the unknowns inside the FOV, so that it may be useful in the future to prove theoretical results for new proposed BCs.

## 5.3 Computational results

The problem of noise sensitivity in image restoration is usually addressed by using regularization methods, where one or some parameters play a key role in the regularization. In literature several techniques are known — like Tikhonov direct regularization [65] — but in large-scale problems the main choice is given by iterative methods, where the parameter is the number of iterations. A suitable stop prevents from reconstruction of noisy components in the approximated solution.

A well-known iterative method for solving the image deblurring problem is Landweber method [84]. A comprehensive description and analysis of it can be found in [113]. The $k$-th iteration step of that method is defined by

$$x_k = x_{k-1} + \tau A^H (b - A x_{k-1}), \tag{5.22}$$

where $A$ is the blurring matrix, $b$ is the recorded image, $\tau$ is the descent parameter, whose convergence interval is $0 < \tau < 2/\|A\|_2^2$; here we set it equal to one. From now on we will use the more familiar symbol $x$ — leaving $f$ — to denote approximated reconstructions and $\bar{x}$ to denote the true image. As one can observe experimentally, the restorations seem to converge in the initial iterations, before they become worse and finally diverge; this phenomenon is called semiconvergence. Indeed it is known that Landweber method is a regularization method [13, 99], where the number of steps $k$ is the regularization parameter. Moreover it has good stability properties, but it is usually very slow to converge to the sought solution. Therefore it is a good candidate for testing the proposed preconditioning technique. Thus we introduce the preconditioned Landweber method [99]

$$x_k = x_{k-1} + \tau D A^H (b - A x_{k-1}), \tag{5.23}$$

where $D$ is the preconditioner. In order to build it, we compute the eigenvalues $\lambda_j$ of the blurring matrix associated to the PSF and to periodic BCs (via FFT) or to the symmetrized PSF and to Reflective BCs (via FCT) or to the symmetrized PSF and to Anti-Reflective BCs (via FST, see Theorem 5.1 and Theorem 5.2 and comments below), then we apply the Tikhonov Filter

$$d_j = \frac{1}{|\lambda_j|^2 + \alpha} \tag{5.24}$$

to determine the eigenvalues $d_j$ of $D$; finally the PSF related to $D$ can be obtained via IFFT or IFCT or IFST (the inverses of the previous transforms, namely inverse FFT, inverse FCT, inverse FST). In these first numerical experiments we have set the parameter $\alpha$ manually, so that we have reached excellent performances both in terms of quality of the restorations and of acceleration of the method.

Actually in our implementation, which is partially based on the MatLab Toolbox RestoreTools [91] by James Nagy, we have never worked with $A^H$, but always with $A'$, that is the matrix related to the PSF rotated by 180 degrees. This approach is known in literature as *reblurring* strategy [47]. The reason behind this choice resides in one of the main problems of Anti-Reflective algebra $\mathcal{AR}$, i.e. the fact that it is not closed under transposition. We stress that $A^H$ and $A'$ are always the same thing in case of periodic and zero boundary conditions, but in general they are different for Reflective and Anti-Reflective ones.

To test these different BCs and preconditioning techniques, we have taken into account two test cases: a) the Cameraman deblurring problem of Figure 5.3, in which the PSF is a slightly non-symmetric portion of a Gaussian blur;

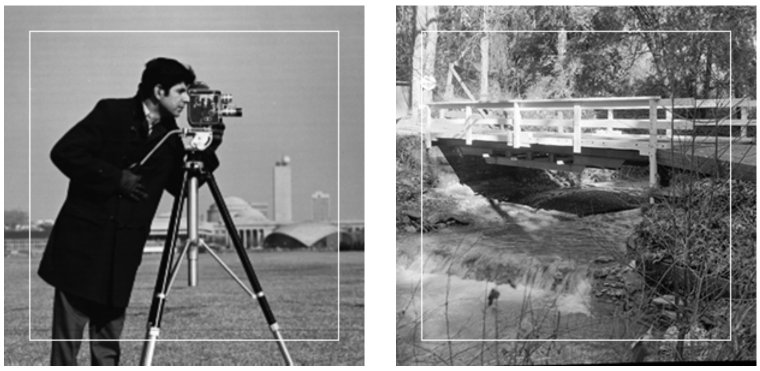


Figure 5.2: FOV delimited by white lines.

b) the Bridge deblurring problem of Figure 5.6, in which the PSF is an highly non-symmetric portion of a Gaussian blur. In both cases we have generated the blurred and noisy data $b$ without imposing any boundary conditions, but by cutting the border of a bigger image (Figure 5.2) — so simulating a real recorded image — and then adding about 0.2% of white Gaussian noise. We have employed BCs in the restoration process, where $A$ and $D$ are structured matrices associated respectively to periodic, reflective and anti-reflective BCs. We have chosen to add a low level of noise to emphasize the importance of boundary conditions, which play a leading role when the noise is low, while they become less decisive when it grows up. Since we know the true image $\bar{x}$, to measure the quality of the deblurred images we compute the Relative Restoration Error (RRE)

$$\|x - \bar{x}\|_{\mathcal{F}} \, / \, \|\bar{x}\|_{\mathcal{F}} \, , \tag{5.25}$$

where $\|\cdot\|_{\mathcal{F}}$ is the Frobenius norm and $x$ is the computed restoration. We report results relative to best restorations, i.e. we stop the iteration method when it reaches its lowest RRE.

As we expected, from Table 5.1 and Table 5.2, we can notice that both Reflective and Anti-Reflective boundary conditions outperform periodic ones, which give rise to poor restorations (see the first image of both Figure 5.4 and Figure 5.7). Furthermore by means of Anti-Reflective BCs (see the third image of both Figure 5.4 and Figure 5.7) we can gain restorations of better quality compared with ones obtained employing Reflective BCs (see the second image of both Figure 5.4 and Figure 5.7). From Tables 5.1, 5.2 and Figures 5.5, 5.8 we can see that all these considerations hold also for $D$-Landweber method — i.e. Landweber method with preconditioning — which for a suitable choice of the parameter $\alpha$ is able to reach restorations of the same quality of the classical Landweber method in much smaller number of steps. In particular the reduction in steps for both Reflective and Anti-Reflective BCs is around $10^2$ times for the Cameraman deblurring problem and around $10^1$ for the Bridge deblurring problem.

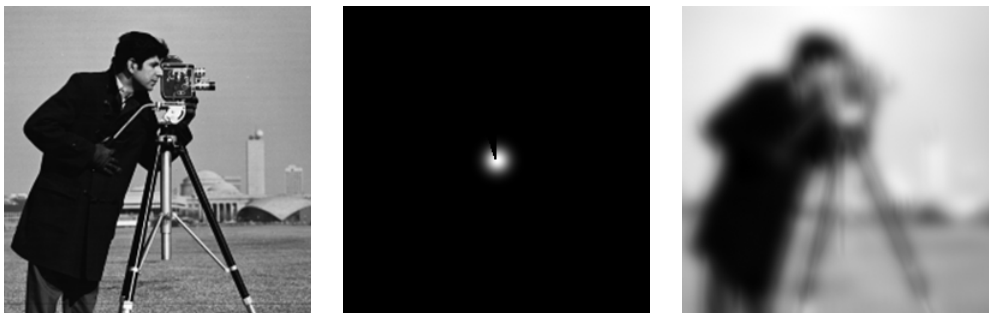


Figure 5.3: Cameraman deblurring problem: true image, PSF, blurred and noisy image.

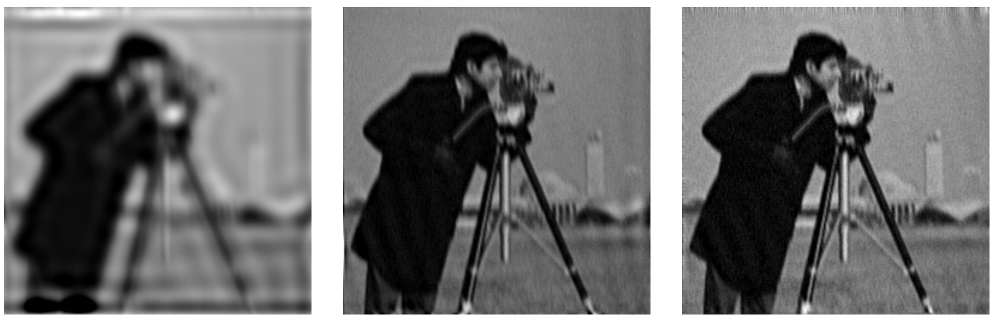


Figure 5.4: Best Landweber restorations, employing periodic, reflective, anti-reflective BCs.

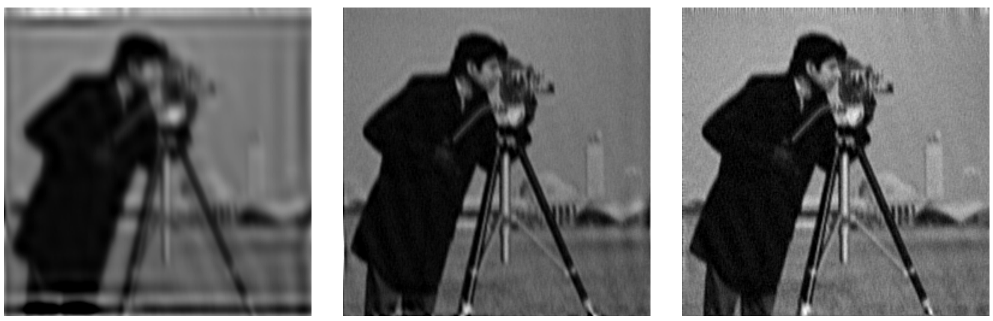


Figure 5.5: Best preconditioned Landweber restorations, employing periodic, reflective, anti-reflective BCs.

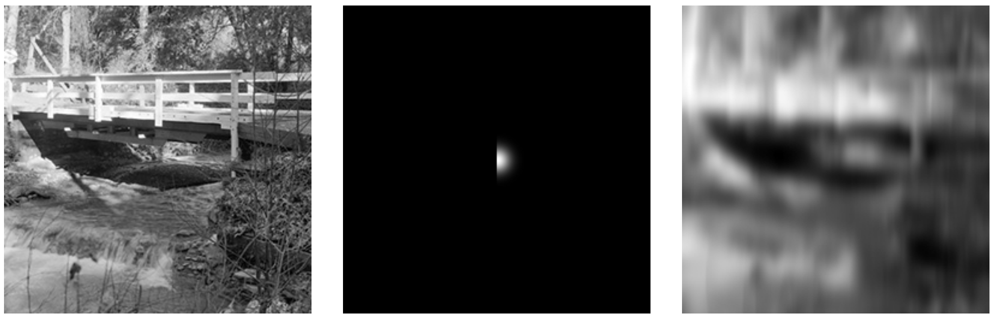


Figure 5.6: Bridge deblurring problem: true image, PSF, blurred and noisy image.

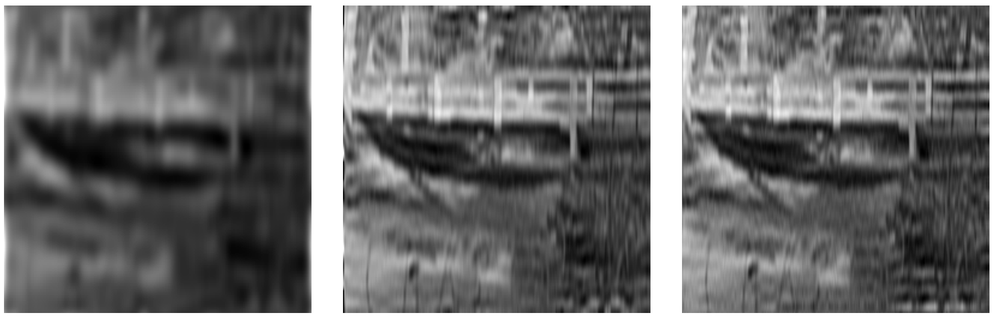


Figure 5.7: Best Landweber restorations, employing periodic, reflective, anti-reflective BCs.

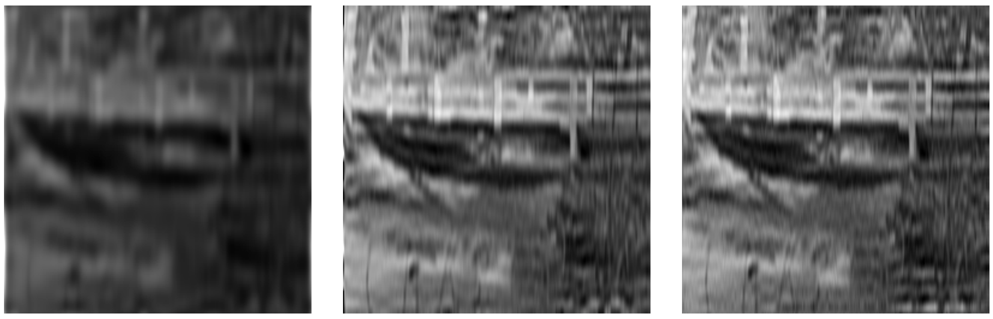


Figure 5.8: Best preconditioned Landweber restorations, employing periodic, reflective, anti-reflective BCs.

We stress that the iteration count — we remind that we stop the iterative method when it gets its best reconstruction in terms of RRE — reported in Table 5.2 in the Anti-Reflective row does not have to deceive, because, as it can be seen from Figure 5.9, if we compare the restorations gained by Landweber (preconditioned or not) at any given fixed iteration, employing

| | Landweber | | $D$-Landweber | |
|---|---|---|---|---|
| | RRE | IT | RRE | IT |
| Periodic | 0.2081 | 60 | 0.2053 | 1 |
| Reflective | 0.1125 | 3234 | 0.1121 | 20 |
| Anti-Reflective | 0.0999 | 4968 | 0.0999 | 50 |

Table 5.1: Results of the classical and preconditioned Landweber method related to the Cameraman deblurring problem, employing different BCs.

| | Landweber | | $D$-Landweber | |
|---|---|---|---|---|
| | RRE | IT | RRE | IT |
| Periodic | 0.2541 | 10 | 0.2535 | 2 |
| Reflective | 0.1983 | 3880 | 0.1983 | 429 |
| Anti-Reflective | 0.1868 | 15545 | 0.1868 | 2034 |

Table 5.2: Results of the classical and preconditioned Landweber method related to the Bridge deblurring problem, employing different BCs.

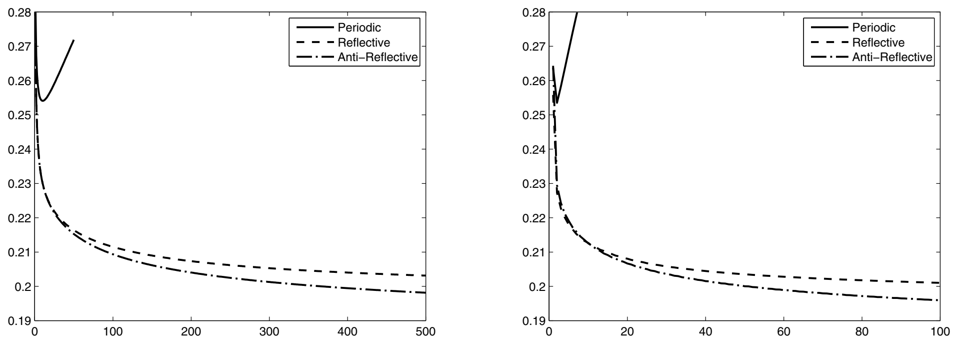


Figure 5.9: Bridge deblurring problem: RRE trends of Landweber (on the left) and $D$-Landweber (on the right) for different BCs.

Reflective BCs or Anti-Reflective BCs, we see that the latter shows always equal or better restoration quality. The same remark holds for the Cameraman deblurring problem (see Table 5.1). In fact Figure 5.9 is very instructive because it tells to the generic user two things: a) the curves for Reflective and Anti-Reflective BCs are very flat; b) the approximation obtained when using Anti-Reflective BCs is always better or equal to that obtained with Reflective BCs. The combined message of the previous two items is that we can safely choose the Anti-Reflective BCs, even when we are unable to estimate precisely the stopping criterion for deciding the optimal iteration: we notice that this observation does not hold for the periodic BCs where a small error in the evaluation of the optimal iteration leads to a substantial deterioration of the quality of the resulting restored image.

In the end, from the results reported in this section we can say that our proposal of the optimal preconditioner in the context of Anti-Reflective BCs is as effective as the one introduced in [92] for Reflective BCs. Therefore the present work represents a theoretical and numerical continuation and strengthening of that line of research.

# Chapter 6

# $Z$ variant

Recalling the notation introduced in the previous chapter, our aim is to find (an approximation of) the true image $\bar{x}$ by solving in some way the following linear system

$$Ax = b, \tag{6.1}$$

where $A$ is the blurring matrix and $b = A\bar{x} + \eta$ is the blurred and noisy data. There are several techniques in literature to do that, but generally in real applications, from which arise large-scale linear systems, the choice falls on iterative algorithms. Usually, instead of (6.1), the associated system of normal equations

$$A^H Ax = A^H b \tag{6.2}$$

is solved in order to find an approximated least squares solution. The main reason of this choice is that $A^H A$, the matrix coefficient of (6.2), has useful properties, first of all symmetry and (semi-)positive definiteness, that usually $A$ does not have. These properties allow to use specific powerful methods — such as Conjugate Gradient and its generalizations — and to improve the behaviour of all iterative methods, which generally become more stable when applied to (6.2) than when applied to (6.1). The least square formulation (6.2) is often implicitly assumed in several iterative methods, for instance Landweber method (5.22). On the other hand an iterative method for solving the equation (6.1) is Van Cittert method [120]

$$x_k = x_{k-1} + \tau(b - Ax_{k-1}). \tag{6.3}$$

In these two methods the step size $\tau > 0$ is fixed, while in Steepest descent method [13]

$$x_k = x_{k-1} + \tau_{k-1} A^H(b - Ax_{k-1}) \tag{6.4}$$

$\tau_{k-1}$ is computed at every step by $\tau_{k-1} = \|r_{k-1}\|_2^2 / \|Ar_{k-1}\|_2^2$ with $r_{k-1} = A^H(b - Ax_{k-1})$, where $\|\cdot\|_2$ is the Euclidean norm. Basically, Van Cittert method is fast, but often does not converge and in general gives rise to stability problems. On the other hand, Landweber and Steepest descent, which have good stability and convergence behaviours, are very slow. Other methods we consider are Lucy-Richardson method (LR) [101, 87, 109]

$$x_k = x_{k-1} \cdot A^H \left( \frac{b}{Ax_{k-1}} \right) \tag{6.5}$$

and Image Space Reconstruction Algorithm (ISRA) [36, 38]

$$x_k = x_{k-1} \cdot \left( \frac{A^H b}{A^H A x_{k-1}} \right), \tag{6.6}$$

that are well-known "statistical" iterative methods, where the multiplication operator $\cdot$ and the division operator $/$ have to be read componentwise. Also these methods have a low convergence rate. Usually to avoid that the denominator has some entries equal to zero, one applies these methods under the hypothesis that there is a small background signal $\beta$, which guarantees that $b_j > 0$, $\forall j$, and so the same for $Ax_{k-1}$ and $A^H A x_{k-1}$, since the initial guess $x_0$ is assumed to be the blurred and noisy data $b$. Really this assumption is not necessary, because the term that rules the zeros pattern is $x_{k-1}$. Hence, by imposing the denominator equal to one for the entries in which it is equal to zero, we can overcome this difficulty and remove the hypothesis related to $\beta$.

## 6.1 The idea

All the algorithms presented here base the update of the iteration on the "key" quantities

$$b - Ax_{k-1} \quad \text{or} \quad \frac{b}{Ax_{k-1}},$$

which both give information on the distance between the blurred data $b$ and the blurred iteration $Ax_{k-1}$. All the methods, except (6.3), require the application of the adjoint operator $A^H$. By an applicative point of view, the $n \times n$ matrix $A^H$ can be seen as a *reblurring* operator, whose role is basically to help the method to manage the noise.

Our idea is to pick a new $n \times n$ matrix $Z$, which will replace $A^H$. A forefather of this idea can be located in [47], where some branches, coming from the reblurring concept, are outlined. This way, (6.2) becomes

$$ZAx = Zb. \tag{6.7}$$

Pure formally, we can reformulate all the previous iterative methods in the new $Z$ context. We will place a $Z$ before the name of a method to distinguish it from the original one. Then, for complex step lengths $\tau, \tau_{k-1} \in \mathbb{C}$, we have the $Z$-Landweber method

$$x_k = x_{k-1} + \tau Z(b - Ax_{k-1}), \tag{6.8}$$

the $Z$-Steepest descent method

$$\begin{aligned} x_k &= x_{k-1} + \tau_{k-1} Z(b - Ax_{k-1}), \\ \tau_{k-1} &= \frac{r_{k-1}^H r_{k-1}}{r_{k-1}^H Z A r_{k-1}} \qquad \text{with} \quad r_{k-1} = Z(b - Ax_{k-1}), \end{aligned} \tag{6.9}$$

the $Z$-LR method

$$x_k = x_{k-1} \cdot Z\left(\frac{b}{Ax_{k-1}}\right), \tag{6.10}$$

and the $Z$-ISRA

$$x_k = x_{k-1} \cdot \left(\frac{Zb}{ZAx_{k-1}}\right). \tag{6.11}$$

Van Cittert algorithm (6.3) does not depend on $A^H$, so that it cannot be directly generalized to $Z$. However, the Van Cittert idea (that is, not to use $A^H$) can be applied to LR and to ISRA. This way, in both the LR and ISRA cases, Van Cittert gives rise to the same method

$$x_k = x_{k-1} \cdot \left(\frac{b}{Ax_{k-1}}\right). \tag{6.12}$$

It is interesting to note that $Z$-Landweber (6.8) can be seen as a generalization of both Landweber method (5.22), where $Z = A^H$, and Van Cittert method (6.3), where $Z = I$. The goal of any $Z$ variant is then to find a sort of "equilibrium" between high convergence speed (such as Van Cittert, i.e. $Z = I$, which is fast and unstable), and good quality of the reconstruction (such as Landweber, i.e. $Z = A^H$, which is slow and stable). We notice that in principle one can think to choose $Z$ as another operator, not necessarily related to a blurring process. This might drive to further developments of the idea proposed here.

### 6.1.1 $Z$-Landweber method

Let us first consider the $Z$-Landweber method, and let $x_0 = 0$ be the initial guess (this assumption is not mandatory, but it simplifies and improves the readability). For our analysis, we only require that the $n\times n$ matrix $M = ZA$ is diagonalizable as

$$M = T\Lambda_M T^{-1}\,,$$

where $\Lambda_M = \mathrm{diag}(m_1,\dots,m_n)$ is the diagonal matrix of the (complex) eigenvalues of $M$ and the columns of $T$ denote the corresponding eigenvectors. Following [25], by rewriting (6.8) as $x_{k+1} = \tau\sum_{i=0}^{k}(I-\tau ZA)^i Zb$, we have that the $n\times n$ iteration matrix $G_k = \tau\sum_{i=0}^{k}(I-\tau ZA)^i$ such that $x_{k+1} = G_k Zb$ can be diagonalized in the same way as follows

$$G_k = \tau P_{k-1}(\tau M) = T\Lambda_{G_k}T^{-1}, \qquad \Lambda_{G_k} = \tau P_{k-1}(\tau\Lambda_M) = \mathrm{diag}(g_1^{(k)},\dots,g_n^{(k)}),$$

for $k \geq 1$, where

$$P_k(\alpha) = \sum_{i=0}^{k}(1-\alpha)^i = \sum_{i=0}^{k}\begin{pmatrix} k+1 \\ i+1 \end{pmatrix}(-\alpha)^i = \frac{1-(1-\alpha)^{k+1}}{\alpha}$$

is a $k$-degree polynomial. Using the last formula, if $m_j \neq 0$ we have

$$g_j^{(k)} = \tau P_{k-1}(\tau m_j) = \frac{1-(1-\tau m_j)^k}{m_j}, \qquad j = 1\dots n\,. \tag{6.13}$$

Now we are ready to study the behaviour of the $k$-th Z-Landweber iteration $x_k$. Let $t_1,\dots,t_n$ be the column vectors of the matrix $T$ and $\hat{t}_1,\dots,\hat{t}_n$ the row vectors of the matrix $T^{-1}$. On one hand $T^{-1}x_k = \Lambda_{G_k}T^{-1}Zb$ and therefore, if we define the (complex) values $\beta_j = \hat{t}_j Zb$, we have $\hat{t}_j x_k = g_j^{(k)}\beta_j$, $j = 1\dots n$. On the other hand, we can define as $x^\dagger$ the vector

$$x^\dagger = (ZA)^\dagger Zb \tag{6.14}$$

where we define $(ZA)^\dagger = T\Lambda_M^\dagger T^{-1}$ with $\Lambda_M^\dagger = \mathrm{diag}(m_1^\dagger,\dots,m_n^\dagger)$, being $m_j^\dagger = m_j^{-1}$ if $m_j \neq 0$ and $m_j^\dagger = 0$ if $m_j = 0$. Note that $(ZA)^\dagger$ is just the Moore-Penrose pseudoinverse if $\Lambda_M$ is invertible or $T$ is unitary. Since $T^{-1}x^\dagger = \Lambda_M^\dagger T^{-1}Zb$, we have

$$\hat{t}_j x^\dagger = \begin{cases} \dfrac{\beta_j}{m_j} & \text{if } \ m_j \neq 0\ , \\ 0 & \text{if } \ m_j = 0\ , \end{cases}$$

for $j = 1,\dots,n$. Thanks to (6.13), we have

$$\frac{\left|\hat{t}_j x_k - \hat{t}_j x^\dagger\right|}{\left|\hat{t}_j x^\dagger\right|} = \frac{\left|g_j^{(k)}\beta_j - \dfrac{\beta_j}{m_j}\right|}{\left|\dfrac{\beta_j}{m_j}\right|} = |m_j|\left|g_j^{(k)} - \frac{1}{m_j}\right| = |1-\tau m_j|^k \quad \text{if } \ \hat{t}_j x^\dagger \neq 0. \tag{6.15}$$

The rate of decay of this error depends on the value of the parameters $\tau$ and $m_j$. For Landweber with normalized matrices such that $\|ZA\|_2 = 1$, we set $\tau = 1$, so that (6.15) only depends on the distance of $m_j$ from 1. Now we give some theoretical results about convergence of $Z$-Landweber method, based on the analysis of (6.15). We underline that (6.15) requires that $m_j \neq 0$ and so we will prove convergence under such an assumption.

**Theorem 6.1** *Let $m_j = \rho_j e^{\mathrm{i}\theta_j}$ , $j = 1, \ldots, n$ , be the non-zero complex eigenvalues of the $n \times n$ global matrix $M = ZA$. Then a necessary condition for the convergence of the $Z$-Landweber iteration (6.8) is that there exists an angle $\delta$ such that*

$$-\pi/2 < \delta + \theta_j < \pi/2\,, \quad j = 1, \ldots, n. \tag{6.16}$$

*Moreover, with complex iteration step length $\tau = \rho_\tau e^{\mathrm{i}\theta_\tau}$, then $Z$-Landweber method converges to $x^\dagger$ of (6.14) if and only if*

$$0 < \rho_\tau < \min_j \left[ \frac{2\cos(\theta_\tau + \theta_j)}{\rho_j} \right]. \tag{6.17}$$

**Proof.** By virtue of (6.13) and (6.15), we have that $\hat{t}_j x_k \to \hat{t}_j x^\dagger$, $\forall j$, as $k \to +\infty$, if and only if $|1 - \tau m_j| < 1$, $\forall j$ . Let $\tau m_j = \rho_\tau e^{\mathrm{i}\theta_\tau} \rho_j e^{\mathrm{i}\theta_j} = \sigma_j e^{\mathrm{i}\gamma_j}$, where $\sigma_j = \rho_\tau \rho_j$ and $\gamma_j = \theta_\tau + \theta_j$. The last condition $|1 - \tau m_j| < 1$, since

$$\left|1 - \sigma_j e^{\mathrm{i}\gamma_j}\right|^2 = (1 - \sigma_j \cos\gamma_j)^2 + (\sigma_j \sin\gamma_j)^2 = 1 + \sigma_j(\sigma_j - 2\cos\gamma_j),$$

is equivalent to $\sigma_j(\sigma_j - 2\cos\gamma_j) < 0$, $\forall j$. This yields $\sigma_j \neq 0$, $\forall j$, which implies $\rho_\tau > 0$, since by hypothesis $\rho_j \neq 0$, and the latter condition becomes

$$0 < \sigma_j < 2\cos\gamma_j, \forall j\,. \tag{6.18}$$

To satisfy (6.18), it is necessary that $\forall j$ , $\cos\gamma_j > 0$, that is $\gamma_j = \theta_\tau + \theta_j \in (-\pi/2, \pi/2)$, which can be satisfied only if (6.16) holds . Finally, (6.18) leads to (6.17). ■

We note that, when the necessary condition (6.16) holds, $\theta_\tau$ has to be chosen in such a way as to satisfy $\theta_\tau + \theta_j \in (-\pi/2, \pi/2)$ . If there exists an index $j$ such that $\theta_\tau \notin (-\pi/2 - \theta_j, \pi/2 - \theta_j)$ , then condition (6.17) on $\rho_\tau$ cannot be satisfied. In this case, $Z$-Landweber method will never converge, independently from the choice of $\tau \in \mathbb{C}$ (on the contrary, we recall that the classical Landweber method always converge for a right choice of the real step length). In addition, we consider some corollaries to this theorem.

**Corollary 6.2** *Let $Z = A^H$. Then*

$$-\pi/2 < \theta_\tau < \pi/2 \qquad \text{and} \qquad 0 < \rho_\tau < \frac{2}{\|A\|_2^2}$$

*are two necessary conditions for the convergence of the $Z$-Landweber method.*

**Proof.** Once noticed that $M = A^H A$ is a symmetric and positive definite matrix, so that $\theta_j = 0\ \forall j$, these results follow from (6.16) and (6.17), using the general inequality $2\cos\theta < 2$ and $\min 1/\rho_j = \max \rho_j = \|A\|_2^2$. ■

**Corollary 6.3** *Let $Z = A^H$. Then the $Z$-Landweber method converges to the solution $x^\dagger$ of (6.14) if and only if*

$$0 < \rho_\tau < \frac{2\cos\theta_\tau}{\|A\|_2^2}\,.$$

**Proof.** By considering the same argument of the proof of the previous corollary, the thesis follows straightforwardly from (6.17). ■

It is interesting to notice that, by Corollary 6.3, if $\tau \in \mathbb{R}$ we reobtain the well-known convergence interval of the classical Landweber method $0 < \tau < 2/\|A\|_2^2$. Anyway, when $\tau \in \mathbb{C}$, we highlight that the analogous condition related to the modulus $\rho_\tau$ is not sufficient, but only necessary.

In some cases (see §6.2), both $A$ and $Z$ have the same base of eigenvectors, that is

$$\begin{aligned} A &= T\Lambda_A T^{-1}, \quad \Lambda_A = \operatorname{diag}(\lambda_1, \dots, \lambda_n), \qquad \text{and} \\ Z &= T\Lambda_Z T^{-1}, \quad \Lambda_Z = \operatorname{diag}(z_1, \dots, z_n), \end{aligned}$$

where the values $\lambda_i$ and $z_i$, for $i = 1 \dots n$, denote the complex eigenvalues of $A$ and $Z$ respectively and the columns of $T$ denote the corresponding eigenvectors. This condition allows us to better discuss the basic properties of the matrix $Z$. For sake of simplicity, let us consider a global matrix $ZA$ such that $\tau = 1$ guarantees convergence according to Theorem 6.1 (a rescaling is often enough to this). Recalling that the classical Landweber method is an iterative regularization method, in order to have that the new $Z$-Landweber method is again a regularization method, $z_j$ has to behave so that $|1 - m_j| = |1 - z_j\lambda_j|$ is close to 1 when $\lambda_j$ is a small eigenvalue — so the reconstruction is slow in “high frequencies”, typically highly corrupted by noise — and close to zero when $\lambda_j$ is a large eigenvalue — so that the reconstruction is fast in “low frequencies”, typically slightly corrupted by noise. From this fact we can deduce that $\forall j$, $z_j$ can be chosen as $|z_j| \approx 1/|\lambda_j|$ when $\lambda_j$ is a large eigenvalue, while $|z_j| \approx 0$ when $\lambda_j$ is a small eigenvalue, with argument $\arg(z_j) = -\arg(\lambda_j)$ (or a suitable approximation). As we will see in §6.3, these constraints can be satisfied, for instance, by choosing $z_j$ by a means of filtering procedures on $\lambda_j$.

### 6.1.2 $Z$-LR and $Z$-ISRA

Now we study the $Z$ variant of the non-linear statistical methods (6.10) and (6.11). Differently from $Z$-Landweber, we now do not provide a rigorous convergence analysis, but we show some analogies with classical linear iterative methods, in order to understand how they work. Anyway, in the following sections we will see that numerical experiments with (6.10) and (6.11) show good performances.

$Z$-ISRA can be rewritten in this way

$$x_k = x_{k-1} + \left(\frac{x_{k-1}}{ZAx_{k-1}}\right) \cdot Z(b - Ax_{k-1}), \tag{6.19}$$

while $Z$-LR can be approximated by the following formula

$$x_k = x_{k-1} + x_{k-1} \cdot Z\left((b - Ax_{k-1}) \cdot \frac{1}{Ax_{k-1}}\right). \tag{6.20}$$

The approximation error for $Z$-LR, that is the difference between (6.20) and (6.10), is

$$x_{k-1} + x_{k-1} \cdot Z\left((b - Ax_{k-1}) \cdot \frac{1}{Ax_{k-1}}\right) - x_{k-1} \cdot Z\left(\frac{b}{Ax_{k-1}}\right) = \\ x_{k-1} - x_{k-1} \cdot Z\left((Ax_{k-1}) \cdot \frac{1}{Ax_{k-1}}\right).$$

If we define

$$[\mathcal{K}(y)]_j = \begin{cases} 1 & \text{if } y_j \neq 0, \\ 0 & \text{if } y_j = 0, \end{cases}$$

the error becomes $x_{k-1} - x_{k-1} \cdot Z\mathcal{K}(Ax_{k-1})$. This approximation error is zero when $[Ax_{k-1}]_j \neq 0$, $\forall j$. Indeed, under this hypothesis we have $x_{k-1} - x_{k-1} \cdot Z\mathcal{K}(Ax_{k-1}) = x_{k-1} - x_{k-1} \cdot Ze = x_{k-1} - x_{k-1} \cdot e = 0$ when $Ze = e$, being $e$ the all-ones vector (we remark that normalized blurring operators $Z$ with periodic, reflective and anti-reflective BCs always satisfy $Ze = e$). From (6.19) and (6.20), we can write this general form

$$x_k = x_{k-1} + S_{k-1}ZT_{k-1}(b - Ax_{k-1}), \tag{6.21}$$

where $S_{k-1}$ and $T_{k-1}$ are diagonal matrices. In particular, for (6.19) we have $S_{k-1} = \mathrm{diag}(\frac{x_{k-1}}{ZAx_{k-1}})$ and $T_{k-1} = I$, where $I$ is the identity matrix of order $n$, while for (6.20) we have $S_{k-1} = \mathrm{diag}\,(x_{k-1})$ and $T_{k-1} = \mathrm{diag}(\frac{1}{Ax_{k-1}})$. The expression (6.21) is interesting since it shows that both Z-ISRA and Z-LR have analogies with the iterative scheme of the Simultaneous Iterative

Reconstruction Techniques (SIRT), in which $S_{k-1} \equiv S$ and $T_{k-1} \equiv T$ are two fixed matrices (see [52] for details). $S_{k-1}$ and $T_{k-1}$ of (6.21) depend on $x_{k-1}$ and can be viewed as two componentwise adaptive descent parameters of the methods (they play a role similar to the step length $\tau$ of Landweber, which, on the contrary, is fixed and equal for all the components). The connection between Landweber method and statistical ones will be analysed in Chapter 7.

## 6.2 $Z$ built by a coarsening technique

Firstly we consider the case of periodic boundary conditions such that both $A$ and $Z$ are Block Circulant with Circulant Blocks (BCCB) matrices. Hence both $A$ and $Z$ have a common eigenvector basis, the 2D discrete Fourier matrix $F$ [97, 21, 27], and we can write

$$\begin{aligned} A &= F\Lambda_A F^H, \quad \Lambda_A = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), & (6.22)\\ Z &= F\Lambda_Z F^H, \quad \Lambda_Z = \mathrm{diag}(z_1, \ldots, z_n), & (6.23)\end{aligned}$$

where $\lambda_i$ and $z_i$ are the complex eigenvalues of $A$ and $Z$. Therefore, substituting $T$ with $F$, the analysis of §6.1.1 can be used in this context.

The issue concerning how to build, in some sense, the "better" $Z$ — that is, we recall, the better compromise between high convergence speed for $Z = I$ and good quality of the restored image for $Z = A^H$ — goes beyond the aim of this thesis and it might be the topic for a future research. Here we only provide some valid roads. The first simple procedure we use to construct $Z$ from $A$ is a coarsening strategy. We need two ingredients for the projection operator:

- Downsampling $\downarrow$ defined as $(\downarrow X)_{i,j} = X_{2i,2j}$ for an image $X$.
- A weighting operator $P$.

In the following $P$ will be usually chosen as the 2D Full Weighting Operator

$$\mathrm{FWO} = \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}. \tag{6.24}$$

On these grounds, by using the original PSF of the deblurring problem denoted by $h_{PSF}$ in (5.3), we generate the following new smaller PSF $w_{PSF}$

$$w_{PSF} := \nu \downarrow (P \circledast h_{PSF} \circledast P), \tag{6.25}$$

where $\circledast$ is the convolution operator and $\nu$ a constant such that $\sum_{j_1,j_2} w_{j_1,j_2} = \sum_{j_1,j_2} h_{j_1,j_2}$, which means that in physical terms the overall light (i.e. the energy) of the PSF is kept as constant. For instance, for $P$ = FWO we fix $\nu = 4$. This is the algebraic procedure for defining coarser matrices in multigrid methods by Galerkin conditions [48]. Thus, the $Z$-matrix is defined as $Z = V^H$, where $V$ is the matrix that results from using the $w_{PSF}$ with, in general, the same BCs of $A$.

According to the analysis in [48], the projection operator has to be a low-pass filter in order to obtain a coarser PSF that is again a blurring operator. Taking $P$ as a weighting operator, this ensures that $w_{PSF}$ has non-negative entries and is again a blurring operator. The size of $w_{PSF}$ is about one half the size of $h_{PSF}$ and repeating the procedure recursively we can get smaller and smaller PSFs. In the following, we will denote by PSF $1/2^j$ the PSF obtained applying $j$ times the coarsening strategy since its size in each direction is a factor $1/2^j$ of the size of $h_{PSF}$.

Unfortunately, it is not sufficient that $w_{PSF}$ is a blurring operator, but it has to share, and possibly enhance, some properties of $A$. A dipper spectral analysis can be performed in the case of periodic BCs and $A$ positive definite. The latter requires that the PSF is at least quadrantally symmetric (i.e. symmetric along the two axis). Under these assumptions, the matrix $A$ has the factorization (6.22) and is associated to a non-negative generating function $f : \mathbb{R}^2 \to \mathbb{R}$ defined by

$$f(x_1,\, x_2) = \sum_{j_1=-q_1}^{q_1} \sum_{j_2=-q_2}^{q_2} h_{j_1,j_2} e^{\mathrm{i}(j_1x_1+j_2x_2)}.$$

The eigenvalues of $A$ are $\lambda_j = f(\frac{2\pi s_1}{n_1}, \frac{2\pi s_2}{n_2})$, for $s_1 = 0, \dots, n_1 - 1$, $s_2 = 0, \dots, n_2-1$, $j = 1, \dots, n_1 n_2$. Moreover, according to the well-known spectral properties of blurring matrices [71, 48], $f$ has the following property.

**Property 6.4 ([48])** *Let $f$ be the generating function associated to a positive definite blurring matrix $A$. Then $f$ is even, non-negative, $2\pi$ periodic in each variable, and monotone non-increasing in $[0,\, \pi] \times [0,\, \pi]$ along each direction ($f$ has maximum at the origin and minimum at $[\pi, \pi]$).*

Therefore, $P$ should to be chosen such that the generating function obtained by $w_{PSF}$ has again the Property 6.4. The filter factor analysis in [49] shows that a multilevel regularization strategy requires that also the generating function of $P$ satisfies the Property 6.4. For such analysis is crucial the following result.

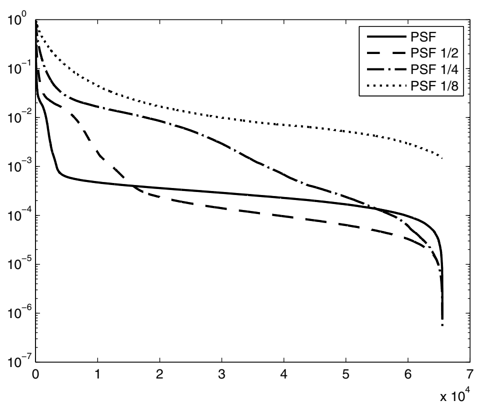


Figure 6.1: $|\lambda_j|$ of $A$ (solid line) for the PSF in Figure 6.4 and $|z_j|$ of different $Z$ variants of the coarsening strategy. The PSF $1/2^j$ are obtained applying the algorithm (6.25) recursively $j$ times, for $j = 1, 2, 3$.

**Proposition 6.5 ([3])** *Let $f$ and $p$ be real trigonometric polynomials associated to $h_{PSF}$ and $P$, respectively. Then, computing $w_{PSF}$ by (6.25), the generating function associated to $w_{PSF}$ is*

$$f_w(x) = \frac{\nu}{4} \sum_{y \in \Omega(x/2)} f(y)p(y)^2,$$

*where $\Omega(x) = \{(x_1, x_2), (x_1, x_2 + \pi), (x_1 + \pi, x_2), (x_1 + \pi, x_2 + \pi)\}$ for $x = (x_1, x_2)$.*

If $P$ is a weighting operator or the identity, then $p$ and hence also $f_w$ share the Property 6.4. For instance, when $P =$ FWO the associated generating function is $p(x) = (1 + \cos(x_1))(1 + \cos(x_2))/4$.

Here we simply consider only weighting operators that hence automatically generate a function $f_w$ which is closer than $f$ to 1 where $f$ is large (close to the origin) and closer than $f$ to zero where $f$ is small (close to $[\pi, \pi]$). However, the Proposition 6.5 could be investigated in the future for choosing $p$ depending on $f$ such that the properties of $f_w$ are further enhanced.

A classical PSF that is included in the previous analysis is the Gaussian blur. However, the following numerical results show that in practice this approach is robust to small perturbations only. Figure 6.1 shows the trend of the modulus of the eigenvalues of $Z$, $|z_j|$, in comparison with the modulus of the eigenvalues of $A$, $|\lambda_j|$, for the PSF in Figure 6.4 (i.e. the Satellite deblurring problem of the next §6.2.1). We note that $|z_j|$ is closer to 1 than $|\lambda_j|$ for large $|\lambda_j|$, i.e. in the signal space (low frequencies). This way, $z_j\lambda_j = m_j$ of the $Z$ variant is closer to 1 than $|\lambda_j|^2$ of the classical least square approach, so that $Z$ variant is able to speed up the iterative method in the signal space with respect to the least square approach.

From Figure 6.2, related to PSF 1/8, we can analyse the behaviour of $\lambda_j$, $z_j$, and $m_j$, in the complex Gauss plane. According to the previous analysis (Proposition 6.5), for the coarsening technique (6.25) the PSF must be quadrantally symmetric, and Theorem 6.1 tell us that it is necessary that all the eigenvalues $m_j$ lie in the same half-plane. This is *numerically* true for Satellite example, i.e. the PSF is near to be symmetric and the eigenvalues $m_j$ stay basically in a half-plane. Indeed, as we will see in §6.2.1, the use of $Z$ variant in numerical tests is successful. On the other hand, for a highly non-symmetric PSF like a motion blur, $z_j$ and $m_j$ have the "bizarre" behaviour reported in Figure 6.3. The necessary condition of Theorem 6.1 does not hold (since there is not any half plane containing all the eigenvalues) and in this case any $Z$ method proposed in this section fails to converge (these arguments will be numerically confirmed at the end of §6.4.2).

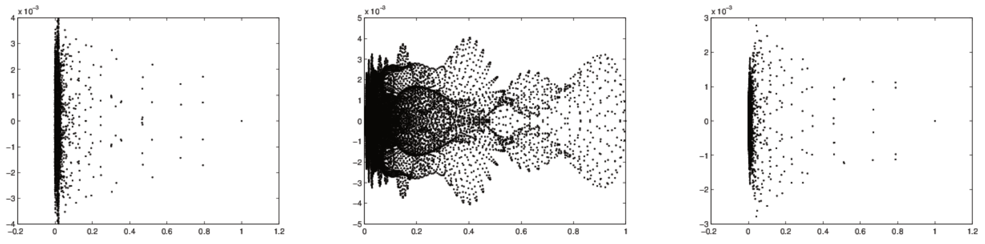


Figure 6.2: $\lambda_j$ (left), $z_j$ (center) and $m_j$ (right) values in the complex plane for the Satellite PSF in Figure 6.4.

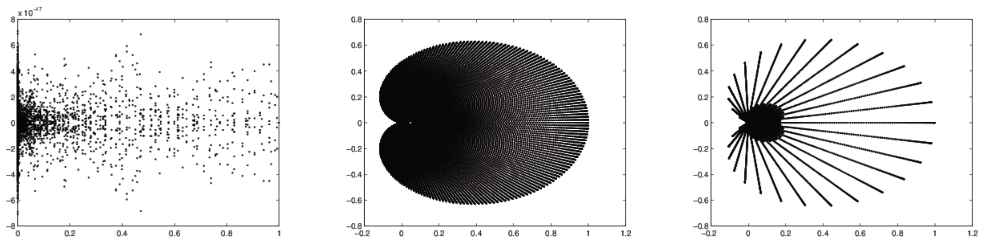


Figure 6.3: $\lambda_j$, $z_j$ and $m_j$ values in the complex plane for a motion blur.

Finally we observe that, although $Z$ variant with $Z$ defined by coarsening (6.25) can be fruitfully applied to very different PSFs, it is mainly conceived for PSFs which have wide distribution width. This is due to the fact that the downsampling technique reduces the "size" of the PSF, so that it is more effective if the original distribution is large. Moreover the PSF has to be (at least numerically) quadrantally symmetric.

### 6.2.1 Computational results relative to $Z$ built by the coarsening technique

We use images of $256 \times 256$ pixels to test the proposed techniques. The first set of data (Figure 6.4) we consider was developed at the US Air Force Phillips Laboratory, Laser and Imaging Directorate, Kirtland Air Force Base, New Mexico, that has been used by several authors with zero BCs [75, 67, 68, 25]. On the contrary we impose periodic BCs, which is the right choice. In fact, if we call $b_p = A\bar{x}$ where $A$ has periodic BCs and $b_z = A\bar{x}$ where $A$ has zero BCs, then we plot one of the first row of the image, we have Figure 6.5, whence it is clear that $b$ comes from $b_p$. The problem is little ill-conditioned ($\mathrm{cond}(A) \approx 1.3 \cdot 10^6$) but it has a considerable presence of noise $\|\eta\|_2 / \|A\bar{x}\|_2 \approx 4.5\%$. By summarizing, in the following tables of this subsection, we will list the minimum RRE (Relative Restoration Error, defined in (5.25)) and the corresponding number of iterations (IT) of different $Z$ methods of §6.2. Since the statistical methods LR and ISRA compute automatically non-negative approximations, in the numerical tests for Landweber method and Steepest descent method we employ the projection on the non-negative cone (see §6.4.3), which can be simply obtained by setting at zero all the negative values.

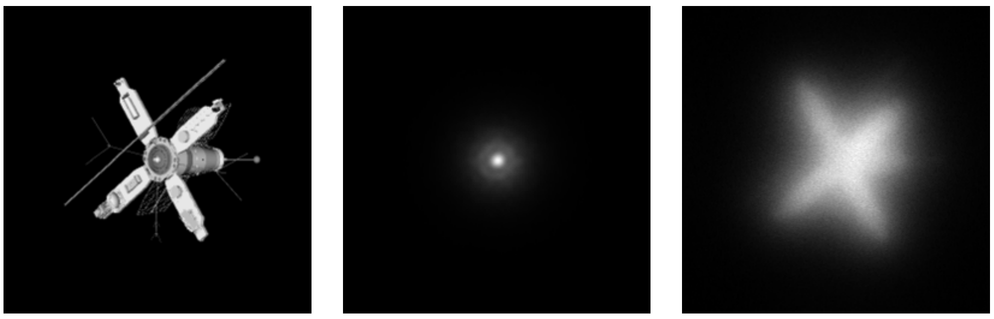


Figure 6.4: Satellite data set: true image, PSF, blurred and noisy image.

In Table 6.1, the numerical results for both Landweber (left) and Steepest descent (right) are reported in the first row, followed by their related $Z$ variants. Table 6.1 shows that $Z$ matrices of §6.2, although they are not able to overcome the best RRE of the classical methods in the first row of the tables, give rise to remarkable acceleration of the convergence speed. In particular the RRE values of $Z$-Landweber (left) are similar to the RRE of

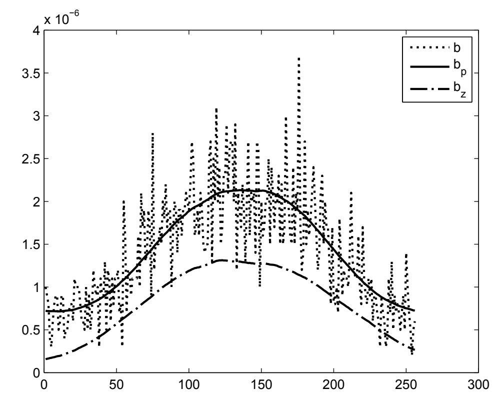


Figure 6.5: Experimental $b$ (blurred and noisy image) compared with $b_p$ (blurred image related to periodic BCs) and $b_z$ (blurred image related to zero BCs).

the corresponding $Z$-Steepest descent ones (right), but the iteration numbers (IT column) show that the $Z$-Steepest descent methods are better. About the Van Cittert method, the numerical results of the last row confirm that, as already remarked, Van Cittert method is unstable, since it starts to diverge very quickly, so that the quality of its reconstruction is really poor. As expected, the more the PSF of $Z$ is small (i.e. technique (6.25) is applied more times), the more the $Z$-method is fast to obtain its best restoration. From end to end, we have the Landweber method ($Z = A^H$) on one side, which is very stable but very low, and Van Cittert method ($Z = I$) on the other side, which is very fast but very unstable; as already sketched $Z$ can be properly chosen in an intermediate way between these two extremes to inherits the good properties of both (see Figure 6.6 where three corresponding restorations are shown).

| | RRE | IT |
|---|---|---|
| Landweber | 0.3278 | 5358 |
| $Z$-Landweber 1/2 | 0.3440 | 1052 |
| $Z$-Landweber 1/4 | 0.3384 | 357 |
| $Z$-Landweber 1/8 | 0.3458 | 111 |
| Van Cittert | 0.5062 | 12 |

| | RRE | IT |
|---|---|---|
| Steepest descent | 0.3278 | 3009 |
| $Z$-Steepest descent 1/2 | 0.3440 | 520 |
| $Z$-Steepest descent 1/4 | 0.3383 | 153 |
| $Z$-Steepest descent 1/8 | 0.3453 | 36 |
| Van.C.-Steepest descent | 0.4985 | 5 |

Table 6.1: $Z$ variants of §6.2 applied to Landweber method (left) and Steepest descent method (right).

Table 6.2 is related to the statistical methods $Z$-LR (left) and $Z$-ISRA (right), with the same organization of Table 6.1. A comparison of Table 6.2 and Table 6.1 shows that in this test statistical methods give restorations

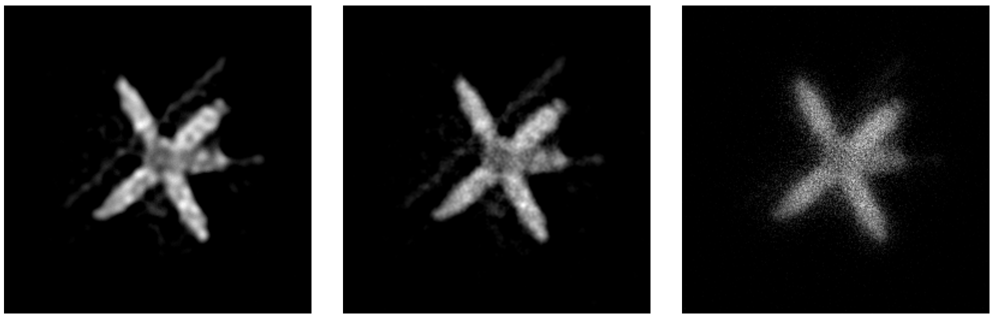


Figure 6.6: Restorations made by Landweber (RRE 0.3278, IT 5358), $Z$-Landweber 1/4 (RRE 0.3384, IT 357) and Van Cittert (RRE 0.5062, IT 12).

that are worse than Landweber and Steepest descent ones. Specifically, RRE of $Z$-LR and RRE of the corresponding $Z$-ISRA are very close each other. It is interesting to notice that as the PSF of $Z$ becomes smaller and smaller, the results of $Z$-ISRA and $Z$-LR tend to become more and more similar: this scenario is expected, since at the end $Z = I$, so that Van Cittert idea applied to LR (6.10) and to ISRA (6.11) yields to the exactly the same method (6.12). Figure 6.7 shows some restorations with $Z$-ISRA for some choices of $Z$.

| | RRE | IT |
|---|---|---|
| LR | 0.3484 | 2128 |
| $Z$-LR 1/2 | 0.3693 | 591 |
| $Z$-LR 1/4 | 0.3588 | 183 |
| $Z$-LR 1/8 | 0.3643 | 55 |
| Van Cittert-LR | 0.5060 | 6 |

| | RRE | IT |
|---|---|---|
| ISRA | 0.3451 | 1926 |
| $Z$-ISRA 1/2 | 0.3672 | 497 |
| $Z$-ISRA 1/4 | 0.3575 | 172 |
| $Z$-ISRA 1/8 | 0.3640 | 55 |
| Van Cittert-ISRA | 0.5060 | 6 |

Table 6.2: $Z$ variants of §6.2 applied to LR method (left) and ISRA method (right).

We have just said that Van Cittert method gives rise to instability and it is not able to give a good reconstruction. This is true for the blurred and (highly) noisy data $b$ of Figure 6.4 (right). However the situation deeply changes if we apply a denoising procedure to $b$, obtaining first a blurred data $b'$ with lower noise. As we can see from Table 6.3, where we consider $b'$ instead of $b$ in the iterative method, Van Cittert has a huge improvement, since it gives good restorations in small number of steps. This shows that Van Cittert method can be used provided that the noise on the blurred data (and the ill-conditioning on the matrix $A$) is low, so that its usual instability does not produce dreadful effects.

As final comment, we now compare different choices of the weighting operator $P$ in the projector. We can chose $P$ as a small Gaussian blur or

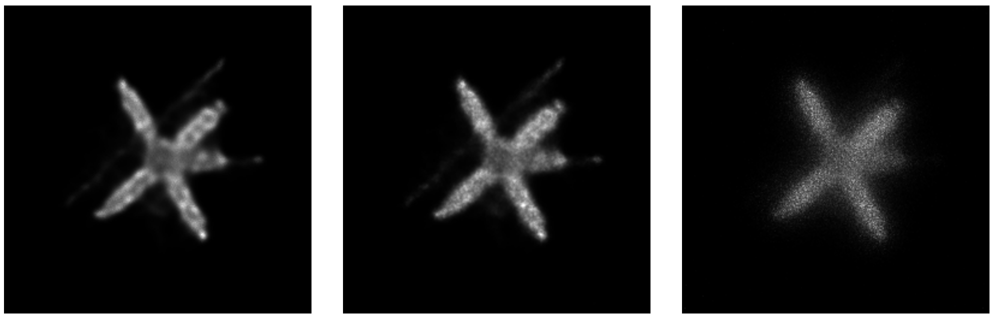


Figure 6.7: Restorations made by ISRA (RRE 0.3451, IT 1926), $Z$-ISRA 1/4 (RRE 0.3575, IT 172) and Van Cittert-ISRA (RRE 0.5060, IT 6).

| | RRE | IT |
|---|---|---|
| Van Cittert | 0.3532 | 75 |
| Van Cittert-Steepest descent | 0.3521 | 23 |
| Van Cittert-LR | 0.3576 | 46 |
| Van Cittert-ISRA | 0.3576 | 46 |

Table 6.3: Van Cittert method with denoised blurred data.

$P = I$ (the identity matrix) instead of the full weighting operator FWO. In these cases, we have that $Z$-Landweber with PSF 1/8 gives the following results: for $P = I$ the RRE is equal to 0.3540 in 88 steps, for $P =$ FWO the RRE is equal to 0.3458 in 111 steps and, for $P$ equal to a Gaussian the RRE is equal to 0.3384 in 168 steps. From these results we can appreciate the higher or lower convergence speed and restoration accuracy of $Z$-Landweber method according to the choice of the weighting operator $P$. In other words, picking $P$ related to PSFs which have increasing wide distribution width ($I <$ FWO $<$ Gaussian), we obtain better restorations, but in an higher number of steps.

## 6.3 $Z$ built by filtering techniques

On one hand the way to build $Z$ of §6.2, which acts directly on the PSF of $A$, has the advantage of being simple and reliable, since it does not require to set any parameter, and the advantage of preserving the non-negativity of the elements of the PSF. On the other hand $Z$ of §6.2 has the drawback that $ZA$ of (6.7) looses the good property of the classical least squares matrix $A^H A$ of having real and non-negative eigenvalues — although the complex eigenvalues of $ZA$ of §6.2 have usually real part which is non-negative and imaginary part which is very small — and usually it gives rise to poor restorations in the non-symmetric cases. In short, obviously $Z$ of §6.2 is not always the "best" choice.

More sophisticated techniques to build $Z$ require to work with the eigenvalues $\lambda_j$ of $A$ — available, for instance, by means of FFT of the PSF in the case of periodic BCs — and to *filter* them in a proper way. These techniques based on filtering, as introduced in the following of this section, lead to $Z$ matrices that have, in some sense, opposite properties of the $Z$ matrices of §6.2 based on downsampling. Indeed, as we will see, the new $Z$ matrices require the choice of a proper threshold parameter $\zeta$ and unfortunately the non-negativity of the elements of the PSF is not preserved, Despite that, the technique of this section is successfully applicable to any PSF and the imaginary part of any eigenvalue of $ZA$ vanishes.

Basically the simplest procedure to compute $Z$ by filtering is the following. By considering the decomposition (6.22), chosen a threshold parameter $\zeta$ in a suitable way, we take $Z = F\text{diag}(z_1, \ldots, z_n)F^H$ with any $z_j$ defined as

$$z_j = \begin{cases} \bar{\lambda}_j = \rho_j e^{-\mathrm{i}\theta_j} & \text{if } |\lambda_j| < \zeta, \\ \dfrac{\bar{\lambda}_j}{|\lambda_j|} = e^{-\mathrm{i}\theta_j} & \text{if } |\lambda_j| \geq \zeta, \end{cases} \tag{6.26}$$

where $\lambda_j = \rho_j e^{\mathrm{i}\theta_j}$ denotes any eigenvalue of $A$. This simple procedure, which we will refer as *VC filtering*, for the largest eigenvalues of $A$ employs (a symmetric generalization of) the Van Cittert approach of using the identity matrix instead of $A^H$ in the normal equations (indeed the modulus of $\bar{\lambda}_j/|\lambda_j|$ is one if $|\lambda_j| \geq \zeta$ , so that $ZA$ can be though as a symmetric approximation of $A$ for those eigencomponents). On the contrary, for the small eigenvalues of $A$, (6.26) gives rise to the classical normal equations approach, since for those eigencomponents (6.7) is equal to (6.2) (indeed therein the eigenvalues $\bar{\lambda}_j$ of $Z$ are exactly the eigenvalues of $A^H$).

In addition to the basic technique (6.26), in the following we consider this filtering approach. Given a real function $f : [0, +\infty) \to [0, +\infty)$, we define an *f filter* for $Z$ such that

$$z_j = \begin{cases} \bar{\lambda}_j f(|\lambda_j|) = \rho_j f(\rho_j) e^{-\mathrm{i}\theta_j} & \text{if } |\lambda_j| f(|\lambda_j|) < 1, \\ \dfrac{\bar{\lambda}_j}{|\lambda_j|} = e^{-\mathrm{i}\theta_j} & \text{if } |\lambda_j| f(|\lambda_j|) \geq 1. \end{cases} \tag{6.27}$$

Heuristically, comparing again $Z$ and $A^H$, the function $f$ is conceived with the aim of modifying the modulus of the eigenvalues of $A^H$ with continuity, as will better explained in the comprehensive §6.3.1. A simple but effective choice might be $f(x) = \gamma x + 1$, with $\gamma \gg 1$. This gives rise to a matrix $Z$ with eigenvalues $z_j$ such that $|z_j| = 1$ if the corresponding eigenvalue $\lambda_j$

of $A$ is large and such that $|z_j|$ is contained in $(|\lambda_j|, 1)$ if the corresponding eigenvalue $\lambda_j$ of $A$ is small. To allow us a comparison between the (small) parameter $\zeta$ of (6.26) and the (large) parameter $\gamma$ for this particular filter $f(x) = \gamma x + 1$, we calculate the intersection between the line $y = 1$, related to the modulus of the largest eigenvalues $z_j$ of (6.26), and $y = \gamma x^2 + x$, related to the largest eigenvalues $z_j$ of (6.27) for $f(x) = \gamma x + 1$, and we impose, on the ground of heuristic considerations, that this point is $(\sqrt{2}\zeta, 1)$. So we have

$$x = \frac{-1 + \sqrt{1 + 4\gamma}}{2\gamma} \approx \frac{1}{\sqrt{\gamma}}, \text{ which gives } \gamma = \frac{1}{2\zeta^2}.$$

Before going on, in Table 6.4 we report some first computational results of $Z$ variant with VC filter (with $\zeta = 0.01$, left side) and $f$ filter (with $\gamma = (2\zeta^2)^{-1} = 5000$, right side) for the same Satellite data set used in §6.2.1 (more meaningful results will be given in §6.3.2). The Table 6.4 shows that performances of $Z$-Landweber with VC filter (left side) are superior to those of $Z$-Landweber $1/2^k$ of Table 6.1, in terms of both quickness and quality of reconstructions. Furthermore results of $Z$-Landweber with filter $f$ are slightly better than those obtained by VC filter.

| **VC filter** ($\zeta = 0.01$) | RRE | IT |
|---|---|---|
| $Z$-Landweber $\zeta$ | 0.3313 | 130 |
| $Z$-Steepest descent $\zeta$ | 0.3311 | 52 |
| $Z$-LR $\zeta$ | 0.3580 | 32 |
| $Z$-ISRA $\zeta$ | 0.3416 | 62 |

| $f$ **filter** ($\gamma = 5000$) | RRE | IT |
|---|---|---|
| $Z$-Landweber $f$ | 0.3298 | 128 |
| $Z$-Steepest descent $f$ | 0.3297 | 49 |
| $Z$-LR $f$ | 0.3439 | 53 |
| $Z$-ISRA $f$ | 0.3407 | 62 |

Table 6.4: $Z$ variants with VC filter (6.26) and with $f$ filter (6.27) for the example in Figure 6.4.

### 6.3.1 $Z$ variant meets regularizing preconditioning

In this section we will illustrate the link between $Z$ variant and classical preconditioning described in literature. By considering again the decomposition (6.22) and taking into account (6.15), if we attempt to reduce the distance of $m_j$ from 1 in the most intuitive way, i.e. by trying to do a kind of *inversion*, just considering $z_j \approx \lambda_j^{-1}$, we meet the classical preconditioning framework developed for Landweber. Basically, the least squares problem (6.2) is replaced by the following linear system

$$DA^H Ax = DA^H b. \tag{6.28}$$

Without deep details, the preconditioner $D$ is basically a regularized approximation of $(A^H A)^{-1}$ whose aim is to speed up the convergence of the components with low noise (i.e. the so called *signal space*). We can notice that the adjoint operator $A^H$ is first applied to (6.1), which stabilizes but also slows down the restoration of all the components related to the small singular values of $A$, and then the circulant preconditioner $D$ is applied to (6.2) to speed up the convergence in the signal space. In other words, in the classical regularizing preconditioning, the preconditioner $D$ has to speed up the slowing down produced by $A^H$. On these grounds, $Z$ can be seen as a single preconditioning operator with the aim of obtaining a preconditioned system $ZAx = Zb$ such that iterative methods become stable (as well as usually obtained through the normal equations involving $A^H$) without slowing the convergence in the signal space (so that no subsequent accelerating operator $D$ is needed). Therefore a natural question arises: which are the differences and the connections between the preconditioning (6.28) based on $D$ and the proposed preconditioning of §6.3 based on $Z$?

To give an answer, in the context of $D = F\text{diag}(d_1, \dots, d_n)F^H$ preconditioning, called $\lambda_j$ the eigenvalues of $A$, $d_j$ the eigenvalues of $D$, we first can itemize the following filtering procedures:

- $f$ Filter
$$d_j = \begin{cases} f(|\lambda_j|) & \text{if } |\lambda_j|\, f(|\lambda_j|) < 1 \\ 1/|\lambda_j| & \text{if } |\lambda_j|\, f(|\lambda_j|) \geq 1 \end{cases} \tag{6.29}$$
- $p$ Low Pass Filter
$$d_j = \begin{cases} 0 & \text{if } |\lambda_j| < \zeta \\ 1/\,|\lambda_j|^p & \text{if } |\lambda_j| \geq \zeta \end{cases} \tag{6.30}$$
- $p$ Hanke Nagy Plemmons Filter [69]
$$d_j = \begin{cases} 1 & \text{if } |\lambda_j| < \zeta \\ 1/\,|\lambda_j|^p & \text{if } |\lambda_j| \geq \zeta \end{cases} \tag{6.31}$$
- $p$ Tyrtyshnikov Yeremin Zamarashkin Filter [118]
$$d_j = \begin{cases} 1/\zeta & \text{if } |\lambda_j| < \zeta \\ 1/|\lambda_j|^p & \text{if } |\lambda_j| \geq \zeta \end{cases} \tag{6.32}$$
- Tikhonov Filter
$$d_j = \frac{1}{|\lambda_j|^2 + \alpha} \tag{6.33}$$
- $q$ Tikhonov Filter
$$d_j = \left( \frac{1}{|\lambda_j|^{2q} + m_q} \right)^{1/q} \tag{6.34}$$

where $m_q = \operatorname*{mean}_j(|\lambda_j|^q)$ is the arithmetic mean.

On the other hand, in the context of $Z = F\text{diag}(z_1, \ldots, z_n)F^H$ variant, recalling the basic example (6.26), by using each filter we can define the eigenvalues of $Z$ as

$$z_j = \bar{\lambda}_j d_j$$

(indeed, (6.26) corresponds to the HNP filter (6.31) with $p = 1$).

This way, the answer of our question about differences and connections between $D$ of (6.28) and $Z$ of (6.7) is now given: in the case of periodic BCs, in which $A$ is BCCB, using filters with $Z$ or $D$, both BCCB, is completely equivalent, since $Z = F\Lambda_Z F^H$, where the diagonal matrix $\Lambda_Z$ is formed by the eigenvalues $z_j = \bar{\lambda}_j \mathcal{F}(\lambda_j)$, $\mathcal{F}$ denoting the filter, and $DA^H = F\Lambda_D\Lambda_{A^H}F^H$, where the diagonal matrix $\Lambda_D$ contains the eigenvalues $d_j = \mathcal{F}(\lambda_j)$. So in both the two cases (6.8) and (5.23), the methods become

$$x_k = x_{k-1} + \tau F\Lambda F^H(b - Ax_{k-1}),$$

where $\Lambda = \Lambda_Z = \Lambda_D\Lambda_{A^H}$. This is no more true if we have zero BCs and the reason is that Toeplitz matrices are no longer an algebra. We will analyse this matter in §6.4, devoted to the zero BCs case, in which we will show how $Z$ variant can improve the traditional circulant preconditioning.

We note that the first filtering technique (6.29) has been defined here in (6.27), while the others, for $p = 2$, come from preconditioning literature; $p$ variants and $q$ variants are our proposals, based on heuristic considerations which are in some sense linked to the recent regularization theory on $L^p$ Banach spaces. According to the classical regularization theory, the key parameter of most of these filters is the *threshold parameter* $\zeta > 0$: in the *noise space* (i.e. eigencomponents with $|\lambda_j| < \zeta$) high filtering effects arise, while in the *signal space* (i.e. $|\lambda_j| \geq \zeta$) there is low or null filtering. Analogously, the key parameter of the Tikhonov filter is the *regularization parameter* $\alpha$. Regarding our proposal of $q$ Tikhonov filter (6.34), with $q > 0$ (usually $1 \leq q \leq 2$), the regularization parameter becomes $q$, since the classical Tikhonov parameter $\alpha$ is now replaced by $m_q$, which depends on $q$. In this case, the convergence speed of iterative method increases as $q$ approaches 0, and in general the quality of the restoration does the opposite. The introduction of the $p$-filters (6.31) and (6.32) (with $1 \leq p \leq 2$) is an attempt to create a link, for example, between VC filter (6.26) (for $p = 1$) and the classical Hanke Nagy Plemmons filter (6.31) (for $p = 2$, see [69]) and, at the same time, to generalize them. The speed of iterative method increases as $p$ approaches 2, and in general the quality of the restoration do the opposite.

On these grounds, we can say that the action of these different filters can be unified in the framework of the two leading ideas of *reblurring* and *inver-*

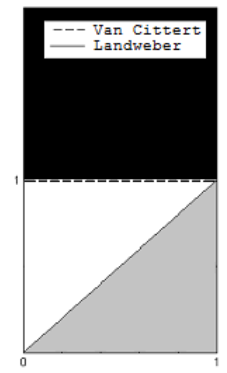


Figure 6.8: Values of $|\lambda_j|$ on the $x$-axis vs $|z_j|$ on the $y$-axis (black: regularizing inversion preconditioner; white: regularizing reblurring preconditioner; gray: slow down the convergence of Landweber).

*sion*. More specifically, looking at Figure 6.8, where we have the modulus of the eigenvalues $\lambda_j$ of a blurring matrix $A$ (normalized as $\|A\|_2 = 1$) on the abscissa and the corresponding modulus of the eigenvalues of the matrix $Z$ on the ordinate, we have drawn three areas:

a) the black one, over the horizontal line of ordinate equal to 1 (this line corresponds to the Van Cittert method, where $Z = I$, so that all the eigenvalues are equal to 1);

b) the white one, between the horizontal line and the bisecting line (this line corresponds to the Landweber method, where $Z = A^H$, so that $|\lambda_j| = |z_j|$);

c) the grey one, under the Landweber method line.

We will talk of *regularizing inversion preconditioner* if $|z_j|$ lie in the black area, while we will talk of *regularizing reblurring preconditioner* if $|z_j|$ lie in the white area. Taking into account that any $|\lambda_j| \leq 1$, in the first case the role of $Z$ is close to an inversion, since $|z_j| \geq 1$, while in the second case, the role of $Z$ is close to a blurring, since $|z_j| \leq 1$ as well as $|\lambda_j|$. The first case should give a strong, and often unstable, acceleration of the convergence; the second case — which we are interested in — should give a convergence speed greater than the slow classical Landweber method, without loosing its stability properties. In the grey area, any iterative methods based on the $Z$ variant equation (6.7) is even slower than the corresponding method based on the normal equation (6.2), since $|z_j| < |\lambda_j|$, as used in the theory of regularizing preconditioning for strongly ill-posed problems [54].

More generally, the relationships between (6.7) and (6.28) can be easily understood by considering the case which $A$ is symmetric and positive definite. Taking into account the basic convergence behaviour of the simplest stationary method (5.22), since $\lambda_j(A^H A) = [\lambda_j(A)]^2 \ll \lambda_j(A)$, if $\lambda_j(A)$ is small, we have that the iterative method based on the normal equation $A^H Ax = A^H b$ are much slower than the same iterative method based on the original equation $Ax = b$ for all the components related to small eigenvalues of $A$. Thus, if this improves the regularization capabilities (the components related to small eigenvalues usually lie in the noise space), often the convergence speed is too much low, so that the inversion preconditioners $D$ (which approximates the inverse of $A^H A$) is applied to quicken. As already briefly sketched, our proposal can be summarized as follows: instead of slowing down with $A^H$ (to obtain stability) and then speeding up via the preconditioner $D$ (to obtain again quickness), we adopt a single preconditioner operator $Z$, which try to simultaneously give rise to stability and quickness.

Finally, before analysing the zero BCs case, we have to complete the arguments for the periodic BCs case, by giving computational results in the next subsection. We also discuss if and how it is possible to use the previous filters for the statistical iterative methods (6.10) and (6.11).

### 6.3.2 Computational results relative to $Z$ built by filtering techniques

As shown in §6.3.1, for periodic deblurring problems the preconditioners $D$ and $Z$ lead exactly to the same analytical method, so that both have to give the same results, net of rounding errors. The aim of this section is to provide a comparison of performances of the filters (6.29)–(6.34). In Table 6.5 we list the results of $Z$-Landweber for the Satellite deblurring problem ($\zeta = 0.013$, $\alpha = 0.005$, $q = 1$). We can argue that the proposed $q$ Tikhonov filter (6.34) gives a better compromise between speed and quality, as shown in the last row of left side where RRE = 0.3369 within only 7 iterations and in Figure 6.9, where three different restorations can be evaluated. Moreover the results of the right side of Table 6.5, all related to different choices of the value of $p$ of the filters (6.30), (6.31) and (6.32), show that, in general, the $p$ strategy can bring a meaningful enhancement of the restored image.

If we use the previous filters for the statistical methods LR and ISRA, we obtain the results reported in Table 6.6 and Table 6.7. We remark that, while filters coming from reblurring idea (i.e. $f$ filter and the other three in the column of $p = 1$) still work well, the others need a quite difficult setting of parameters and sometimes give rise to convergence issues (see Table 6.6).

| | RRE | IT |
|---|---|---|
| $f$ filter | 0.3298 | 128 |
| Low pass | 0.3504 | 5 |
| H.N.P. | 0.3504 | 5 |
| T.Y.Z. | 0.3496 | 5 |
| Tikhonov | 0.3320 | 27 |
| $q$ Tikhonov | 0.3369 | 7 |

| | $p=2$ | | $p=1.5$ | | $p=1$ | |
|---|---|---|---|---|---|---|
| | RRE | IT | RRE | IT | RRE | IT |
| $p$ Low pass | 0.3504 | 5 | 0.3367 | 35 | 0.3311 | 250 |
| $p$ H.N.P. | 0.3504 | 5 | 0.3367 | 36 | 0.3307 | 251 |
| $p$ T.Y.Z. | 0.3496 | 5 | 0.3346 | 30 | 0.3360 | 85 |

Table 6.5: Results of the $Z$-Landweber method with $Z$ built by filtering techniques of §6.3.1 ($\zeta = 0.013$, $\alpha = 0.005$, $q = 1$) for the example in Figure 6.4.

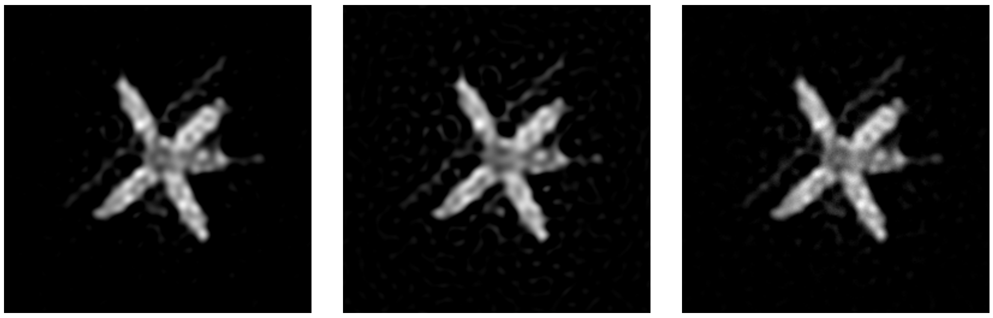


Figure 6.9: Restorations made by $Z$-Landweber with $f$ filter (RRE 0.3298, IT 128), with Hanke, Nagy, Plemmons filter (RRE 0.3504, IT 5) and with $q$ Tikhonov filter (RRE 0.3369, IT 7).

| | RRE | IT |
|---|---|---|
| $f$ filter | 0.3439 | 53 |
| Low pass | 0.4655 | 1 |
| H.N.P. | 0.4655 | 1 |
| T.Y.Z. | 0.4654 | 1 |
| Tikhonov | 0.3474 | 12 |
| $q$ Tikhonov | 0.3473 | 12 |

| | $p=2$ | | $p=1.5$ | | $p=1$ | |
|---|---|---|---|---|---|---|
| | RRE | IT | RRE | IT | RRE | IT |
| $p$ Low pass | 0.4655 | 1 | 0.3664 | 5 | 0.3582 | 32 |
| $p$ H.N.P. | 0.4655 | 1 | 0.3664 | 5 | 0.3580 | 32 |
| $p$ T.Y.Z. | 0.4654 | 1 | 0.3645 | 5 | 0.3614 | 32 |

Table 6.6: $Z$-LR with $Z$ built by different filters ($\zeta = 0.01$, $\alpha = 0.005$, $q = 1.5$) for the example in Figure 6.4.

| | RRE | IT |
|---|---|---|
| $f$ filter | 0.3407 | 62 |
| Low pass | 0.3603 | 1 |
| H.N.P. | 0.3603 | 1 |
| T.Y.Z. | 0.3602 | 1 |
| Tikhonov | 0.3505 | 1 |
| $q$ Tikhonov | 0.3588 | 3 |

| | $p=2$ | | $p=1.5$ | | $p=1$ | |
|---|---|---|---|---|---|---|
| | RRE | IT | RRE | IT | RRE | IT |
| $p$ Low pass | 0.3603 | 1 | 0.3441 | 11 | 0.3416 | 62 |
| $p$ H.N.P. | 0.3603 | 1 | 0.3441 | 11 | 0.3416 | 62 |
| $p$ T.Y.Z. | 0.3602 | 1 | 0.3442 | 10 | 0.3534 | 40 |

Table 6.7: $Z$-ISRA with $Z$ built by different filters ($\zeta = 0.01$, $\alpha = 0.0002$, $q = 0.52$) for the example in Figure 6.4.

Choosing $\zeta = 0.05$, we can obtain a quite good restoration for T.Y.Z. filter (RRE 0.3568, IT 105), while for Low pass filter and H.N.P. filter we are able to get only poor reconstructions also for higher values of $\zeta$.

## 6.4 Deblurring problems with zero boundary conditions

So far we have studied a first applications of the $Z$ variant in the simple context of deblurring problems with periodic boundary conditions, i.e. when both the matrices $A$ and $Z$ are BCCB. This BCCB assumption is useful to simplify the theoretical analysis, since both $A$ and $Z$ belong to the same algebra, so that the link between the matrix $Z$ and the classical regularizing preconditioner $D$ can be easily derived, as described in § 6.3.1.

From now on, we will consider deblurring problems with zero boundary conditions, so that $A$ is a Block Toeplitz with Toeplitz Blocks (BTTB) matrix, which is the classical assumption for isolate objects on a black background. Our aim is to improve classical BCCB preconditioning techniques for BTTB matrices by using $Z$ variants defined again by filtering procedures in the BCCB algebra like those in §6.3. Differing from the case of periodic BCs for $A$ discussed in the previous section, now the two matrices $Z$ of (6.7) and $DA^H$ of (6.28) are really different because $DA^H$ is not longer BCCB. Before going on, we can just observe that any iterative method based on the inversion preconditioned system $DA^HAx = DA^Hb$ requires the BTTB matrix-vector product with $A^H$ plus the BCCB matrix-vector product with $D$ in the same place where the corresponding method based on the reblurring preconditioned system $ZAx = Zb$ requires only the BCCB matrix-vector product with $Z$. Recalling that the BTTB matrix-vector product is more involving than the BCCB one, with $Z$ variant we have a significant reduction of computational costs. These comments show that $Z$ is different from $DA^H$ not only in the form, but also in the substance.

### 6.4.1 Form and substance

Since $A$ is a BTTB matrix (and so its eigenvalues are not directly available by the FFT of the PSF), instead of the eigenvalues of $A^HA$, we now consider the eigenvalues of $\tilde{A}^H\tilde{A}$, where $\tilde{A}$ is a BCCB approximation of $A$ (for example the T. Chan optimal approximation [33], see [54] for references and details). In other words, if we want to obtain a BCCB preconditioner for a BTTB problem, we have only to replace the eigenvalues of $A$, $\lambda_j$, with the eigenvalues of $\tilde{A}$, say $\kappa_j$, in all the filters and the techniques used to create $Z$ and

$D$. The idea behind the classical inversion filters, such as (6.30)-(6.34) with $p = 2$, consists in choosing eigenvalues of $D$ (resp. $Z$) close to eigenvalues of $\left(A^H A\right)^{-1}$ (resp. $A^{-1}$), so that $DA^H A$ (resp. $ZA$) is "near" to the identity matrix. The reason of inverted commas is that, as we described in §6.3.1, this is done only in the *signal space*, i.e. where $|\lambda_j| \geq \zeta$, being $\zeta$ the threshold parameter, or with the introduction of the regularization parameter $\alpha$, associated with Tikhonov filter. In addition, differently from the periodic BCs case, $D$ and $A^H A$ (resp. $Z$ and $A$) are no longer in the same algebra. Thus, using *eig* to indicate the eigenvalues of a matrix, once we have arranged the eigenvectors of $A^H A$ (resp. $A$) according to the Fourier basis, whose vectors are the eigenvectors of $D$ (resp. $Z$), we have (see [123] for the approximation of the eigenvectors of Toeplitz matrices by means of the Fourier vectors)

$$\mathrm{eig}_j(DA^H A) \approx \mathrm{eig}_j(D)\mathrm{eig}_j(A^H A) = |\kappa_j|^{-2} |\lambda_j|^2 , \tag{6.35}$$

$$\mathrm{eig}_j(ZA) \approx \mathrm{eig}_j(Z)\mathrm{eig}_j(A) = |\kappa_j|^{-2} \bar{\kappa}_j \lambda_j . \tag{6.36}$$

Therefore, with regard to the comparison between $Z$ variant and inversion preconditioning, the point is to evaluate if (6.36), which comes from $Z$ approach, is closer to one than (6.35), which is related to $D$ approach.

Regarding (6.36), we have

$$\mathrm{dist}(1, |\kappa_j|^{-2} \bar{\kappa}_j \lambda_j) = |1 - \theta \, |\lambda_j| \, / \, |\kappa_j|| , \tag{6.37}$$

where $\theta = (\bar{\kappa}_j \lambda_j)/(|\kappa_j| \, |\lambda_j|)$ and its value depends on how good is the approximation of the *angle* of the complex eigenvalue $\lambda_j$ made by $\kappa_j$. Regarding (6.35), we have

$$\mathrm{dist}(1, |\kappa_j|^{-2} |\lambda_j|^2) = \left|1 - (|\lambda_j| \, / \, |\kappa_j|)^2\right| = t \cdot |1 - |\lambda_j| \, / \, |\kappa_j|| , \tag{6.38}$$

where $t = 1 + |\lambda_j| \, / \, |\kappa_j|$ and its value depends on how good is the approximation of the *modulus* of the complex eigenvalue $\lambda_j$ made by $\kappa_j$. We notice that if the angle of $\kappa_j$ is near to the angle of $\lambda_j$, then $\theta \approx 1$, hence

$$\mathrm{dist}(1, |\kappa_j|^{-2} |\lambda_j|^2) \approx t \cdot \mathrm{dist}(1, |\kappa_j|^{-2} \bar{\kappa}_j \lambda_j), \tag{6.39}$$

where $t > 1$ ($t \approx 2$ if also the approximation of the modulus is close to be exact). So by comparing (6.37) and (6.38), we can summarize that the reblurring preconditioner $Z$ gives the possibility to manage and control the angle $\theta$ (which is basically fixed for $D$) between its eigenvalues and the corresponding eigenvalues of the system matrix. Figure 6.10 (left) shows that for $\theta = 1$, $Z$ variant beats $D$ preconditioning everywhere, since the distance (6.37) (dotted line) is always lower than (6.38) (solid line). As $\theta$ goes away

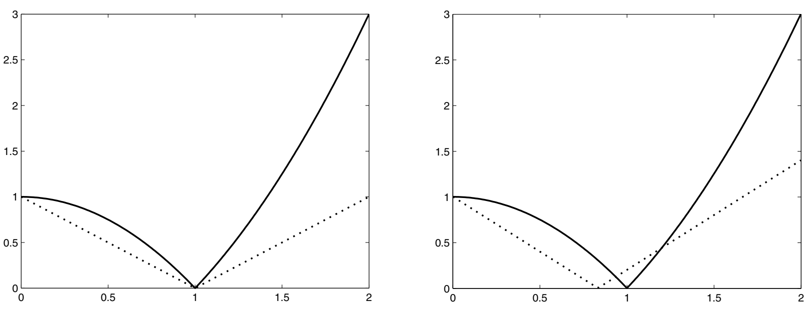


Figure 6.10: Plot, for $\theta = 1$ (left) and $\theta \approx 1$ (right), of the distance (6.38) in solid line, and for the distance (6.37) in dotted line, with the ratio $|\lambda_j| \,/\, |\kappa_j|$ on the $x$-axis.

from 1, the interval in which (6.38) is lower than (6.37) spreads slowly, as shown by Figure 6.10 (right). From this simple analysis we may argue to really reach an improvement of the performance of preconditioned iterative methods by using the proposed $Z$ strategy for the more involving zero BCs too.

## 6.4.2 Computational results

To test BCCB preconditioning and $Z$ variant for BTTB systems, we take into account the Saturno deblurring problem of Figure 6.11. The PSF is created as a uniform sampling of 101 points in the domain $[-5, 5] \times [-5, 5]$ of the Gaussian kernel $e^{-\sqrt{x^2+y^2}}$ of 101 points in $[-5, 5] \times [-5, 5]$ (as in [48]).

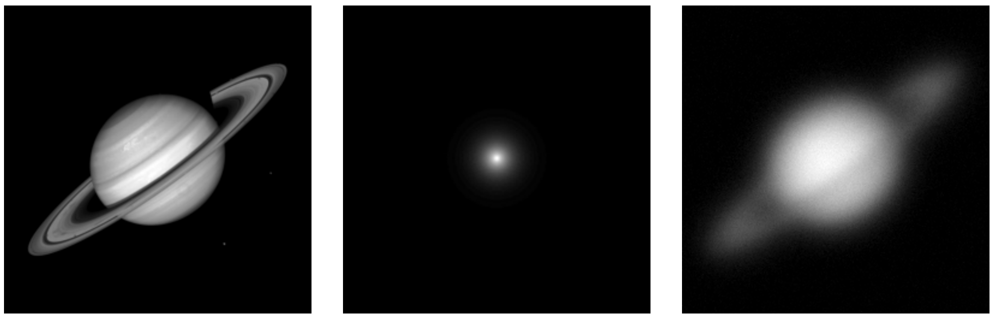


Figure 6.11: True image, PSF, blurred and noisy image.

We have generated the blurred and noisy data $b$, adding about 4% of white Gaussian noise. First we compare the performances of $D$-Landweber method (5.23) against the $Z$-Landweber method (6.8), both using $q$ Tikhonov filtering (6.34). In this case, the top side of Table 6.8 shows some improvements given

| | **$q$ Tikhonov filter** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $q = 1$ | | $q = 2$ | | $q = 3$ | | $q = 3.5$ | |
| | RRE | IT | RRE | IT | RRE | IT | RRE | IT |
| $D$-Landweber | 0.5964 | 1 | 0.2275 | 2 | 0.1645 | 32 | 0.1424 | 104 |
| $Z$-Landweber | 0.1522 | 3 | 0.1426 | 27 | 0.1400 | 80 | 0.1394 | 111 |

| | **$p$ T.Y.Z. filter** | | | | | | **Reblurring filters** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $p = 2$ | | $p = 1.5$ | | $p = 1$ | | VC filter | | $f$ filter | |
| | RRE | IT | RRE | IT | RRE | IT | RRE | IT | RRE | IT |
| $D$-Landweber | 0.3372 | 1 | 0.2023 | 4 | 0.1542 | 42 | 0.1799 | 16 | 0.1793 | 16 |
| $Z$-Landweber | 0.1414 | 57 | 0.1413 | 60 | 0.1426 | 67 | 0.1506 | 35 | 0.1472 | 38 |

Table 6.8: Results for $D$-Landweber and $Z$-Landweber, with $D$ and $Z$ built by different filters for example in Figure 6.11.

by $Z$ variant. As intermediate result of the $q = 3.5$ column, we have that $D$-Landweber needs 93 steps — so the preconditioning influence is weak — to reach the same quality that $Z$-Landweber reaches after 27 iterations for $q = 2$. $Z$-Landweber is preferable both in terms of speed (i.e. number of iterations) and accuracy (i.e. RRE). We enforce this comment on the computational time, by reminding that one single iteration of $Z$-Landweber is even cheaper than one single iteration of $D$-Landweber. In Figure 6.12, some restorations relative to the top side of Table 6.8 are shown, highlighting the difference between the two strategies. For the same level $q = 2$, the $Z$-Landweber restoration (center) is better than the $D$ one (left). By comparing a number of iteration similar to the $Z$ one, for $D$-Landweber we show the restoration with $q = 3$ on the right, where again a lower reconstruction of the details appears. In Figure 6.13 we report the related convergence histories, that is, the RRE vs the iteration index, for $q = 2$ (left) and $q = 3$ (right). We notice that $Z$ approach and $D$ approach walk together for the initial iterations, then $D$-Landweber (solid line) starts to diverge, whereas $Z$-Landweber (dashed–dotted line) continue its iterative process going to an valuable better approximated solution. This behaviour of $Z$ is useful since it simplifies the choice of the stopping iteration.

The bottom-right side of Table 6.8 concerns the VC filter (6.26) and the reblurring filters of type (6.27) ($\zeta = 0.01$, $\gamma = 5000$). The difference between the $D$-Landweber and $Z$-Landweber methods is again strong, and the $Z$ restorations is better in terms of RRE. The same fact can be observed for the $p$ Tyrtyshnikov Yeremin Zamarashkin filter, with $\zeta = 0.05$, reported in the bottom-left side of Table 6.8. The $D$-Landweber suffers often from convergence problems: if we pick a lower $\zeta$ value ($\zeta = 0.01$), $D$-Landweber does not converge at all for any value of $p$, while our $Z$-Landweber works

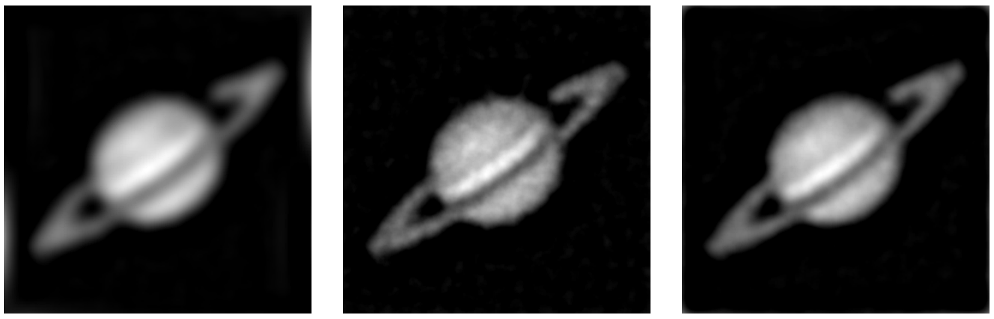


Figure 6.12: Restorations by $q$ Tikhonov filter: $D$-Landweber restoration with $q = 2$ (RRE 0.2275, IT 2), $Z$-Landweber restoration with $q = 2$ (RRE 0.1426, IT 27), $D$-Landweber restoration with $q = 3$ (RRE 0.1645, IT 32).

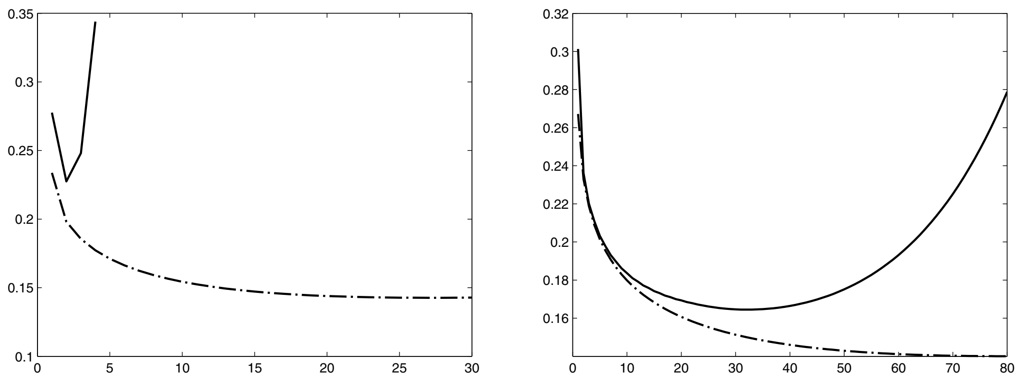


Figure 6.13: Convergence history of $D$-Landweber (solid line) and $Z$-Landweber (dashed–dotted line) with $q$ Tikhonov filter: RRE vs iteration number, for $q = 2$ (left) and $q = 3$ (right).

fine, since it gives RRE $= 0.1675$ after 5 steps for $p = 1.5$, and RRE $= 0.1580$ after 26 steps for $p = 1$. Anyway, a similar negative influence starts to affect also $Z$-Landweber as $p$ becomes greater, even if in a less dramatic way.

Finally we report also here the $Z$-Landweber method presented in §6.2, which has the great advantage that it does not need any parameter setting. For $Z$-Landweber 1/4 we get an RRE equal to 0.1470 after 60 steps, which is a quite good result compared to ones reached by sophisticated filters, which, on the contrary, require the tuning of the threshold parameter.

As already remarked, the efficiency of $Z$ variant by filtering techniques of §6.3 does not depend on the symmetry of the PSF. This is important since, differing from the $D$ preconditioning (which is always applied to the normal equation with the symmetric matrix $A^H A$), the preconditioner $Z$ is applied to possibly non-symmetric matrix $A$. To evaluate this fact, we consider a test having the same true Saturno image but a highly non-symmetric linear motion blur as PSF. The resulting Saturno deblurring problem has about 3.6% of white Gaussian noise (see the blurred and noisy image on the left of Figure 6.14). We remind that the simple coarsening technique introduced in §6.2 fails in the non-symmetric case, while $D$-Landweber and $Z$-Landweber exhibit now a behaviour similar to one showed in the symmetric case, i.e. they still work and $Z$ is again better than $D$. In Figure 6.14, we report two restorations made by those methods with $q$ Tikhonov filter, where $q = 1.5$. $Z$-Landweber restoration (right) is good enough, while $D$-Landweber one is still blurred (center) and shows stains along the borders.

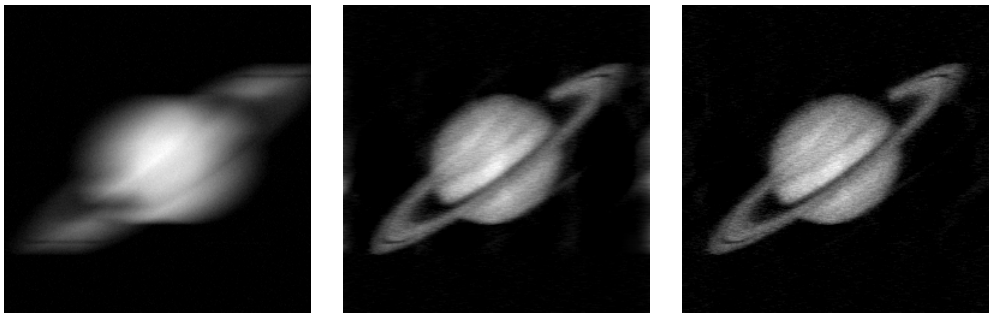


Figure 6.14: Blurred and noisy image, $D$-Landweber restoration (RRE 0.1771, IT 7), $Z$-Landweber restoration (RRE 0.1352, IT 14), with $q$ Tikhonov filter, where $q = 1.5$.

### 6.4.3 A note on projection

Since our PSFs and images are non-negative, as usual, all our numerical tests consider the projected variant of our methods, i.e. we apply the projection $\mathcal{P}$ on the non-negative cone ($c_1 = 0$ and $c_2 = +\infty$)

$$[\mathcal{P}(x)]_j = \begin{cases} c_1 & \text{if } x_j < c_1, \\ x_j & \text{if } c_1 \le x_j \le c_2, \\ c_2 & \text{if } x_j > c_2, \end{cases} \tag{6.40}$$

after any step of the iterative method. To be more precise, instead of the basic iteration $x_k$ of any iterative method, we take $x_k := \mathcal{P}(x_k)$. We notice that statistical methods do not need the projection on the non-negative cone, because they act pixel by pixel, hence they naturally preserve non-negativity. About BCCB preconditioning of BTTB problems, we have to make an interesting remark. As just said, usually projection $\mathcal{P}$ is applied in this manner $x_k = x_{k-1} + \tau D A^H (b - A x_{k-1})$, $\quad x_k := \mathcal{P}(x_k)$, and so we have done in our computational tests. Nevertheless in this particular case we have experimentally found that the following new projection strategy

$$x_k = x_{k-1} + \tau D A^H (b - A x_{k-1}^+), \qquad x_{k-1}^+ = \mathcal{P}(x_{k-1}), \tag{6.41}$$

can improve the performances of the method. The new strategy (6.41) consists to do an iteration step with simultaneously both projected and non-projected iterations $x_{k-1}$ and finally to pick $x_k^+$ as computed solution. In particular, we have numerically tested that this strategy is able to reduce instability. This behaviour can be seen in the convergence history of Figure 6.15, related to the new iteration (6.41), which can be compared with the corresponding of Figure 6.13 (left) of the classical iteration. Both the figures Figure 6.13 (left) and 6.15 are related to the same $q$ Tikhonov filter ($q = 2$) for Saturno deblurring problem. Furthermore, even though general considerations on the superiority of $Z$ variant are still valid, this new projection technique brings performances of $D$-Landweber with reblurring filters close to those of $Z$-Landweber.

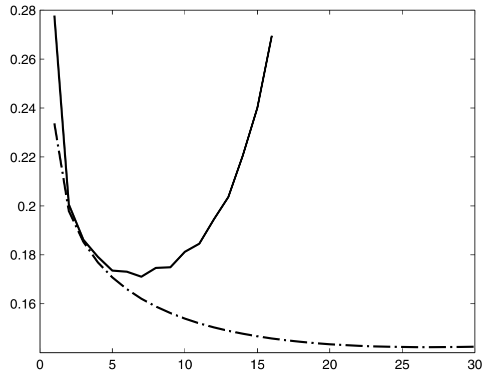


Figure 6.15: Convergence history of $D$-Landweber (solid line) and $Z$-Landweber (dashed–dotted line) with $q$ Tikhonov filter, employing the new projection strategy (6.41): RRE vs iteration number, for $q = 2$.

## 6.5 Deblurring problems with accurate boundary conditions

We recall that, when we build a BCCB preconditioner $D$ for $A$ BTTB, we make use of a circulant approximation (in our case the T. Chan optimal approximation [33]) $\tilde{A}$ of $A$, that is

$$A = \tilde{A} + N + R, \tag{6.42}$$

where $N$ is a low norm correction and $R$ is a low rank correction. This means that we can employ the same idea and so use $Z$ and $D$ both BCCB, if the matrix $A$, having other boundary conditions, can be written in this form. This is what happens for instance with Reflective and Anti-Reflective BCs. We stress that for these accurate boundary conditions, the matrix-vector product with $A^H$ could be unavailable or computationally expensive. Hence instead of $A^H$ one usually considers $A'$, which is the matrix related to the PSF rotated by 180 degrees (see Chapter 5). So in this context the use of $Z$ instead of $DA^H$ could be very useful, since it avoid to resort to $A^H$ or to replace it with $A'$.

To test BCCB preconditioning for problems with reflective BCs, we take into account the Cameraman deblurring problem of Figure 6.16. The PSF is a strongly non-symmetric portion of a Gaussian blur. We have generated the blurred and noisy data $b$, adding about 2% of white Gaussian noise. Numerical results are reported in Table 6.9, while some restorations are reported in Figure 6.17. We note that for $Z$-Landweber the regularization parameter affects only the speed of convergence, while for $D$-Landweber a wrong estimation of the regularization parameter leads to a poor computed approximation. Such behaviour is clearly shown in Figure 6.18, where we can see that the restoration error for $Z$-Landweber remains stable also for small values of the regularization parameter. Of course, for small values of the regularization parameter the method converges faster and a good restoration can be computed within few iterations. To test BCCB preconditioning for problems with anti-reflective BCs, we take into account the Bridge deblurring problem of Figure 6.19. The PSF is a strongly non-symmetric portion of a defocus blur. We have generated the blurred and noisy data $b$, adding about 0.75% of white Gaussian noise. Numerical results are reported in Table 6.10, while some restorations are reported in Figure 6.20. In these tests we have considered only strongly non-symmetric PSFs, since if the PSF is near to be symmetric, one usually takes the reflective (resp. anti-reflective) blurring matrix associated to the symmetrized PSF as preconditioner, as shown in Chapter 5.

As one could expected, even if what we said in the beginning of this section is true, circulant preconditioning for deblurring problems with reflective or anti-reflective BCs show lower efficacy than one exhibited for problems having zero BCs. In particular filters parameters are very difficult to set and some filters, like Low Pass filter or Hanke, Nagy, Plemmons one, do not work well, i.e. for some values of the threshold parameter they give poor restorations and for other values they do not meaningfully speed up the method (as preconditioning should do). As observed before, also for these boundary conditions $Z$ variant is able to be more stable than classical preconditioning

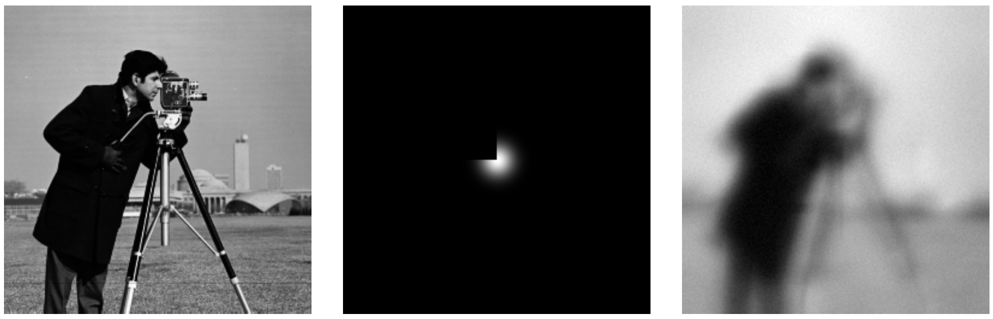


Figure 6.16: True image, PSF, blurred and noisy image.

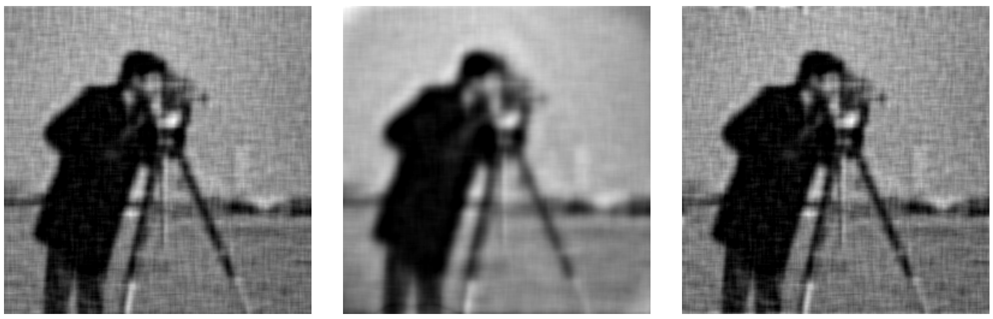


Figure 6.17: Employing Tikhonov filter, $D$-Landweber restorations for $\alpha = 0.3$ (RRE 0.1632, IT 137), $\alpha = 0.25$ (RRE 0.1876, IT 24), and $Z$-Landweber restoration for $\alpha = 0.03$ (RRE 0.1656, IT 15).

| | **Tikhonov filter** | | | | | | **T.Y.Z. filter** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\alpha = 0.3$ | | $\alpha = 0.1$ | | $\alpha = 0.03$ | | $\zeta = 0.5$ | | $\zeta = 0.3$ | | $\zeta = 0.13$ | |
| | RRE | IT | RRE | IT | RRE | IT | RRE | IT | RRE | IT | RRE | IT |
| $D$-Landweber | 0.1632 | 137 | 0.3638 | 1 | 0.8198 | 1 | 0.1629 | 227 | 0.3185 | 1 | 0.7119 | 1 |
| $Z$-Landweber | 0.1650 | 144 | 0.1651 | 48 | 0.1656 | 15 | 0.1649 | 239 | 0.1649 | 143 | 0.1649 | 62 |

Table 6.9: Cameraman deblurring problem: $D$-Landweber and $Z$-Landweber with $D$ and $Z$ built by different filters.

| | **Tikhonov filter** | | | | | | **$f$ filter** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\alpha = 0.45$ | | $\alpha = 0.4$ | | $\alpha = 0.12$ | | $\gamma = 150$ | | $\gamma = 80$ | | $\gamma = 12$ | |
| | RRE | IT | RRE | IT | RRE | IT | RRE | IT | RRE | IT | RRE | IT |
| $D$-Landweber | 0.1520 | 198 | 0.2023 | 29 | 0.3361 | 1 | 0.3804 | 1 | 0.3339 | 1 | 0.2306 | 8 |
| $Z$-Landweber | 0.1611 | 184 | 0.1611 | 163 | 0.1607 | 46 | 0.1663 | 40 | 0.1685 | 77 | 0.1645 | 255 |

Table 6.10: Bridge deblurring problem: $D$-Landweber and $Z$-Landweber with $D$ and $Z$ built by different filters.

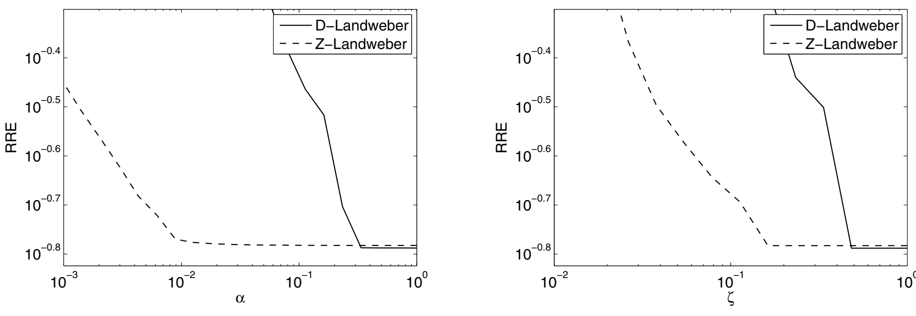


Figure 6.18: RRE vs regularization parameter ($\alpha$ or $\zeta$) for example in Figure 6.16 and Tikhonov filter (left) and T.Y.Z. filter (right).

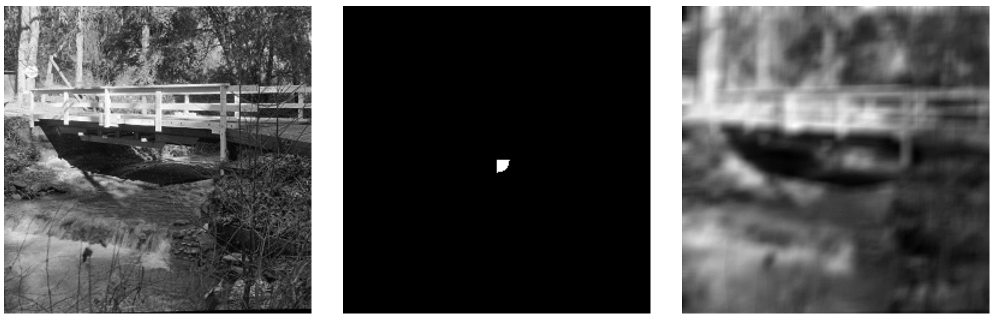


Figure 6.19: True image, PSF, blurred and noisy image.

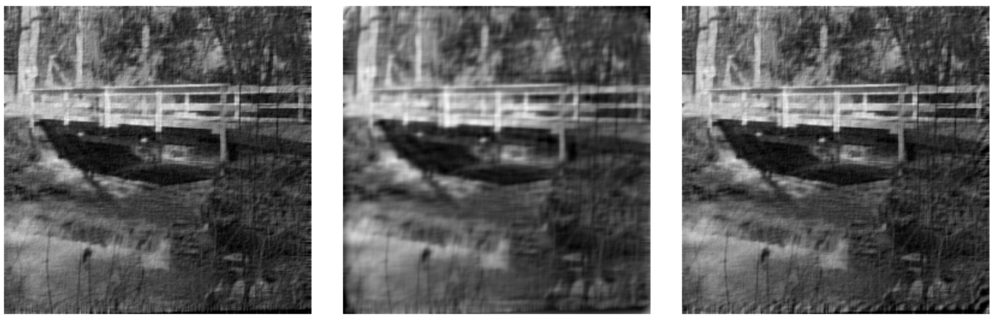


Figure 6.20: Employing Tikhonov filter, $D$-Landweber restorations for $\alpha = 0.45$ (RRE 0.1520, IT 198), $\alpha = 0.4$ (RRE 0.2023, IT 29) and $Z$-Landweber restoration for $\alpha = 0.12$ (RRE 0.1607, IT 46).

and so to give rise to an higher acceleration when it is applied to an iterative method. The only drawback of $Z$ variant is that the quality of the restoration is sometimes slightly worse than one obtained by classical preconditioning (see first column of Table 6.10). This may be caused by the fact that, since $Z$ is chosen BCCB, we are imposing periodic BCs, so we completely lose information on the boundary, while $DA'$ keep this information thanks to $A'$, which is a reflective (resp. anti-reflective) blurring matrix. Indeed in the third image of Figure 6.17 and of Figure 6.20 we can notice that the errors are mainly placed along the borders (right and bottom border in the case of Bridge reconstruction). As delineated in the next section, it is possible to avoid this by constructing $Z$ with the same BCs of $A$.

## 6.6 A general $Z$ algorithm

We remind that till now we have compared $D$ BCCB with $Z$ BCCB, so have put $Z$ and $D$, in the wide matter of preconditioning, on the same footing. Nevertheless, since the original idea is to replace $A^H$ with $Z$, the logical consequence is that $Z$ has to inherit its BCs from $A$. This means that it is possible to go along another course, i.e. to use circulant algebra only to apply filters and then to come back and to employ the original BCs of the problem, no more only periodic BCs as in the circulant preconditioning theory. In other words, **for any BCs**, we can perform the next ALGORITHM. In the sequel we will indicate with $\mathbf{C}_{n^2}$ the BCCB matrix associated with ($h_{PSF}$, ‘periodic’) of size $n \times n$ and with $c_j$ its eigenvalues, while FFT and IFFT will denote respectively the two-dimensional Fast Fourier Transform and its Inverse.

```
$Z \longleftarrow$ ALGORITHM($h_{PSF}$, BCs)
------------------------------------------------
· get $\{c_j\}_{j=1}^{n^2}$ by computing FFT of $h_{PSF}$
· get $z_j$ by applying a filter to $c_j$
· get $w_{PSF}$ by computing IFFT of $\{z_j\}_{j=1}^{n^2}$
· generate $Z$ from ($w_{PSF}$, BCs)
```

We highlight that this algorithm is consistent, in fact if the filter is identity, i.e. there is no filtering, we have $Z = A^H$. Here we present the algorithm for building $Z$, but clearly, since the filters for $Z$ are related with those for $D$, an analogous algorithm can be applied to create the preconditioner $D$. We underline that the filters can be chosen whether of inversion or of reblurring type.

### 6.6.1 Zero BCs

When boundary conditions are the zero one, it is quite natural to seek a link between the proposed algorithm and Toeplitz preconditioning. We briefly present the preconditioner proposed in [32, 67], that we call *Chan, Ng* preconditioner. For simplicity, we do this in the one-dimensional case. Given the Toeplitz matrix $A$, associated to the PSF of the problem

$$A = \begin{pmatrix} h_0 & \dots & h_{-q} & 0 & 0 & 0 & 0 \\ \vdots & h_0 & & h_{-q} & 0 & 0 & 0 \\ h_q & & h_0 & & h_{-q} & 0 & 0 \\ 0 & h_q & & h_0 & & h_{-q} & 0 \\ 0 & 0 & h_q & & h_0 & & h_{-q} \\ 0 & 0 & 0 & h_q & & h_0 & \vdots \\ 0 & 0 & 0 & 0 & h_q & \dots & h_0 \end{pmatrix}_{n\times n},$$

for our algorithm we simply consider the circulant matrix associated to the same PSF

$$\mathbf{C}_n = \begin{pmatrix} h_0 & \dots & h_{-q} & 0 & 0 & h_q & \dots \\ \vdots & h_0 & & h_{-q} & 0 & 0 & h_q \\ h_q & & h_0 & & h_{-q} & 0 & 0 \\ 0 & h_q & & h_0 & & h_{-q} & 0 \\ 0 & 0 & h_q & & h_0 & & h_{-q} \\ h_{-q} & 0 & 0 & h_q & & h_0 & \vdots \\ \dots & h_{-q} & 0 & 0 & h_q & \dots & h_0 \end{pmatrix}_{n\times n}.$$

Then, denoting by $-\mathcal{F}$ the *inversion* made by the filters, we choose $D = (\mathbf{C}_n)^{-\mathcal{F}}$. Chan, Ng consider a circulant embedding of $A$

$$\mathbf{C}_{2n} = \begin{pmatrix} A & R \\ R & A \end{pmatrix}_{2n\times 2n}, \quad (\mathbf{C}_{2n})^{-\mathcal{F}} = \begin{pmatrix} C_1 & C_2 \\ C_2 & C_1 \end{pmatrix}_{2n\times 2n} \tag{6.43}$$

and choose $D = C_1$, that is the $n \times n$ leading principal submatrix of $(\mathbf{C}_{2n})^{-\mathcal{F}}$. In pratice, both techniques try to approximate $1/f$, where $f$ is the generating function of the Toeplitz matrix $A$.

#### One-dimensional test

We test these different approaches for the one-dimensional blurring problem reported in Figure 6.21, having a level of noise that is about 1.6%. As always we compare the results of preconditioning techniques with those of

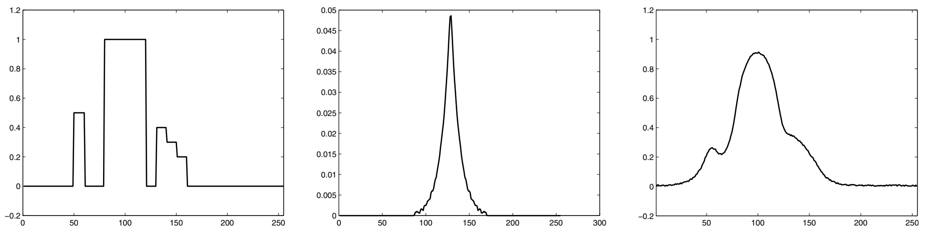


Figure 6.21: True signal, PSF, blurred and noisy signal.

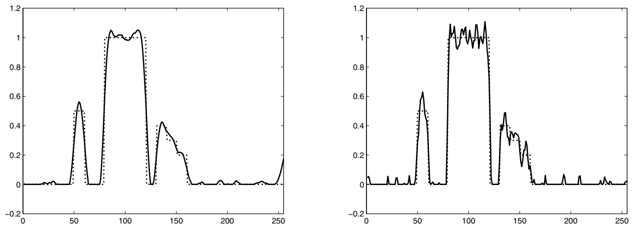


Figure 6.22: Chan restoration (RRE 0.1849, IT 6) and $Z$ Chan restoration (RRE 0.1424, IT 85).

the basic Landweber method, with non-negative projection, which reaches an RRE equal to 0.1397 in 4086 iterations without preconditioning. We employ Tikhnov filter for all the available possibilities, including T. Chan optimal preconditioner [33]

$$c_k = \frac{(n - |k|)h_k + |k|\, h_{k-n}}{n}, \quad - n < k < n \tag{6.44}$$

and its $Z$ variant, which are circulant matrices. Table 6.11 confirm again the strength of $Z$ strategy, which converges when other approaches fail and gets always low RREs. Moreover, as already observed, the improvement given by $Z$ variant in the case of circulant preconditioning is huge (see Figure 6.22). Also Chan, Ng Toeplitz preconditioner improves the performances of Chan circulant preconditioner, in particular it gives rise to computational results analogous to ones gained by our $D$ Toeplitz preconditioner, which is generated by the algorithm presented in this section, the same algorithm of $Z$.

### Two-dimensional test

To test BTTB preconditioning — where both $D$ and $Z$ are created by the proposed algorithm — for restoration problems having $A$ BTTB, we take into account the Saturno problem of Figure 6.11. We report Table 6.12, that has to be compared with Table 6.8 in §6.4.2 (we have chosen the regularization

| | **Tikhonov filter** | | | | | |
|---|---|---|---|---|---|---|
| | $\alpha = 0.001$ | | $\alpha = 0.02$ | | $\alpha = 0.04$ | |
| | RRE | IT | RRE | IT | RRE | IT |
| Chan | 0.2121 | 1 | 0.1849 | 6 | 0.1577 | 44 |
| $Z$ Chan | 0.1581 | 4 | 0.1424 | 85 | 0.1412 | 168 |
| Chan, Ng | 0.3222 | 2 | 0.1424 | 84 | 0.1414 | 166 |
| $D$ | 0.3222 | 2 | 0.1424 | 84 | 0.1414 | 166 |
| $Z$ | 0.1568 | 4 | 0.1425 | 84 | 0.1414 | 167 |

Table 6.11: Landweber method with different preconditioners and $Z$ variants.

parameters in the same way). We can observe that the distance between $D$-preconditioning approach and $Z$ variant one is higher for $q = 1$ and $p = 2$. About $p$ Tyrtyshnikov Yeremin Zamarashkin filter (6.32), if we pick $\zeta = 0.01$, $D$-Landweber converges to an acceptable restoration only for values of $p$ near to 1, for $p = 1$ it reaches an RRE equal to 0.1614 after 23 steps, while $Z$-Landweber reaches an RRE equal to 0.1670 after 5 steps for $p = 1.5$ and an RRE equal to 0.1578 after 26 steps for $p = 1$. About VC filter, if we pick $\zeta = 0.02$, $D$-Landweber reaches an RRE equal to 0.1472 in 56 steps, while $Z$-Landweber reaches an RRE equal to 0.1414 in 68 steps. More in general, once left out the instances in which $D$-Landweber shows instability, we can notice that the two strategies give similar results both in terms of best restoration and of iterations number. Nevertheless we remind that one iteration of an iterative method that employs $Z$-variant has a computational cost which is lower than one iteration of a method that employs $D$-preconditioning. Therefore $Z$ variant is again preferable.

| | $q$ **Tikhonov filter** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $q = 1$ | | $q = 2$ | | $q = 3$ | | $q = 3.5$ | |
| | RRE | IT | RRE | IT | RRE | IT | RRE | IT |
| $D$-Landweber | 0.7259 | 1 | 0.1504 | 12 | 0.1396 | 87 | 0.1390 | 121 |
| $Z$-Landweber | 0.1512 | 3 | 0.1421 | 29 | 0.1398 | 86 | 0.1392 | 120 |

| | $p$ **T.Y.Z. filter** | | | | | | **Reblurring filters** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $p = 2$ | | $p = 1.5$ | | $p = 1$ | | VC filter | | $f$ filter | |
| | RRE | IT | RRE | IT | RRE | IT | RRE | IT | RRE | IT |
| $D$-Landweber | 0.4157 | 1 | 0.1403 | 61 | 0.1421 | 68 | 0.2773 | 4 | 0.1470 | 37 |
| $Z$-Landweber | 0.1480 | 41 | 0.1411 | 60 | 0.1422 | 68 | 0.1501 | 35 | 0.1463 | 39 |

Table 6.12: $D$-Landweber and $Z$-Landweber with $D$ and $Z$ (both BTTB) built by different filters.

| | $D$-Landweber | | $Z$-Landweber | |
|---|---|---|---|---|
| | RRE | IT | RRE | IT |
| Reflective | 0.1107 | 18 | 0.1031 | 5 |
| Anti-Reflective | 0.0995 | 25 | 0.0999 | 4 |

Table 6.13: Results of $D$-Landweber and $Z$-Landweber method, related to the Cameraman deblurring problem, employing different BCs.

### 6.6.2 Accurate BCs

Now we study our proposal of §6.6 applied to Reflective and Anti-Reflective boundary conditions. We recall that for both these BCs the optimal preconditioner is associated to the symmetrized PSF, as shown in Chapter 5. To make tests we use the same two data set of §5.3, i.e. Cameraman and Bridge ones. As done in that section, we consider Landweber method and Tikhonov filter, for which we set the regularization parameter $\alpha$ manually, in order to obtain excellent performances both in terms of RRE and of number of steps. Table 6.13 (Cameraman) and Table 6.14 (Bridge) clearly show that this new preconditioner $D$ gives rise to computational results which are better (see in particular the IT column) than ones gained by the optimal preconditioner, reported in Table 5.1 and Table 5.2. As expected, the distance between the two strategies is huge in the case of strongly non-symmetric PSF, since our proposal can manage directly non-symmetric structures, differently from optimal preconditioning, which is based on symmetrization. In this situation $Z$ variant represents a further improvement, which allows to obtain a more powerful acceleration than $D$ and often also better restorations; see in particular the Cameraman restoration problem for Reflective BCs (first image of Figure 6.23 and Figure 6.24). For this example in Figure 6.24 we observe that, from the point of view of the human eye, maybe Reflective restoration may be preferable to Anti-Reflective one, since it does not have the annoying ringing effect along the upper border which is present in the other one. This is caused by the fact that Anti-Reflective BCs are more accurate but they are more sensitive to non-symmetry of the PSF and to noise. A low level of noise — we remind that for both these problems it is about 0.2% — combined with slightly non-symmetric PSF, is sufficient to produce unpleasant ringing phenomenon along the borders. Finally we can say that, since for a suitable choice of the parameter $\alpha$ we have that $Z$-Landweber converges in few steps, one can think to apply $Z$ variant also in the framework of direct regularization method. To this topic is devoted the next section.

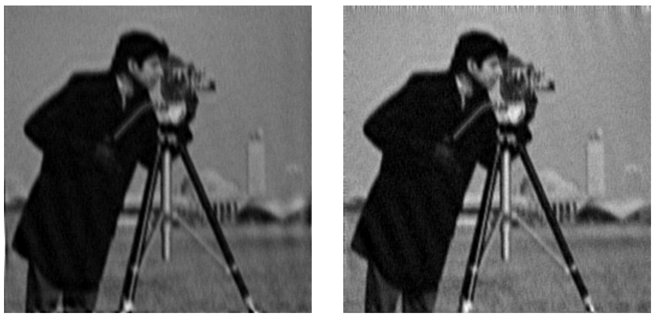


Figure 6.23: Best $D$-Landweber restorations, employing reflective and anti-reflective BCs.

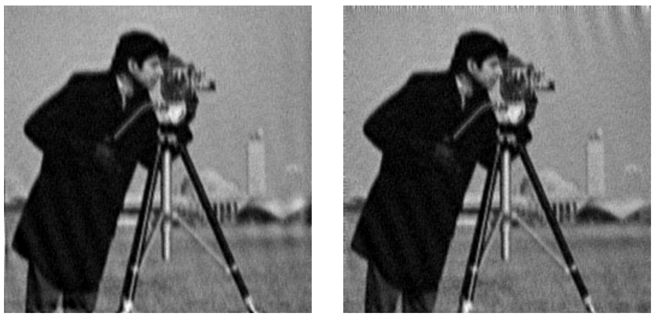


Figure 6.24: Best $Z$-Landweber restorations, employing reflective and anti-reflective BCs.

| | $D$-Landweber | | $Z$-Landweber | |
|---|---|---|---|---|
| | RRE | IT | RRE | IT |
| Reflective | 0.1982 | 2 | 0.1922 | 1 |
| Anti-Reflective | 0.1868 | 311 | 0.1858 | 1 |

Table 6.14: Results of $D$-Landweber and $Z$-Landweber method, related to the Bridge deblurring problem, employing different BCs.

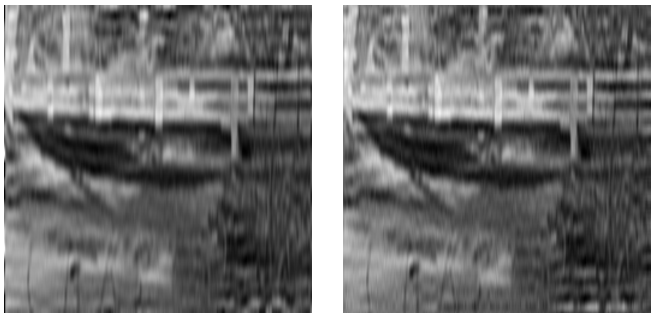


Figure 6.25: Best $D$-Landweber restorations, employing reflective and anti-reflective BCs.

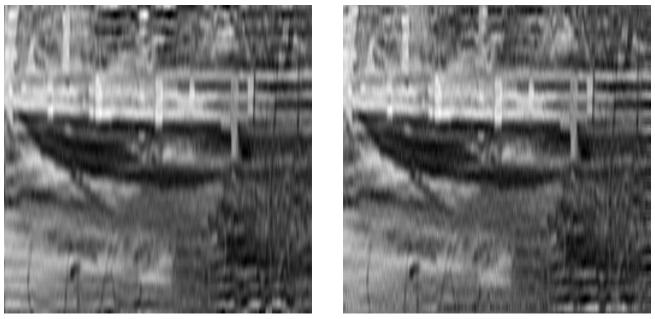


Figure 6.26: Best $Z$-Landweber restorations, employing reflective and anti-reflective BCs.

## 6.7 Direct regularization methods

In this section we outline the possible use of $Z$ variant for direct regularization methods. Here we examine Tikhonov regularization, but all the considerations made for it can be extended to other similar regularization methods, like those conceived by Hanke, Nagy, Plemmons [69] and by Tyrtyshnikov, Yeremin, Zamarashkin [118]. For discrete ill-posed problems, *Tikhonov regularization* in general form leads to the solution of the minimization problem

$$\min \left\{ \|Ax - b\|_2^2 + \alpha \|x\|_2^2 \right\}, \tag{6.45}$$

where the regularization parameter $\alpha > 0$ controls the weight given to minimization of the regulation term; $\|x\|_2^2$ is usually called the penalty term. An alternative formulation of (6.45) is

$$(A^H A + \alpha I)x = A^H b \tag{6.46}$$

whose solution is unique and it is given by

$$x = A_\alpha^\dagger b, \tag{6.47}$$

where $A_\alpha^\dagger = (A^H A + \alpha I)^{-1} A^H$. This matrix $A_\alpha^\dagger$ is called the *Tikhonov generalized inverse*. In literature there are fundamentally three cases in which this approach can be applied:

a) if BCs are periodic, solve (6.46) by FFT (in other words we can compute $A_\alpha^\dagger$ since we are in the Circulant algebra);

b) if BCs are reflective and the PSF of the problem is strongly symmetric, solve (6.46) by FCT (in other words we can compute $A_\alpha^\dagger$ since we are in the Reflective algebra);

c) if BCs are anti-reflective and the PSF of the problem is strongly symmetric, solve (6.46) by FST (in other words we can compute $A_\alpha^\dagger$ since we are in the Anti-Reflective algebra).

We remind that really, since we employ reblurring strategy, we solve $(A'A+\mu I)x = A'b$, where $A'$ is the matrix related to the PSF rotated by 180 degrees. Making use of $Z$ variant, $A_\alpha^\dagger$ is replaced by $Z_\alpha$, i.e.

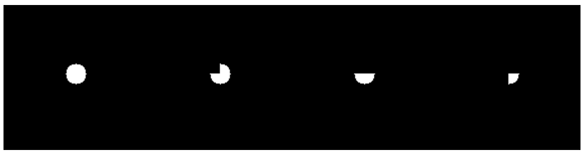
$$x = Z_\alpha b \tag{6.48}$$

and we have *$Z$-Tikhonov regularization*. We underline that an important difference between this strategy and $Z$-Landweber method is that for the last, since it is an iterative method, the regularization parameter is the number of steps $k$, while for $Z$-Tikhonov regularization the key parameter is $\alpha$. However, if we consider $Z$-Landweber method, in which the initial guess is chosen equal to zero and $\tau$ is chosen equal to one, and for which $Z$ is built using Tikhonov filter (with the same $\alpha$ employed in the Tikhonov regularization), we can observe that its first step

$$x_k = x_{k-1} + \tau Z(b - Ax_{k-1}) = Zb \tag{6.49}$$

is equivalent to $Z$-Tikhonov regularization. Therefore, inspired by general $Z$ variant utilized in §6.6 for iterative methods, instead of (a), (b), (c), we have only: **for any BCs** and **for any PSF**, create $Z_\alpha$ by the algorithm presented at the beginning of §6.6 and find the solution by (6.48). Clearly this strategy, if BCs are periodic, is the same of (a). If BCs are reflective or anti-reflective, with strongly symmetric PSF, it gets reconstructions very near to those gained by (b) or (c). On the other hand, in all the remaining cases, it represents a new technique that allows to have always at our disposal an approximation of $A_\alpha^\dagger$ and so it permits to us to solve the linear systems related to Tikhonov regularization by (6.48). We remind that for zero, periodic, reflective, anti-reflective boundary conditions there is an algorithm, based on Fast Fourier Transform, for the matrix vector product, so the computational cost is $O(n^2 \log(n))$. Also the algorithm that generates $Z_\alpha$ requires only one FFT and one IFFT. Therefore the overall cost consists only in few FFTs.

To have a picture of the situation, we report the results relative to four restoration problems, having the same true image and for which we have imposed reflective BCs, related to four different defocus blur, all with similar level of noise (about 2%). We compare the best restoration performed by the classical Landweber method, typically in one hundred or two hundred iteration steps, with $Z$-Tikhonov regularization, for which we picked $\alpha = 0.005$ for the first problem and $\alpha = 0.008$ elsewhere. Also in this case we employ non-negative projection.



| **Landweber** | 0.1609 | 0.1493 | 0.1466 | 0.1272 |
|---|---|---|---|---|
| **Z-Tikhonov** | 0.1659 | 0.1542 | 0.1511 | 0.1329 |

Table 6.15: $Z$ variant of Tikhonov regularization vs Landweber method.

Table 6.15 shows that in all instances $Z$-Tikhonov regularization reaches an RRE that is only about 0.5% worse than the one gained by Landweber method. Furthermore, as we have advanced, the RRE of $Z$-Tikhonov regularization in the first column, i.e. the strongly symmetric case, is very close to that obtained by approach described in (b), since it is different only from the four decimal digit. Finally we can assert that also in the field of direct methods, $Z$ variant represents a valid way to strengthen regularization techniques, releasing them by ties represented by BCs and symmetry of the PSF.

## 6.8 Merits of $Z$ variant

After this general survey on $Z$ variant for iterative and direct methods, we can make some comments concerning the obtained results and stress the merits of $Z$ variant. Despite the closeness sometimes observed between performances of methods with $D$-preconditioning and with $Z$ variant, the former has several disadvantages. The more important ones that we have singled out are the next three:

a) $DA^H$ loses the *matrix structure*, while $Z$ preserves it;

b) *computational cost* of $Z$ variant remains fundamentally the same of the standard iterative method to which variant is applied, while the cost of a preconditioned method is in general higher than the standard one;

c) for all filters $Z$ variant shows an higher *stability*, and with this word we mean that iterative methods compute a good restoration also when regularization parameters $\zeta$ and $\alpha$ are very small.

Finally we provide an example that shows as our $Z$ approach gives an acceleration of the speed of convergence higher than whose provided by classical acceleration techniques widely used in the literature. Concerning the

| | RRE | IT |
|---|---|---|
| LR aut acc | 0.3646 | 83 |
| $Z$-LR | 0.3643 | 55 |
| $Z$-LR aut acc | 0.3587 | 17 |

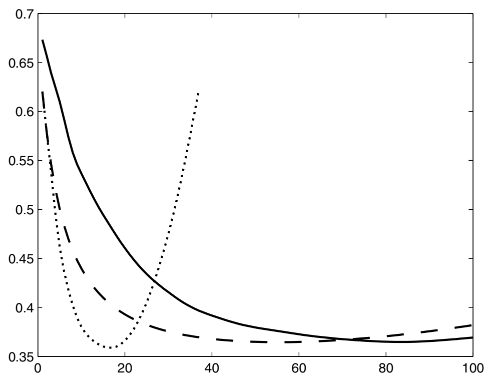


Table 6.16: Convergence history of LR with automatic acceleration (solid line), $Z$-LR without any acceleration (dashed line), $Z$-LR with automatic acceleration (dotted line).

LR method, at our knowledge, the most popular acceleration technique is that introduced in 1997 by Biggs and Andrews [16], called *automatic acceleration* by the authors. It exploits a vector extrapolation to determine the point on which the LR iteration has to be applied, thus it introduces a little computational overhead at each iteration of the LR method (for details see Chapter 7). Recalling the Satellite problem of Figure 6.4, created $Z$ by applying three times the coarsening technique of §6.2, Table 6.16 shows that in this case $Z$ variant overcomes automatic acceleration. Nevertheless, a more important fact is that by using automatic acceleration and $Z$ variant together we get an acceleration which is more powerful than both ones (see third row and dotted line). In conclusion, beyond the intrinsic strength of $Z$ approach, it can be naturally combined with other acceleration techniques. Future developments will be focused in these directions. For the moment, in the next chapters we will leave the preconditioning and reblurring field in order to analyse thoroughly acceleration techniques.

# Chapter 7

# $\nu$ acceleration

In literature there are many papers, more or less recent, that deal with acceleration techniques for LR method or analogous ones [16, 88, 1, 72, 76, 15, 85, 111, 20, 11]. At this time, to the best of our knowledge, the most popular is described in [16]. It is a form of vector extrapolation that predicts subsequent points based on previous points. In detail:

$$\begin{aligned} y_k &= x_k + \alpha_k(x_k - x_{k-1}), \\ \alpha_k &= \frac{(g_{k-1})^T g_{k-2}}{(g_{k-2})^T g_{k-2}}, \\ g_{k-1} &= x_k - y_{k-1}, \\ g_{k-2} &= x_{k-1} - y_{k-2}, \\ x_{k+1} &= \text{it.method}(y_k), \end{aligned}$$

where $\alpha_1 = 0$, $\alpha_2 = 0$, $0 \leq \alpha_k \leq 1$, $\forall k$. It is worth to note that it.method indicates a very general iterative method. In fact, as written in [16], "acceleration can be applied when the restoration changes slowly between each iteration and also if the algorithm is insensitive to the changes introduced". These are both properties of several iterative methods, for instance LR and Landweber method. The extrapolation, consisting in a shift along the direction given by the difference between the current iteration and the previous iteration (see Figure 7.1, taken from [16]), introduces a little computational overhead. The cost per iteration of the method is only slightly larger than it.method without acceleration. The goodness of this technique — called *automatic acceleration* by the authors, since it is exactly automatic, i.e. there is no parameter that needs to be set — is known. Indeed it is included in the LR implementation of Image Processing MatLab toolbox (see the function 'deconvlucy').

In this chapter we present a "new" acceleration, which actually is not

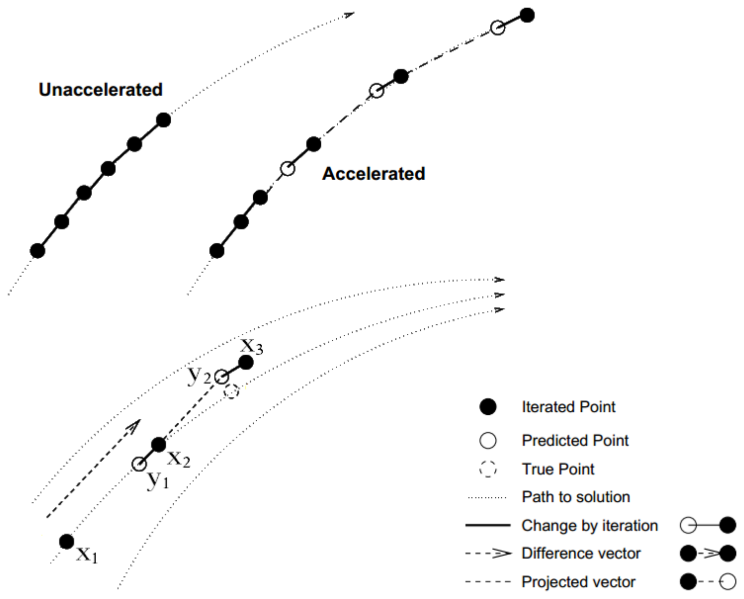


Figure 7.1: Conceptual difference between the unaccelerated and accelerated method (up) and vector diagram illustrating the acceleration method in geometric terms (down).

properly new, since it exists in another context. We are talking about $\nu$-method [22, 66], which is a semi-iterative method that speed up the classical Landweber method. The so-called $\nu$-method is defined as follows

$$x_k = \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k A^H(b - Ax_{k-1}), \tag{7.1}$$

where the coefficients $\mu_k$ and $\omega_k$ are given by

$$\mu_k = 1 + \frac{(k-1)(2k-3)(2k+2\nu-1)}{(k+2\nu-1)(2k+4\nu-1)(2k+2\nu-3)}, \tag{7.2}$$

$$\omega_k = \frac{4(2k+2\nu-1)(k+\nu-1)}{(k+2\nu-1)(2k+4\nu-1)}, \tag{7.3}$$

for $k > 1$, and with $\mu_1 = 1$, $\omega_1 = 1$. We notice that $\mu_k$ and $\omega_k$ do not depend on $A$, $b$, $x_k$, related to the particular restoration problem, but they depend only on the choice of $\nu$ and on the step $k$, so they are, in this sense, quite general. Figure 7.2 shows the trend of $\mu_k$ and of $\omega_k$, which grow rapidly in the first steps and then approach their respective limit value. In particular

$$\lim_{k\to\infty} \mu_k = 2\ , \quad \lim_{k\to\infty} \omega_k = 4\ , \quad \lim_{k\to\infty} \frac{\omega_k}{\mu_k} = 2\ ,$$

independently from the value of the acceleration parameter $\nu$. We remark that the ratio $\frac{\omega_k}{\mu_k}$ converges to its limit value more quickly both than $\mu_k$ and $\omega_k$.

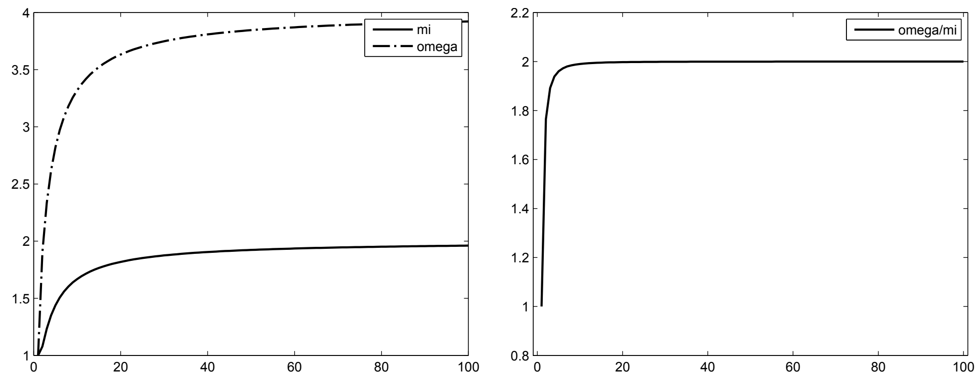


Figure 7.2: Trend, for $\nu = 1$, of $\mu_k$, $\omega_k$ and of $\omega_k/\mu_k$.

We notice that $\nu$-method is based on the fact that the $k$-th iteration of Landweber method can be written as $x_{k-1} + \text{update}_{k-1}$. Therefore, to use the same idea — from now on we call it $\nu$ *acceleration* — for LR, we rewrite (6.5) in this way

$$x_k = x_{k-1} + \left[ x_{k-1} \cdot A^H \left( \frac{b}{Ax_{k-1}} \right) - x_{k-1} \right],$$

whence, by a formal usage of (7.1), we write

$$\begin{aligned} x_k &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[ x_{k-1} \cdot A^H \left( \frac{b}{Ax_{k-1}} \right) - x_{k-1} \right] \\ &= (\mu_k - \omega_k)x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[ x_{k-1} \cdot A^H \left( \frac{b}{Ax_{k-1}} \right) \right]. \end{aligned} \tag{7.4}$$

An analogous formula holds for ISRA (6.6)

$$x_k = (\mu_k - \omega_k)x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[ x_{k-1} \cdot \left( \frac{A^H b}{A^H A x_{k-1}} \right) \right]. \tag{7.5}$$

## 7.1 Projection

Statistical methods do not need the projection on the non-negative cone, because they act pixel by pixel without any subtraction, hence they naturally preserve non-negativity whenever the matrix $A$ and the initial guess $x_0$ are non-negative. This is not true any more for the accelerated versions (7.4) and (7.5), so that, for keeping non-negativity and obtaining again functioning methods, is of fundamental importance to apply the projection $\mathcal{P}$ on the non-negative cone (6.40) after any step of the iterative method, i.e. instead of $x_k$ we take $x_k := \mathcal{P}(x_k)$ as iteration $k$. In the end, (7.4) and (7.5) become respectively

$$\begin{aligned} x_k &= \mathcal{P}\left\{(\mu_k - \omega_k)x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[x_{k-1} \cdot A^H \left(\frac{b}{Ax_{k-1}}\right)\right]\right\}, \\ x_k &= \mathcal{P}\left\{(\mu_k - \omega_k)x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[x_{k-1} \cdot \left(\frac{A^H b}{A^H A x_{k-1}}\right)\right]\right\}. \end{aligned}$$

Moreover, inspired by what we have observed in §6.4.3, we propose a new projection strategy

$$\begin{aligned} x_k &= (\mu_k - \omega_k)x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[x_{k-1}^+ \cdot A^H \left(\frac{b}{Ax_{k-1}^+}\right)\right], \\ x_k &= (\mu_k - \omega_k)x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k \left[x_{k-1}^+ \cdot \left(\frac{A^H b}{A^H A x_{k-1}^+}\right)\right], \end{aligned}$$

where $x_{k-1}^+ := \mathcal{P}(x_{k-1})$, which consists to do an iteration step with both projected and non-projected $x_{k-1}$ and finally to pick $x_k^+$ as computed solution. We remark that a similar idea can be used also for automatic acceleration, since the standard method provides $y_k := \mathcal{P}(y_k)$ and $x_{k+1} =$ it.method$(y_k)$, then we can introduce the next alteration, that is $y_k^+ = \mathcal{P}(y_k)$ and $x_{k+1} =$ it.method$(y_k^+)$; this means that acceleration algorithm works with non-projected $y_k$, while projection occurs only when we have to move from acceleration routine to iterative method. We will indicate these two alternative types of projection for $\nu$ acceleration and automatic acceleration by the symbol $\mathcal{Q}$, while for the standard projection we will use the symbol $\mathcal{P}$.

## 7.2 Connection between Landweber method and statistical methods

To justify the proposed approach, following the lines drawn in §6.1.2, we show the link between Landweber method (5.22) and LR (6.5), ISRA (6.6), from which follows the link between $\nu$-method (7.1) and $\nu$-accelerated LR (7.4), ISRA (7.5). LR can be approximated by the following formula

$$x_k = x_{k-1} + x_{k-1} \cdot A^H \left((b - Ax_{k-1}) \cdot \frac{1}{Ax_{k-1}}\right). \tag{7.6}$$

The difference between (7.6) and (6.5) is $x_{k-1} - x_{k-1} \cdot A^H \left( (Ax_{k-1}) \cdot \frac{1}{Ax_{k-1}} \right)$. If we define

$$[\mathcal{K}(y)]_j = \begin{cases} 1 & \text{if } y_j \neq 0, \\ 0 & \text{if } y_j = 0, \end{cases}$$

it becomes $x_{k-1} - x_{k-1} \cdot A^H \mathcal{K}(Ax_{k-1})$. Clearly it is zero when $[Ax_{k-1}]_j \neq 0$, $\forall j$, in fact under this hypothesis, denoting by $e$ the all-ones vector, we obtain

$$x_{k-1} - x_{k-1} \cdot A^H e = x_{k-1} - x_{k-1} \cdot e = 0. \tag{7.7}$$

We notice that (7.7) holds for periodic, reflective and anti-reflective BCs, while it does not hold for zero BCs, for which $A^H e \neq e$. When the stated hypothesis falls down, for example in the case of images with black background — i.e. images with many zeros — we have to consider the next result.

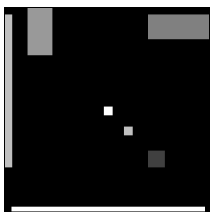
**Proposition 7.1** *If BCs are periodic, then* $x_{k-1} \cdot A^H \mathcal{K}(Ax_{k-1}) = x_{k-1}$.

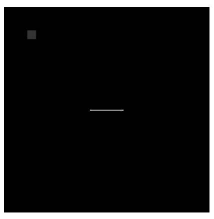
**Proof.** $\mathcal{K}(Ax_{k-1})$ is equal to 1 in some areas, whose position and extent depend on $x_{k-1}$ and on $A$, while is equal to 0 elsewhere. Since BCs are periodic, by direct calculation we get

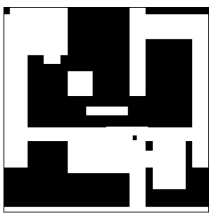
$$\left[A^H \mathcal{K}(Ax_{k-1})\right]_j = \begin{cases} 1 & \text{if } [x_{k-1}]_j \neq 0, \\ v_j & \text{if } [x_{k-1}]_j = 0, \end{cases} \tag{7.8}$$

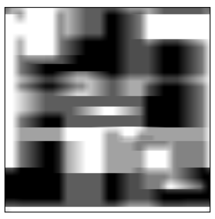
where $v_j$ is a certain value, that do not interest us. The thesis is a direct consequence of (7.8), so the proof is concluded. ■

**Example 7.2** *Some pictures that illustrate Proposition 7.1.*


$x_{k-1}$


*PSF*


$\mathcal{K}(Ax_{k-1})$


$A^H \mathcal{K}(Ax_{k-1})$

*We remind that white color corresponds to one, while black color corresponds to zero. By comparing the figures relative to* $x_{k-1}$ *and to* $A^H \mathcal{K}(Ax_{k-1})$ *with the formula (7.8), we can really see that* $A^H \mathcal{K}(Ax_{k-1})$ *has value equal to one (white) where* $x_{k-1}$ *is higher than zero (non-black). Therefore* $x_{k-1} \cdot A^H \mathcal{K}(Ax_{k-1}) = x_{k-1}$.

Therefore, from (7.4) and (7.6), for LR with $\nu$ acceleration we have

$$\begin{aligned} x_k &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \\ &\quad +\omega_k \left[ x_{k-1} + x_{k-1} \cdot A^H \left( (b - Ax_{k-1}) \cdot \frac{1}{Ax_{k-1}} \right) - x_{k-1} \right] \\ &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k y_k \cdot A^H((b - Ax_{k-1}) \cdot z_k), \end{aligned} \tag{7.9}$$

where $y_k = x_{k-1}$ and $z_k = \frac{1}{Ax_{k-1}}$.

Similarly for ISRA the following formula holds

$$x_k = x_{k-1} + \left( \frac{x_{k-1}}{A^H A x_{k-1}} \right) \cdot A^H(b - Ax_{k-1}), \tag{7.10}$$

so, from (7.5) and (7.10), for ISRA with $\nu$ acceleration we can write

$$\begin{aligned} x_k &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \\ &\quad +\omega_k \left[ x_{k-1} + \left( \frac{x_{k-1}}{A^H A x_{k-1}} \right) \cdot A^H(b - Ax_{k-1}) - x_{k-1} \right] \\ &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k y_k \cdot A^H(b - Ax_{k-1}). \end{aligned} \tag{7.11}$$

where $y_k = \frac{x_{k-1}}{A^H A x_{k-1}}$.

In other words, in the classical $\nu$-method $y_k$ and $z_k$ are always equal to $e$, while here they can be viewed as dynamic descent parameters, point by point. From an intuitive point of view, $y_k$ and $z_k$, as well as the ratios in (6.5) and in (6.6), vary according to the more or less closeness of $x_{k-1}$ to the sought solution, and since it is unknown, this is made by comparing $Ax_{k-1}$ with $b$. For each pixel, if $Ax_{k-1}$ is smaller than $b$, than the action of parameters will increase the value of that point, on the contrary if $Ax_{k-1}$ is higher than $b$, that point will be decreased by the action of parameters.

## 7.3 About convergence

First of all, we highlight that, as far as we know, no convergence proof of automatic acceleration is available. For $\nu$-accelerated ISRA, from (7.11) we have

$$\begin{aligned} x_k &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k A^H(b - Ax_{k-1}) + \\ &\quad +\omega_k (y_k - e) \cdot A^H(b - Ax_{k-1}) \\ &= x_k^{\nu-\text{method}} + \omega_k \left[ x_{k-1} \left( \frac{A^H b}{A^H A x_{k-1}} \right) - (x_{k-1} + A^H(b - Ax_{k-1})) \right] \\ &= x_k^{\nu-\text{method}} + \omega_k (x_k^{\text{ISRA}} - x_k^{\text{Landweber}}) \end{aligned} \tag{7.12}$$

and from (7.9) and by Proposition 7.1, for $\nu$-accelerated LR we have

$$\begin{aligned} x_k &= \mu_k x_{k-1} + (1-\mu_k)x_{k-2} + \omega_k A^H(b - Ax_{k-1}) + \\ &\quad +\omega_k \left[ y_k \cdot A^H((b - Ax_{k-1}) \cdot z_k) - A^H(b - Ax_{k-1}) \right] \\ &= x_k^{\nu-\text{method}} + \omega_k \left[ x_{k-1} \cdot A^H \left( \frac{b}{Ax_{k-1}} \right) - (x_{k-1} + A^H(b - Ax_{k-1})) \right] \\ &= x_k^{\nu-\text{method}} + \omega_k (x_k^{\text{LR}} - x_k^{\text{Landweber}}), \end{aligned} \tag{7.13}$$

where $x_k^{\text{it.method}}$ indicates the approximated solution that one gets doing one step of it.method, having $x_{k-1}$ as "initial guess". Therefore, even if these formulas do not represent a formal proof of convergence, we can say that LR (resp. ISRA) with $\nu$ acceleration in a certain sense inherits convergence properties from $\nu$-method (see [66]), Landweber method and LR (resp. ISRA). In fact, given $x_{k-1}$, all these methods make an iteration in order to approach the sought solution, so $\nu$-accelerated LR (resp. ISRA), which is a linear combination of those iterations, should have similar convergence properties. We stress that a rigorous proof of this fact cannot be obtained by using the same techniques of [66]. More in general, such a proof (if it exists) is not straightforward, since non-negative projection is an indispensable ingredient of $\nu$ acceleration and in literature, as far as we know, there is absence of effective mathematical tools for investigating the convergence of projected methods.

## 7.4 Computational cost

We underline that the cost per iteration of an accelerated method is only slightly larger than the same iterative method without acceleration. This is true for automatic acceleration, as we said at the beginning of this chapter, and also for $\nu$ acceleration. In fact the only additional costs are due to the computation of $\mu_k$ and $\omega_k$, that can be done with few operations by their formulas, and to the sum of the three vectors $(\mu_k - \omega_k)x_{k-1}$, $(1-\mu_k)x_{k-2}$ and $\omega_k \left[ x_{k-1} \cdot \left( \frac{A^H b}{A^H A x_{k-1}} \right) \right]$ (or $\omega_k \left[ x_{k-1} \cdot A^H \left( \frac{b}{Ax_{k-1}} \right) \right]$) which compose the accelerated iteration.

## 7.5 Computational results

To test the algorithms we have used images of $256 \times 256$ pixels and we have considered problems with periodic BCs. Several problems — having different true images, PSFs, level of noise — have been tested. Here we report an example — the Satellite restoration problem of Figure 6.4 — which is representative of the general behaviour of the methods and of the relations among them.

To estimate the level of acceleration gained by methods that employ an acceleration technique, we compare their performances (Table 7.2 and Table 7.3) with those of classical methods (Table 7.1) by calculating the Acceleration Factor (AF). AF is defined as the ratio between the number of iterations that are necessary to the original method to get a reconstructed image with some $\text{RRE}_{\text{AF}}$ (see Relative Restoration Error (RRE) defined in (5.25)) and the number of iterations that are necessary to the accelerated method to do the same thing. In particular, if we call $\text{RRE}_1$ the best RRE related to the first method and $\text{RRE}_2$ the best RRE related to the second one, we pick $\text{RRE}_{\text{AF}} = \max(\text{RRE}_1, \text{RRE}_2)$ to compute AF. We report in Figure 7.3 an example of this. We remind that the symbol $\mathcal{P}$ is related to the standard projection on the non-negative cone, while $\mathcal{Q}$ is related to the alternative type of projection described in §7.1. We point out that ISRA with automatic acceleration and $\mathcal{Q}$ projection has errors mainly placed around the shape of the satellite, while ISRA has errors mainly placed along the edges of the figure. This is the reason why the former restoration seems to have more inner details and the latter appears more clear in the outer part. We make a list of the different methods and their best RRE.

| | RRE | IT |
|---|---|---|
| LR | 0.3484 | 2128 |
| ISRA | 0.3451 | 1926 |

Table 7.1: LR and ISRA.

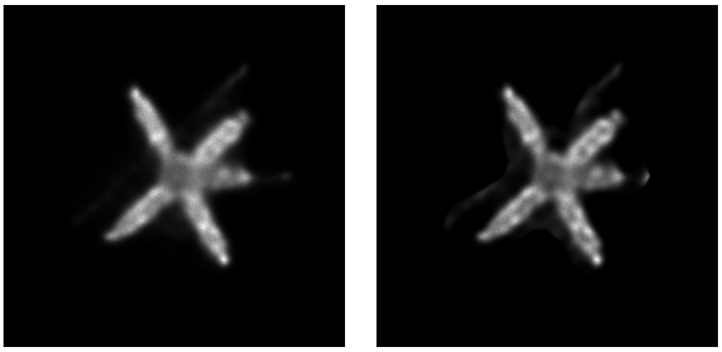


Figure 7.3: ISRA (RRE 0.3560, IT 912), ISRA with automatic acceleration and $\mathcal{Q}$ projection (RRE 0.3560, IT 109).

Recalling what we said at the beginning of this chapter, Table 7.2 and Table 7.3 confirm the effectiveness of $\nu$ acceleration and tell us that automatic acceleration is dethroned by it. $\nu$ acceleration overcomes automatic acceleration both in terms of AFs gained (for all three $\nu$ values) and of restorations performed (for $\nu$ equal to 1 and 2). Really for $\nu = 2$ it reaches an RRE that is even lower than one gained by classical LR or ISRA (Table 7.1). If we compare these two acceleration techniques looking at charts of Figure 7.4, where $\nu$ is taken equal to 1, we see that they have similar behaviour with regard to convergence speed; the huge difference of AF values is mainly due to the fact that $\nu$ acceleration reaches a lower RRE.

| | | | | | $\nu$ acc | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | aut acc | | | $\nu = 0.7$ | | | $\nu = 1$ | | | $\nu = 2$ | | |
| | | RRE | IT | AF | RRE | IT | AF | RRE | IT | AF | RRE | IT | AF |
| LR | $\mathcal{P}$ | 0.3646 | 83 | 10.1 | 0.3650 | 51 | 16.4 | 0.3526 | 69 | 20.4 | 0.3472 | 95 | 24.7 |
| | $\mathcal{Q}$ | 0.3555 | 123 | 9.9 | 0.3601 | 55 | 18.2 | 0.3478 | 70 | 32.2 | 0.3471 | 95 | 25 |

Table 7.2: LR with automatic acceleration and $\nu$ acceleration.

| | | | | | $\nu$ acc | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | aut acc | | | $\nu = 0.7$ | | | $\nu = 1$ | | | $\nu = 2$ | | |
| | | RRE | IT | AF | RRE | IT | AF | RRE | IT | AF | RRE | IT | AF |
| ISRA | $\mathcal{P}$ | 0.3695 | 75 | 7.5 | 0.3840 | 24 | 15 | 0.3484 | 62 | 21.1 | 0.3443 | 89 | 23.8 |
| | $\mathcal{Q}$ | 0.3560 | 109 | 8.4 | 0.3539 | 47 | 21.1 | 0.3458 | 65 | 24.9 | 0.3442 | 89 | 23.8 |

Table 7.3: ISRA with automatic acceleration and $\nu$ acceleration.

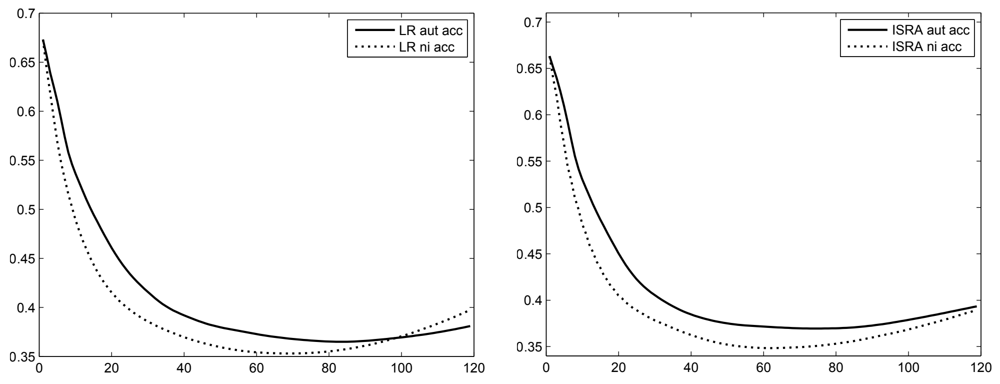


Figure 7.4: LR and ISRA with automatic acceleration and $\nu$ acceleration.

Modified projection $\mathcal{Q}$ always allows accelerated methods to reach a little better reconstructed image. Nevertheless, in the case of $\nu$ acceleration, as shown by Figure 7.5, in which we have plotted the non-zeros pattern of the computed approximated solution, using traditional projection $\mathcal{P}$ we get

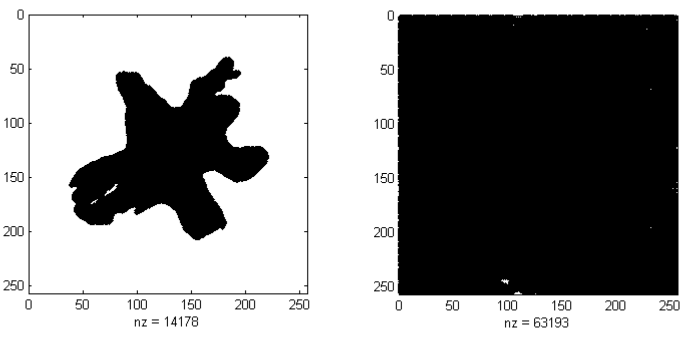


Figure 7.5: $\nu$ acceleration with $\mathcal{P}$ projection and $\mathcal{Q}$ projection.

| | RRE | IT | AF |
|---|---|---|---|
| Landweber | 0.3278 | 5358 | |
| Landweber aut acc | 0.3284 | 179 | 24.4 |
| Landweber $\nu$ acc | 0.3280 | 107 | 44.2 |

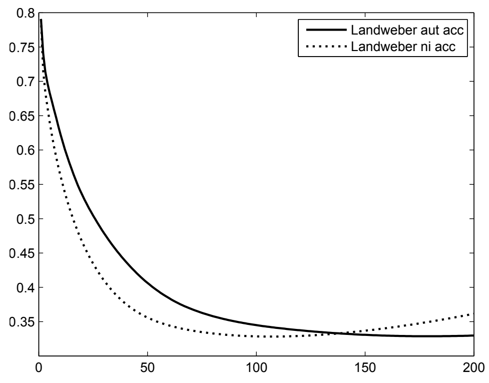


Table 7.4: Landweber method with automatic acceleration and $\nu$ acceleration.

non-zeros elements only in proximity to the true object, while using our proposal we obtain an image that has non-zero values almost everywhere. For automatic acceleration instead, the non-zeros pattern related to each projection is similar, and it is near the true object.

Finally it is worth to note that it is also possible to use automatic acceleration for Landweber method. For the Satellite problem we obtain results reported in Table 7.4, which reaffirm the goodness of $\nu$ acceleration ($\nu$ is chosen equal to 1). The only drawback of this strategy, that on the contrary is totally absent in automatic acceleration, consists in the choice of $\nu$. Even though here it does not appear to be a big problem, if we use acceleration for an iterative method that already employs acceleration techniques, like those based on preconditioning (see Chapter 5 and Chapter 6) or on Banach spaces (see Chapter 8), we might run into a different scenario.

# Chapter 8

# Banach spaces and Meinel acceleration

We highlight that the contents of this chapter are ideas in the early stages, which seem to be nice and effective, but which need to be analysed and studied in depth from the theoretical point of view. We want to investigate the acceleration of LR method gained by the introduction of an exponent, proposed in 1986 by Meinel [88] and recently reconsidered in [111]. Using this technique, LR method becomes

$$x_k = x_{k-1} \cdot \left[ A^H \left( \frac{b}{Ax_{k-1}} \right) \right]^p, \tag{8.1}$$

while ISRA becomes

$$x_k = x_{k-1} \cdot \left[ \frac{A^H b}{A^H A x_{k-1}} \right]^p, \tag{8.2}$$

where the selection of the exponent $p$ (usually $1 < p < 3$) has to be done manually by hit and trial. This strategy — that from now on we will call *Meinel acceleration* — has two fundamental drawbacks. The first is that we obtain an improvement in the convergence rate at the expense of a lack of stability in the convergence. To remedy this, in [111] the authors proposed a dynamic choice of the parameter $p$, which consequently becomes $p_k$, and they called Adaptively Accelerated Lucy-Richardson (AALR) the resulting method. This accelerated LR method emphasizes speed at the beginning stages of iterations by forcing $p_k$ around 3. As steps increase, the algorithm forces $p_k$ toward the value of 1, which leads to stability of iteration. Moreover, in order to avoid instability at the start of iteration, $p_k$ is fixed equal to 1 for the first two steps. The second drawback is that this technique, compared for instance with automatic acceleration described in Chapter 7, reduce the number of steps in a less important way; generally it cannot even halve it.

## 8.1 Meinel acceleration

We start noting that the basic schemes (6.5) and (6.6) can be rewritten in this general form

$$x_k = x_{k-1} \cdot \alpha_k \tag{8.3}$$

and the same can be done for the Meinel variants (8.1) and (8.2)

$$x_k^{\text{Meinel}} = x_{k-1}^{\text{Meinel}} \cdot (\beta_k)^p \, . \tag{8.4}$$

We remark that, even though $\alpha_k$ and $\beta_k$ are defined by the same formula — $A^H\left(\frac{b}{Ax_{k-1}}\right)$ in the case of LR method and $\frac{A^H b}{A^H A x_{k-1}}$ in the case of ISRA — we have to use different symbols to denote them, since they depend respectively on $x_{k-1}$ and on $x_{k-1}^{\text{Meinel}}$, which are different. In the following we will consider $p$ as an integer, but the same argument can be used for real $p$, and we set $q = p^{-1}$. Moreover we will use $\text{RRE}(x) = \|x - \bar{x}\|_2 / \|\bar{x}\|_2$ to measure the goodness of restored images and the acronym BR to denote the best restoration gained by the algorithm we are examining.

Recalling the semiconvergence behaviour, suppose that the algorithm (8.3) converges to its BR in $n$ steps, that is

$$\begin{aligned} \text{RRE}(x_0) \geq \text{RRE}(x_1) \geq \ldots &\geq \text{RRE}(x_n) \\ \text{RRE}(x_k) &> \text{RRE}(x_n), \forall k > n \end{aligned}$$

and the algorithm (8.4) does the same in $m$ steps. Then we can express the BRs performed by (8.3) and (8.4) in this manner

$$\begin{aligned} x_n &= b \cdot \alpha_1 \cdot \ldots \cdot \alpha_n, \\ x_m^{\text{Meinel}} &= b \cdot (\beta_1)^p \cdot \ldots \cdot (\beta_m)^p \, , \end{aligned}$$

since here the initial guess is always assumed to be the blurred and noisy data $b$, i.e. $x_0 = x_0^{\text{Meinel}} = b$. We know that algorithm (8.3) is in general very slow to converge, so chosen $p$ and $n'$ in a suitable way, we have that, $\forall k > n'$, $\alpha_{(k-1)p+1} \approx \ldots \approx \alpha_{kp}$. Thus the idea that lies behind Meinel acceleration is that, for any $k$, the product $\alpha_{(k-1)p+1} \cdot \ldots \cdot \alpha_{kp}$ can be approximated by $(\beta_k)^p$. If this is true, we get

$$x_n = b \cdot \alpha_1 \cdot \ldots \cdot \alpha_n \approx b \cdot (\beta_1)^p \cdot \ldots \cdot \left(\beta_{\lfloor qn \rfloor}\right)^p = x_{\lfloor qn \rfloor}^{\text{Meinel}},$$

which means that the number of steps of (8.4) is equal to $\lfloor qn \rfloor$ and its BR is close to that of (8.3). We introduce this concept formally through the next two definitions.

**Definition 8.1** *Fixed $p$, an accelerated algorithm is called* stable *if it converges to its BR in $\lfloor qn \rfloor$ steps, where $q = p^{-1}$.*

**Definition 8.2** *An accelerated algorithm is called $\epsilon$-*accurate *if*

$$\mathrm{RRE}(x_m^{\mathrm{Acc}}) - \mathrm{RRE}(x_n) < \epsilon,$$

*where $x_n$ and $x_m^{\mathrm{Acc}}$ are the iterations related to the BR of the basic algorithm and of the accelerated algorithm respectively.*

In plain words, Definition 8.1 tell us that algorithm maintains the same stability properties of (8.3), whereas Definition 8.2 tell us that the error introduced by the approximation done by $\beta_k$ is, in some sense, negligible. Therefore if (8.4) is stable and accurate, we get a restoration of the same quality of (8.3) in a lower number of steps. The sore point is that, as we said at the start of the chapter, while for $p$ near to 1 Definitions 8.1 and 8.2 hold, moving away from 1 this becomes soon untrue.

**Remark 8.3** *Numerical tests show that the two definitions are strongly linked, i.e. if we want that the algorithm is $\epsilon$-accurate — with small $\epsilon$, in particular we pick $\epsilon = 10^{-4}$ — then it must be stable.*

## 8.2 Acceleration by map

Here we present a new acceleration scheme — which is a further variant of Meinel one — based on a map, so we call it *map acceleration*. Similarly to (8.3) and (8.4), chosen a suitable value $p > 0$, we define the iterations as follows

$$\begin{aligned} \check{x}_k^{\mathrm{Map}} &= \check{x}_{k-1}^{\mathrm{Map}} \cdot \gamma_k, \\ x_k^{\mathrm{Map}} &= \mathbf{J}_p(\check{x}_k^{\mathrm{Map}}), \end{aligned} \tag{8.5}$$

where $\check{x}_0^{\mathrm{Map}} = \mathbf{J}_q(x_0^{\mathrm{Map}})$, $q = p^{-1}$ and in general $\mathbf{J}_s$ is the (non-linear) map

$$\mathbf{J}_s(x) = \|x\|_{s+1}^{1-s} \, x^s, \tag{8.6}$$

which has to be read componentwise. As before, $\gamma_k$ denotes $A^H\left(\frac{b}{A\check{x}_{k-1}}\right)$ in the case of LR method and $\frac{A^H b}{A^H A \check{x}_{k-1}}$ in the case of ISRA. Even if at the first look it seems to be very dissimilar to Meinel acceleration, it can be rewritten in this way

$$x_k^{\mathrm{Map}} = x_{k-1}^{\mathrm{Map}} \cdot c_k (\gamma_k)^p, \tag{8.7}$$

where the constants $c_k$ are given by this formula

$$c_k = \left[ \frac{\left\| x_{k-1}^{\mathrm{Map}} \right\|_{q+1}}{\left\| \check{x}_k^{\mathrm{Map}} \right\|_{p+1}} \right]^{p-1} . \tag{8.8}$$

In fact, since $\check{x}_{k-1}^{\mathrm{Map}} = \mathbf{J}_q(x_{k-1}^{\mathrm{Map}})$ and $(1-q)p = p-1$, we have

$$\begin{aligned} \check{x}_k^{\mathrm{Map}} &= \check{x}_{k-1}^{\mathrm{Map}} \cdot \gamma_k = \mathbf{J}_q(x_{k-1}^{\mathrm{Map}}) \cdot \gamma_k = \left\| x_{k-1}^{\mathrm{Map}} \right\|_{q+1}^{1-q} (x_{k-1}^{\mathrm{Map}})^q \cdot \gamma_k, \\ x_k^{\mathrm{Map}} &= \left\| \check{x}_k^{\mathrm{Map}} \right\|_{p+1}^{1-p} \left\| x_{k-1}^{\mathrm{Map}} \right\|_{q+1}^{(1-q)p} x_{k-1}^{\mathrm{Map}} \cdot (\gamma_k)^p = \left[ \frac{\left\| x_{k-1}^{\mathrm{Map}} \right\|_{q+1}}{\left\| \check{x}_k^{\mathrm{Map}} \right\|_{p+1}} \right]^{p-1} x_{k-1}^{\mathrm{Map}} \cdot (\gamma_k)^p. \end{aligned}$$

Comparing (8.7) with (8.4), we notice that the only difference (very important one) is that while in the former each $c_k$ is suitably computed by (8.8), in the latter $c_k$ are all equal to 1. As before, it is possible to express the restoration performed by (8.7) after $m$ steps in this manner

$$x_m^{\mathrm{Map}} = b \cdot c_1 \, (\gamma_1)^p \cdot \ldots \cdot c_m \, (\gamma_m)^p \, ,$$

since $x_0^{\mathrm{Map}} = b$.

The key point is that, differently from (8.4), (8.7) has a wider range of $p$ values in which the properties of Definitions 8.1 and 8.2 hold, as we will appreciate in §8.4. In particular we think that it can be useful to report here the chart related to $c_k$ values, for Meinel acceleration and map one, both having $q = 1/3$, that is $p = 3$, in the first one hundred iterations of a test that involves LR (see Test 1 in §8.4).

We can notice that $c_k \to 1$ as iteration number grows, becoming very near to Meinel values, even though the most important work is done by $c_k$ in the initial steps. Furthermore from this we can affirm that AALR approach, which emphasizes speed at the beginning stages, setting $p$ near to 3, and then, as iterations increase, forcing it to 1 for stability, is not the best possible. So in our shy adaptive attempts we will do the opposite: $p$, starting from 1, will be chosen according to some increasing sequence. Note that, while usually $c_k$ curve is smooth, in Figure 8.1 there is a jump. This can be explained by the fact that for $q = 1/3$ instability starts to reveal itself and then, for lower values of $q$, its influence becomes more dramatic, as we will see in §8.4.

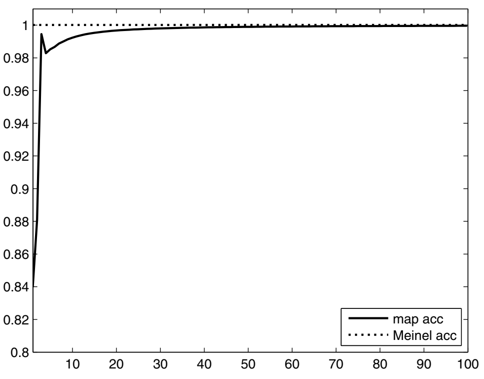


Figure 8.1: (Test 1) $c_k$ for Meinel acceleration and map acceleration.

## 8.3 Landweber method in Banach spaces

The technique described in §8.2 is inspired by duality map — whose role is to associate in a proper way the elements in a Banach space $B$ to the elements in its dual space $B^*$ — used for Landweber method in Banach spaces [104]. In this field, to develop all the theory, which requires a solid background in functional analysis, one considers a continuous linear operator

$$A : X \to Y \tag{8.9}$$

associated to the linear problem (6.1). In the classical case, $X$ and $Y$ are Hilbert spaces and, if we denote by $^*$ dual spaces and dual operator, we have

$$A^* : Y^* \to X^* \tag{8.10}$$

where $X = X^*$ and $Y = Y^*$; the duality map is identity.

Now, if we move to Banach spaces, for $p > 0$, we consider $X = Y = L^{q+1}$ so that $X^* = Y^* = L^{p+1}$, and we get Landweber method as follows

$$\begin{array}{ll} \check{x}_0 = \mathbf{J}_q(x_0), & \mathbf{J}_q : X \to X^*, \\ \check{x}_k = \check{x}_{k-1} + \tau A^* \mathbf{J}_q \left(b - Ax_{k-1}\right), & \mathbf{J}_q : Y \to Y^*, \\ x_k = \mathbf{J}_p(\check{x}_k), & \mathbf{J}_p : X^* \to X. \end{array} \tag{8.11}$$

where $\mathbf{J}$ is the same map introduced in (8.6), $\mathbf{J}_q : L^{q+1} \to L^{p+1}$ and $\mathbf{J}_p : L^{p+1} \to L^{q+1}$. We call (8.11) Landweber Banach method (LB), in opposition to classical Landweber method (5.22) in Hilbert spaces that we call Landweber Hilbert method (LH). It is known that often LB allows to gain restorations of higher quality than LH ones. Unfortunately, if we consider the version of LH and LB with the projection $\mathcal{P}$ on the non-negative cone (6.40), i.e. at each step $x_k := \mathcal{P}(x_k)$, this in general is not true any more.

In particular, we give this experimental conjecture (i.e. that is based on computational experience) about the incapacity of LB to get restored images that are better than LH ones.

**Conjecture 8.4** *Let $r_q$ the RRE of the best restoration made by Landweber Banach method (8.11) in $L^{q+1}$. If $a > b$ $(0 < a, b \leq 1)$, then $r_a < r_b$.*

The obvious consequence of this conjecture is that the best restoration is performed by LH ($r_1$). To realize the closeness of this statement to the truth, have a look at §8.4. It is worth to note that statistical methods naturally preserve non-negativity, because they act pixel by pixel without any subtraction, so they do not need any projection.

Deviating from the main furrow of this chapter, now we propose a modification of the strategy described in this section, in order to increase its effectiveness. Instead of the classical LB method (8.11), we consider the following modification

$$\begin{aligned} &\Delta_{k-1} = A^H\left(b - Ax_{k-1}\right), \\ &\check{x}_k = \mathbf{J}_q(x_{k-1}) + \tau \mathbf{J}_q\left(\Delta_{k-1}\right), \quad && \mathbf{J}_q : X \to X^*, \\ &x_k = \mathbf{J}_p(\check{x}_k), && \mathbf{J}_p : X^* \to X. \end{aligned} \tag{8.12}$$

Thus, in the first step of the algorithm we suppose that $X$ and $Y$ are both Hilbert spaces, so $A$ and $A^*$ can act directly, without needing any duality map. In that case we use the notation $A^H$ instead of $A^*$. Then we work with $X = L^{q+1}$ and its dual. We call (8.12) Modified Landweber Banach method (MLB). One possible way to obtain theoretical results about this method is to study the intersection of spaces $L^2$ and $L^{q+1}$, but for the moment we are not able to provide results in that direction.

One can think to further generalize this approach in the next fashion

$$\begin{aligned} &\Delta_{k-1} = \mathbf{J}_u\left[A^H \mathbf{J}_v\left(b - Ax_{k-1}\right)\right], \\ &\check{x}_k = \mathbf{J}_q(x_{k-1}) + \tau \mathbf{J}_q\left(\Delta_{k-1}\right), \\ &x_k = \mathbf{J}_p(\check{x}_k), \end{aligned} \tag{8.13}$$

where $v = u^{-1}$, $\mathbf{J}_v : L^{v+1} \to L^{u+1}$ and $\mathbf{J}_u : L^{u+1} \to L^{v+1}$. We notice that if we pick $v = q$ and $u = p$, as $\mathbf{J}_q(\mathbf{J}_p(x)) = x$, we obtain again LB (8.11) and if we pick $v = u = 2$, we obtain MLB (8.12). We call (8.13) Generalized Landweber Banach method (GLB). Finally we state the next experimental conjecture, which expresses the fact that the best RRE is no longer related to LH, as in Conjecture 8.4, but to MLB for a suitable value of $q < 1$.

**Conjecture 8.5** *Let $r_q$ the RRE of the best restoration made by Modified Landweber Banach method (8.12) in $L^{q+1}$. Let $r_i = \min_{0<j\leq 1} r_j$. Then $i < 1$.*

## 8.4 Computational results

To test the algorithms we have used images of $256 \times 256$ pixels and we have considered problems with periodic BCs. The set of data for Test 1 is Satellite of Figure 6.4. For Test 2 — that is Moon deblurring problem — we have generated $b$ by a motion blur, adding about 1.4% of white Gaussian noise (see Figure 8.2). We make a list of the different methods and their best RRE. To estimate the level of acceleration we use AF, defined in §7.5.

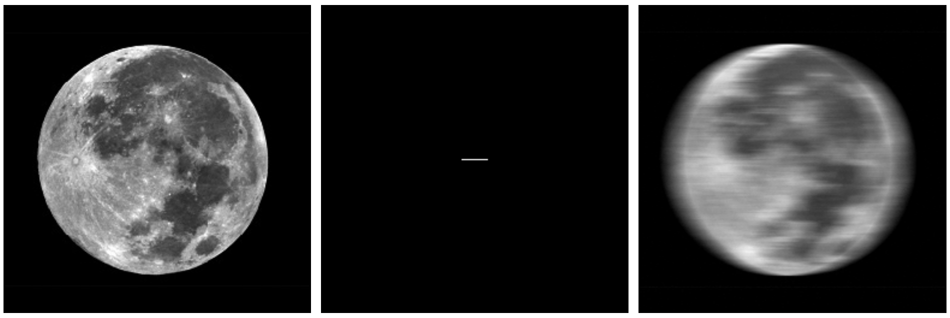


Figure 8.2: (Test 2) True image, PSF, blurred and noisy image.

We report the results relative to LR and ISRA with Meinel acceleration or with map acceleration proposed by us. The symbol '–' means that there is no acceleration. As we said, from Test 1 (Table 8.1 and Table 8.2) it clearly comes out that, while Meinel acceleration becomes instable very soon, map acceleration converges to a solution of the same quality of classical LR, gaining an AF that is around 3. In other words, the interval in which Definitions 8.1 and 8.2 hold is much wider for (8.7) than for (8.4). If we choose $q$ adaptively, in a way opposite to AALR, performances of Meinel acceleration get better. In fact we are able to gain results near to those of map acceleration for $q = 1/2$, namely IT 1073, RRE 0.3484, but nothing more. The same holds for ISRA, for which we have IT 989, RRE 0.3451. If we try to obtain an higher AF by this adaptive choice of $q$, we meet again instability. In Test 2 (Table 8.3 and Table 8.4) performances of map acceleration and Meinel one are very similar and both start to display stability problems when $q$ is equal to $1/2$ for LR and even before for ISRA. In this example it can be useful to use an adaptive strategy not only for Meinel acceleration, but also for map acceleration. In particular, choosing the increasing sequence in two different manners, for LR we have the results reported in Table 8.5.

As observed in Chapter 7, LH is linked with statistical methods. So we guess that it can exist a parallelism between the acceleration gained by the introduction of an exponent in that context and this approach based

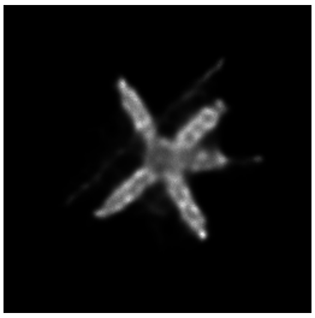


Figure 8.3: Best ISRA restoration (RRE 0.3484) for Test 1.

| | map acc | | | Meinel acc | | |
|---|---|---|---|---|---|---|
| | RRE | IT | AF | RRE | IT | AF |
| $q = 1$ | 0.3484 | 2128 | | 0.3484 | 2128 | |
| $q = 3/4$ | 0.3484 | 1596 | 4/3 | 0.3484 | 1596 | 4/3 |
| $q = 1/2$ | 0.3484 | 1064 | 2 | 0.5656 | 7 | 2.2 |
| $q = 1/3$ | 0.3484 | 709 | 3 | 0.6440 | 1 | 4 |
| $q = 1/4$ | 0.5043 | 10 | 4 | 0.6661 | 1 | 2 |
| $q = 1/5$ | 0.5720 | 2 | 7 | 0.7301 | 1 | − |

Table 8.1: LR for Test 1.

| | map acc | | | Meinel acc | | |
|---|---|---|---|---|---|---|
| | RRE | IT | AF | RRE | IT | AF |
| $q = 1$ | 0.3451 | 1926 | | 0.3451 | 1926 | |
| $q = 3/4$ | 0.3451 | 1445 | 4/3 | 0.3451 | 1445 | 4/3 |
| $q = 1/2$ | 0.3451 | 963 | 2 | 0.6458 | 1 | 2 |
| $q = 1/3$ | 0.3451 | 642 | 3 | 0.6831 | 1 | − |
| $q = 1/4$ | 0.4701 | 18 | 3.6 | 0.8114 | 1 | − |
| $q = 1/5$ | 0.5472 | 4 | 5 | 0.9766 | 2 | − |

Table 8.2: ISRA for Test 1.

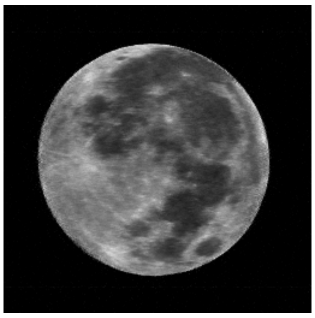


Figure 8.4: Best LR restoration (RRE 0.0974) for Test 2.

| | map acc | | | Meinel acc | | |
|---|---|---|---|---|---|---|
| | RRE | IT | AF | RRE | IT | AF |
| $q = 1$ | 0.0974 | 76 | | 0.0974 | 76 | |
| $q = 3/4$ | 0.0974 | 57 | 4/3 | 0.0974 | 57 | 4/3 |
| $q = 1/2$ | 0.0975 | 38 | 2 | 0.0976 | 38 | 2 |
| $q = 1/3$ | 0.1768 | 3 | 2 | 0.1867 | 2 | 2 |

Table 8.3: LR for Test 2.

| | map acc | | | Meinel acc | | |
|---|---|---|---|---|---|---|
| | RRE | IT | AF | RRE | IT | AF |
| $q = 1$ | 0.1090 | 178 | | 0.1090 | 178 | |
| $q = 3/4$ | 0.1090 | 133 | 4/3 | 0.1090 | 133 | 4/3 |
| $q = 1/2$ | 0.1144 | 51 | 2 | 0.1485 | 15 | 2 |
| $q = 1/3$ | 0.2185 | 2 | – | 0.2222 | 1 | 2 |

Table 8.4: ISRA for Test 2.

| map acc | | Meinel acc | |
|---|---|---|---|
| RRE | IT | RRE | IT |
| 0.0995 | 24 | 0.1006 | 22 |
| 0.1086 | 14 | 0.1093 | 13 |

Table 8.5: LR with two adaptive strategies for Test 2.

on Banach spaces. From LB row of Table 8.6 and Table 8.7, related to Test 1 and Test 2 respectively, we can notice that this Banach technique, differently from our proposal for statistical methods, has a very small range of $q$ values in which the iterative method gets a RRE equal to the one of LH, then the quality of the restoration gets gradually worse as $q$ approaches zero (see Conjecture 8.4). On the other hand, called $n$ the number of iterations of LH, that of LB is near to $qn$, when $q$ is very near to 1, then in Test 1 it gradually decreases as $q$ approaches zero (Figure 8.5) — therefore displaying a behaviour in line with Definition 8.1 given for statistical methods with Meinel acceleration and map acceleration — whereas in Test 2 it has a more irregular trend (Table 8.7). In Test 1 and Test 2 MLB seems to be preferable to LB both in terms of restorations performed and of acceleration factors gained. In particular in Test 2 LB is not able to get any acceleration, while MLB can do that (for $q = 0.5$). As we just said for Conjecture 8.4, we highlight that numerical results also confirm the Conjecture 8.5 stated in §8.3. Finally we say that, in the context of Landweber method, adaptive choice of $q$ seems to be able to bring only small improvements of performances, so we do not report any results in that direction.

In conclusion, the two drawbacks of Meinel acceleration, which are partially cured by the map acceleration proposed here, are instability and the

| | | $q = 1$ | $q = 0.9$ | $q = 0.75$ | $q = 0.5$ | $q = 0.3$ | $q = 0.2$ | $q = 0.1$ |
|---|---|---|---|---|---|---|---|---|
| **LB** | IT | 5358 | 4843 | 3984 | 2411 | 1202 | 680 | 1 |
| | RRE | 0.3278 | 0.3278 | 0.3283 | 0.3311 | 0.3361 | 0.3398 | 0.7731 |
| | AF | | 1.1 | 1.1 | 1.4 | 2 | 3.1 | – |
| **MLB** | IT | 5358 | 3999 | 2529 | 1006 | 600 | 585 | 5 |
| | RRE | 0.3278 | 0.3271 | 0.3271 | 0.3326 | 0.3374 | 0.3779 | 0.5877 |
| | AF | | 1.6 | 2.5 | 3 | 3.9 | 1.3 | 7.4 |

Table 8.6: LB and MLB for Test 1.

| | | $q = 1$ | $q = 0.9$ | $q = 0.75$ | $q = 0.5$ | $q = 0.3$ | $q = 0.2$ | $q = 0.1$ |
|---|---|---|---|---|---|---|---|---|
| **LB** | IT | 182 | 171 | 151 | 96 | 2 | 1 | 1 |
| | RRE | 0.1058 | 0.1068 | 0.1089 | 0.1125 | 0.2247 | 0.2383 | 0.2343 |
| | AF | | – | – | – | – | – | – |
| **MLB** | IT | 182 | 162 | 130 | 124 | 2 | 2 | 1 |
| | RRE | 0.1058 | 0.1067 | 0.1092 | 0.1034 | 0.2260 | 0.2549 | 0.2588 |
| | AF | | – | – | 2.1 | – | – | – |

Table 8.7: LB and MLB for Test 2.

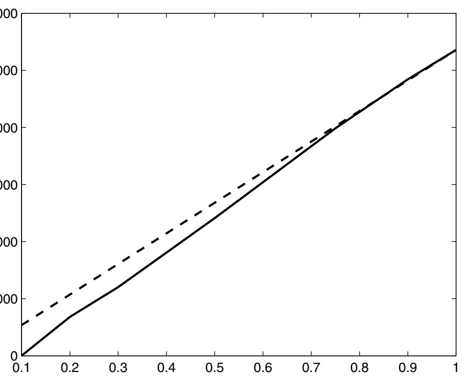


Figure 8.5: (Test 1) Iterations of LB (solid line), with $q$ on the $x$-axis, compared with $qn$ (dashed line), where $n$ is the number of iterations of LH.

fact that, compared with other acceleration techniques, it reduces the number of steps in a less important way; notice that these are also drawbacks of Banach spaces technique for Landweber method. Taken note of this, one could think in principle to seek other tools which can further improve the stability and the strength of that type of acceleration, or more realistically to set aside those techniques. Nevertheless they can be again useful if we make use of them together with other more efficient strategies, for instance preconditioning (see [26]). Future developments will be focused in this direction.

# The pocket

In this bitter ending, that is the black hole in which all the contents are sucked into, we summarize them before they sink into oblivion.

In Part I we have considered multi-iterative techniques of multigrid type for the numerical solution of large linear systems with (weighted) structure of graph Laplacian operators. We have combined efficient coarser-grid operators with iterative techniques used as smoothers, by showing that the most effective smoothers typically are Krylov type with subgraph-based preconditioners, while the projectors have to be designed for maintaining as much as possible the graph structure of the projected matrix at the inner levels. Some necessary and sufficient conditions have been proved. As a guiding principle, we have observed that coarse-grid operators that preserve the fine-level structure work satisfactorily, otherwise the performances are degraded. Interestingly enough, this framework is useful for explaining the reason why the classical projectors inherited from differential equations are good in the differential context and why they behave unsatisfactorily for unstructured graphs. Several numerical experiments have been conducted showing that our approach is effective in a uniform way, independently from the conditioning of the considered matrices, from the type of the problems, from the IP (Interior Point) step in which the considered matrices arise. We think that this robustness is the most relevant result obtained in our work. In fact, while very sophisticated PCG techniques exist that often work quite well, there are some cases where the convergence rate is slow. Along the lines of [82, 42, 43], the objective of our work has been to integrate "simple" tree-based PCG approaches with multigrid type algorithms: as already mentioned, the resulting methods combine a relatively simple implementation and robustness, in the sense of uniformly delivering good performances without the need of complex parameter tuning. Several research lines seems to be worth future investigation. From the theoretical viewpoint, it would be very interesting to obtain a rigorous characterization of the convergence speed of the proposed multigrid techniques with varying pre- and post-smoothers and choices of the projector. Regarding the latter, a better understanding of the effect of the projection step on the spectral properties of the matrices at lower level would be required to design more effective approaches to choosing $R$, possibly simultaneously taking into account a selection of an appropriate subgraph for preconditioning techniques. Another important challenge would be the extension of our proposal to a non-symmetric setting, as it occurs when dealing with the Google problem [86].

Part II has been devoted to several techniques to improve speed and stability of iterative methods in the context of image deblurring problem. In Chapter 5, inspired by the theoretical results on optimal preconditioning stated in [92] for the Reflective BCs, we have presented analogous results for Anti-Reflective BCs. In both cases the optimal preconditioner is the blurring matrix associated to the symmetrized PSF. We stress that our proof is based on a geometrical idea, which allows to greatly simplify the job. Moreover that idea is very powerful in its generality and it may be useful in the future to prove theoretical results for new BCs. Computational results have shown that the proposed preconditioning strategy is effective and it is able to give rise to a meaningful acceleration both for slightly and highly non-symmetric PSFs. On the other hand, symmetrization is efficient when we have a PSF that is near to be symmetric and it becomes more and more ineffective as the PSF departs from symmetry. In this case, other techniques, like those described in §6.6, which can manage directly non-symmetric structures, can gain better performances. In Chapter 6 we have presented iterative methods based on a variant of the normal equations approach, called $Z$ variant, and we have analysed its features and its similarities with classical regularizing preconditioning. We have proposed two strategies for defining $Z$: by coarsening of the PSF and by filtering of its eigenvalues. Since the latter approach requires the spectral decomposition of $Z$, this is chosen in the BCCB algebra. The blurring matrix $A$ has been first taken BCCB (periodic BCs) for a simple theoretical analysis. Afterwards, the use of zero, reflective, and anti-reflective BCs for $A$ has highlighted that our $Z$ variant provides accurate restorations with a computational cost lower than regularizing preconditioning and without requiring an accurate estimation of the threshold parameter. We remark that this good behaviour is opposed to the one of classical regularizing preconditioning, where an accurate estimation of such parameter is crucial for obtaining good restorations. As a natural continuation of the work done in the circulant environment, we have explored the potential of $Z$ variant idea by giving a general algorithm which allows to choose $Z$ as a blurring matrix with the same BCs (so with the same matrix structure) of the blurring operator $A$. Moreover the use of the $Z$ variant idea proposed here can be successfully employed for direct filtering methods, like Tikhonov. In Chapter 7 we have used $\nu$ acceleration, that is nothing but the application of $\nu$-method, conceived for speed up Landweber method, in the framework of statistical methods. Numerical results have highlighted the effectiveness of this strategy, which, compared with automatic acceleration, gives rise to higher acceleration factors and better restorations. In Chapter 8 we have introduced a map acceleration for statistical methods, inspired by theory on Landweber method in Banach spaces, and we have shown the link between it and Meinel acceleration. We have illustrated the relation between the number of steps performed by accelerated and non-accelerated LR method. Moreover we have experimentally observed that also Landweber method in Banach spaces follows a similar behaviour. We emphasize that the contents of Chapter 7 and Chapter 8 are ideas in the early stages, which work well from the computational point of view, but which need to be analysed and studied in depth from the theoretical one.

A future line of research may consist in analysing how to combine the different acceleration techniques inspected here in a suitable way. A further idea which goes in this direction and whose main aim is not to speed up the methods, but to improve the quality of the restoration, is the following *iterative denoising*. With these terms we mean that we want to solve the linear system

$$By = b', \tag{8.14}$$

where $b' = Bb = B\,(A\bar{x}) + B\eta$, to find an approximation of $\bar{b} = A\bar{x}$, the blurred image without noise. So for this problem the definition of RRE becomes $\left\|y - \bar{b}\right\|_{\mathcal{F}} / \left\|\bar{b}\right\|_{\mathcal{F}}$. There are several possible choices of $B$, for instance $B = A$. Once found $y$, instead of (6.1) the restoration problem becomes

$$Ax = y. \tag{8.15}$$

As a final greeting, we report an example relative to Satellite problem of Figure 6.4. For iterative denoising we make use of $\nu$-method (see Figure 8.6 and Figure 8.7) and then we employ $Z$-LR and $Z$-ISRA with $\nu$ acceleration to compute high quality restorations (see Table 8.8 and Figure 8.8).

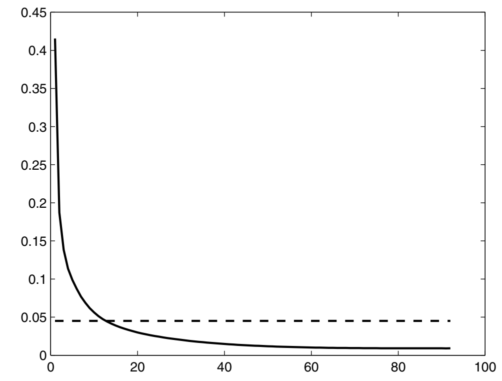


Figure 8.6: Original level of noise (dashed line) and iterative denoising made by $\nu$-method, with $\nu = 1.5$ (solid line).

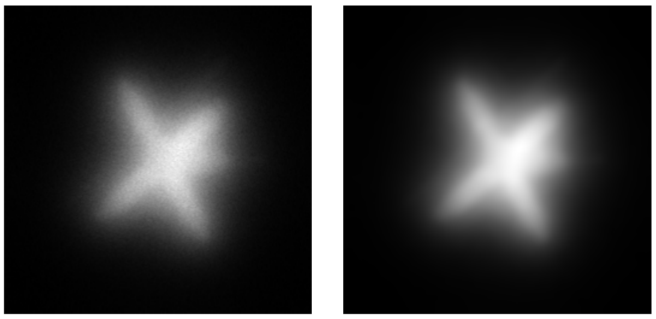


Figure 8.7: Original blurred and noisy image (RRE 0.0451) and the same data after iterative denoising (RRE 0.0091).

| | RRE | IT |
|---|---|---|
| $Z$-LR 1/4 | 0.3172 | 57 |
| $Z$-ISRA 1/4 | 0.3181 | 59 |

Table 8.8: $Z$-LR and $Z$-ISRA with $\nu$ acceleration ($\nu = 1.5$).

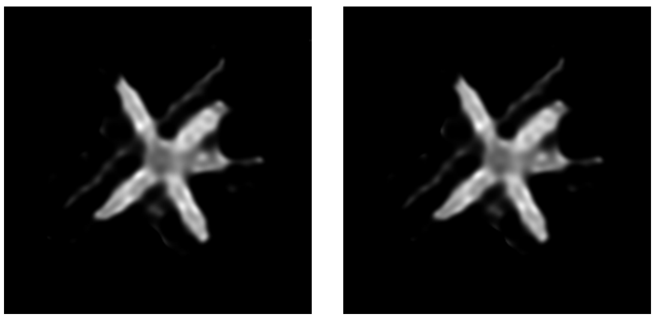


Figure 8.8: $Z$-LR restoration and $Z$-ISRA restoration.

# Bibliography

[1] H. M. Adorf, R. N. Hook, L. B. Lucy, F. D. Murtagh. *Accelerating the Richardson-Lucy restoration algorithm.* In 4th ESO/ST-ECF Data Analysis Workshop, 99–103, 1992.

[2] R.K. Ahuja, T.L. Magnanti, J.B. Orlin. *Network flows: theory, algorithms and applications.* Prentice Hall, Englewood Cliffs, NJ, 1993.

[3] A. Aricò, M. Donatelli. *A V-cycle multigrid for multilevel matrix algebras: proof of optimality.* Numer. Math. 105, 511–547, 2007.

[4] A. Aricò, M. Donatelli, J. Nagy, S. Serra Capizzano. *The anti-reflective transform and regularization by filtering.* Special volume *Numerical Linear Algebra in Signals, Systems, and Control.*, in Lecture Notes in Electrical Engineering, Springer Verlag., Vol. 80, 1–21, 2011.

[5] A. Aricò, M. Donatelli, S. Serra Capizzano *V-cycle optimal convergence for certain (multilevel) structured linearsystems.* SIAM J. Matrix Anal. Appl. 26–1, 186–214, 2004.

[6] A. Aricò, M. Donatelli, S. Serra Capizzano. *The Antireflective Algebra: Structural and Computational Analyses with Application to Image Deblurring and Denoising.* Calcolo 45–3, 149–175, 2008.

[7] A. Aricò, M. Donatelli, S. Serra Capizzano. *Spectral analysis of the anti-reflective algebra.* Linear Algebra Appl. 428, 657–675, 2008.

[8] O. Axelsson, G. Lindskög. *The rate of convergence of the preconditioned conjugate gradient method.* Numer. Math. 52, 499–523, 1986.

[9] O. Axelsson, M. Neytcheva. *The algebraic multilevel iteration methods – theory and applications.* Proc. of the 2nd Int. Coll. on Numerical Analysis, D. Bainov Ed., Plovdiv, Bulgaria, 13–23, 1993.

[10] M.S. Bazaraa, J.J. Jarvis, H.D. Sherali. *Linear programming and network flows.* Wiley, New York, NY, 1990.

[11] F. Benvenuto, R. Zanella, L. Zanni, M. Bertero. *Nonnegative least-squares image deblurring: improved gradient projection approaches.* Inverse Problems 26–2, 025004, 2010.

[12] M. Bern, J. Gilbert, B. Hendrickson, N. Nuygen, and S. Toledo. *Support-graph preconditioners.* SIAM J. Matrix Anal. & Appl. 27–4, 930–951, 2006.

[13] M. Bertero, P. Boccacci. *Introduction to inverse problems in imaging.* Institute of Physics Publ., Bristol, 1998.

[14] R. Bhatia. *Matrix Analysis.* Springer Verlag, New York, 1997.

[15] D. S. C. Biggs, M. Andrews. *Conjugate gradient acceleration of maximum-likelihood image restoration.* Electronics Letters 31, No. 23, 1985–1986, 1995.

[16] D. S. C. Biggs, M. Andrews. *Acceleration of iterative image restoration algorithms.* Applied Optics 36, No. 8, 1766–1775, 1997.

[17] D. Bini, M. Capovani. *Spectral and computational properties of band symmetric Toeplitz matrices.* Linear Algebra Appl. 52/53, 99–125, 1983.

[18] E.G. Boman, D. Chen, B. Hendrickson, S. Toledo. *Maximum-weight-basis preconditioners.* Numer. Linear Algebra Appl. 11, 8-9, 695–721, 2004.

[19] E.G. Boman, B. Hendrickson. *Support theory for preconditioning.* SIAM J. Matrix Anal. Appl. 25–3, 694–717, 2003.

[20] S. Bonettini, R. Zanella, L. Zanni. *A scaled gradient projection method for constrained image deblurring.* Inverse Problems 25–1, 015002, 2009.

[21] R. N. Bracewell. *The Fourier transform and its application.* New York, MacGraw-Hill, 1965.

[22] H. Brakhage. *On ill-posed problems and the method of conjugate gradient.* In: H. W. Engl, C. W. Groetsch, eds., Inverse and ill-posed problems. Academic press, Boston, New York, London, 165–175, 1987.

[23] M. Brezina, R.D. Falgout, S. MacLachlan, T.A. Manteuffel, S.F. McCormick, J. Ruge. *Adaptive smoothed aggregation (SA) multigrid.* SIAM Review 47–2, 317–346, 2005.

[24] M. Brezina, R.D. Falgout, S. MacLachlan, T.A. Manteuffel, S.F. McCormick, J. Ruge. *Adaptive algebraic multigrid.* SIAM J. Sci. Comput. 27–4, 1261–1286, 2006.

[25] P. Brianzi, F. Di Benedetto, C. Estatico. *Improvement of space-invariant image deblurring by preconditioned Landweber iterations.* SIAM J. Sci. Comput. 13, 1430–1458, 2008.

[26] P. Brianzi, F. Di Benedetto, C. Estatico. *Preconditioned iterative regularization in Banach spaces*, submitted.

[27] W. L. Briggs, V. E. Henson. *The DFT.* SIAM, Philadelphia, 1995.

[28] W.L. Briggs, V.E. Henson, S.F. McCormick. *A multigrid tutorial.* SIAM, 2nd edition, 2000.

[29] J. Castro *A specialized interior-point algorithm for multicommodity network flows.* SIAM J. Opt. 10, 852–877, 2000.

[30] J. Castro, A. Frangioni. *A parallel implementation of an interior-point algorithm for multicommodity network flows.* in Vector and Parallel Processing – VECPAR 2000, J.M. Palma, J. Dongarra and V. Hernandez eds., Lecture Notes in Computer Science Vol. 1981, Springer-Verlag, 301–315, 2001.

[31] A. Cayley. *A theorem on trees.* Quart. J. Math. 23, 376–378, 1889.

[32] R. H. Chan, K. P. Ng. *Toeplitz preconditioners for hermitian Toeplitz systems.* Linear Algebra Appl. 190, 181-208, 1993.

[33] T. Chan. *An optimal circulant preconditioner for Toeplitz systems.* SIAM J. Sci. Comput. 9, 766–771, 1988.

[34] D. Cherubini, A. Fanni, A. Frangioni, A. Mereu, C. Murgia, M.G. Scutellà, P. Zuddas. *A Linear Programming Model for Traffic Engineering in 100% Survivable Networks under combined IS-IS/OSPF and MPLS-TE Protocols.* Computers & Operations Research 38–12, 1805-1815, 2011.

[35] D. Cvetkovic, M. Doob, H. Sachs. *Spectra of Graphs.* Academic Press, New York, 1979.

[36] M. E. Daube-Witherspoon, G. Muehllehner. *An iterative image space reconstruction algorithm suitable for volume ECT.* IEEE Transactions on Medical Imaging MI-5, 61–66, 1986.

[37] P. J. Davis. *Circulant Matrices.* Wiley, New York, 1979.

[38] A. R. De Pierro. *On the convergence of the iterative image space reconstruction algorithm for volume ECT.* IEEE Transactions on Medical Imaging MI-6, 174–175, 1987.

[39] H. De Sterck, T.A. Manteuffel, S.F. McCormick, Q. Nguyen, J. Ruge. *Multilevel adaptive aggregation for Markov chains, with application to web ranking.* SIAM J. Sci. Comput. 30–5, 2235–2262, 2008.

[40] H. De Sterck, T.A. Manteuffel, S.F. McCormick, K. Miller, J. Pearson, J. Ruge, G. Sanders. *Smoothed aggregation multigrid for Markov chains.* SIAM J. Sci. Comput. 32–1, 40–61, 2010.

[41] H. De Sterck, T.A. Manteuffel, S.F. McCormick, K. Miller, J. Ruge, G. Sanders. *Algebraic multigrid for Markov chains.* SIAM J. Sci. Comput. 32–2, 544–562, 2010.

[42] H. De Sterck, T.A. Manteuffel, K. Miller, G. Sanders. *Top-level acceleration of adaptive algebraic multilevel methods for steady-state solution to Markov chains.* Advances in Computational Math. 35, 375403, 2010.

[43] H. De Sterck, K. Miller, G. Sanders, M. Winlaw. *Recursively accelerated multilevel aggregation for Markov chains.* SIAM J. Sci. Comput. 32–3, 1652–1671, 2010.

[44] G. Del Corso, A. Gullí, F. Romani. *Fast PageRank computation via a sparse linear system.* Internet Math. 3–2, 259–281, 2005.

[45] M. Donatelli. *An algebraic generalization of local Fourier analysis for grid transfer operators in multigrid based on Toeplitz matrices.* Numer. Linear Algebra Appl. 17, 179–197, 2010.

[46] M. Donatelli, M. Semplice, S. Serra Capizzano. *Analysis of Multigrid preconditioning for implicit PDE solvers for degenerate parabolic equations.* SIAM J. Matrix Anal. Appl., 32–4, 1125–1148, 2011.

[47] M. Donatelli, S. Serra Capizzano. *Anti-reflective boundary conditions and re-blurring.* Inverse Problems 21, 169–182, 2005.

[48] M. Donatelli, S. Serra Capizzano. *On the regularizing power of multigrid-type algorithms.* SIAM J. Sci. Comput. 27, No. 6, 2053–2076, 2006.

[49] M. Donatelli, S. Serra Capizzano. *Filter factor analysis of an iterative multilevel regularizing method.* Electron. Trans. Numer. Anal. 29, 163–177, 2007/2008.

[50] M. Donatelli, C. Estatico, A. Martinelli, S. Serra Capizzano. *Improved image deblurring with anti-reflective boundary conditions and re-blurring.* Inverse Problems 22, 2035–2053, 2006.

[51] M. Donatelli, C. Estatico, J. Nagy, L. Perrone, S. Serra Capizzano. *Anti-reflective boundary conditions and fast 2D deblurring models.* Proceeding to SPIE's 48th Annual Meeting, San Diego, CA USA, F. Luk Ed, 5205, 380–389, 2003.

[52] T. Elfving, P. C. Hansen, T. Nikazad. *Semi-convergence and relaxation parameters for projected sirt algorithms.* Electronic Transactions on Numerical Analysis 37, 321–336, 2010.

[53] H. W. Engl, M. Hanke, A. Neubauer. *Regularization of inverse problems.* Kluwer Academic Publishers, 1996.

[54] C. Estatico. *A classification scheme for regularizing preconditioners, with application to Toeplitz systems.* Linear Algebra and its Applications 397, 107–131, 2005.

[55] C. Estatico. *Regularization processes for real functions and ill-posed Toeplitz problems.* Operator Theory: Advances and Applications 160, 161–178, 2005.

[56] A. Frangioni, G. Gallo *A bundle type dual-ascent approach to linear multicommodity Min Cost Flow problems.* INFORMS J. Comput. 11, 370–393, 1999.

[57] A. Frangioni, B. Gendron *0-1 reformulations of the multicommodity capacitated network design problem.* Disc. Appl. Math. 157, 1229–1241, 2009.

[58] A. Frangioni, C. Gentile. *New Preconditioners for KKT Systems of Network Flow Problems.* SIAM J. Opt. 14, 894–913, 2004.

[59] A. Frangioni, C. Gentile. *Prim-based BCT preconditioners for Min-Cost Flow Problems.* Comput. Opt. Appl. 36, 271–287, 2007.

[60] A. Frangioni, A. Manca. *A Computational Study of Cost Reoptimization for Min Cost Flow Problems.* INFORMS J. On Comput. 18–1, 61–70, 2006.

[61] A. Frangioni, S. Serra Capizzano. *Spectral analysis of (sequences of) graph matrices.* SIAM J. Matrix Anal. Appl. 23–2, 339–348, 2001.

[62] G.H. Golub, C.F. Van Loan. *Matrix computations.* North Oxford Academic, 1983.

[63] K. Gremban. *Combinatorial Preconditioners for Sparse, Symmetric, Diagonally Dominant Linear Systems.* PhD Thesis, Carnegie Mellon University, CMU CS Tech Report CMU-CS-96-123, 1996.

[64] A. Greenbaum. *Analysis of a multigrid method as an iterative technique for solving linear systems.* SIAM J. Numerical Anal. 21–3, 473–485, 1984.

[65] C. W. Groetsch. *The theory of Tikhonov regularization for Fredholm equations of first kind.* Research Notes in Mathematics 105, Pitman Publishing, 1984.

[66] M. Hanke. *Accelerated Landweber iterations for the solutions of ill-posed equations.* Numer. Math. 60, 341–373, 1991.

[67] M. Hanke, J. Nagy. *Inverse Toeplitz preconditioners for ill-posed problems.* Linear Algebra Appl. 284, 177–192, 1998.

[68] M. Hanke, J. Nagy, C. Vogel. *Quasi-Newton approach to nonnegative image restorations.* Linear Algebra Appl. 316, 223–236, 2000.

[69] M. Hanke, J. Nagy, R. Plemmons. *Preconditioned iterative regularization for ill-posed problems.* Numerical Linear Algebra. Proceedings of the Conference in Numerical Linear Algebra and Scientific Computation, Kent, Ohio, March 13-14 1992, de Gruyter, 141–163, 1993.

[70] P. C. Hansen. *Rank-deficient and discrete ill-posed problems.* SIAM, Philadelphia, 1998.

[71] P. C. Hansen, J. G. Nagy, D. P. O'Leary. *Deblurring images: matrices, spectra and filtering.* SIAM, Philadelphia, 2006.

[72] T. J. Holmes, Y. H. Liu. *Acceleration of maximum-likelihood image restoration for fluorescence microscopy and other noncoherent imagery.* Journal of the Optical Society of America A 8, No. 6, 893–907, 1991.

[73] R. Horn, S. Serra Capizzano. *A general setting for the parametric Google matrix.* Internet Math. 3–4, 385–411, 2008.

[74] G. Horton, S.T. Leutenegger. *A multi-level solution algorithm for steady-state Markov chains.* Proceedings of the Acm Sigmetrics Conference on Measurement and Modeling of Computer Systems, 191–200, 1994.

[75] J. Kamm, J. Nagy. *Kronecker product and SVD approximations in image restoration.* Linear Algebra Appl. 284, 137–156, 1998.

[76] L. Kaufman. *Implementing and accelerating the EM algorithm for positron emission tomography.* IEEE Transactions on Medical Imaging 6, No. 1, 37–51, 1987.

[77] H.B. Keller. *Numerical methods for two-points boundary-value problems.* Blaisdell, London, 1968.

[78] G. Kirchhoff. *Uber die Auflosung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Strome gefuhrt wird.* Ann. Phys. Chem. 72, 497–508, 1847.

[79] I. Koutis. *Combinatorial and algebraic algorithms for optimal multilevel algorithms.* PhD Thesis, Carnegie Mellon University, CMU CS Tech Report CMU-CS-07-131, 2007.

[80] I. Koutis, G.L. Miller. *Graph partitioning into isolated, high conductance clusters: theory, computation and applications to preconditioning.* Symposiun on Parallel Algorithms and Architectures SPAA, 2008.

[81] I. Koutis, G.L. Miller. *A linear work, $O(n^{1/6})$ time, parallel algorithm for solving planar Laplacians.* Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms, SODA '07, 2007.

[82] I. Koutis, G.L. Miller, D. Tolliver. *Combinatorial Preconditioners and Multilevel Solvers for Problems in Computer Vision and Image Processing.* International Symposium of Visual Computing, 1067–1078, 2009.

[83] R. L. Lagendijk, J. Biemond. *Iterative Identification and Restoration of Images.* Springer-Verlag New York, Inc., 1991.

[84] L. Landweber. *An iteration formula for Fredholm integral equations of the first kind.* Amer. J. Math. 73, 615–624, 1951.

[85] R. G. Lane. *Methods for maximum likelihood deconvolution.* Journal of the Optical Society of America A 13, 1992–1998, 1996.

[86] A. Langville, C. Meyer. *A survey of eigenvector methods for WEB information retrieval.* SIAM Review 47–1, 135–161, 2005.

[87] L. B. Lucy. *An iterative technique for the rectification of observed images.* The Astronomical Journal 79, No.6, 745–754, 1974.

[88] E. S. Meinel. *Origins of linear and non-linear recursive restoration algorithms.* Journal of the Optical Society of America A 3, No. 6, 787–799, 1986.

[89] B. Mohar. *Some Applications of Laplace Eigenvalues of Graphs.* Graph Symmetry: Algebraic Methods and Applications, G. Hahn and G. Sabidussi eds., NATO ASI Ser. C 497, Kluwer, 225–275, 1997.

[90] R.D.C. Monteiro, J.W. O'Neal, T. Tsuchiya. *Uniform boundedness of a preconditioned normal matrix used in interior-point methods.* SIAM J. Opt. 15–1, 96–100, 2004.

[91] J. G. Nagy, K. Palmer, L. Perrone. *Iterative Methods for Image Deblurring: A Matlab Object Oriented Approach.* Numer. Algorithms 36, 73–93, 2004.

[92] M. K. Ng, R. H. Chan, W. C. Tang. *A fast algorithm for deblurring models with Neumann boundary conditions.* SIAM J. Sci. Comput. 21, no. 3, 851–866, 1999.

[93] M. Ng, S. Serra Capizzano, C. Tablino Possio. *Multigrid preconditioners for symmetric Sinc systems.* ANZIAM J. 45–E, 857–869, 2004.

[94] D. Noutsos, S. Serra Capizzano, P. Vassalos. *The conditioning of FD matrix sequences coming from semi-elliptic Differential Equations.* Linear Algebra Appl. 428–2/3, 600–624, 2008.

[95] Y. Notay. *An aggregation-based algebraic multigrid method.* Electronic Trans. Num. An. 37, 123–146, 2010.

[96] R. Olfati-Saber, R.M. Murray. *Consensus problems in networks of agents with switching topology and time-dealays.* IEEE Trans. Automatic Control 49–9, 1520–1533, 2004.

[97] A. Papoulis. *The Fourier integral and its application.* New York, MacGraw-Hill, 1962.

[98] L. Perrone. *Kronecker Product Approximations for Image Restoration with Anti-Reflective Boundary Conditions.* Numer. Linear Algebra Appl. 13–1, 1–22, 2006.

[99] M. Piana, M. Bertero. *Projected Landweber method and preconditioning.* Inverse Problems 13, 441–464, 1997.

[100] L.F. Portugal, M.G.C. Resende, G. Veiga, J.J. Jùdice. *A truncated primal-infeasible dual-feasible network interior point method* Networks 35, 91–108, 2000.

[101] W. H. Richardson. *Bayesian-based iterative method of image restoration.* Journal of the Optical Society of America 62, No. 1, 55–59, 1972.

[102] J.W. Ruge, K. Stüben. *Algebraic multigrid.* in Multigrid methods, vol. 3 of Frontiers Appl. Math., SIAM, Philadelphia, 73–130, 1987.

[103] Y. Saad. *Iterative Methods for Sparse Linear Systems.* PWS, Boston, 1996.

[104] F. Schopfer, A. K. Louis, T. Schuster. *Nonlinear iterative methods for linear ill-posed problems in Banach spaces.* Inverse Problems 22, 311-329, 2006.

[105] J. Schwartz, A. Steger, A. Weissl. *Fast Algorithms for Weighted Bipartite Matching.* in *Experimental and Efficient Algorithms*, Lecture Notes in Computer Science 3503, 476–487, 2005.

[106] S. Serra Capizzano. *A note on anti-reflective boundary conditions and fast deblurring models.* SIAM J. Sci. Comput. 25–3, 1307–1325, 2003.

[107] S. Serra Capizzano. *Multi-iterative methods.* Comput. Math. Appl. 26–4, 65–87, 1993.

[108] S. Serra Capizzano, C. Tablino Possio. *Multigrid methods for multilevel circulant matrices.* SIAM J. Sci. Comput. 26–1, 55–85, 2004.

[109] L. A. Sheep, Y. Vardi. *Maximum likelihood reconstruction for emission tomography.* IEEE Transactions on Medical Imaging MI-1, No. 2, 113–122, 1982.

[110] Y. Shi, Q. Chang. *Acceleration methods for image restoration problem with different boundary conditions.* Applied Numerical Mathematics, Volume 58, Issue 5, 602-614, 2008.

[111] M. K. Singh, U. S. Tiwary, Y. H. Kim. *An adptively accelerated Lucy-Richardson method for image deblurring.* EURASIP Journal on Advances in Signal Processing, 365021, 2008.

[112] D.A. Spielman, S.H. Teng. *Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems.* Proceedings of the 36th Annual ACM Symposium on Theory of Computing, 81–90, 2004.

[113] O. N. Strand. *Theory and methods related to the singular-function expansion and Landweber's iteration for integral equations of the first kind.* SIAM J. Numer. Anal. 11, 798–825, 1974.

[114] G. Strang. *The Discrete Cosine Transform.* SIAM Review 41–1, 135–147, 1999.

[115] K. Stüben. *A review of algebraic multigrid.* J. Comput. Appl. Math. 128, 281–309, 2001.

[116] C. Tablino Possio. *Truncated decompositions and filtering methods with Reflective/Anti-Reflective boundary conditions: a comparison,* in Matrix methods: theory, algorithms, applications. Dedicated to the Memory of Gene Golub, V. Olshevsky, E. Tyrtyshnikov Eds., World Scientific Publishing, 382–408, 2010.

[117] U. Trottenberg, C. Oosterlee, A. Schuller. *Multigrid.* Academic Press, 2001.

[118] E. E. Tyrtyshnikov, A. Y. Yeremin, N. L. Zamarashkin. *Clusters, preconditioners, convergence.* Linear Algebra Appl. 263, 25–48, 1997.

[119] P. M. Vaidya. *Solving linear equations with symmetric diagonally dominant matrices by constructing good preconditioners.* Unpublished manuscript. A talk based on the manuscript was presented at the IMA Workshop on Graph Theory and Sparse Matrix Computation, Minneapolis, 1991.

[120] P. H. Van Cittert. *Zum einfluss der Spaltbreite auf die Intensitatsverteilung in Spektrallinen II.* Z. Phys. 69, 298, 1931.

[121] P. Vanek, J. Mandel, M. Brezina. *Algebraic multigrid on unstructured meshes.* Technical Report X, Center for Computational Mathematics, Mathematics Department, 1994.

[122] R.S. Varga. *Matrix Iterative Analysis.* Prentice Hall, Englewood Cliffs, 1962.

[123] N. L. Zamarashkin, E. E. Tyrtyshnikov. *Distribution of eigenvalues and singular values of Toeplitz matrices under weakened conditions on the generating function.* Russian Acad. Sci. Sb. Math. 188, 1191–1021, 1997.