ON THE COMPUTATION OF SPARSE REFLEXIVE GENERALIZED INVERSES

Gabriel Oliveira da Ponte

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientadores: Marcia Helena Costa Fampa
Jon Lee

Rio de Janeiro
Dezembro de 2024

ON THE COMPUTATION OF SPARSE REFLEXIVE GENERALIZED INVERSES

Gabriel Oliveira da Ponte

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: Marcia Helena Costa Fampa
Jon Lee

Aprovada por: Profa. Marcia Helena Costa Fampa
Prof. Jon Lee
Prof. Nelson Maculan Filho
Prof. Renan Vicente Pinto
Profa. Fernanda Maria Pereira Raupp

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2024

Ponte, Gabriel Oliveira da

On the computation of sparse reflexive generalized inverses/Gabriel Oliveira da Ponte. – Rio de Janeiro: UFRJ/COPPE, 2024.

X, 87 p.: il.; 29, 7cm.

Orientadores: Marcia Helena Costa Fampa
Jon Lee

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2024.

Referências Bibliográficas: p. 86 – 87.

1. Generalized inverse. 2. Sparse optimization. 3. Approximation algorithm. I. Fampa, Marcia Helena Costa *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Agradecimentos

Gostaria de expressar minha gratidão, primeiramente a Deus, por todo auxílio ao longo dessa jornada. O período de mestrado é desafiador e, em muitos momentos, nos confrontamos com tribulações. A Igreja desempenhou um papel importante em me dar forças para continuar.

Aos meus pais acadêmicos, Marcia Fampa e Jon Lee, agradeço por trazerem leveza às dificuldades diárias da pesquisa, por seu apoio e paciência diante das minhas limitações, por todos os ensinamentos compartilhados e por serem acessíveis. Tenho muita sorte de tê-los como orientadores, eles criam um ambiente de família e união, fico muito contente em fazer parte do *lab*. Também gostaria de agradecer ao meu irmão acadêmico, Luze Xu, pela sua contribuição nos projetos desta dissertação.

Agradeço a Laura Balzano e Ahmad Mousavi por sugerirem a minimização da norma-2,1 com o intuito de induzir esparsidade estruturada e ao Fei Wang pela sugestão da reformulação de um problema de otimização neste trabalho a partir da decomposição em valores singulares para obtenção de melhor convergência numérica.

Agradeço ao ICERM pelo auxílio financeiro que me permitiu visitar a Universidade de Brown e participar do Workshop Discrete Optimization: Mathematics, Algorithms, and Computation, durante duas semanas do meu mestrado. Essa experiência foi essencial para dar continuidade ao projeto desta dissertação.

Agradeço ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), GM-GD 161501/2022-2, pelo suporte financeiro recebido.

Expresso minha gratidão a Nelson Maculan, Renan Pinto e Fernanda Raupp por gentilmente aceitarem fazer parte da banca examinadora.

Agradeço aos meus pais, Andréa e Robson, por todo o suporte ao longo da minha vida. Eles sempre se empenharam em me proporcionar uma boa educação e me mostraram a importância de ser uma pessoa ética e de bom coração. Minha gratidão também se estende à minha irmã, Isabela, por todos os momentos compartilhados ao longo desses anos, que trouxeram leveza ao meu cotidiano. Agradeço à minha avó, Algenora, por seu carinho e afeto.

Por fim, à minha amada noiva, Bruna, agradeço por ser uma pessoa tão meiga e gentil, e pelo companheirismo constante. Valorizo profundamente o seu apoio, auxílio, conselhos e carinho. Sou muito grato por tê-la em minha vida.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CALCULANDO INVERSAS REFLEXIVAS GENERALIZADAS ESPARSAS

Gabriel Oliveira da Ponte

Dezembro/2024

Orientadores: Marcia Helena Costa Fampa
Jon Lee

Programa: Engenharia de Sistemas e Computação

A pseudo-inversa de Moore-Penrose (M-P) pode ser usada em várias aplicações de álgebra linear; por exemplo, para calcular soluções de mínimos quadrados de sistemas lineares inconsistentes. Entretanto, independentemente de uma dada matriz ser esparsa, sua pseudo-inversa pode ser completamente densa, levando potencialmente a um alto custo computacional e dificuldades numéricas, especialmente ao lidarmos com matrizes de alta dimensão. A pseudo-inversa M-P é caracterizada por quatro propriedades, mas nem todas precisam ser atendidas para algumas aplicações. Nesta dissertação, aplicamos otimização matemática para induzir esparsidade geral e esparsidade estruturada em inversas generalizadas de uma dada matriz que satisfaz somente subconjuntos específicos das propriedades M-P. Utilizamos minimização da norma-1 (vetorial) para induzir esparsidade geral (não-estruturada) e a minimização da norma-2,1 para induzir esparsidade (estruturada) em linhas. A esparsidade estruturada é útil, não apenas por causa da eficiência computacional, mas também pela explicabilidade. No contexto da aplicação dos mínimos quadrados, é desejável ter esparsidade em linhas, ou seja, ter poucas linhas não nulas na inversa generalizada, pois assim o modelo linear associado é mais explicável. Mais especificamente, a teoria dos mínimos quadrados conecta variáveis explicativas a variáveis de resposta (observações), por meio de um modelo de regressão linear no qual os parâmetros desconhecidos da relação linear são estimados pela solução dos mínimos quadrados. A esparsidade em linhas corresponde à seleção de um pequeno número de variáveis explicativas para determinar o modelo linear. Nós também investigamos a aplicação de procedimentos de busca local para construir inversas generalizadas com esparsidade estruturada, posto pequeno e com controle da magnitude das entradas.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ON THE COMPUTATION OF SPARSE REFLEXIVE GENERALIZED INVERSES

Gabriel Oliveira da Ponte

December/2024

Advisors: Marcia Helena Costa Fampa
Jon Lee

Department: Systems Engineering and Computer Science

The well-known Moore-Penrose (M-P) pseudoinverse is used in several linear-algebra applications; for example, to compute least-squares solutions of inconsistent systems of linear equations. Irrespective of whether a given matrix is sparse, its M-P pseudoinverse can be completely dense, potentially leading to high computational burden and numerical difficulties, especially when we are dealing with high-dimensional matrices. The M-P pseudoinverse is uniquely characterized by four properties, but not all of them need to be satisfied for some applications. In this dissertation, we apply mathematical optimization to induce general sparsity and structured sparsity on generalized inverses of a given matrix, which satisfy only specific subsets of the M-P properties. We use 1-norm (vector) minimization to induce (unstructured) sparsity and 2,1-norm minimization to induce (structured) row-sparsity. Structured sparsity is useful, not only because of computational efficiency, but also for explainability. In the context of the least-squares application it is desirable to have row-sparsity, i.e., to have few non-zero rows on the generalized inverse, as then the associated linear model is more explainable. More specifically, least-squares theory connects explanatory variables to predicted variables (observations), through a linear regression model in which the unknown parameters of the linear relation are estimated by the least-squares solution. Row-sparsity corresponds to the selection of a small number of explanatory variables to determine a linear model. We also consider the application of local-search procedures that produce generalized inverses with guaranteed structured sparsity, with low rank, and with entries under control.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The well-known M-P (Moore-Penrose) pseudoinverse, independently discovered by E.H. Moore and R. Penrose, is used in several linear-algebra applications — for example, to compute least-squares solutions of inconsistent systems of linear equations. If $A = U\Sigma V^\intercal$ is the real singular-value decomposition of A (see Golub and Van Loan (1996), for example), then the M-P pseudoinverse of A can be defined as $A^\dagger := V\Sigma^\dagger U^\intercal$, where $\Sigma^\dagger$ has the shape of the transpose of the diagonal matrix Σ, and is derived from Σ by taking reciprocals of the non-zero (diagonal) elements of Σ (i.e., the non-zero singular values of A).

The following theorem gives a fundamental characterization of the M-P pseudoinverse.

Theorem 1 (Penrose (1955))**.** *For $A \in \mathbb{R}^{m\times n}$, the M-P pseudoinverse $A^\dagger$ is the unique $H \in \mathbb{R}^{n\times m}$ satisfying:*

$$AHA = A \tag{P1}$$

$$HAH = H \tag{P2}$$

$$(AH)^\intercal = AH \tag{P3}$$

$$(HA)^\intercal = HA \tag{P4}$$

The first three M-P properties are particularly important for our purposes.

P1: Following Rohde (1964), we say that a *generalized inverse* of A is any H satisfying P1. Note that without P1, even the all-zero matrix (which carries no information about A) satisfies the other M-P properties.

P2: A generalized inverse is *reflexive* if it satisfies P2. Theorem 3.14 in Rohde (1964) states that: (i) if H is a generalized inverse of A, then $\text{rank}(H) \geq \text{rank}(A)$, and (ii) a generalized inverse H of A is reflexive if and only if $\text{rank}(H) = \text{rank}(A)$. Therefore, enforcing P2 gives us the lowest possible rank of a generalized inverse of A.

P3: Following Xu, Fampa, Lee, and Ponte (2021), we say that H is *ah-symmetric* if it satisfies P3. That is, ah-symmetric means that AH is symmetric. If H is an ah-symmetric generalized inverse (i.e., H satisfies P1 and P3), then $\hat{x} := Hb$ solves the least-squares problem $\min\{\|Ax-b\|_2 : \ x \in \mathbb{R}^n\}$. So, not all of the M-P properties are required for a generalized inverse to solve this key problem. Especially important in our context, we have that by imposing property P3 on the generalized inverse when solving linear systems, we guarantee least-square solutions. If additionally, P2 is imposed, we have the advantage of working with generalized inverses of minimum rank. Low rank is a desirable property for an ah-symmetric generalized inverse H, in the context of the least-squares application, as it corresponds to a type of "explainability" for the associated linear model $\hat{x} := Hb$.

P4: Following Xu, Fampa, Lee, and Ponte (2021), we say that H is *ha-symmetric* if it satisfies P4. That is, ha-symmetric means that HA is symmetric. If H is an ha-symmetric generalized inverse (i.e., H satisfies P1 and P4, then $\hat{x} := Hb$ solves $\min\{\|x\|_2 : \ Ax = b, \ x \in \mathbb{R}^n\}$ (see Campbell and Meyer (2009); Fuentes, Fampa, and Lee (2016)). Equivalently to *ah-symmetric generalized inverses*, if additionally, P2 is imposed, we have the advantage of working with generalized inverses of minimum rank.

Even if a given input matrix A is sparse, its M-P pseudoinverse can be dense, leading to a high computational burden, especially in applications with high-dimensional matrices involving many response vectors b. Over the past ten years, sparse-optimization techniques have been widely studied to find sparse matrices H leading to efficiency in calculating $\hat{x} := Hb$. Dokmanić and Gribonval (2017a,b); Dokmanić, Kolundžija, and Vetterli (2013) introduced the idea of seeking to induce sparsity and row-sparsity in H by applying the standard minimization of its (vector) 1-norm and of its 2,1-norm over the set of left inverses (i.e., $HA = I_n$) or right inverses (i.e., $AH = I_m$). Fuentes, Fampa, and Lee (2020, 2016) suggested using relaxations based 1-norm minimization on subsets M-P properties aiming at sparse generalized inverses. Fampa and Lee (2018) investigated one such kind of sparse generalized inverse, with particular interest in rank-deficient matrices and suggested a polynomial-time local search with a performance guarantee, aiming at minimizing the 1-norm over block-structured generalized inverses.

The present work started in 2018, as an undergraduate research project and the goal was to follow up the work of Fampa and Lee (2018). Initially, I joined the ongoing project of Prof. Marcia Fampa, Prof. Jon Lee, and Luze Xu, to which I contributed by implementing the proposed algorithms and performing the computational experiments. The outcome of this work is described in Chapter §2. This dissertation reflects the continuity of our research on this topic to date, with new insights developed based on previous results. In the following, I further describe the

organization of this dissertation and the publications derived from our work.

1. In Chapter §2 we present part of the work in:
   Luze Xu, Marcia Fampa, Jon Lee, Gabriel Ponte. Approximate 1-norm minimization and minimum rank structured sparsity for various generalized inverses via local search. SIAM Journal on Optimization 31(3), 1722-1747, 2021. https://doi.org/10.1137/19M1281514
   We investigate a column block construction method to produce an ah-symmetric reflexive generalized inverse that is structured and has guaranteed sparsity. We provide a theoretically efficient and practical local-search algorithm to column-block construct an approximate 1-norm minimizing ah-symmetric reflexive generalized inverse.

2. In Chapter §3 we present the work in:
   Marcia Fampa, Jon Lee, Gabriel Ponte, Luze Xu. Experimental analysis of local searches for sparse reflexive generalized inverses. Journal of Global Optimization 81, 1057-1093, 2021. https://doi.org/10.1007/s10898-021-01087-y
   Fampa and Lee (2018) and Xu, Fampa, Lee, and Ponte (2021) proposed local-search procedures to construct sparse block-structured generalized inverses that satisfy only some of the M-P properties. We have implemented several local-search procedures based on results presented in these two papers and make an experimental analysis of them, considering their application to randomly generated matrices of varied dimensions, ranks, and densities. Further, we carried out a case study on a real-world data set.

3. In Chapter §4 we present the work in:
   Marcia Fampa, Jon Lee, Gabriel Ponte. Trading off 1-norm and sparsity against rank for linear models using mathematical optimization. Open Journal of Mathematical Optimization, 2(4), 14 p, 2021. https://doi.org/10.5802/ojmo.6
   We investigate the trade-off between low 1-norm and low rank for generalized inverses that can be used in the computation of least-squares solutions. We propose several algorithmic approaches that start from a 1-norm minimizing generalized inverse that satisfies the two key M-P properties, and gradually decrease its rank, by iteratively imposing the reflexive property. The algorithms iterate until the generalized inverse has the least possible rank. During the iterations, we produce intermediate solutions, trading off low 1-norm (and typically high sparsity) against low rank.

4. In Chapter §5 we present the work in:

Gabriel Ponte, Marcia Fampa, Jon Lee, Luze Xu. On computing sparse generalized inverses. Operations Research Letters 52, 2024. https://doi.org/10.1016/j.orl.2023.107058

We show that a 2,1-norm minimizing generalized inverse satisfies two additional M-P properties, including one needed for computing least-squares solutions. We present formulations related to finding row-sparse generalized inverses that can be solved very efficiently, which we verify numerically.

I gave talks about this work in:

1. 10th Week of Academic Integration at UFRJ, 2019;
2. LII Brazilian Symposium of Operations Research (SOBRAPO), 2020;
3. XLII Giulio Massarani Scientific, Technological, Artistic and Cultural Initiation Journey (JICTAC 2020 - Special Edition), 2021;
4. 11th Week of Academic Integration at UFRJ, 2021;

and I received the following awards:

1. Honorable Mention at 10th Week of Academic Integration at UFRJ, 2019;
2. SOBRAPO Undergraduate Research Project Award, 2021;
3. Honorable Mention at 11th Week of Academic Integration at UFRJ, 2021.

We introduce the notations that will be used throughout this work in Table 1.1.

Notation	**Description**
$\mathrm{vec}(X)$	column vector from a matrix X by stacking the column vectors of X
$A[S,T]$	submatrix of A with row indices S and column indices T
$A[S,:]$	submatrix of A formed by the rows S
$A[:,T]$	submatrix of A formed by the columns T
$A[S]$	submatrix $A[S,S]$ when A is symmetric
$A_{i\cdot}$	$A[\{i\},:]$
$A_{\cdot j}$	$A[:,\{j\}]$
$\|H\|_0$	number of nonzeros in the matrix H
$\|H\|_1$	$\|\mathrm{vec}(H)\|_1$
$\|H\|_{2,1}$	$\sum_i \|H_{i\cdot}\|_2$
$\|H\|_{\max}$	$\|\mathrm{vec}(H)\|_{\max}$
$\mathbf{e}$	vector of all-ones
$\mathbf{e}_i$	i-th standard unit vector
I	identity matrix
J	all-ones matrix
$\langle X,Y\rangle$	$\mathrm{trace}(X^\intercal Y) := \sum_{ij} x_{ij}y_{ij}$
$r(X)$	rank of X
det	determinant
$X \geq Y$	$X - Y$ is nonnegative
$X \succeq Y$	$X - Y$ is positive-semidefinite
$\sigma_{\min}(X)$	minimum singular value of X
$\otimes$	Kronecker product

Table 1.1: Summary of notations used in this dissertation

Chapter 2

Approximate 1-norm minimization and minimum-rank structured sparsity for various generalized inverses via local search

This chapter is part of the work that has been published as:
Luze Xu, Marcia Fampa, Jon Lee, Gabriel Ponte. Approximate 1-norm minimization and minimum rank structured sparsity for various generalized inverses via local search. SIAM Journal on Optimization 31(3), 1722-1747, 2021. https://doi.org/10.1137/19M1281514

This chapter is also part of Luze's Xu PhD Thesis, (see https://deepblue.lib.umich.edu/bitstream/handle/2027.42/172649/xuluze_1.pdf, Chapter 5). My specific contribution for this project was the implementation of the algorithms proposed and the performance of the numerical experiments.

2.1 Introduction

It is hard to find a generalized inverse (i.e., a solution of P1) having the minimum number of nonzeros, subject to various subsets of {P2, P3, P4} (but not all of them). We let $\|H\|_0$ (resp., $\|x\|_0$) be the number of nonzeros in the matrix H (resp., vector x). Dokmanić and Gribonval (2017b) established that $\min\{\|H\|_0 : \ P1\}$ is **NP**-hard as follows: for full row-rank $A \in \mathbb{R}^{m\times n}$ $(m < n)$, we have $\min\{\|H\|_0 : \ AHA = A\} = \min\{\|H\|_0 : \ AH = I\}$, and computing a minimizer can be done column-wise, as a collection of sparse optimization problems $\min\{\|x\|_0 : \ Ax = \mathbf{e}_i\}$. These latter sparse optimization problems are known to be **NP**-hard (see Natarajan (1995)) for a general right-hand side $b \neq 0$. But with A having full row rank, we can reduce

any general right-hand side $b \neq 0$, to a problem with $b = \mathbf{e}_i$, by left-multiplying A and b by an appropriate square and invertible matrix. Using the same idea, we can show the following hardness result.

Proposition 2. *The following problems are* **NP***-hard:*

$$\min\{\|H\|_0 : \ P1 + P2\}; \tag{$SGI12$}$$

$$\min\{\|H\|_0 : \ P1 + P3\}; \tag{$SGI13$}$$

$$\min\{\|H\|_0 : \ P1 + P2 + P3\}; \tag{$SGI123$}$$

$$\min\{\|H\|_0 : \ P1 + P4\}; \tag{$SGI14$}$$

$$\min\{\|H\|_0 : \ P1 + P2 + P4\}. \tag{$SGI124$}$$

Proof. For full row-rank $A \in \mathbb{R}^{m\times n}$ $(m < n)$, we have $AHA = A \ \Leftrightarrow AH = I$, and thus $HAH = H$ and $(AH)^\intercal = AH$ are also satisfied. Therefore, $(SGI12)$, $(SGI13)$, $(SGI123)$ are all equivalent to $\min\{\|H\|_0 : \ AH = I\}$, which is **NP**-hard. Similarly, with full column-rank A, we have that $(SGI14)$, $(SGI124)$ are **NP**-hard. □

We note that we have not been able to resolve the complexity of

$$\min\{\|H\|_0 : \ P1 + P3 + P4\}. \tag{$SGI134$}$$

Because of Proposition 2, we take the standard approach of minimizing $\|H\|_1$ to induce sparsity, subject to P1 and various subsets of {P2, P3, P4} (but not all).

It is a very important point that minimizing $\|H\|_1$ (or any norm), serves to keep the entries of H under control. This is very useful for applications, because it leads to more reasonable models (e.g., in the least-squares application) and with better numerics. Minimizing $\|H\|_0$ does not have any such property (as $\|\cdot\|_0$ is not a norm). Indeed, the cost of 10^{-8} and 10^{8} are the same under $\|\cdot\|_0$; but we can effectively round entries on the order of 10^{-8} to 0 in H, while many entries on the order of 10^{8} in H will lead to unstable computations using H. It might seem that minimizing $\|H\|_{\max}$ would more naturally keep *entries* of H under control, but there is a strong preference for minimizing $\|\cdot\|_1$ because it empirically induces sparsity, and it captures the lower envelope of $\|\cdot\|_0$ when the argument entries are in $[-1, 1]$. Moreover, $\|H\|_{\max}$ sees no benefit for reducing entries of H that are not largest.

Considering the tractability of minimizing $\|H\|_1$, we see that P1, P3 and P4 are linear constraints, which are easy to handle, while P2 is a non-convex quadratic, hence rather nasty. But, as we have noted, P2 is very useful for a generalized inverse, as it is equivalent to the rank of H being equal to the rank of A. Therefore, we are particularly interested in situations where, without solving a mathematical-programming formulation via a generic method (like linear programming (LP) or

non-convex quadratically-constrained programming), we can construct a minimizer or approximate minimizer of $\|H\|_1$, subject to P1, P2, and one or none of P3 and P4. In fact, our methods will do this and more. Additionally, we will get *structured sparsity* for H.

Fampa and Lee (2018) gave some results in this direction, when neither P3 nor P4 is enforced. In particular, Fampa and Lee (2018) gave a "block construction" of a generalized inverse H of rank-r A that is always reflexive, is "somewhat-sparse", having at most r^2 nonzeros and all confined to a choice of r rows and r columns (hence, structured). We note that any generalized inverse of A must have at least r nonzeros (because its rank is always at least r). Therefore, for any choice of block, the construction of Fampa and Lee (2018) has the number of nonzeros within a factor of r of the minimum number of nonzeros.

Fampa and Lee (2018) also demonstrated that there exists an easy-to-find block construction of a 1-norm minimizing reflexive generalized inverse, for rank-1 matrices and rank-2 nonnegative matrices. Finally, for general rank-r matrices, Fampa and Lee (2018) gave an efficient local-search based approximation algorithm, that efficiently finds a generalized inverse following the block construction, and that has its 1-norm within a factor of (almost) r^2 of the minimum 1-norm of any generalized inverse. In fact, experimentally, we see much better performance for the local search than this guarantee (see Fampa, Lee, Ponte, and Xu (2021)), while we establish here that the guarantee of the local search is best possible; see (Xu, Fampa, Lee, and Ponte, 2021, Appendix A).

In what follows, we aim at finding sparse ah-symmetric (or ha-symmetric) reflexive generalized inverses. Note that Proposition 2 ($SGI123$, $SGI124$) establishes that finding an ah-symmetric (or ha-symmetric) reflexive generalized inverse with minimum number of nonzeros is **NP**-hard even for the full row (or column) rank matrix A. So we aim at construction of an ah-symmetric (or ha-symmetric) reflexive generalized inverse with minimum (or approximately minimum) 1-norm.

In §2.2, we provide a local-search based (almost) r-approximation algorithm for general rank r. With an observation of the connection between ah-symmetric (reflexive) generalized inverses and ha-symmetric (reflexive) generalized inverses, we can easily extend all the results in §2.2 to the ha-symmetric case. In §2.3, we present results of numerical experiments for ah-symmetric generalized inverses, aimed at illustrating our results and confirming their applicability. Finally, in §2.4, we make some brief concluding remarks.

2.2 ah-symmetric results

In this section, let A be an arbitrary $m \times n$ real matrix. We seek to obtain a solution to $\min\{\|H\|_1 : P1 + P2 + P3\}$ (that is, a 1-norm minimizing ah-symmetric reflexive generalized inverse). As we have mentioned, ah-symmetric generalized inverses play a key role in solving least square problems. We develop an approximation approach for this problem that has many benefits, which we later summarize in Figure 2.1.

Proposition 3. *(see, for example, (Fuentes, Fampa, and Lee, 2016, Proposition 4.3)) If H satisfies P1 and P3, then $AH = AA^\dagger$.*

Proof.

$$\begin{aligned} AHA &= AA^\dagger A \qquad \text{(by P1)} \\ H^\intercal A^\intercal A &= (A^\dagger)^\intercal A^\intercal A \qquad \text{(by P3)} \\ A^\intercal AH &= A^\intercal AA^\dagger \\ (A^\dagger)^\intercal A^\intercal AH &= (A^\dagger)^\intercal A^\intercal AA^\dagger \\ AH &= AA^\dagger, \end{aligned}$$

the last equation following directly from a well-known property of $A^\dagger$. □

Note from Proposition 3 that if H is an ah-symmetric generalized inverse, then $AH = AA^\dagger$, where $A^\dagger$ is the M-P pseudoinverse. Therefore, P2 ($HAH = H$) becomes a linear constraint $HAA^\dagger = H$, which implies that $\min\{\|H\|_1 : P1 + P2 + P3\}$ can be cast as an LP. However, the extreme solutions of this (linear program) LP only have a guaranteed bound of $mr + (m-r)(n-r)$ for the number of nonzeros, while the extreme solutions of $\min\{\|H\|_1 : P1 + P3\}$ have at most mr nonzeros.

Proposition 4.
Suppose that $A \in \mathbb{R}^{m \times n}$ has rank r.

(1) Extreme solutions of the LP for $\min\{\|H\|_1 : P1 + P3\}$ have at most mr nonzeros. Furthermore, the bound is sharp for all $m \geq n \geq r \geq 1$.

(2) Extreme solutions of the LP for $\min\{\|H\|_1 : P1 + P2 + P3\}$ have at most $mr + (m-r)(n-r)$ nonzeros.

Proof. We have $\min\{\|H\|_1 : P1 + P3\} = \min\{\|H\|_1 : \ AH = AA^\dagger\} =$

$$\min\{\|\text{vec}(H)\|_1 : \ (I_m \otimes A)\text{vec}(H) = \text{vec}(AA^\dagger)\},$$

with $\text{rank}(I_m \otimes A) = \text{rank}(I_m)\,\text{rank}(A) = mr$. To see that the bound is sharp, let $\hat{A}$ be a random dense $r \times m$ matrix (with iid entries taken from any absolutely

continuous density), and then take A to be all zero except for $\hat{A}^\dagger$ in the western r columns. Then with probability one: $\hat{A}$ is dense, $\hat{A}$ has rank r, and A has a unique generalized inverse which is the M-P pseudoinverse $A^\dagger$, which is all zero except for the dense $r \times m$ block $\hat{A}$ in the northern r rows. Thus (1) holds.

As for $\min\{\|H\|_1 : P1 + P2 + P3\}$, it can be written as

$$\min\{\|\mathrm{vec}(H)\|_1 : \ (I_m \otimes A)\mathrm{vec}(H) = \mathrm{vec}(AA^\dagger), \ [(AA^\dagger \otimes I_n) - I_{mn}]\mathrm{vec}(H) = 0\},$$

with

$$\begin{aligned} &\mathrm{rank}\left(\begin{bmatrix} I_m \otimes A \\ (AA^\dagger \otimes I_n) - I_{mn} \end{bmatrix}\right) = \mathrm{rank}\left(\begin{bmatrix} I_m \otimes A \\ (AA^\dagger - I_m) \otimes I_n \end{bmatrix}\right) \\ = \ &\mathrm{rank}\left(\begin{bmatrix} I_m \otimes A \\ (AA^\dagger - I_m) \otimes (I_n - A^\dagger A) \end{bmatrix}\right) = mr + (m-r)(n-r). \end{aligned}$$

The second-to-last equation follows from the fact that $(AA^\dagger - I_m) \otimes A^\dagger A = ((AA^\dagger - I_m) \otimes A^\dagger)(I_m \otimes A)$. Thus (2) holds. □

We seek to do better than what Proposition 4, part (2) provides. We want fewer nonzeros, and we want block structure. To get these properties, we give a new *column block construction*, producing an ah-symmetric reflexive generalized inverse that has at most mr nonzeros.

Theorem 5. *For $A \in \mathbb{R}^{m\times n}$, let $r := \mathrm{rank}(A)$. For any T, an ordered subset of r elements from $\{1, \ldots, n\}$, let $\hat{A} := A[:, T]$ be the $m \times r$ submatrix of A formed by columns T. If $\mathrm{rank}(\hat{A}) = r$, let $\hat{H} := \hat{A}^\dagger = (\hat{A}^\intercal \hat{A})^{-1}\hat{A}^\intercal$. The $n \times m$ matrix H with all rows equal to zero, except rows T, which are given by $\hat{H}$, is an ah-symmetric reflexive generalized inverse of A.*

Proof. Without loss of generality, assume that $T := (1, 2, \ldots, r)$, so we may write

$$A = \begin{bmatrix} \hat{A} & \hat{B} \end{bmatrix}, \ H = \begin{bmatrix} \hat{H} \\ 0 \end{bmatrix}.$$

We have that H satisfies:

- P1, as $AHA = [\hat{A}\hat{H}\hat{A} \quad \hat{A}\hat{H}\hat{B}] = [\hat{A} \quad \hat{B}] = A$, where $\hat{A}\hat{H}\hat{A} = \hat{A}$ because $\hat{H}\hat{A}$ is the $r \times r$ identity matrix, and $\hat{A}\hat{H}\hat{B} = \hat{B}$ because, as A (and $\hat{A}$) has rank r, the columns of $\hat{B}$ are in the range of $\hat{A}$ and $\hat{A}\hat{H}$ is the projection matrix on the range of $\hat{A}$.

- P2, as

$$HAH = \begin{bmatrix} \hat{H}\hat{A}\hat{H} \\ 0 \end{bmatrix} = \begin{bmatrix} \hat{H} \\ 0 \end{bmatrix} = H,$$

where we again use the fact that $\hat{H}\hat{A}$ is the $r \times r$ identity matrix.

- P3, as $AH = \hat{A}\hat{H} = \hat{A}(\hat{A}^\intercal\hat{A})^{-1}\hat{A}^\intercal$ is symmetric.

□

Remark 6. *We have already mentioned that if H is an ah-symmetric generalized inverse, then $AH = AA^\dagger$. Therefore, P2 ($HAH = H$) becomes a linear constraint $HAA^\dagger = H$. In fact, rather than linearize using the M-P pseudoinverse $A^\dagger$, we can take any column block ah-symmetric generalized inverse $\hat{H}$ of A, and linearize more efficiently via $HA\hat{H} = H$. The cost of calculating such an $\hat{H}$ is the cost of calculating the M-P pseudoinverse of an $m \times r$ matrix, rather than the M-P pseudoinverse of the $m \times n$ matrix A.*

We note that it is useful to consider relaxing P2, arriving at $\min\{\|H\|_1 : P1 + P3\} = \min\{\|H\|_1 : AHA = A,\ (AH)^\intercal = AH\}$, which we re-cast as a linear-optimization problem (P_{ah}) and its dual (D_{ah}):

$$\begin{array}{ll} \text{minimize} & \langle J, H^+\rangle + \langle J, H^-\rangle \\ \text{subject to} & A(H^+ - H^-)A = A, \\ & (H^+ - H^-)^\intercal A^\intercal = A(H^+ - H^-), \\ & H^+, H^- \geq 0. \end{array} \tag{P_{ah}}$$

$$\begin{array}{ll} \text{maximize} & \langle A, W\rangle \\ \text{subject to} & -J \leq A^\intercal W A^\intercal + A^\intercal(V^\intercal - V) \leq J \end{array} \tag{D_{ah}}$$

We can see (D_{ah}) as: $\max\{\langle A, W\rangle :\ \|A^\intercal W A^\intercal + A^\intercal U\|_{\max} \leq 1,\ U^\intercal = -U\}$.

When $\text{rank}(A) = 1$, construction of a 1-norm minimizing ah-symmetric reflexive generalized inverse can be based on the column block construction over a column $\hat{a}$ that minimizes $\|\hat{a}^\dagger\|_1$ (see https://arxiv.org/abs/1903.05744).

Generally, when $\text{rank}(A) = 2$, we cannot construct a 1-norm minimizing ah-symmetric reflexive generalized inverse based on the column block construction. Even under the condition that A is nonnegative, we have the following example:

$$A = \begin{bmatrix} 1 & 3 & 8 \\ 2 & 2 & 8 \\ 3 & 1 & 8 \end{bmatrix}.$$

Note that $\text{rank}(A) = 2$ because $a_3 = 2a_1 + 2a_2$. We have an ah-symmetric reflexive generalize inverse with 1-norm $\frac{9}{8}$,

$$H := \begin{bmatrix} -\frac{1}{4} & 0 & \frac{1}{4} \\ \frac{1}{4} & 0 & -\frac{1}{4} \\ \frac{1}{24} & \frac{1}{24} & \frac{1}{24} \end{bmatrix}.$$

However, the three ah-symmetric reflexive generalized inverses based on our column block construction have 1-norm $\frac{31}{24}, \frac{31}{24}, \frac{7}{6}$, respectively. Nevertheless, under an efficiently-checkable technical condition, when $\text{rank}(A) = 2$, construction of a 1-norm minimizing ah-symmetric reflexive generalized inverse can be based on the column block construction (see https://arxiv.org/abs/1903.05744).

For general $r := \text{rank}(A)$, we will efficiently find an ah-symmetric reflexive generalized inverse following our column block construction that is within a factor $r(1+\epsilon)$ of the 1-norm of the ah-symmetric reflexive generalized inverse having minimum 1-norm.

Definition 7. *Let A be an arbitrary $m \times n$, rank-r matrix, and let S be an ordered subset of r elements from $\{1, \ldots, m\}$ such that these r rows of A are linearly independent. For T an ordered subset of r elements from $\{1, \ldots, n\}$, and fixed $\epsilon \geq 0$, if $|\det(A[S,T])|$ cannot be increased by a factor of more than $1+\epsilon$ by swapping an element of T with one from its complement, then we say that $A[S,T]$ is a $(1+\epsilon)$-local maximizer for the absolute determinant on the set of $r \times r$ nonsingular submatrices of $A[S,:]$.*

Lemma 8. *Let T be an ordered subset of r elements from $\{1, \ldots, n\}$ and $\hat{A} := A[:,T]$ be the $m \times r$ submatrix of an $m \times n$ matrix A formed by columns T, and $\text{rank}(\hat{A}) = r$. There exists an $m \times n$ matrix W and a skew-symmetric $m \times m$ matrix U such that*

$$\hat{A}^\intercal W A^\intercal + \hat{A}^\intercal U = E,$$

where $E := sign(\hat{A}^\dagger)$. Furthermore, $\langle A, W \rangle = \|\hat{A}^\dagger\|_1$.

Proof. Suppose that $\tilde{A} := A[S,T]$ is the nonsingular $r \times r$ submatrix of $\hat{A}$ formed by rows $S := \{i_1, i_2, \ldots, i_r\}$. Let $\hat{W}$ be an $r \times r$ matrix and W be an $m \times n$ matrix with all elements equal to zero, except the ones in rows S and columns T, which are given by the respective elements in $\hat{W}$. If we choose $\hat{W}$ and U to be

$$\hat{W} := \tilde{A}^{-\intercal} E \hat{A} (\hat{A}^\intercal \hat{A})^{-1} = \tilde{A}^{-\intercal} E (\hat{A}^\intercal)^\dagger$$

and

$$U := \hat{A}\hat{W}^\intercal D - D^\intercal \hat{W} \hat{A}^\intercal + D^\intercal \tilde{A}^{-\intercal} E - E^\intercal \tilde{A}^{-1} D \ ,$$

where D is an $r \times m$ matrix with all elements equal to zero, except $D_{1i_1} = D_{2i_2} = \cdots = D_{ri_r} = 1$.

Because $D\hat{A} = \tilde{A}$, we have

$$\begin{aligned}\hat{A}^\intercal U &= \hat{A}^\intercal \hat{A}\hat{W}^\intercal D - \tilde{A}^\intercal \hat{W}\hat{A}^\intercal + E - \hat{A}^\intercal E^\intercal \tilde{A}^{-1} D \\ &= E - \tilde{A}^\intercal \hat{W}\hat{A}^\intercal + (\hat{W}(\hat{A}^\intercal \hat{A}) - \tilde{A}^{-\intercal} E\hat{A})^\intercal D \\ &= E - \tilde{A}^\intercal \hat{W}\hat{A}^\intercal.\end{aligned}$$

Hence, $\hat{A}^\intercal W A^\intercal + \hat{A}^\intercal U = \tilde{A}^\intercal \hat{W}\hat{A}^\intercal + \hat{A}^\intercal U = E$. Furthermore,

$$\langle A, W\rangle = \operatorname{trace}(\tilde{A}^\intercal \hat{W}) = \operatorname{trace}(E(\hat{A}^\intercal)^\dagger) = \langle \hat{A}^\dagger, E\rangle = \|\hat{A}^\dagger\|_1 \ .$$

□

Theorem 9. *Let A be an arbitrary $m \times n$, rank-r matrix, and let S be an ordered subset of r elements from $\{1, \ldots, m\}$ such that these r rows of A are linearly independent. Choose $\epsilon \geq 0$, and let $\tilde{A} := A[S, T]$ be a $(1+\epsilon)$-local maximizer for the absolute determinant on the set of $r \times r$ nonsingular submatrices of $A[S, :]$. Then the $n \times m$ matrix H constructed by Theorem 5 over $\hat{A} := A[:, T]$, is an ah-symmetric reflexive generalized inverse of A satisfying $\|H\|_1 \leq r(1+\epsilon)\|H^r_{opt}\|_1$, where H^r_{opt} is a 1-norm minimizing ah-symmetric reflexive generalized inverse of A.*

Proof. We prove a stronger result $\|H\|_1 \leq r(1+\epsilon)\|H^{ah}_{opt}\|_1$, where H^{ah}_{opt} is an optimal solution of (P_{ah}), which implies $\|H\|_1 \leq r(1+\epsilon)\|H^{ah}_{opt}\|_1 \leq r(1+\epsilon)\|H^r_{opt}\|$. We will construct a dual feasible solution with objective value $\frac{1}{r(1+\epsilon)}\|H\|_1$. By weak duality for linear optimization, we will then have $\frac{1}{r(1+\epsilon)}\|H\|_1 \leq \|H^{ah}_{opt}\|_1$.

By Lemma 8, we can choose W and a skew-symmetric matrix U such that $\hat{A}^\intercal W A^\intercal + \hat{A}^\intercal U = E$. and $\langle A, W\rangle = \|\hat{A}^\dagger\|_1 = \|H\|_1$.

So it is sufficient to demonstrate that $\|A^\intercal W A^\intercal + A^\intercal U\|_{\max} \leq r(1+\epsilon)$, then $\frac{1}{r(1+\epsilon)}W, \frac{1}{r(1+\epsilon)}U$ is dual feasible and $\langle A, \frac{1}{r(1+\epsilon)}W\rangle = \frac{1}{r(1+\epsilon)}\|H\|_1$.

First, it is clear that $\|\hat{A}^\intercal W A^\intercal + \hat{A}^\intercal U\|_{\max} = \|E\|_{\max} = 1 \leq r(1+\epsilon)$. Next, we consider any column $\hat{b}$ of $\hat{B}$, because $\operatorname{rank}(\hat{A}) = r = \operatorname{rank}(A)$, we know that $\hat{b} = \hat{A}\beta$, $\beta \in \mathbb{R}^r$, which implies $\tilde{b} = \tilde{A}\beta$. By Cramer's rule, where $\tilde{A}_i(\tilde{b})$ is $\tilde{A}$ with column i replaced by $\tilde{b}$, we have

$$|\beta_i| = \frac{|\det(\tilde{A}_i(\tilde{b}))|}{|\det(\tilde{A})|} \leq 1 + \epsilon,$$

because $\tilde{A}$ is a $(1+\epsilon)$-local maximizer for the absolute determinant of $A[S, :]$. Therefore

$$\begin{aligned}\|\hat{b}^\intercal W A^\intercal + \hat{b}^\intercal U\|_{\max} &= \|\beta^\intercal(\hat{A}^\intercal W A^\intercal + \hat{A}^\intercal U)\|_{\max} \\ &= \|\beta^\intercal E\|_{\max} \leq \sum_{i=1}^{r} |\beta_i| \leq r(1+\epsilon).\end{aligned}$$

□

Remark 10. *In Theorem 9, we could have required the stronger condition that $\tilde{A}$ is a global maximizer for the absolute determinant on the set of $r \times r$ nonsingular submatrices of A_σ. But we prefer our hypothesis, both because it is weaker and because we can find an $\tilde{A}$ satisfying our hypothesis by a simple finitely-terminating local search. Moreover, if A is rational, and we choose ϵ positive and fixed, then our local search is efficient.*

Theorem 11. *Let A be rational. We have an FPTAS for calculating an ah-symmetric reflexive generalized inverse H of A that has $\|H\|_1$ within a factor of r of $\|H^r_{opt}\|_1$, where H^r_{opt} is a 1-norm minimizing ah-symmetric reflexive generalized inverse of A.*

Proof. Following the proof in (Fampa and Lee, 2018, Theorem 10), we have that the local search reaches a $(1+\epsilon)$-local maximizer for the absolute determinant on the set of $r \times r$ nonsingular principal submatrices of A in at most $\mathcal{O}(\text{poly}(\text{size}(A)))(1 + \frac{1}{\epsilon})$ iterations, where size(A) is the number of bits in a binary encoding of A. Along with Theorem 9, we conclude that the local search is an FPTAS. □

As we have mentioned, ah-symmetric generalized inverses have the key use for solving least-squares problems. In Figure 2.1, we compare various possibilities for calculating ah-symmetric generalized inverses, highlighting the excellent properties of the solution produced by our local search.

Considering Figure 2.1, we dismiss methods based on minimizing the 0-norm as we do not have nice computational methods for them, and they suffer from not being able to control the magnitudes of entries. Concentrating now on tractable optimization methods (that seek to keep the magnitude of entries under control), we have LP-based methods and our local search.

We can see some very important advantages of our local search: (i) comparing just to LP-based methods, our local search has (block) structure, while the LP-based methods have no guaranteed structure; (ii) comparing further to the LP based on P1+P3, our local search has a low-rank guarantee, while the LP method does not. (iii) instead comparing further to the LP based on P1+P2+P3, our local search has a much better sparsity guarantee than the LP method.

Remark 12. *H is a ha-symmetric (reflexive) generalized inverse of A if and only if $H^\intercal$ is an ah-symmetric (reflexive) generalized inverse of $A^\intercal$. Following this observation, we can extend all the results in section §2.2 to the ha-symmetric case.*

Figure 2.1: Comparing options for ah-symmetric generalized inverses

arbitrary ah-sym	arbitrary reflexive ah-sym	0-norm min ah-sym	0-norm min reflexive ah-sym	LP:P1+P3	LP:P1+P2+P3	arbitrary block	our local search	
✗	✗	✗	✗	✓	✓	✗	✓	entries under control[a]
✗	✓	✗	✓	✗	✓	✓	✓	guaranteed low rank ($= r$[b])
✗	✗	✗	✗	✗	✗	✓	✓	structured[c]
✗	✗	✓	✓	✓[d]	✗[e]	✓	✓	guaranteed sparsity ($\leq rm$ nonzeros[f])
✗	✗	✓	✓	✓	✓	✗	✓	induced sparsity[g]
✓	✓	✗[h]	✗[i]	✓	✓	✓	✓	calculate efficiently

[a] via 1-norm pressure
[b] via P2
[c] via column block construction
[d] see Proposition 4, part (1)
[e] no more than $rm + (m-r)(n-r)$ nonzeros: see Proposition 4, part (2)
[f] via column block construction
[g] via 1-norm or 0-norm pressure
[h] see Proposition 2, $SGI13$
[i] see Proposition 2, $SGI123$

2.3 Numerical experiments

Next, we report on some numerical results to illustrate and confirm the applicability of our proposed approach for constructing generalized inverses. For that, we have selected the ah-symmetric case and implemented a local-search algorithm based on Theorem 5, Definition 7, and Theorem 9. Specifically, we choose a set of $r := \text{rank}(A)$ row indices S (from $\{1, 2, \ldots, m\}$). Within $A[S, :]$, we choose an $r \times r$ nonsingular submatrix $A[S, T]$. Then we do a local search (repeatedly considering pairwise swaps of elements between T and $\{1, 2, \ldots, n\} \setminus T$) to locally maximize $|\det(A[S, T])|$; for the purpose of computations, the parameter ϵ in Theorem 9 was chosen to be zero. Using the resulting T from the local search, let $\hat{A} := A[:, T]$, let $\hat{H} := \hat{A}^\dagger = (\hat{A}^\intercal \hat{A})^{-1} \hat{A}^\intercal$, and finally, our output H is the $n \times m$ matrix with all rows equal to zero, except rows T, which are given by $\hat{H}$.

The algorithm was coded in Matlab R2020a, and to evaluate its performance, we also solved the linear programs LP:P1+P3 and LP:P1+P2+P3 for the smaller instances, with Gurobi v.9.0.2. We ran our experiments on a 16-core machine (running Windows Server 2016 Standard): two Intel Xeon CPU E5-2667 v4 processors running at 3.20GHz, with 8 cores each, and 128 GB of memory.

Our test matrices were randomly generated with varied dimensions and ranks. We used the Matlab function *sprand*, which generates a random $m \times n$ dimensional matrix A with singular values given by a nonnegative input vector rc. We generated dense matrices and selected the r nonzeros of rc as the decreasing vector $M \times (\rho^1, \rho^2, \ldots, \rho^r)$, where $M = 2$, and $\rho = (1/M)^{(2/(r+1))}$.

Average results for our first experiment are reported in Table 2.1. We solved LP:P1+P3 and LP:P1+P2+P3 for five instances of each dimension/rank indicated in the first column of the table, limiting the computational time to solve each instance to 2 hours (i.e., 7200 seconds). Our purpose is to demonstrate how fast the time to solve these problems increases as we increase the dimension/rank of our test matrices. In the third column of Table 2.1, we give the number of instances solved to optimality within the time limit. The average times in the second column, only take into account the instances solved to optimality. We note that for $m, n, r = 200, 100, 50$, we could only solve one instance with each LP model. The results demonstrate that computing ah-symmetric generalized inverses by solving the LP problems does not scale well and is not a practical approach for instances of moderate size, even when the reflexive property P2 is not imposed.

	Time (sec)		Instances solved	
m, n, r	LP:P1+P3	LP:P1+P2+P3	LP:P1+P3	LP:P1+P2+P3
40, 20, 10	1.76	1.98	5	5
80, 40, 20	41.39	40.19	5	5
120, 60, 30	384.34	390.34	5	5
160, 80, 40	4130.99	4248.34	4	3
200, 100, 50	4197.86	4707.34	1	1

Table 2.1: Computation of minimum 1-norm H with LP models

In Table 2.2, we compare the optimal solution of LP:P1+P2+P3 to the reflexive ah-symmetric generalized inverse obtained by the local search, showing the average ratios between the 1-norm of the solutions of the local search (H) and of the solutions of LP (H_{123}). The same ratios are shown considering the 0-norm (computed with tolerance 10^{-6}). In this experiment we use 30 instances of each dimension/rank indicated in the first column of the table, and report the mean and standard deviation (in parenthesis) of the norms for each group. The results confirm the advantage of the local search over the LP solution in obtaining sparser matrices (via our column block construction), while keeping the magnitude of the entries reasonably small (via our approximate 1-norm minimization).

In Tables 2.3 and 2.4 we investigate the performance of the local search. In the second column of these tables we show the relative decrease on the 1-norm of the reflexive ah-symmetric generalized inverse, comparing the solution H obtained by the local search to the matrix H^0 used to initialize the algorithm.

m, n, r	$\|H\|_1/\|H_{123}\|_1$	$\|H\|_0/\|H_{123}\|_0$
40, 20, 10	1.14 (0.06)	0.71 (0.05)
80, 40, 20	1.30 (0.09)	0.65 (0.03)
120, 60, 30	1.35 (0.10)	0.64 (0.02)

Table 2.2: Comparison between local search and LP (Mean(Std Dev))

To construct the matrix H^0, we need to select an $r \times r$ nonsingular submatrix $\bar{A} := A[S, T]$ of A. For that, we first apply a fast phase-one algorithm that selects an $\ell \times \ell$ nonsingular submatrix of A. If $\ell < r$, we then apply a greedy algorithm to obtain $\bar{A}$, initializing it with the ℓ rows selected from A.

For the phase-one algorithm, we construct an $(m + r) \times n$ matrix $\tilde{A}$. On the m first rows of $\tilde{A}$, we have the matrix A with the columns randomly reordered. The last r rows of $\tilde{A}$ are all zero except for its south-west corner, where we have the identity matrix scaled by a small factor (10^{-6}). We apply our ah-symmetric local-search algorithm to $\tilde{A}^\intercal$ at most five times, each time the columns of A are ordered differently. The local search in phase-one is easily initialized with the scaled identity matrix and converges to an $r \times r$ submatrix $\tilde{A}[\tilde{S}, \tilde{T}]$ of $\tilde{A}$, which is a local maximizer for the absolute determinant on the set of $r \times r$ nonsingular submatrices of $\tilde{A}[\tilde{S}, :]$.

If $\tilde{S} \subset \{1, \ldots, m\}$, we set $\bar{A} = \tilde{A}$, and the algorithm stops. Otherwise, we consider the rows of A indexed by $\tilde{S} \cap \{1, \ldots, m\}$ and apply a greedy algorithm, which initially selects the remaining rows from A to compose the set S, and then, from the chosen set S of rows, it selects r columns to compose the set T. Each row and column selected is the first obtained (with the least indices), which keeps the least singular value of the partially constructed submatrix greater than a given positive tolerance τ. In practice, we start with a relatively large value of τ, and then we decrease it whenever we are not able to find sufficient rows/columns.

In the two last columns of the tables we report the total computational time and number of column swaps performed by the local search. The time to compute the initial matrix H^0 and to perform the local search are both included.

In Table 2.3, we consider 30 instances of each dimension/rank, and we present the mean and standard deviation for each group. We note that the local search is effective in reducing the 1-norm of the initial matrix and is much faster than solving LP problems of smaller dimensions, as can be observed from the results in Table 2.1. The average number of column swaps and the standard deviation for the norm decrease demonstrates that the algorithm is very stable, converging to similar solutions after swapping about 60% of the columns in the matrix.

In Table 2.4, we consider five instances of each dimension/rank, and present average results. Our purpose with this last experiment is to show the scalability of the local search. The algorithm is able to construct sparse reflexive ah-symmetric

m, n, r	$\frac{\|H^0\|_1 - \|H\|_1}{\|H^0\|_1}$	Time (sec)	Swaps
250, 125, 25	0.90 (0.08)	0.03 (0.01)	73.03 (11.11)
500, 250, 50	0.94 (0.06)	0.10 (0.03)	159.67 (20.84)
1000, 500, 100	0.91 (0.13)	0.84 (0.29)	293.33 (106.47)

Table 2.3: Performance of the local search - medium-size instances (30 of each dimension)(Mean(Std Dev))

generalized inverses for our test matrices with up to 10000 rows, 1000 columns and rank 100, in less than 1.1 second on average.

m, n, r	$\frac{\|H^0\|_1 - \|H\|_1}{\|H^0\|_1}$	Time (sec)	Swaps
5000, 500, 50	0.91	0.21	121.8
7500, 750, 75	0.89	0.65	172.4
10000, 1000, 100	0.89	1.09	204.8

Table 2.4: Performance of the local search - large-size instances (5 of each dimension)(Mean)

2.4 Concluding remarks

Generalized inverses have a wide variety of uses in matrix algebra and its applications. Sparsity of a generalized inverse is highly preferred for efficiency in its use; structured sparsity and low rank (=reflexivity) are both preferred for explainability. (Approximate) 1-norm minimization is useful for keeping entries under control and for inducing sparsity.

Ah-symmetric (resp., ha-symmetric) generalized inverses have the key use in solving least-squares (resp., minimum-norm) problems. Reflexive generalized inverses have low rank (same as the input matrix), and this is usually preferred in applications.

We have given a local-search algorithm that efficiently produces reflexive ah-symmetric (ha-symmetric) generalized inverses. Our algorithm produces generalized inverses with guaranteed structured sparsity, with low rank (same as the input matrix), and with entries under control (by approximate 1-norm minimization). No other known methods have all of these nice properties.

Of course giving efficient algorithms to improve any of our approximation ratios is a nice challenge. Even for special classes of matrices, this could be interesting. It would be nice to resolve the complexity of $\min\{\|H\|_0 : \ P1 + P3 + P4\}$.

Chapter 3

Experimental analysis of local searches for sparse reflexive generalized inverses

This chapter correspond to the work that has been published as:
Marcia Fampa, Jon Lee, Gabriel Ponte, Luze Xu. Experimental analysis of local searches for sparse reflexive generalized inverses. Journal of Global Optimization 81, 1057-1093, 2021. https://doi.org/10.1007/s10898-021-01087-y

Part of this chapter was presented in my undergraduate final project. More specifically, §3.1, §3.3, §3.5, and §3.6. Here, we present these sections with more details and add §3.2 and §3.4, where we present a complete analysis of other types of generalized inverses.

3.1 Introduction

Fampa and Lee (2018) and Xu, Fampa, Lee, and Ponte (2021) propose local-search procedures to construct reflexive generalized inverses, ah-symmetric reflexive generalized inverses, and in case A is symmetric, symmetric reflexive generalized inverses. The purpose of the procedures is the construction of sparser matrices than the M-P pseudoinverse, without losing some of its important properties. In Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021), (vector) 1-norm minimization is used to induce sparsity (leading to less computational burden in applications) and to keep the magnitude of the entries under control (leading to better numerical stability in applications). Therefore, at each iteration of the local-search procedures, the overall goal is to decrease the 1-norm of the constructed matrix H.

The generalized inverses constructed by the procedures have the following very nice features: they have block structure, i.e., they have all non-zero entries confined

to a selected choice of columns (and, sometimes, also of rows), they are reflexive, they have a bounded number of non-zero entries, and they have 1-norm within a provable factor of the minimum 1-norm of generalized inverses with corresponding properties.

Our goal in this chapter is to develop and analyze through numerical experiments, the performance of local-search procedures based on the ideas presented in Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021), and to see how tight are the bounds presented for the 1-norms of the constructed matrices H, considering randomly generated input matrices A with varied dimensions, ranks, and densities. We have implemented different local-search procedures for each case studied, more specifically, the cases where we construct (i) a reflexive generalized inverse, (ii) an ah-symmetric reflexive generalized inverse, and (iii) a symmetric reflexive generalized inverse. We propose a method for constructing an initial solution for the local searches; interestingly, this turns out to be a rather difficult numerical task at large scale, even though in theory it is rather trivial. We propose and compare local searches with updates performed with the best improvement ('BI') obtained in the neighborhood of the starting solution, and with updates performed with the first improvement ('FI') obtained. We analyze local searches that consider as the criterion for improvement, the increase in the absolute value of the determinant of an $r \times r$ non-singular submatrix of the given rank-r matrix A, which are based on theoretical results presented in Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021). These procedures are identified in the paper with the notation 'det'. We also propose a local search that considers a more natural criterion for improvement, the decrease in the 1-norm of the constructed matrix H, and is identified with 'norm'. Observing the behavior of these local searches leads us to combine the 'det' with the 'norm' searches. Aiming at reaching matrices with smaller norms, we apply hybrid procedures that perform local searches based on the decrease of the 1-norm of H, starting from the output of a local search based on the increase of the absolute determinant of the submatrix of A.

The algorithms proposed were coded in Matlab R2019b. To evaluate the solutions obtained by them, we solve the linear programming (LP) problems described in the next sections, with Gurobi v.9.0.2. We ran the experiments on a 16-core machine (running Windows Server 2016 Standard): two Intel Xeon CPU E5-2667 v4 processors running at 3.20GHz, with 8 cores each, and 128 GB of memory.

In §3.2, we present our results for generalized inverses. In §3.3, we present our results for ah-symmetric generalized inverses. In §3.4, we present our results for symmetric generalized inverses (applied to symmetric input matrices). In §3.5, we present a case study where we apply our algorithm for ah-symmetric generalized inverses to real data. In §3.6, we make some brief concluding remarks.

Before continuing, we wish to mention that an earlier approach to constructing sparse generalized inverses was developed in Fuentes, Fampa, and Lee (2020). Unfortunately those methods, based on solving convex relaxations (linear programming (LP) and convex quadratic programming (QP)), scale very poorly. The failure of those methods to scale efficiently led to the investigations in Fampa and Lee (2018) and Xu, Fampa, Lee, and Ponte (2021), which in turn motivated our present work. Dokmanić and Gribonval (2017a,b); Dokmanić, Kolundžija, and Vetterli (2013) presents an additional prior approach, based also on LP, for constructing sparse left and right pseudoinverses.

3.2 Generalized inverse

The local-search procedures for the reflexive generalized inverse are based on the block construction procedure proposed in Fampa and Lee (2018). More specifically they are based on Theorem 13, Definition 14, and Theorem 15, presented next.

Theorem 13 (Fampa and Lee (2018))**.** *For $A \in \mathbb{R}^{m\times n}$, let $r := \text{rank}(A)$. Let $\tilde{A}$ be* any *$r \times r$ non-singular submatrix of A. Let $H \in \mathbb{R}^{n\times m}$ be such that its submatrix that corresponds in position to that of $\tilde{A}$ in A is equal to $\tilde{A}^{-1}$, and other positions in H are zero. Then H is a reflexive generalized inverse of A.*

Definition 14 (Fampa and Lee (2018))**.** *For $A \in \mathbb{R}^{m\times n}$, let $r := \text{rank}(A)$. For σ an ordered subset of r elements from $\{1,\ldots,m\}$ and τ an ordered subset of r elements from $\{1,\ldots,n\}$, let $A[\sigma,\tau]$ be the $r \times r$ submatrix of A with row indices σ and column indices τ. For fixed $\epsilon \geq 0$, if $|\det(A[\sigma,\tau])|$ cannot be increased by a factor of more than $1+\epsilon$ by either swapping an element of σ with one from its complement or swapping an element of τ with one from its complement, then we say that $A[\sigma,\tau]$ is a* $(1+\epsilon)$-local maximizer for the absolute determinant *on the set of $r \times r$ non-singular submatrices of A.*

Theorem 15 (Fampa and Lee (2018))**.** *For $A \in \mathbb{R}^{m\times n}$, let $r := \text{rank}(A)$. Choose $\epsilon \geq 0$, and let $\tilde{A}$ be a $(1+\epsilon)$-local maximizer for the absolute determinant on the set of $r \times r$ non-singular submatrices of A. Construct H as per Theorem 13. Then H is a (reflexive) generalized inverse (having at most r^2 non-zeros), satisfying $\|H\|_1 \leq r^2(1+\epsilon)^2\|H_{opt}\|_1$, where H_{opt} is a 1-norm minimizing generalized inverse of A .*

We note that the ϵ of Definition 14 and Theorem 15 is used in Fampa and Lee (2018) to gain polynomial running time in $1/\epsilon$. For the purpose of actual computations, our observation has been that ϵ can be chosen to be zero. We further note that in Xu, Fampa, Lee, and Ponte (2021), we demonstrated that the bound

in Theorem 15 is the best possible. However, we will see in our experiments that the bound is overly pessimistic by a wide margin.

The idea of our algorithms is to select an $r \times r$ non-singular submatrix $\tilde{A}$ of A, and construct the reflexive generalized inverse with the inverse of this submatrix, as described in Theorem 13. The non-zero entries of H will be the non-zero entries of $\tilde{A}^{-1}$. Guided by the result in Theorem 15, the 'det' searches aim at selecting a submatrix $\tilde{A}$ that is a local maximizer for the absolute value of the determinant on the set of $r \times r$ non-singular submatrices of the given matrix A. In an attempt to construct matrices H with smaller 1-norm, the 'norm' searches more directly try to decrease the 1-norm of the matrix constructed at each iteration.

In the following, we discuss how the test matrices A used in our computational experiments were generated, how we select the initial submatrix of A to initialize the local searches, and we give details of the algorithms and present numerical results.

To analyze the local-search procedures proposed, we compare their solutions to the solution of a natural LP problem, identified below as P_1. Its optimal solution value corresponds to $\|H_{opt}\|_1$, where as defined in Theorem 15, H_{opt} is a 1-norm minimizing generalized inverse of A.

$$\begin{aligned} (P_1)\ z_{P_1} := \quad & \min \quad \langle J, T \rangle \ , \\ & \text{s.t.:} \quad T - H \geq 0 \ , \\ & \qquad\ \ T + H \geq 0 \ , \\ & \qquad\ \ AHA = A \ . \end{aligned}$$

3.2.1 Our test matrices

To test the proposed local-search procedures, we randomly generated 462 matrices with varied dimensions, ranks, and densities, with the Matlab function *sprand*. The function generates a random $m \times n$ dimensional matrix A with approximate density d and singular values given by the non-negative input vector rc. The number of non-zero singular values in rc is of course the desired rank r. The matrix is generated by sprand using random plane rotations applied to a diagonal matrix with the given singular values. For our experiments, we selected the r nonzeros of rc as the decreasing vector $M \times (\rho^1, \rho^2, \ldots, \rho^r)$, where $M = 2$, and $\rho = (1/M)^{(2/(r+1))}$. The shape of this distribution is concave (as is the case for many matrices that one encounters), and moreover, the entries are not extreme (always between 1/2 and 2), and the product is unity, so we can reasonably hope that the numerics may not be terrible.

We divide our instances into the following three categories:

- Small: 90 instances. 5 with each of the 18 combinations of the following parameters: $m = n = 50, 80, 100$; $r = 0.1 \times n, 0.5 \times n$; $d = 0.25, 0.50, 1.00$.

- Medium: 360 instances. 30 with each of the 12 combinations of the following parameters: $m = n = 1000, 2000$; $r = 0.05 \times n, 0.1 \times n$; $d = 0.25, 0.50, 1.00$.
- Large: 12 instances. 3 with each of the 4 combinations of the following parameters: $m = 5000, 10000$; $n = 1000$; $r = 0.05 \times n, 0.1 \times n$; $d = 1.00$.

The numerical experiments with each category had different purposes. The tests with the 'Small' instances have the main purpose of checking how tight are the bounds presented in Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021), for the norms of the constructed matrices H. We note that this analysis requires the solution of the LP P_1. These are not easy LPs because they are rather dense. The tests with the 'Medium' instances have the main purpose of comparing the different local searches and initialization procedures that we have proposed. Finally, the tests with the 'Large' instances have the main purpose of demonstrating the scalability of our methodology.

3.2.2 Selecting an initial block for the local search

Our algorithm to construct the initial $r \times r$ non-singular submatrix of A for our local searches is called 'NSub', where NSub stands for non-singular submatrix. It comprises the Phase-One and Greedy algorithms described below.

In the Phase-One algorithm (see Algorithm 1), we consider an $r \times n$ matrix $\tilde{I}_\delta$ with all elements equal to zero except the elements $[i, T(i)]$, for all $i = 1, \ldots, r$, where T is a randomly selected set of r indices from $\{1, \ldots, n\}$. The nonzero elements of $\tilde{I}_\delta$ are all equal to δ, a parameter initially set to 1. Then, we define $\tilde{A} := \begin{bmatrix} \tilde{I}_\delta \\ A \end{bmatrix}$, and

Algorithm 1: Algorithm Phase-One.

Input: $A \in \mathbb{R}^{m \times n}$, such that rank$(A)$= r.
Output: $S \subset M := \{1, \ldots, m\}$, such that rank$(A[S, :])$= $|S| \leq r$, and $T \subset N := \{1, \ldots, n\}$, such that $|T| = r$.

$\delta := 1$; $\tilde{I}_\delta := 0_{r \times n}$;
randomly select $T \subset \{1, \ldots, n\}$, such that $|T| = r$;
$S := \{1, \ldots, r\}$;
while $\delta > 10^{-4}$ & $S \cap \{1, \ldots, r\} \neq \emptyset$ **do**
    $\tilde{I}_\delta[i, T(i)] := \delta$, for $i = 1, \ldots, r$;
    $\tilde{A} := \begin{bmatrix} \tilde{I}_\delta \\ A \end{bmatrix}$;
    $[S] :=$ FI(det)($\tilde{A}[:, T]^\mathsf{T}$, $\tilde{A}[S, T]^\mathsf{T}$) (FI(det) is presented in Alg. 7);
    $\delta := \delta/10$;
$S := S \setminus \{1, \ldots, r\}$; $S(i) := S(i) - r$, $\forall i$;

iteratively apply the local search 'FI(det)' (presented in Algorithm 7), to obtain a set S of linearly-independent rows of $\tilde{A}[:, T]$, aiming at increasing the absolute value of the determinant of the submatrix $\tilde{A}[S, T]$. The local search is initialized at every

iteration with the transpose of an updated $r \times r$ non-singular submatrix $\tilde{A}[S,T]$. At the first iteration, we set $S = \{1,\ldots,r\}$, so $\tilde{A}[S,T]$ is the identity matrix. At each subsequent iteration, S is updated with the solution of the local search, and the indices of S still in $\{1,\ldots,r\}$ are made less attractive to be in the next solution by decreasing δ by a constant factor. The Phase-One algorithm stops when δ becomes 10^{-4} or when all the indices in S are greater than r.

We execute the Phase-One algorithm up to a maximum number of times. Each time, a set of r columns T of $\tilde{A}$ is randomly selected, and a set of r rows S is obtained. We then separate the indices from S that correspond to rows of the matrix A, i.e., we set $S := S \setminus \{1,\ldots,r\}$ and $S(i) := S(i) - r,\ \forall i$. We finally check if $|S| = r$; if so, we stop executing the algorithm and output $A[S,T]$ as the $r \times r$ non-singular submatrix of A.

We note that if all sets T randomly selected in the executions of the Phase-One algorithm correspond to linearly-dependent columns of A, the final set S will certainly contain less than r indices. In this case, we select from all the sets S obtained in the executions of the Phase-One algorithm, the one with largest cardinality and starting from it, we successively execute the Greedy algorithm (see Algorithm 2). At each execution, we iteratively add row indices to S. Each row selected is the

Algorithm 2: Algorithm Greedy.

Input: $A \in \mathbb{R}^{m\times n}$, such that $\text{rank}(A) = r$, $\tau > 0$, $S \subset M := \{1,\ldots,m\}$, such that $\text{rank}(A[S,:]) = |S| < r$.
Output: $S \subset M$, such that $\text{rank}(A[S,:]) = |S| \leq r$.

```
while |S| < r do
    Choose the least i ∈ M \ S such that σ_min(A[S ∪ {i}, :]) > τ;
    S := S ∪ {i};
```

first (with the least index), which keeps the minimum singular value of the partially constructed submatrix $A(S,:)$ greater than a given tolerance τ. The iterations are repeated until no row is obtained or $|S| = r$. If $|S| < r$, τ is decreased by a constant factor and the algorithm is executed once more. We note that as A has rank r, the convergence of this procedure is assured. We start the executions of the Greedy algorithm with τ slightly smaller than the minimum singular value of $A(S,:)$ if the initial set S is nonempty, or with $\tau = 1$ otherwise.

Finally, once we obtain the set S with r linearly-independent rows of A with the Greedy algorithm, the last step of the NSub algorithm consists of obtaining r linearly-independent columns of A. For that we rerun Algorithm 1 (and also Algorithm 2, if necessary), but considering $A[S,:]^\intercal$ as the input matrix. In this case, as the input matrix has only r columns, the set T in Algorithm 1 is given by the indices of all columns, instead of being randomly selected. Therefore, the Phase-One algorithm is applied only once.

The NSub algorithm is depicted in Algorithm 3. Our Matlab implementation of NSub is now available at https://www.mathworks.com/matlabcentral/fileexchange/83638-linear-independent-rows-and-columns-generator and can be used to either obtain r linearly-independent rows of a given matrix A with rank not smaller than r, or to compute an $r \times r$ non-singular submatrix of A.

Algorithm 3: Algorithm NSub.

Input: $A \in \mathbb{R}^{m \times n}$, such that rank$(A)= r$, and $k_{\max} > 0$
Output: $S \subset M := \{1, \ldots, m\}$, $T \subset N := \{1, \ldots, n\}$, such that $|S| = |T| = r$, and $A[S, T]$ is non-singular.
$S^1 := \emptyset$; $k := 1$;
while $|S^k| < r$ & $k < k_{\max}$ **do**
    $[S^{k+1}, T] :=$ Algorithm Phase-One(A);
    $k := k + 1$;
$S :=$argmax$_k\{|S^k|\}$;
if $|S| < r$ **then**
    if $|S| > 0$ **then**
        $\tau := \sigma_{\min}(A[S, :])/10$;
    else
        $\tau := 1$;
    while $|S| < r$ **do**
        $[S] :=$Algorithm Greedy(A, S);
        $\tau := \tau/10$;
$[\bar{S}, \bar{T}]:=$Algorithm Phase-One$(A[S, :]^\intercal)$;
$T := \bar{S}$;
if $|T| > 0$ **then**
    $\tau := \sigma_{\min}(A[S, T])/10$;
else
    $\tau := 1$
while $|T| < r$ **do**
    $[T] :=$Algorithm Greedy$(\ A[S, :]^\intercal, T)$;
    $\tau := \tau/10$;

We note that we could directly apply the Greedy algorithm to compute the initial non-singular submatrix for the local searches, without calling the Phase-One algorithm. However, in our numerical experiments, we significantly improved the performance of NSub, when calling Phase-One as depicted in Algorithm 3. We also observe that our best numerical results were obtained by applying the NSub algorithm to A when $m \geq n$, and to $A^\intercal$, otherwise. In other words, when $m < n$, we initially choose the linearly-independent columns of A.

Given the submatrix computed by NSub, we perform a local search with the goal of reducing the 1-norm of the matrix H, by replacing rows and columns of the submatrix, as explained in the next subsection.

3.2.3 The local-search procedures

In Algorithm 4 and 5, we present the local search procedures that consider as the

Algorithm 4: 'FI(det)' ('FI$^+$(det)') for generalized inverses.

Input: $A \in \mathbb{R}^{m\times n}$, with $r :=$rank(A),
$S \subset M := \{1,\ldots,m\}$, $T \subset N := \{1,\ldots,n\}$, such that $|S| = |T| = r$, and $A[S,T]$ is non-singular.
Output: possibly updated sets S, T.

1 $\tilde{A} := A[S,T]$;
2 $\bar{M} := M \setminus S$, $\bar{N} := N \setminus T$, $R := A[\bar{M},T]$, $C := A[S,\bar{N}]$;
3 $[L,U] := LU(\tilde{A})$ (Compute the LU factorization of $\tilde{A}$);
4 $cont = true$;
5 **while** *(cont)* **do**
6 $cont = false$;
7 **for** $\ell = 1,\ldots,n-r$ **do**
8 Solve $Ly = C[S,\ell]$, with solution $\hat{y}$;
9 Solve $U\alpha = \hat{y}$, with solution $\hat{\alpha}$;
10 **if** $|\hat{\alpha}| \not\leq (1,\ldots,1)^\mathsf{T}$ **then**
11 $\hat{\jmath} := \min\{j : |\hat{\alpha}_j| > 1\}$ for 'FI(det)', or $\hat{\jmath} := \text{argmax}_j\{|\hat{\alpha}_j|\}$ for 'FI$^+$(det)';
12 $aux := \tilde{A}[S,\hat{\jmath}]$;
13 $\tilde{A}[S,\hat{\jmath}] := C[S,\ell]$;
14 $C[S,\ell] := aux$;
15 $T := T \cup \{\bar{N}(\ell)\} \setminus \{T(\hat{\jmath})\}$;
16 $\bar{N} := \bar{N} \setminus \{\bar{N}(\ell)\} \cup \{T(\hat{\jmath})\}$;
17 $[L,U] := LU(\tilde{A})$ (update LU factorization of previous iteration);
18 $cont = true$;
19 **for** $\ell = 1,\ldots,m-r$ **do**
20 Solve $U^\mathsf{T} y = R[\ell,T]^\mathsf{T}$, with solution $\hat{y}$;
21 Solve $L^\mathsf{T}\alpha = \hat{y}$, with solution $\hat{\alpha}$;
22 **if** $|\hat{\alpha}| \not\leq (1,\ldots,1)^\mathsf{T}$ **then**
23 $\hat{\jmath} := \min\{j : |\hat{\alpha}_j| > 1\}$ for 'FI(det)', or $\hat{\jmath} := \text{argmax}_j\{|\hat{\alpha}_j|\}$ for 'FI$^+$(det)' ;
24 $aux := \tilde{A}[\hat{\jmath},T]$;
25 $\tilde{A}[\hat{\jmath},T] := R[\ell,T]$;
26 $R[\ell,T] := aux$;
27 $S := S \cup \{\bar{M}(\ell)\} \setminus \{S(\hat{\jmath})\}$;
28 $\bar{M} := \bar{M} \setminus \{\bar{M}(\ell)\} \cup \{S(\hat{\jmath})\}$;
29 $[L,U] := LU(\tilde{A})$ (update LU factorization of previous iteration);
30 $cont = true$;

criterion for improvement of the given solution, the increase in the absolute value of the determinant of the $r \times r$ non-singular submatrix of A.

Based on Theorem 15, for a given rank-r matrix A, the procedure starts from a set S of r rows and a set T of r columns of A, such that $A[S,T]$ is non-singular.

In the first loop of Algorithm 4 (lines 7–18), a column of $A[S, N \setminus T]$ replaces a column of $A[S,T]$ if the absolute value of the determinant increases with the replacement. To evaluate how much the determinant changes when each column of $A[S,T]$ is replaced by a given column γ of $A[S, N \setminus T]$, we use the result in Remark 16 (i.e., using Cramer's Rule).

Remark 16. *Let $\gamma \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$ with $\det(A) \neq 0$. Let $A_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column of A by γ. If $\hat{\alpha} \in \mathbb{R}^n$ solves the linear system of equations $A\alpha = \gamma$, then we have $\det(A_{\gamma/j}) = \hat{\alpha}_j \times \det(A)$.*

Similarly, in the second loop of the algorithm (lines 19–30), a row of $A[M \setminus S, T]$ replaces a row of $A[S, T]$ if the absolute value of the determinant increases with the replacement. In this case, to evaluate how much the determinant changes when each row of $A[S, T]$ is replaced by a given row γ of $A[M \setminus S, T]$, we use the equivalent result in Remark 17.

Remark 17. *Let $\gamma \in \mathbb{R}^{1\times n}$ and $A \in \mathbb{R}^{n\times n}$ with $\det(A) \neq 0$. Let $A_{\gamma/i}$ be the matrix obtained by replacing the i^{th} row of A by γ. If $\hat{\alpha} \in \mathbb{R}^{1\times n}$ solves the linear system of equations $\alpha A = \gamma$, then we have $\det(A_{\gamma/i}) = \hat{\alpha}_i \times \det(A)$.*

Use of Cramer's rule greatly improves the performance of these local searches.

Two algorithms, 'FI(det)' and 'FI$^+$(det)', are presented in Algorithm 4. The only differences between them are shown in lines 11 and 23. For 'FI$^+$(det)' ("first improvement plus"), we iteratively select a column (row) that is not in $A[S, T]$ and exchange it with the column (row) of $A[S, T]$ that leads to the greatest increase in the absolute value of the determinant of the submatrix. For 'FI(det)' ("first improvement"), the column (row) of $A[S, T]$ selected for the replacement is the one of least index, that leads to an increase in the absolute value of determinant.

We also present in Algorithm 5, the algorithm 'BI(det)' ("best improvement"), where the pair of rows or columns exchanged at each iteration is selected as the pair that leads to the greatest increase in the absolute value of the determinant, among all possibilities.

Algorithms 'FI(det)', 'FI$^+$(det)', and 'BI(det)' stop when no replacement of a row or column of $A[S, T]$ would lead to an increase in the absolute value of the determinant, i.e., when we reach a local maximizer for the absolute value of the determinant, according to Definition 14.

Algorithm 6 represents the local search 'FI(norm)'. In this case, we consider as the criterion for improvement of the given solution, the decrease in the 1-norm of H, or equivalently, the decrease in the 1-norm of the inverse of the $r \times r$ non-singular submatrix of A being considered. To evaluate how much the 1-norm of the inverse of the submatrix changes when each column (row) of $A[S, T]$ is replaced by a given column (row) γ of $A[S, N \setminus T]$ ($A[M \setminus S, T]$), we use the result in Remark 18.

Algorithm 5: 'BI(det) for generalized inverses'.

Input: $A \in \mathbb{R}^{m\times n}$, with $r :=$rank(A),
$S \subset M := \{1,\dots,m\}$, $T \subset N := \{1,\dots,n\}$, such that $|S| = |T| = r$, and $A[S,T]$ is non-singular.
Output: possibly updated sets S, T.

$\tilde{A} := A[S,T]$;
$\bar{M} := M \setminus S$, $\bar{N} := N \setminus T$, $R := A[\bar{M},T]$, $C := A[S,\bar{N}]$;
$[L,U] := LU(\tilde{A})$ (Compute the LU factorization of $\tilde{A}$);
$biggest.\alpha_r = biggest.\alpha_c := 1$;
$cont = true$;
while *(cont)* **do**
  $cont = false$;
  for $\ell = 1,\dots,n-r$ **do**
    Solve $Ly = C[S,\ell]$, with solution $\hat{y}$;
    Solve $U\alpha = \hat{y}$, with solution $\hat{\alpha}$;
    $\hat{\alpha}_{\max} := \max_j\{|\hat{\alpha}_j|\}$;
    if $\hat{\alpha}_{\max} > biggest.\alpha_r$ **then**
      $biggest.\alpha_r := \hat{\alpha}_{\max}$;
      $\hat{\jmath}_r := \text{argmax}_j\{|\hat{\alpha}_j|\}$;
      $\hat{\ell}_r := \ell$;
  for $\ell = 1,\dots,m-r$ **do**
    Solve $U^{\mathsf{T}}y = R[\ell,T]^{\mathsf{T}}$, with solution $\hat{y}$;
    Solve $L^{\mathsf{T}}\alpha = \hat{y}$, with solution $\hat{\alpha}$;
    $\hat{\alpha}_{\max} := \max_j\{|\hat{\alpha}_j|\}$;
    if $\hat{\alpha}_{\max} > biggest.\alpha_c$ **then**
      $biggest.\alpha_c := \hat{\alpha}_{\max}$;
      $\hat{\jmath}_c := \text{argmax}_j\{|\hat{\alpha}_j|\}$;
      $\hat{\ell}_c := \ell$;
  if $\max\{biggest.\alpha_r, biggest.\alpha_c\} > 1$ **then**
    $cont = true$;
    if $biggest.\alpha_r > biggest.\alpha_c$ **then**
      $aux := \tilde{A}[S,\hat{\jmath}_r]$;
      $\tilde{A}[S,\hat{\jmath}_r] := C[S,\hat{\ell}_r]$;
      $C[S,\hat{\ell}_r] := aux$;
      $T := T \cup \{\bar{N}(\hat{\ell}_r)\} \setminus \{T(\hat{\jmath}_r)\}$;
      $\bar{N} := \bar{N} \setminus \{\bar{N}(\hat{\ell}_c)\} \cup \{T(\hat{\jmath}_r)\}$;
      $[L,U] := LU(\tilde{A})$ (update LU factorization of previous iteration);
    else
      $aux := \tilde{A}[\hat{\jmath}_c,T]$;
      $\tilde{A}[\hat{\jmath}_c,T] := R[\hat{\ell}_c,T]$;
      $R[\hat{\ell}_c,T] := aux$;
      $S := S \cup \{\bar{M}(\ell)\} \setminus \{S(\hat{\jmath})\}$;
      $\bar{M} := \bar{M} \setminus \{\bar{M}(\hat{\ell}_c)\} \cup \{S(\hat{\jmath}_c)\}$;
      $[L,U] := LU(\tilde{A})$ (update LU factorization of previous iteration);

Algorithm 6: 'FI(norm)' for generalized inverses.

Input: $A \in \mathbb{R}^{m\times n}$, with $r :=\text{rank}(A)$,
$S \subset M := \{1,\dots,m\}$, $T \subset N := \{1,\dots,n\}$, such that $|S| = |T| = r$, and $A[S,T]$ is non-singular.

Output: possibly updated sets S, T.

$\tilde{A} := A[S,T]$;
$\bar{M} := M \setminus S$, $\bar{N} := N \setminus T$, $R := A[\bar{M},T]$, $C := A[S,\bar{N}]$;
$cont = true$;
while *(cont)* **do**
  $cont = false$;
  $B = A[S,T]$;
  $Binv := B^{-1}$;
  $nBinv = \|Binv\|_1$;
  for $\ell = 1,\dots,n-r$ **do**
    $\gamma := A[S,\bar{N}(\ell)]$;
    for $j = 1,\dots,r$ **do**
      Let $B_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column of B by γ;
      $nBinv^+ := \|B^{-1}_{\gamma/j}\|_1$ (Computed with the result in Remark 18);
      if $nBinv^+ < nBinv$ **then**
        $B := B_{\gamma/j}$;
        $Binv := B^{-1}_{\gamma/j}$ (Computed with the result in Remark 18);
        $nBinv = nBinv^+$;
        $T := T \cup \{\bar{N}(\ell)\} \setminus \{T(j)\}$;
        $\bar{N} := N \setminus T$;
        $cont = true$;
        break ;
  $B = A[S,T]^\intercal$;
  $Binv := B^{-1}$;
  $nBinv = \|Binv\|_1$;
  for $\ell = 1,\dots,m-r$ **do**
    $\gamma := A[\bar{M}(\ell),T]^\intercal$;
    for $j = 1,\dots,r$ **do**
      Let $B_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column of B by γ;
      $nBinv^+ := \|B^{-1}_{\gamma/j}\|_1$ (Computed with the result in Remark 18);
      if $nBinv^+ < nBinv$ **then**
        $B := B_{\gamma/j}$;
        $Binv := B^{-1}_{\gamma/j}$ (Computed with the result in Remark 18);
        $nBinv = nBinv^+$;
        $S := S \cup \{\bar{M}(\ell)\} \setminus \{S(j)\}$;
        $\bar{M} := M \setminus S$;
        $cont = true$;
        break;

Remark 18. *Let $\gamma \in \mathbb{R}^n$ and $A := (a_1, \ldots, a_j, \ldots, a_n) \in \mathbb{R}^{n\times n}$ with $\det(A) \neq 0$. Let $A_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column of A by γ, and $v = (v_1, \ldots, v_j, \ldots, v_n)^\intercal := A^{-1}\gamma$. If $v_j \neq 0$, define*

$$\bar{v} := \left(-\frac{v_1}{v_j}, \ldots, -\frac{v_{j-1}}{v_j}, \ \frac{1}{v_j}, \ -\frac{v_{j+1}}{v_j}, \ldots, -\frac{v_n}{v_j}\right)^\intercal.$$

Then

$$A^{-1}_{\gamma/j} = \Theta \ A^{-1},$$

where

$$\Theta = (e_1, \ldots, e_{j-1}, \ \bar{v}, \ e_{j+1}, \ldots, e_n),$$

and e_i are the standard unit vectors.

Use of Remark 18 greatly improves the performance of these local searches.

Remark 19. *In Algorithms 4 and 5 (and later in Algorithms 7 and 8), we need to update LU factorizations of an $r \times r$ matrix B under low-rank changes. Practical and numerically-stable algorithms for LU factorizations employ "partial or complete pivoting", and Matlab provides this functionality, calculating such factorizations in $\mathcal{O}(r^3)$ floating-point operations (in the dense case). But Matlab does not have functionality for efficiently updating these factorizations, while in theory they can be updated in $\mathcal{O}(r^2)$ floating-point operations (in the dense case); see for example, Elble and Sahinidis (2012) or Gondzio (1992)). In principle, we do advocate a proper updating approach, but we computed our new LU factorizations (with partial pivoting) from scratch each time for two reasons: (i) the updating procedures are not available in Matlab, and (ii) our algorithms turn out to be very fast even without performing fast LU updates.*

3.2.4 Numerical results

We initially consider the experiments done with the 90 instances in the 'Small' category, which had the main purpose of analyzing the ratios between the 1-norm of the matrices H computed by the three local searches based on the determinant, with the minimum 1-norm of a generalized inverse given by the solution of the LP problem P_1 ($\|H\|_1/z_{P_1}$). We aim at checking how close these ratios are from the upper bound given by Theorem 15.

In Figure 3.1, we present the average ratios for the matrices with the same dimension, rank, and density. From Theorem 15, we know that these ratios cannot be greater than r^2, and we see from the results, that for the matrices considered in our tests, we stay quite far from this upper bound (even though the upper bound

is the best possible). In general, the ratios increase with the rank r, the dimension $m = n$, and the density d of the matrices, but even for $r = 50$, we obtain ratios less than 2. So, our conclusion is that the worst-case bound, while best possible, is extremely pessimistic.

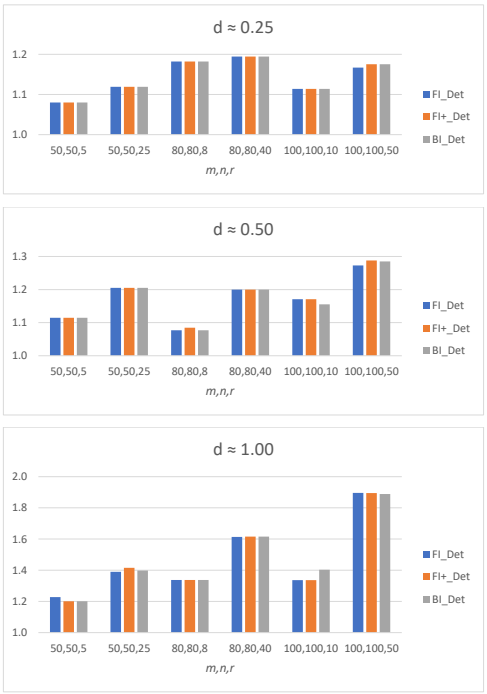


Figure 3.1: $\|H\|_1/z_{P_1}$ (Small) (generalized inverse)

In Table 3.1, besides presenting the average ratios depicted in Figure 3.1, we also present the average running time to compute the generalized inverses. In case of the local searches, the total time to compute the generalized inverse is given by the sum of the time to generate the initial matrix H by the NSub algorithm (Algorithm 3), and the time of the local search (FI(det), FI$^+$(det), or BI(det)).

	$\|H\|_1/z_{P_1}$			Time(sec)				
m, r, d	FI(det)	FI$^+$(det)	BI(det)	P_1	NSub	FI(det)	FI$^+$(det)	BI(det)
50,05,0.25	1.080	1.080	1.080	0.495	0.091	0.004	0.003	0.005
50,25,0.25	1.119	1.119	1.119	4.711	0.094	0.003	0.002	0.004
80,08,0.25	1.182	1.182	1.182	1.734	0.077	0.003	0.003	0.006
80,40,0.25	1.195	1.195	1.195	27.284	0.149	0.009	0.006	0.020
100,10,0.25	1.114	1.114	1.114	3.297	0.090	0.004	0.003	0.011
100,50,0.25	1.167	1.175	1.175	65.156	0.266	0.005	0.004	0.016
50,05,0.50	1.115	1.115	1.115	0.489	0.065	0.004	0.004	0.005
50,25,0.50	1.205	1.205	1.205	4.995	0.084	0.004	0.003	0.007
80,08,0.50	1.077	1.085	1.077	1.705	0.068	0.005	0.004	0.008
80,40,0.50	1.200	1.200	1.200	29.604	0.131	0.006	0.004	0.013
100,10,0.50	1.171	1.171	1.155	3.806	0.094	0.005	0.004	0.010
100,50,0.50	1.273	1.288	1.285	69.810	0.288	0.007	0.004	0.021
50,05,1.00	1.228	1.201	1.201	0.687	0.003	0.007	0.006	0.010
50,25,1.00	1.390	1.416	1.399	6.691	0.036	0.008	0.005	0.014
80,08,1.00	1.337	1.337	1.337	2.391	0.018	0.005	0.004	0.011
80,40,1.00	1.614	1.615	1.615	43.598	0.088	0.010	0.006	0.029
100,10,1.00	1.337	1.337	1.403	4.792	0.020	0.006	0.006	0.018
100,50,1.00	1.896	1.895	1.889	124.856	0.139	0.030	0.017	0.089

Table 3.1: Local Searches for generalized inverse vs. P_1

We see from Figure 3.1 and Table 3.1, that the three local searches converge to solutions of similar quality on most of the experiments.

We observe in Table 3.1 that the running times to solve the LP P_1 increase quickly with the dimension of the matrix, and are much higher than the times for the local searches. Therefore, we can already see that the LP P_1 is not useful as a computational alternative to our local searches when we consider larger instances (and additionally, as we have mentioned, the solutions produced by the LP do not have the reflexive property, nor are they nicely block structured).

Next, we consider the experiments done with the 360 instances in the 'Medium' category, which had the main purpose of comparing the different local searches proposed. We present in Table 3.2 average results for each group of 30 instances with the same configuration, described in the first column. In the next three columns we present statistics for the local searches based on the determinant, which are initialized with the solutions given by the NSub algorithm, and in the last three columns we consider the application of the local searches based on the 1-norm of H, which are initialized with the solutions given by the three first local searches. In the first half of the table, we show the mean and standard deviation of the relative difference between the 1-norm of the matrix H obtained by each local search and the minimum value among all of them, denoted by $||H_{best}||_1$. H_{best} is naturally obtained with the application of one of the local searches based on the 1-norm.

m, r, d	FI(det)	FI$^+$(det)	BI(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	BI(det) FI(norm)
	$(\|H\|_1 - \|H_{best}\|_1)/\|H_{best}\|_1$					
1000,050,0.25	0.073/0.034	0.072/0.034	0.071/0.036	0.000/0.000	0.000/0.000	0.000/0.000
1000,050,0.50	0.098/0.031	0.100/0.030	0.097/0.030	0.000/0.001	0.000/0.001	0.001/0.002
1000,050,1.00	0.086/0.026	0.086/0.026	0.086/0.026	0.001/0.004	0.001/0.004	0.000/0.002
1000,100,0.25	0.134/0.024	0.132/0.025	0.133/0.026	0.000/0.001	0.000/0.001	0.001/0.002
1000,100,0.50	0.089/0.047	0.088/0.047	0.088/0.048	0.000/0.003	0.000/0.000	0.000/0.000
1000,100,1.00	0.132/0.042	0.131/0.046	0.131/0.047	0.002/0.004	0.002/0.004	0.001/0.003
2000,100,0.25	0.116/0.035	0.115/0.035	0.114/0.036	0.000/0.002	0.000/0.001	0.000/0.000
2000,100,0.50	0.171/0.036	0.170/0.035	0.171/0.034	0.001/0.002	0.001/0.003	0.001/0.003
2000,100,1.00	0.128/0.054	0.130/0.057	0.129/0.054	0.002/0.006	0.002/0.006	0.002/0.005
2000,200,0.25	0.202/0.050	0.210/0.065	0.213/0.060	0.005/0.010	0.004/0.005	0.007/0.011
2000,200,0.50	0.160/0.046	0.161/0.045	0.161/0.044	0.001/0.003	0.001/0.004	0.001/0.003
2000,200,1.00	0.217/0.046	0.219/0.047	0.220/0.048	0.003/0.007	0.002/0.005	0.002/0.005
	Time(sec)					
1000,050,0.25	0.116/0.035	0.109/0.032	0.469/0.116	1.888/0.426	1.899/0.439	1.843/0.381
1000,050,0.50	0.267/0.041	0.241/0.050	1.111/0.337	13.806/2.242	14.163/2.440	14.097/2.442
1000,050,1.00	0.372/0.061	0.348/0.091	1.755/0.268	29.225/5.751	28.888/5.567	28.990/5.817
1000,100,0.25	1.320/0.293	0.975/0.273	8.871/3.565	317.976/59.536	314.073/60.679	319.427/52.886
1000,100,0.50	0.104/0.032	0.110/0.030	0.369/0.133	1.995/0.322	1.983/0.336	1.994/0.330
1000,100,1.00	0.246/0.041	0.236/0.064	1.112/0.529	14.737/2.585	14.865/2.729	14.697/2.646
2000,100,0.25	0.337/0.054	0.318/0.093	1.459/0.390	30.938/5.014	30.503/5.153	30.220/5.091
2000,100,0.50	1.301/0.258	1.070/0.206	8.537/5.039	312.795/60.716	323.046/67.272	320.428/67.722
2000,100,1.00	0.113/0.043	0.132/0.042	0.771/0.536	2.303/0.562	2.340/0.620	2.281/0.608
2000,200,0.25	0.279/0.083	0.258/0.063	2.215/1.311	20.197/4.851	21.013/4.596	20.791/4.553
2000,200,0.50	0.290/0.105	0.319/0.088	1.945/2.843	38.609/7.940	38.806/8.670	38.700/8.634
2000,200,1.00	1.153/0.454	0.912/0.320	10.513/13.156	395.193/62.373	416.49/88.601	398.360/67.740

Table 3.2: Local Searches for generalized inverse (Mean/Std Dev) (Medium)

Comparing to the tests with the 'Small' instances, we see that on this larger group of instances of higher dimension, the solutions obtained by the three local searches based on the determinant are still of similar quality. Consequently, the solutions obtained by the searches based on the 1-norm are also of similar quality. By applying these 1-norm searches, we are able to improve the solutions from

determinant searches by approximately 7 to 22%. Furthermore, we see that this improvement comes with a high computational cost. The necessity of computing the inverse of the $r \times r$ submatrix of A at each iteration, significantly increases the time of these searches. Even though we use the result in Remark 18 to accelerate this computation, it still makes the norm searches slower than the determinant searches. We finally note that the 1-norm searches are able to improve more the solutions from the determinant searches as the rank and the dimension of the matrix increase.

In Table 3.3, we present the number of swaps for each local search. Combining these results with the running time of the procedures, we conclude that, despite the fact that the best improvement is commonly pointed as a good criterion for local searches in the literature, in our case, BI(det) could be discarded. Comparing it to the other determinant searches, we see that, although it converges to solutions of similar quality performing fewer swaps, it is much more time consuming. Comparing the two other searches based on the determinant, FI$^+$(det) preforms slightly better, with a smaller number of swaps and the average computational time a bit smaller. We can also observe the high cost of the swaps performed by the searches based on the 1-norm. These observations are pointed out in Figure 3.2, where we show the relation between the average relative 1-norm difference to the minimum norm obtained by all searches and the average running time of the local searches for the larger instances in the 'Medium' category. The hollow circle indicates that the local search had worse average result and longer average running time than another procedure, and therefore, should not be adopted. We note that we use a logarithmic scale for the running time. Once again we see some improvement given by the norm searches, but with a very high computational cost.

m, r, d	FI(det)	FI$^+$(det)	BI(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	BI(det) FI(norm)
1000,050,0.25	27.833	23.367	12.800	14.300	14.000	13.900
1000,050,0.50	46.300	36.467	20.433	31.600	32.033	31.400
1000,050,1.00	42.333	35.367	18.333	31.867	31.900	32.000
1000,100,0.25	79.767	60.233	33.467	75.433	74.600	74.833
1000,100,0.50	20.133	16.933	8.867	17.033	16.933	16.933
1000,100,1.00	41.900	32.367	20.633	37.600	37.767	37.600
2000,100,0.25	32.200	26.933	13.600	40.300	40.267	39.967
2000,100,0.50	67.600	54.300	34.867	87.933	88.033	87.967
2000,100,1.00	116.667	73.633	26.467	23.767	23.967	24.267
2000,200,0.25	180.833	104.067	50.933	60.800	62.033	61.233
2000,200,0.50	103.700	59.700	21.200	57.067	57.167	56.600
2000,200,1.00	139.700	81.300	43.400	116.267	116.767	116.833

Table 3.3: Number of swaps (Medium) (generalized inverse)

Finally, our last experiments intend to show the scalability of the procedures, considering matrices with up to 10000 rows, 1000 columns, rank up to 100, and density equal to one. As the BI(det) procedure was not successful in the previous test, we did not run it on the 'Large' category. In Table 3.4 we see that the average relative differences between the 1-norm of the matrix H obtained by each local search based on the determinant and the minimum 1-norm among the four local searches are approximately between 9 and 15%. So, comparing to best solutions found, the

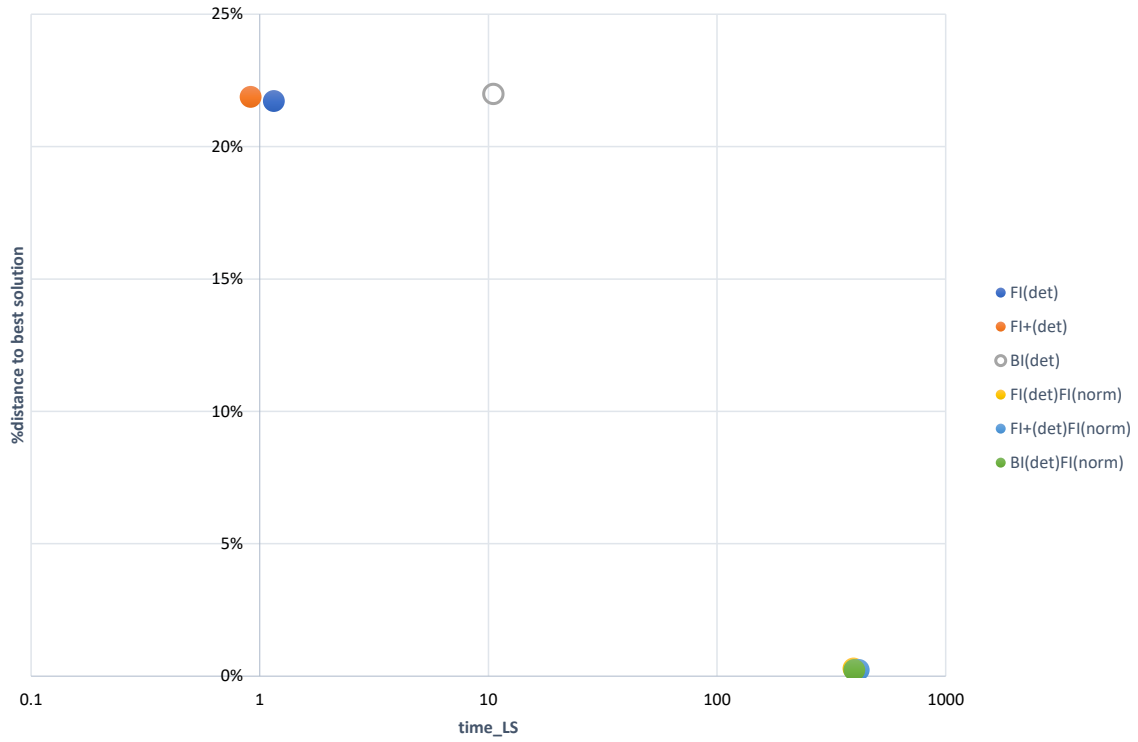


Figure 3.2: Local searches ($m = 2000$, $r = 200$, $d = 1$) (generalized inverse)

searches based on the determinant keep the same quality observed on the smaller instances. Furthermore, these larger matrices are obtained in less than 5 seconds on average. The NSub algorithm had a very good performance on these large instances, being able to compute the initial non-singular submatrices for the local searches in less than 2 seconds on average.

	$(\|H\|_1 - \|H_{best}\|_1)/\|H_{best}\|_1$				
m, n, r	FI(det)	FI$^+$(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	
5000,1000,050	0.129	0.152	0.034	0.000	
5000,1000,100	0.098	0.143	0.006	0.038	
10000,1000,050	0.088	0.108	0.001	0.040	
10000,1000,100	0.104	0.090	0.000	0.015	
	Time(sec)				
m, n, r	FI(det)	FI$^+$(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	NSub
5000,1000,050	0.541	0.578	9.614	14.551	0.641
5000,1000,100	1.288	1.461	94.748	68.547	1.139
10000,1000,050	1.088	1.351	10.882	10.800	1.897
10000,1000,100	2.623	2.512	166.830	131.578	1.942

Table 3.4: Local searches (Large) (generalized inverse)

3.3 ah-symmetric generalized inverse

Recall the key use of an ah-symmetric generalized inverse: if H is an ah-symmetric generalized inverse of A, then $\hat{x} := Hb$ solves $\min\{\|Ax - b\|_2 : \ x \in \mathbb{R}^n\}$. The local-search procedures for the ah-symmetric reflexive generalized inverse are based on the block construction procedure presented in Xu, Fampa, Lee, and Ponte (2021). More specifically, on Theorem 5, Definition 7, and Theorem 9.

In the context of the least-square problem, such a "column block solution" amounts to choosing a set of r "explanatory variables" in the context of multicolinearity (i.e., dependent columns of A), which is highly desirable in terms of explainability. It remains to choose a good column block solution, by which we mean having entries under control (via approximate 1-norm minimization).

As before, the ϵ of Definition 7 and Theorem 9 is used in Xu, Fampa, Lee, and Ponte (2021) to gain polynomial running time in $1/\epsilon$. For the purpose of actual computations, our observation has been that ϵ can be chosen to be zero. We further note that in Xu, Fampa, Lee, and Ponte (2021), we demonstrated that the bound in Theorem 9 is the best possible. However, we will see in our experiments that the bound is overly pessimistic by a wide margin.

The idea of the algorithms considered in this section is to select an $m \times r$ rank-r submatrix of A, and construct an ah-symmetric reflexive generalized inverse of A with the M-P pseudoinverse of this submatrix, as described in Theorem 5.

Next, we give details of the algorithms and present numerical results. The test matrices used in the computational experiments are the same 462 matrices considered in the previous section, and the same $r \times r$ non-singular submatrices of A constructed by the NSub algorithm, were used to initialize the local-search procedures discussed in this section.

To analyze the local-search procedures proposed, we compare their solutions to the solutions of LP problem identified below as P_{123}. Its solutions corresponds to $\|H^{ah,r}_{opt}\|_1$. As defined in Theorem 9, $H^{ah,r}_{opt}$ is a 1-norm minimizing ah-symmetric *reflexive* generalized inverse of A. In order to formulate P_{123} as an LP problem, we linearize P2, using the following result.

From Proposition 3, we note that if H satisfies P1 and P3, then P2 can be reformulated as the linear equation

$$HAA^\dagger = H. \tag{3.1}$$

Considering (3.1), we then have

$$\begin{aligned} (P_{123})\ z_{P_{123}} := \quad & \min \quad \langle J, T\rangle \ , \\ & \text{s.t.:} \quad T - H \geq 0 \ , \\ & \qquad\ \ T + H \geq 0 \ , \\ & \qquad\ \ AHA = A \ , \\ & \qquad\ \ (AH)^\intercal = (AH) \ , \\ & \qquad\ \ HAA^\dagger = H \ . \end{aligned}$$

It is important to note that in the case of a generalized inverse, we could only compare the 1-norm quality of our solution to the optimal value of the LP P_1,

ignoring the reflexivity condition P2. Here, we can compare to the optimal value of th LP P_{123} because P2 can be linearized when P3 is imposed.

3.3.1 The local-search procedures

In Algorithms 7 and 8, we present the local-search procedures that consider as the criterion for improvement of the given solution, the increase in the absolute determinant of the current $r \times r$ non-singular submatrix of A. Based on Theorem 9, the procedures start from a set S of r rows and a set T of r columns of A, such that $A[S,T]$ is non-singular. We note that unlike what is done in Algorithms 4 and 5, in Algorithms 7 and 8 only columns of $A[S,T]$ are considered to be exchanged in order to increase the determinant. From the result in Theorem 9, we see that *any* set S of r linearly-independent rows of A could be used in the search for a local maximizer for the absolute determinant on the set of $r \times r$ non-singular submatrices of $A[S,:]$. The initial submatrix $A[S,T]$ is obtained by the NSub algorithm (Algorithm 3).

Algorithm 7: 'FI(det)' ('FI$^+$(det)') for ah-symmetric generalized inverses

Input: $A \in \mathbb{R}^{m\times n}$, with $r :=$rank(A),
$S \subset M := \{1,\ldots,m\}$, $T \subset N := \{1,\ldots,n\}$, such that $|S| = |T| = r$, and $A[S,T]$ is non-singular.
Output: possibly updated set T.

$\tilde{A} := A[S,T]$;
$\bar{N} := N \setminus T$;
$C := A[S,\bar{N}]$;
$[L,U] := LU(\tilde{A})$ (Compute the LU factorization of $\tilde{A}$);
$cont = true$;
while *(cont)* **do**
  $cont = false$;
  for $\ell = 1,\ldots,n-r$ **do**
    Solve $Ly = C[S,\ell]$, with solution $\hat{y}$;
    Solve $U\alpha = \hat{y}$, with solution $\hat{\alpha}$;
    $[\hat{\alpha}^{\ell}_{\max}, \hat{\jmath}] := \max_i\{|\hat{\alpha}_i|\}$;
    if $|\hat{\alpha}| \not\leq (1,\ldots,1)^{\mathsf{T}}$ **then**
      $\hat{\jmath} := \min\{j : |\hat{\alpha}_j| > 1\}$ for 'FI(det)', or $\hat{\jmath} := \text{argmax}_j\{|\hat{\alpha}_j|\}$ for 'FI$^+$(det)';
      $aux := \tilde{A}[S,\hat{\jmath}]$;
      $\tilde{A}[S,\hat{\jmath}] := C[S,\ell]$;
      $C[S,\ell] := aux$;
      $T := T \cup \{\bar{N}(\ell)\} \setminus \{T(\hat{\jmath})\}$;
      $\bar{N} := \bar{N} \setminus \{\bar{N}(\ell)\} \cup \{T(\hat{\jmath})\}$;
      $[L,U] := LU(\tilde{A})$ (update LU factorization of previous iteration);
      $cont = true$;

We present in Algorithm 9 the 'FI(norm)' for ah-symmetric generalized inverses. The algorithm is obtained by excluding from Algorithm 6, the loop where row exchanges are performed, and also by replacing the inverses of the matrices B and $B_{\gamma/j}$, with their pseudoinverses.

Algorithm 8: 'BI(det)' for ah-symmetric generalized inverses

Input: $A \in \mathbb{R}^{m\times n}$, with $r :=\text{rank}(A)$,
$S \subset M := \{1,\dots,m\}$, $T \subset N := \{1,\dots,n\}$, such that $|S| = |T| = r$, and $A[S,T]$ is non-singular.

Output: possibly updated set T.

$\tilde{A} := A[S,T]$;
$\bar{N} := N \setminus T$;
$C := A[S,\bar{N}]$;
$[L,U] := LU(\tilde{A})$ (Compute the LU factorization of $\tilde{A}$);
$biggest.\alpha_r := 1$;
$cont := true$;
while *(cont)* **do**
    $cont := false$;
    for $\ell = 1,\dots,n-r$ **do**
        Solve $Ly = C[S,\ell]$, with solution $\hat{y}$;
        Solve $U\alpha = \hat{y}$, with solution $\hat{\alpha}$;
        $\hat{\alpha}_{\max} := \max_j\{|\hat{\alpha}_j|\}$;
        if $\hat{\alpha}_{\max} > biggest.\alpha_r$ **then**
            $biggest.\alpha_r := \hat{\alpha}_{\max}$;
            $\hat{\jmath}_r := \text{argmax}_j\{|\hat{\alpha}_j|\}$;
            $\hat{\ell}_r := \ell$;
    if $biggest.\alpha_r > 1$ **then**
        $cont := true$;
        $aux := \tilde{A}[S,\hat{\jmath}_r]$;
        $\tilde{A}[S,\hat{\jmath}_r] := C[S,\hat{\ell}_r]$;
        $C[S,\hat{\ell}_r] := aux$;
        $T := T \cup \{\bar{N}(\hat{\ell}_r)\} \setminus \{T(\hat{\jmath}_r)\}$;
        $\bar{N} := \bar{N} \setminus \{\bar{N}(\hat{\ell}_c)\} \cup \{T(\hat{\jmath}_r)\}$;
        $[L,U] := LU(\tilde{A})$ (update LU factorization of previous iteration);

To evaluate how much the 1-norm of the M-P pseudoinverse of the submatrix changes when each column of $A[:, T]$ is replaced by a given column γ of $A[:, N \setminus T]$, we use the result in Remark 20.

Remark 20. *(see, for example, Meyer (1973)) Let $\gamma = Av \in \mathbb{R}^m$ and $A := (a_1, \ldots, a_j, \ldots, a_r) \in \mathbb{R}^{m \times r}$ with $\text{rank}(A) = r$. Let $A_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column of A by γ. If $v_j \neq 0$, define*

$$\bar{v} := \left(-\frac{v_1}{v_j}, \ldots, -\frac{v_{j-1}}{v_j}, \ \frac{1}{v_j}, \ -\frac{v_{j+1}}{v_j}, \ldots, -\frac{v_r}{v_j} \right)^{\intercal}.$$

Then

$$A^{\dagger}_{\gamma/j} = \Theta \ A^{\dagger},$$

where

$$\Theta = (e_1, \ldots, e_{j-1}, \ \bar{v}, \ e_{j+1}, \ldots, e_r),$$

and e_i are the standard unit vectors.

Proof. Notice that $A_{\gamma/j} = A\Theta^{-1}$. We could verify that

$$A^{\dagger}_{\gamma/j} A_{\gamma/j} = I_r, \ A_{\gamma/j} A^{\dagger}_{\gamma/j} = AA^{\dagger},$$

which implies $A^{\dagger}_{\gamma/j}$ is the M-P pseudoinverse of $A_{\gamma/j}$. □ □

3.3.2 Numerical results

Similarly to the previous section, we initially consider the experiments done with the 90 instances in the 'Small' category, with the purpose of analyzing the ratios between the the 1-norm of the matrices H computed by the three local searches based on the determinant, with the minimum 1-norm of a ah-symmetric generalized inverse given by the solution of the LP problem P_{123} ($\|H\|_1 / z_{P_{123}}$). We now aim at checking how close these ratios are from the upper bound given by Theorem 9.

In Figure 3.3, we present the average ratios for the matrices with the same dimension, rank, and density. From Theorem 9, we know that these ratios cannot be greater than r, and we also see from the results, that for the matrices considered in our tests, we stay quite far from this upper bound. In general, the ratios increase with the rank r, the dimension $m = n$, and the density d of the matrices, but even for $r = 50$, we obtain ratios smaller than 1.5.

In Table 3.5, besides presenting the average ratios depicted in Figure 3.3, we also present the average running time to compute the generalized inverses. In case of the local searches, the total time to compute the generalized inverse is given by the sum

Algorithm 9: 'FI(norm)' for ah-symmetric generalized inverses.

Input: $A \in \mathbb{R}^{m\times n}$, with $r :=$rank(A),
$T \subset N := \{1,\ldots,n\}$, such that $|T| = r$, and $A[:,T]$ has rank r.
Output: possibly updated set T.

$\bar{N} := N \setminus T$;
$B = A[:,T]$;
$Bpinv := (B^{\mathsf{T}}B)^{-1}B^{\mathsf{T}}$;
$nBpinv = \|Bpinv\|_1$;
$cont = true$;
while *(cont)* **do**
  $cont = false$;
  for $\ell = 1,\ldots,n-r$ **do**
    $\gamma := A[:,\bar{N}(\ell)]$;
    for $j = 1,\ldots,r$ **do**
      Let $B_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column of B by γ;
      $Bpinv^+ := (B_{\gamma/j}^{\mathsf{T}}B_{\gamma/j})^{-1}B_{\gamma/j}^{\mathsf{T}}$;
      $nBpinv^+ := \|Bpinv^+\|_1$;
      if $nBpinv^+ < nBpinv$ **then**
        $B := B_{\gamma/j}$;
        $Bpinv := Bpinv^+$;
        $nBpinv = nBpinv^+$;
        $T := T \cup \{\bar{N}(\ell)\} \setminus \{T(j)\}$;
        $\bar{N} := N \setminus T$;
        $cont = true$;
        break ;

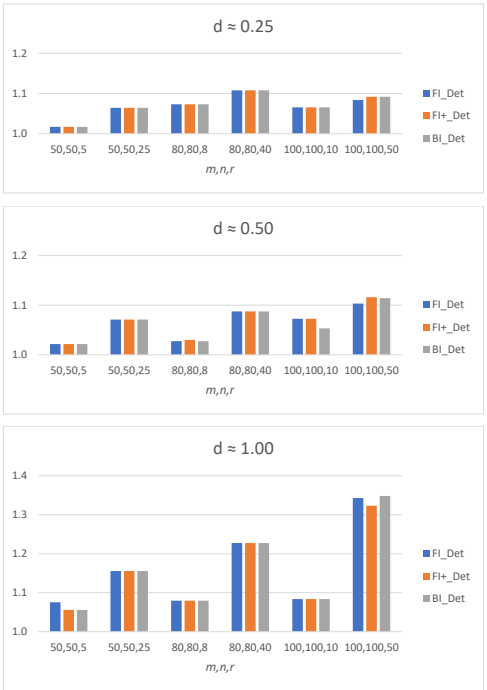


Figure 3.3: $\|H\|_1/z_{P_{123}}$ (Small) (ah-symmetric generalized inverse)

of the time to generate the initial matrix H by the NSub algorithm (Algorithm 3), and the time of the local search (FI(det), FI$^+$(det), or BI(det)).

	$\|H\|_1/z_{P_{123}}$			Time(sec)				
m, r, d	FI(det)	FI$^+$(det)	BI(det)	P_{123}	NSub	FI(det)	FI$^+$(det)	BI(det)
50,05,0.25	1.017	1.017	1.017	1.526	0.091	0.005	0.004	0.005
50,25,0.25	1.065	1.065	1.065	5.506	0.094	0.002	0.001	0.001
80,08,0.25	1.073	1.073	1.073	5.325	0.077	0.002	0.002	0.003
80,40,0.25	1.108	1.108	1.108	34.729	0.149	0.003	0.003	0.006
100,10,0.25	1.066	1.066	1.066	11.695	0.090	0.003	0.003	0.006
100,50,0.25	1.084	1.092	1.092	93.004	0.266	0.007	0.005	0.011
50,05,0.50	1.022	1.022	1.022	1.807	0.065	0.002	0.001	0.002
50,25,0.50	1.071	1.071	1.071	7.873	0.084	0.002	0.002	0.003
80,08,0.50	1.027	1.030	1.027	8.427	0.068	0.002	0.002	0.004
80,40,0.50	1.088	1.088	1.088	102.428	0.131	0.005	0.003	0.008
100,10,0.50	1.073	1.073	1.053	24.495	0.094	0.003	0.003	0.005
100,50,0.50	1.103	1.116	1.114	481.614	0.288	0.011	0.006	0.023
50,05,1.00	1.075	1.056	1.056	2.699	0.003	0.004	0.003	0.005
50,25,1.00	1.155	1.155	1.155	41.476	0.036	0.004	0.003	0.007
80,08,1.00	1.079	1.079	1.079	32.213	0.018	0.003	0.002	0.007
80,40,1.00	1.227	1.227	1.227	692.151	0.088	0.009	0.005	0.019
100,10,1.00	1.084	1.084	1.084	169.194	0.020	0.004	0.004	0.012
100,50,1.00	1.343	1.323	1.348	3672.983	0.139	0.017	0.010	0.040

Table 3.5: Local Searches for ah-symmetric generalized inverse vs. P_{123}

We see from Figure 3.3 and Table 3.5, that the three local searches converge to solutions of similar quality on most of the experiments. We also see in Table 3.5 that the running times to solve P_{123} increase quickly with the dimension of the matrix, and are much higher than the times for the local searches.

Next, we consider the experiments done with the 360 instances in the 'Medium' category, which had the main purpose of comparing the different local searches proposed. We present in Table 3.6 average results for each group of 30 instances with the same configuration, described in the first column. In the next three columns, we present statistics for the local searches based on the determinant, which are initialized with the solutions given by the NSub algorithm, and in the last three columns we consider the application of the local searches based on the 1-norm of H, which are initialized with the solutions given by the three first local searches. In the first half of the table, we show the mean and standard deviation of the relative difference between the 1-norm of the matrix H obtained by each local search and the minimum value among all of them, denoted by $||H_{best}||_1$.

Comparing to the tests with the 'Small' instances, we note that we still have solutions of similar quality obtained by the three local searches based on the determinant, for this larger group of instances of higher dimension. The average relative difference between the norms of the solutions obtained by these searches and the minimum norms approximately goes from 3 to 10%, increasing with the rank and the dimension. We also see that the improvement on the solutions found by the local searches based on the 1-norm of H, when compared to the determinant searches comes once more with a high computational cost.

In Table 3.7, we present the number of swaps for each local search. Combining these results with the running time of the procedures, we conclude that the BI(det) procedure could be discarded. Comparing it to the other determinant searches, we see again that, although it converges to solutions of similar quality performing

m, r, d	FI(det)	FI$^+$(det)	BI(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	BI(det) FI(norm)
	$(\|H\|_1 - \|H_{best}\|_1)/\|H_{best}\|_1$					
1000,050,0.25	0.029/0.026	0.029/0.026	0.029/0.026	0.000/0.000	0.000/0.000	0.000/0.000
1000,050,0.50	0.045/0.022	0.045/0.022	0.044/0.022	0.000/0.002	0.000/0.002	0.000/0.000
1000,050,1.00	0.042/0.022	0.041/0.020	0.041/0.020	0.000/0.000	0.000/0.000	0.000/0.000
1000,100,0.25	0.064/0.017	0.064/0.017	0.064/0.018	0.000/0.000	0.000/0.001	0.001/0.004
1000,100,0.50	0.042/0.028	0.042/0.028	0.042/0.028	0.000/0.000	0.000/0.000	0.000/0.000
1000,100,1.00	0.064/0.028	0.066/0.030	0.066/0.030	0.001/0.003	0.001/0.003	0.001/0.003
2000,100,0.25	0.056/0.020	0.056/0.020	0.056/0.020	0.000/0.000	0.000/0.000	0.000/0.000
2000,100,0.50	0.082/0.026	0.083/0.027	0.083/0.027	0.000/0.001	0.001/0.001	0.000/0.001
2000,100,1.00	0.052/0.030	0.054/0.032	0.053/0.029	0.001/0.003	0.002/0.007	0.003/0.006
2000,200,0.25	0.089/0.037	0.095/0.041	0.095/0.035	0.002/0.005	0.004/0.007	0.006/0.010
2000,200,0.50	0.062/0.017	0.062/0.016	0.061/0.017	0.001/0.003	0.000/0.001	0.000/0.001
2000,200,1.00	0.098/0.024	0.100/0.025	0.100/0.029	0.002/0.005	0.001/0.004	0.002/0.004
	Time(sec)					
1000,050,0.25	0.051/0.010	0.048/0.005	0.079/0.059	8.505/2.592	8.338/2.798	8.220/2.626
1000,050,0.50	0.119/0.040	0.098/0.023	0.215/0.229	32.498/10.200	32.487/10.583	32.339/10.694
1000,050,1.00	0.205/0.047	0.199/0.026	0.325/0.115	148.958/37.366	147.906/38.710	145.106/36.993
1000,100,0.25	0.643/0.205	0.490/0.107	1.559/1.769	995.64/257.79	978.29/227.30	991.33/232.10
1000,100,0.50	0.049/0.012	0.048/0.009	0.076/0.071	14.223/3.432	14.289/3.758	14.151/3.722
1000,100,1.00	0.129/0.043	0.106/0.024	0.397/0.310	55.756/12.981	56.009/13.701	56.973/12.637
2000,100,0.25	0.176/0.048	0.201/0.032	0.350/0.174	233.947/35.658	234.675/36.255	236.703/39.453
2000,100,0.50	0.594/0.184	0.441/0.155	2.975/3.004	1467.39/275.77	1445.03/311.96	1454.70/312.04
2000,100,1.00	0.069/0.030	0.058/0.019	0.332/0.239	23.153/6.454	22.469/6.044	22.610/6.376
2000,200,0.25	0.216/0.077	0.155/0.045	1.219/0.706	85.532/13.034	89.602/18.831	87.646/20.069
2000,200,0.50	0.208/0.092	0.238/0.060	1.050/1.472	349.917/75.253	348.767/75.847	350.218/76.554
2000,200,1.00	0.651/0.348	0.447/0.175	4.683/6.211	2085.29/345.92	1974.42/361.11	1990.26/350.17

Table 3.6: Local Searches for ah-symmetric generalized inverse (Mean/Std Dev) (Medium)

fewer swaps, it is much more time consuming. We can also observe the high cost of the swaps performed by the searches based on the 1-norm. This observation is illustrated in Figure 3.4, where we show the relation between the average relative 1-norm difference to the minimum norm obtained by all searches and the average running time of the local searches for the larger instances in the 'Medium' category. The hollow circle indicates that the local search had worse average result and longer average running time than another procedure, and therefore, should not be adopted. We note that we use a logarithmic scale for the running time. Once more we see some improvement given by the norm searches, but with a very high computational cost.

m, r, d	FI(det)	FI$^+$(det)	BI(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	BI(det) FI(norm)
1000,050,0.25	4.133	3.667	2.833	6.267	6.267	6.267
1000,050,0.50	8.700	6.800	5.067	14.733	14.733	14.700
1000,050,1.00	5.633	4.967	2.833	16.200	16.100	16.100
1000,100,0.25	15.033	12.033	9.133	35.367	35.233	35.167
1000,100,0.50	4.867	4.233	2.833	7.500	7.500	7.500
1000,100,1.00	18.833	15.533	11.967	17.633	17.900	17.867
2000,100,0.25	5.433	4.900	3.100	19.767	19.767	19.667
2000,100,0.50	31.500	27.333	21.700	44.067	43.800	43.833
2000,100,1.00	113.267	70.700	24.867	10.067	10.033	9.600
2000,200,0.25	177.067	100.800	48.867	23.767	24.533	24.467
2000,200,0.50	99.867	56.567	19.833	23.600	23.400	22.933
2000,200,1.00	134.500	76.933	41.267	52.333	52.400	51.900

Table 3.7: Number of swaps (Medium) (ah-symmetric generalized inverse)

Finally, our last experiments intend to show the scalability of the procedures, considering matrices with up to 10000 rows, 1000 columns, rank up to 100, and density equal to one. As the BI(det) procedure, was not successful in the previous test, we did not run it on the 'Large' category. In Table 3.8, we see that the average relative differences between the 1-norm of the matrix H obtained by each local search based on the determinant and the minimum value among all the four local

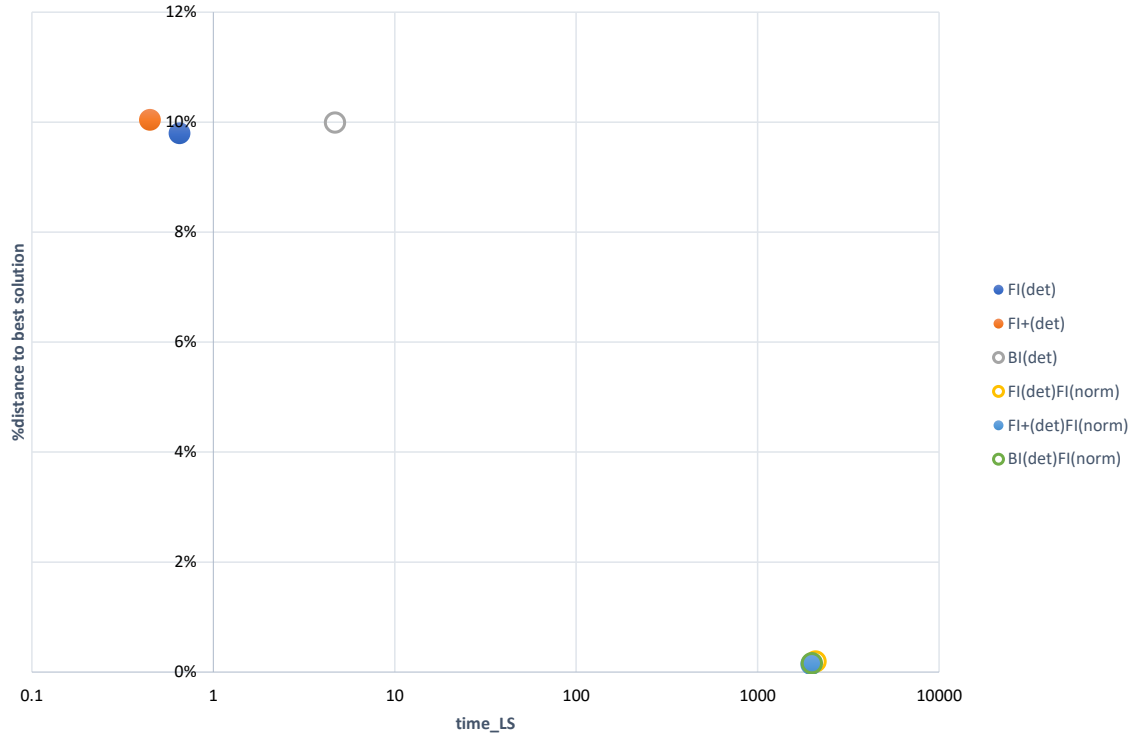


Figure 3.4: Local searches ($m = 2000$,$r = 200$,$d = 1$) (ah-symmetric generalized inverse)

searches are approximately between 3 and 6%. Comparing to best solutions found, the searches based on the determinant keep the quality observed on the smaller instances. Furthermore, theses matrices are obtained in less than 2.5 seconds on average, even for the largest instances.

	$(\|H\|_1 - \|H_{best}\|_1)/\|H_{best}\|_1$				
m,n,r	FI(det)	FI$^+$(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	
5000,1000,050	0.053	0.045	0.020	0.021	
5000,1000,100	0.034	0.040	0.011	0.024	
10000,1000,050	0.045	0.059	0.000	0.025	
10000,1000,100	0.030	0.028	0.012	0.005	
	Time(sec)				
m,n,r	FI(det)	FI$^+$(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	NSub
5000,1000,050	0.100	0.109	221.678	157.442	0.641
5000,1000,100	0.252	0.291	534.947	395.332	1.139
10000,1000,050	0.084	0.090	424.338	307.216	1.897
10000,1000,100	0.317	0.343	987.229	1145.040	1.942

Table 3.8: Local searches (Large) (ah-symmetric generalized inverse)

3.4 Symmetric generalized inverse

Now we assume that A is symmetric, and we are interested in finding a good symmetric reflexive generalized inverse. The local-search procedures for the symmetric reflexive generalized inverse are based on the block construction procedure presented in Xu, Fampa, Lee, and Ponte (2021). More specifically, on Theorem 21, Definition 22, and Theorem 23, presented next.

Theorem 21 (Xu, Fampa, Lee, and Ponte (2021)). *For a symmetric matrix $A \in \mathbb{R}^{n\times n}$, let $r := \operatorname{rank}(A)$. Let $\tilde{A} := A[S]$ be any $r \times r$ non-singular principal submatrix of A. Let $H \in \mathbb{R}^{n\times n}$ be equal to zero, except its submatrix with row/column indices S equal to $\tilde{A}^{-1}$. Then H is a symmetric reflexive generalized inverse of A.*

Definition 22 (Xu, Fampa, Lee, and Ponte (2021)). *Let A be an arbitrary $n \times n$, rank-r matrix. For S an ordered subset of r elements from $\{1, \ldots, n\}$ and fixed $\epsilon \geq 0$, if $|\det(A[S])| > 0$ cannot be increased by a factor of more than $1 + \epsilon$ by swapping an element of S with one from its complement, then we say that $A[S]$ is a $(1+\epsilon)$-local maximizer for the absolute determinant on the set of $r \times r$ non-singular principal submatrices of A.*

Theorem 23 (Xu, Fampa, Lee, and Ponte (2021)). *For a symmetric matrix $A \in \mathbb{R}^{n\times n}$, let $r := \operatorname{rank}(A)$. Choose $\epsilon \geq 0$, and let $\tilde{A} := A[S]$ be a $(1+\epsilon)$-local maximizer for the absolute determinant on the set of $r \times r$ non-singular principal submatrices of A. The $n \times n$ matrix H constructed by Theorem 21 over $\tilde{A}$, is a symmetric reflexive generalized inverse (having at most r^2 non-zeros), satisfying $\|H\|_1 \leq r^2(1+\epsilon)\|H^{sym,r}_{opt}\|_1$, where $H^{sym,r}_{opt}$ is a 1-norm minimizing symmetric reflexive generalized inverse of A.*

The idea of our algorithms in this section is then to select an $r \times r$ non-singular *principal* submatrix $\tilde{A}$ of A, and construct the symmetric reflexive generalized inverse with the inverse of this submatrix, as described in Theorem 21. The non-zero entries of H will be the non-zero entries of $\tilde{A}^{-1}$. Guided by the result in Theorem 23, the 'det' searches aim at selecting a principal submatrix $\tilde{A}$ that is a local maximizer for the absolute determinant on the set of $r \times r$ non-singular principal submatrices of A. We also apply 'norm' searches, as done in the two previous sections.

In the following, we discuss how the symmetric test matrices A used in our computational experiments were generated, how we select the initial principal submatrix of A to initialize the local searches, we describe the algorithms, and we present numerical results.

To analyze the local-search procedures, we compare their solutions to the solution of LP problem P_1^{sym}. Its solution corresponds to $\|H^{sym}_{opt}\|_1$, where H^{sym}_{opt} is a 1-norm minimizing symmetric generalized inverse of A.

$$
\begin{aligned}
(P_1^{sym})\ z_{P_1^{sym}} := \quad \min \quad & \langle J, T\rangle \ , \\
\text{s.t.:} \quad & T - H \geq 0 \ , \\
& T + H \geq 0 \ , \\
& AHA = A \ , \\
& H = H^{\intercal} \ .
\end{aligned}
$$

We note that the result in Theorem 23 is not related to H_{opt}^{sym}, but to $H_{opt}^{sym,r}$, a 1-norm minimizing symmetric reflexive generalized inverse of A. However, from the proof in Xu, Fampa, Lee, and Ponte (2021), we can conclude that the result is still valid if we replace $H_{opt}^{sym,r}$ by H_{opt}^{sym} in the theorem. The reason of computing the second in our experiments is that, unlike the first, it can be efficiently computed by the solution of an LP problem, specifically P_1^{sym}.

3.4.1 Our test matrices

To test the local-search procedures proposed in this section, we randomly generated 360 symmetric matrices A with varied dimensions, ranks, and densities.

The matrices were generated with the Matlab function *sprandsym*. The function generates a random $m \times m$ dimensional symmetric matrix A with approximate density d and eigenvalues rc. The eigenvalues of A are given as the input vector rc,. The number of non-zero elements of rc is of course the desired rank r. For our experiments, we selected the r nonzeros of rc as before, $M \times (\pm\rho^1, \pm\rho^2, \ldots, \pm\rho^r)$, where $M := 2$, and $\rho := (1/M)^{(2/(r+1))}$, and the signs were randomly selected.

We divide our instances into the following three categories:

- Small: 90 instances. 5 with each of the 18 combinations of the following parameters: $m = n = 50, 80, 100$; $r = 0.1 \times n, 0.5 \times n$; $d = 0.25, 0.50, 1.00$.
- Medium: 360 instances. 30 with each of the 12 combinations of the following parameters: $m = n = 1000, 2000$; $r = 0.05 \times n, 0.1 \times n$; $d = 0.25, 0.50, 1.00$.

3.4.2 Selecting an initial block for the local search

To construct an $r \times r$ non-singular principal submatrix of A to initialize the local searches when A is symmetric, we consider the following result.

Proposition 24. *Let A be a symmetric $m \times m$ matrix with rank r. Suppose that the r columns of A indexed by $j_1, j_2, \ldots j_r$ are linear independent. Then the principal submatrix $A[j_1, j_2, \ldots, j_r]$ has rank r.*

Proof. Without loss of generality, we assume that $j_1 = 1, j_2 = 2, \ldots, j_r = r$, and

$$A = \begin{pmatrix} \hat{A} & B \\ B^\top & D \end{pmatrix},$$

with $\hat{A}$ being an $r \times r$ symmetric submatrix. Then

$$\operatorname{rank} \begin{pmatrix} \hat{A} \\ B^\top \end{pmatrix} = r.$$

This implies that there exists an $r \times (m-r)$ matrix X, such that $B = \hat{A}X$, $D = B^\intercal X$, as the r first columns of A form a basis for the column space of A. Therefore,

$$r = \text{rank}\begin{pmatrix} \hat{A} \\ B^\intercal \end{pmatrix} = \text{rank}\left(\begin{pmatrix} I \\ X^\intercal \end{pmatrix} \cdot \hat{A}\right) \leq \text{rank}(\hat{A}),$$

which implies that $\text{rank}(\hat{A}) = r$. □ □

Based on Proposition 24, we apply the same ideas described in Algorithms 1 and 2, but now to select only the set S of r linear independent rows of A. The set of columns T is then selected to be equal to S.

3.4.3 The local-search procedures

In Algorithm 10, we present the 'first improvement' local-search procedure 'FI(det)', which considers as the criterion for improvement of the given solution, the increase in the absolute determinant of the $r \times r$ non-singular principal submatrix of A. Based on Theorem 23, for a given rank-r matrix A, the procedure starts from a set S of r indices, such that $A[S]$ is non-singular.

Algorithm 10: 'FI(det)' for symmetric generalized inverses.

Input: $A \in \mathbb{R}^{m \times m}$, such that $r :=\text{rank}(A)$, $A = A^\intercal$
$S \subset M := \{1, \ldots, m\}$, such that $|S| = r$, and $A[S]$ is non-singular.
Output: possibly updated set S.

```
1  B := A[S];
2  detB := det(B);
3  M̄ := M \ S;
4  cont = true;
5  while (cont) do
6      cont = false;
7      for ℓ = 1, ..., m − r do
8          for j = 1, ..., r do
9              Saux := S ∪ {M̄(ℓ)} \ {S(j)};
10             B+ := A[Saux];
11             detB+ := det(B+);
12             if |detB+| > |detB| then
13                 B := B+;
14                 detB := detB+;
15                 S := Saux;
16                 M̄ := M \ S;
17                 cont = true;
18                 break ;
```

In the loop of Algorithm 10 (lines 7–18), a column and row of $A[S, S]$ is replaced if the absolute determinant increases with the replacement. To evaluate how much the determinant changes with the replacement, we consider the result in Remark 25.

Remark 25. *Let $\gamma \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$ with $\det(A) \neq 0$. Let $A_{\gamma/j}$ be the matrix obtained by replacing the j^{th} column and row of A by γ and $\gamma^\intercal$, respectively. If $\hat{\alpha} \in \mathbb{R}^n$ solves the linear system of equations $A\alpha = \gamma$, then we have*

$$\det(A_{\gamma/j}) = \hat{\alpha}_j^2 \times \det(A).$$

The result follows from

$$A_{\gamma/j} = [I + \mathbf{e}_j(\hat{\alpha} - \mathbf{e}_j)^\intercal]A[I + (\hat{\alpha} - \mathbf{e}_j)\mathbf{e}_j^\intercal].$$

The algorithm for 'FI$^+$(det)' differs from Algorithm 10, concerning the choice of the index to be replaced in the current set S. For the 'FI$^+$' local search, instead of considering the first increase in the absolute value of the determinant of $A[S]$, obtained in the loop described in lines 8–18, the algorithm computes the modification in the absolute value of the determinant obtained for each index j, and selects the index that leads to the greatest increase. For the algorithm 'BI(det)', the pair of indices (ℓ, j), in the two loops described in lines 7–18, that leads to the greatest increase in the absolute value of the determinant is considered for the modification in the matrix.

For the 'norm' searches, instead of computing the determinant of B and B^+ in lines 2 and 11 of Algorithm 10, we compute the 1-norm of their inverses. The update in the index set S occurs when the 1-norm decreases.

3.4.4 Numerical results

We initially consider the experiments done with the 90 instances in the 'Small' category, which had the main purpose of analyzing the ratios between the the 1-norm of the matrices H computed by the three local searches based on the determinant, with the minimum 1-norm of a ah-symmetric generalized inverse given by the solution of the LP problem P_1^{sym} ($\|H\|_1/z_{P_1^{sym}}$). We aim at checking how close these ratios are from the upper bound given by Theorem 23.

In Figure 3.5, we present the average ratios for the matrices with the same dimension, rank, and density. From Theorem 23, we know that these ratios cannot be greater than r^2, and we see from the results, that for the matrices considered in our tests, we stay quite far from this upper bound. In general, the ratios increase with the rank r, the dimension $m = n$, and the density d of the matrices, but even for $r = 50$, we obtain ratios smaller than 2.1.

In Table 3.9, besides presenting the average ratios depicted in Figure 3.5, we also present the average running times to compute the generalized inverses. In case of the local searches, the total time to compute the generalized inverse is given by the

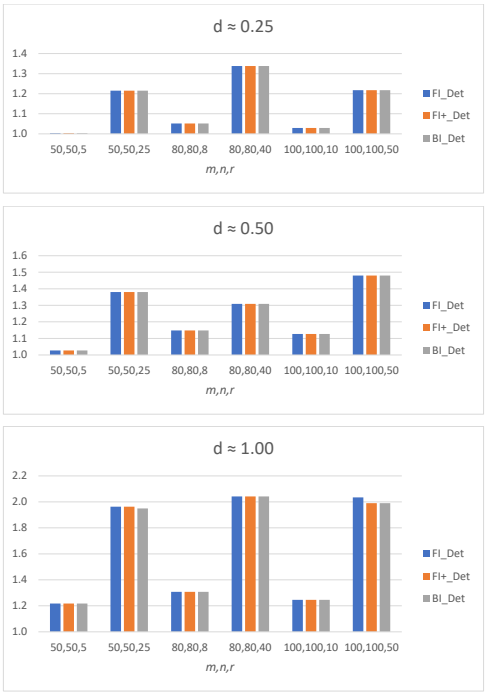


Figure 3.5: $\|H\|_1/z_{P_1^{sym}}$ (Small) (symmetric generalized inverse)

sum of the time to generate the initial matrix H by the simplified version of the NSub algorithm (Algorithm 3) discussed in §3.4.2, and the time of the local search (FI(det), FI$^+$(det), or BI(det)).

	$\|H\|_1/z_{P_1^{sym}}$			Time(sec)				
m, r, d	FI(det)	FI$^+$(det)	BI(det)	P_1^{sym}	NSub	FI(det)	FI$^+$(det)	BI(det)
50,05,0.25	1.003	1.003	1.003	0.623	0.031	0.004	0.004	0.003
50,25,0.25	1.215	1.215	1.215	6.238	0.004	0.028	0.025	0.028
80,08,0.25	1.051	1.051	1.051	2.041	0.029	0.007	0.004	0.004
80,40,0.25	1.338	1.338	1.338	46.916	0.027	0.134	0.115	0.140
100,10,0.25	1.029	1.029	1.029	4.132	0.020	0.008	0.007	0.009
100,50,0.25	1.218	1.218	1.218	69.780	0.038	0.222	0.196	0.174
50,05,0.50	1.027	1.027	1.027	0.641	0.003	0.003	0.003	0.003
50,25,0.50	1.380	1.380	1.380	7.511	0.003	0.034	0.028	0.054
80,08,0.50	1.148	1.148	1.148	2.785	0.005	0.005	0.004	0.004
80,40,0.50	1.309	1.309	1.309	46.378	0.027	0.125	0.095	0.108
100,10,0.50	1.127	1.127	1.127	4.704	0.005	0.010	0.008	0.010
100,50,0.50	1.480	1.480	1.480	97.951	0.022	0.256	0.235	0.186
50,05,1.00	1.218	1.218	1.218	0.908	0.003	0.003	0.002	0.003
50,25,1.00	1.963	1.963	1.950	10.777	0.003	0.041	0.034	0.048
80,08,1.00	1.307	1.307	1.307	3.628	0.004	0.006	0.004	0.005
80,40,1.00	2.042	2.042	2.042	62.370	0.008	0.200	0.170	0.386
100,10,1.00	1.246	1.246	1.246	6.318	0.006	0.012	0.010	0.015
100,50,1.00	2.034	1.991	1.991	942.775	0.016	0.315	0.298	0.398

Table 3.9: Local Searches for symmetric generalized inverse vs. P_1^{sym}

We see from Figure 3.5 and Table 3.9, that the three local searches converge to solutions of similar quality on most of the experiments. We also see in Table 3.9 that the running times to solve P_1^{sym} increase quickly with the dimension of the matrix, and are much higher than the times for the local searches.

Next, we consider the experiments done with the 360 instances in the 'Medium' category, which had the main purpose of comparing the different local searches pro-

posed. We present in Table 3.10 average results for each group of 30 instances with the same configuration, described in the first column. In the next three columns, we present statistics for the local searches based on the determinant, which are initialized with the solutions given by the NSub procedure, and in the last three columns we consider the application of the local searches based on the 1-norm of H, which are initialized with the solutions given by the three first local searches. In the first half of the table, we show the mean and standard deviation of the relative difference between the 1-norm of the matrix H obtained by each local search and the minimum value among all of them, denoted by $||H_{best}||_1$.

m, r, d	FI(det)	FI$^+$(det)	BI(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	BI(det) FI(norm)
	$(\|H\|_1 - \|H_{best}\|_1)/\|H_{best}\|_1$					
1000,050,0.25	0.040/0.037	0.040/0.037	0.040/0.037	0.000/0.000	0.000/0.000	0.000/0.000
1000,050,0.50	0.098/0.053	0.097/0.052	0.097/0.052	0.000/0.000	0.000/0.000	0.000/0.000
1000,050,1.00	0.073/0.038	0.073/0.038	0.073/0.038	0.001/0.004	0.001/0.004	0.000/0.000
1000,100,0.25	0.119/0.039	0.120/0.038	0.119/0.039	0.001/0.002	0.001/0.003	0.001/0.003
1000,100,0.50	0.077/0.049	0.077/0.049	0.077/0.049	0.000/0.000	0.000/0.000	0.000/0.000
1000,100,1.00	0.144/0.086	0.146/0.085	0.146/0.085	0.002/0.013	0.004/0.014	0.001/0.006
2000,100,0.25	0.107/0.049	0.107/0.049	0.107/0.049	0.000/0.000	0.000/0.001	0.000/0.000
2000,100,0.50	0.203/0.074	0.204/0.076	0.204/0.076	0.000/0.002	0.001/0.007	0.001/0.008
2000,100,1.00	0.187/0.126	0.187/0.126	0.187/0.126	0.000/0.000	0.000/0.000	0.000/0.000
2000,200,0.25	0.210/0.117	0.207/0.117	0.206/0.118	0.002/0.008	0.000/0.000	0.001/0.004
2000,200,0.50	0.218/0.094	0.216/0.093	0.218/0.094	0.001/0.003	0.001/0.007	0.003/0.009
2000,200,1.00	0.293/0.088	0.287/0.090	0.287/0.090	0.000/0.001	0.000/0.000	0.000/0.000
	Time(sec)					
1000,050,0.25	5.404/1.694	3.810/0.663	4.592/1.348	31.492/9.511	31.145/8.401	31.662/8.791
1000,050,0.50	23.957/6.956	17.517/4.616	26.754/12.052	266.19/71.67	268.51/71.74	268.53/70.63
1000,050,1.00	69.181/27.258	36.835/5.228	70.744/21.313	520.68/163.62	519.80/152.22	512.77/145.89
1000,100,0.25	577.53/190.48	310.02/44.63	734.92/207.46	5668.8/1073.3	5558.6/974.5	5559.5/1019.7
1000,100,0.50	5.320/2.249	3.372/0.593	4.191/1.303	31.881/9.784	30.835/8.241	31.580/9.405
1000,100,1.00	21.921/9.387	15.336/3.281	20.791/8.659	334.73/76.07	334.18/76.32	335.50/76.70
2000,100,0.25	66.531/18.351	37.912/5.941	73.078/27.199	667.30/196.24	658.59/197.53	665.20/199.51
2000,100,0.50	549.92/274.73	318.89/85.71	527.86/185.11	7116.8/1690.3	7010.6/1683.7	6844.4/1521.1
2000,100,1.00	4.421/2.456	3.251/1.033	3.923/1.855	45.350/10.200	45.420/10.154	45.121/11.686
2000,200,0.25	19.292/9.230	13.978/4.679	20.583/15.295	437.376/128.407	437.66/126.50	433.88/112.52
2000,200,0.50	48.824/21.987	35.423/9.395	41.201/18.144	921.35/280.99	935.07/310.43	935.30/297.93
2000,200,1.00	387.26/180.38	275.56/74.36	373.22/200.87	9525.2/2202.6	9675.2/2388.8	9645.7/2182.5

Table 3.10: Local Searches for symmetric generalized inverse (Mean/Std Dev) (Medium)

Comparing to the tests with the 'Small' instances, we see that on this larger group of instances of higher dimension, the similarity among the quality of the solutions obtained by the three local searches based on the determinant is still present. The average relative difference between the norms of the solutions obtained by these searches and the minimum norms goes from approximately 4 to 30%, increasing with the rank and the dimension. We also see that the improvement on the solutions found by the local searches based on the 1-norm of H, when compared to the determinant searches comes with a high computational cost.

In Table 3.11, we present the number of swaps for each local search. Combining these results with the running time of the procedures, we conclude that the FI$^+$(det) procedure is the best search based on the determinant, presenting smaller average computational times than the other two. We can also observe a relative increase in the number of swaps when compared to the non-symmetric cases discussed in the previous sections. This increase is also reflected in the greater improvement obtained with these searches. However, the improvement continues to come with a

very high computational cost.

m, r, d	FI(det)	FI$^+$(det)	BI(det)	FI(det) FI(norm)	FI$^+$(det) FI(norm)	BI(det) FI(norm)
1000,050,0.25	3.633	2.067	1.633	4.467	4.467	4.533
1000,050,0.50	4.833	3.267	2.267	12.800	12.700	12.700
1000,050,1.00	7.133	3.767	3.000	11.667	11.667	11.633
1000,100,0.25	9.233	5.600	3.967	28.733	28.833	28.867
1000,100,0.50	3.533	1.733	1.500	6.200	6.200	6.200
1000,100,1.00	3.367	2.000	1.567	16.967	16.933	16.900
2000,100,0.25	7.167	3.900	3.100	15.533	15.533	15.533
2000,100,0.50	6.433	3.433	2.500	40.733	40.467	40.267
2000,100,1.00	3.100	1.867	1.333	12.767	12.767	12.767
2000,200,0.25	3.333	1.967	1.533	25.333	25.333	25.033
2000,200,0.50	3.000	1.833	1.333	30.000	29.900	29.867
2000,200,1.00	3.200	1.933	1.467	60.700	60.067	59.800

Table 3.11: Number of swaps (Medium) (symmetric generalized inverse)

3.5 Case Study

In this section, we report on a case study that we undertook on a real-world data set. We sought to validate our techniques, in the ah-symmetric case. We applied the ideas as we described, but additionally in a somewhat more general way. In our presentation, we always worked with r equal to the rank of the input data matrix. But we can also take *smaller* r, with the benefit of gaining even sparser (block) ah-symmetric generalized inverses. Of course, with smaller r, we give up something in the least-squares fit, and so we explored this trade off in our case study.

We applied our techniques to the "Communities and Crime Data Set" (https://archive.ics.uci.edu/ml/datasets/Communities+and+Crime) obtained from the UCI Machine Learning Repository, at the Center for Machine Learning and Intelligent Systems, University of California at Irvine. The data set combines socioeconomic data from the 1990 US Census, law enforcement data from the 1990 US LEMAS (Law Enforcement Management and Administrative Statistics) survey, and crime data from the 1995 FBI UCR (Uniform Crime Reporting) Program.

The data is for $1,994$ communities, 128 variables: 122 predictive, 5 non-predictive, 1 goal. The goal variable is the total number of violent crimes per 100,000 population. Data was incomplete for one community and for 22 predictive variables. So we settled on a (rather dense) $(m = 1,993) \times (n = 100)$ matrix A to work with, and a corresponding goal $b \in \mathbb{R}^m$. Considering the real singular value decomposition $A = \sum_{i=1}^{n} \sigma_i u_i v_i^\intercal$ of A, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$, orthonormal $u_i \in \mathbb{R}^m$, and orthonormal $v_i \in \mathbb{R}^n$, we define low-rank versions of A, namely $A_r := \sum_{i=1}^{r} \sigma_i u_i v_i^\intercal$, for $r = 50, 49, \ldots, 10$. In Figure 3.6, we plot the singular values of A. We note that A_r is the closest rank-r matrix in Frobenius norm to A. For $r = 50$, we have $\|A_{50}\|_F^2 = 40,353$ as compared to $\|A\|_F^2 = 40,480$, so we can say that A_{50} is a very close approximation of A. But considering the sharp decay in the plot, we can even say that A_r is a good approximation of A for all $r = 50, 49, \ldots, 10$.

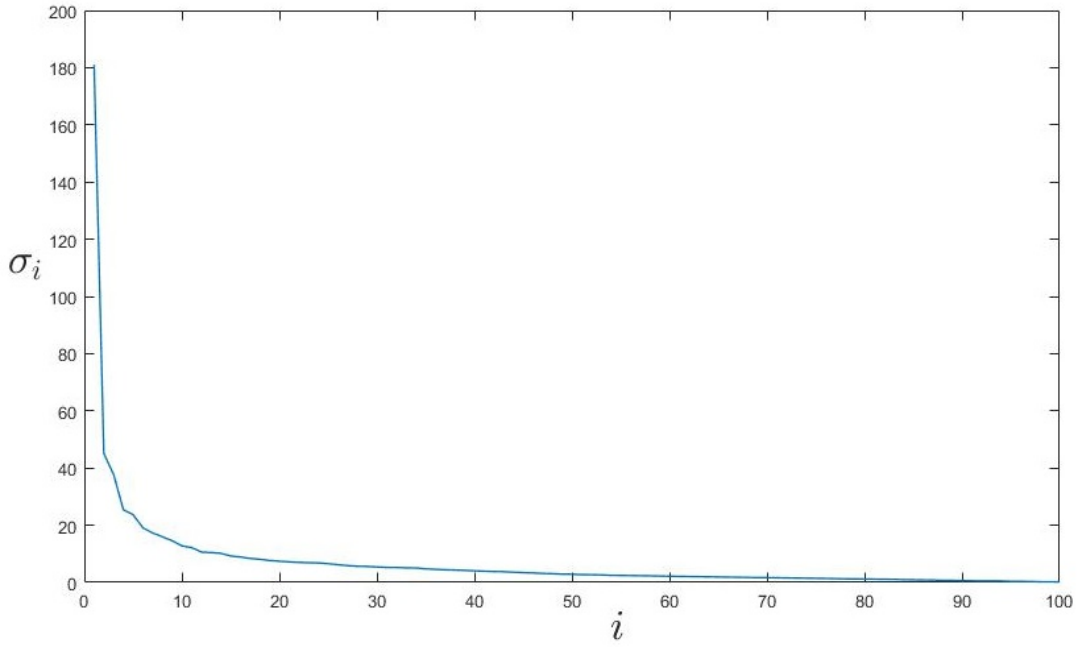


Figure 3.6: Singular values of A

In Figures 3.7–3.10, we present the R-squared[1] statistical measure computed by considering:

- all 100 columns of the matrix A_{50}: indicated by the gold line in all four figures. The R-squared for A_{50} is 0.6690, only a bit lower than the R-squared of 0.6957 for A;
- the r columns of A_{50} selected by applying our local search to A_{50}, for $r = 50, 49, \ldots, 10$: indicated by the blue plot in Figures 3.7 and 3.8;
- the r columns of A_{50} selected by applying our local search to A_r, for $r = 50, 49, \ldots, 10$, indicated by the orange plot in Figures 3.8 and 3.10;
- the r columns of A selected by applying our local search to A_{50}, for $r = 50, 49, \ldots, 10$: indicated by the black plot in Figures 3.7 and 3.9;
- the r columns of A selected by applying our local search to A_r, for $r = 50, 49, \ldots, 10$, indicated by the red plot in Figures 3.9 and 3.10;

The local search procedure applied in these experiments was $\text{FI}^+(\det)$, which had the best performance on the tests presented in §3.3.2.

We wish to emphasize that our local searches do not consider the goal variable. Rather, our local searches aim to find a good *small* set of columns of A (corresponding to a sparse block-structured ah-symmetric generalized inverse), that can be good for *any* realizations of the goal variable.

[1]percentage of variation of the goal variable that is linearly explained by the regression

Figure 3.7 considers our local search for finding r columns, for $r = 50, 49, \ldots, 10$, always applying the local search to A_{50}. In the end, we compare the performance of the selected columns indices, doing the regressions on the chosen columns of both A_{50} and A. We can see that it does not matter much whether we do the regressions on A_{50} or A; that is, A_{50} is a reasonable low-rank approximation of A. Also, we see only a slow deterioration in R-squared as we decrease r (using either A_{50} or A); so our algorithms do well even for $r = 10$.

Figure 3.8 compares two different local searches for finding r columns, for $r = 50, 49, \ldots, 10$. One always does the local search on A_{50}, while the other does it on A_r. In the end, we evaluate the local searches by doing regressions on the chosen columns of A_{50}. We can see that both local searches perform similarly (possibly the one using A_r is a bit better), with slow deterioration in R-squared as we decrease r.

Figure 3.9 again compares the two local searches, but we evaluate the local searches by doing regressions on the chosen columns of A (rather than A_{50}. We reach the same conclusion as we did for Figure 3.8.

Finally, Figure 3.10, considers our local search for finding r columns, for $r = 50, 49, \ldots, 10$, always applying the local search to A_r. In the end, we compare the performance of the selected columns indices, doing the regressions on both the chosen columns of A_{50} but also on A. We reach the same conclusion as we did for Figure 3.7.

Overall, we find that our local-search algorithm for finding $r = 50,\ 49, \ldots, 10$ good regression variables: (i) it is quite robust to versions of the input matrix, working well on A_{50} or A_r, (ii) it is quite robust to how we evaluate the chosen r column indices (treating either A or its low-rank counterpart A_{50} as "the truth"), and (iii) we get very little deterioration in the quality of the least-square fits (as measured by R-squared), as we decrease r.

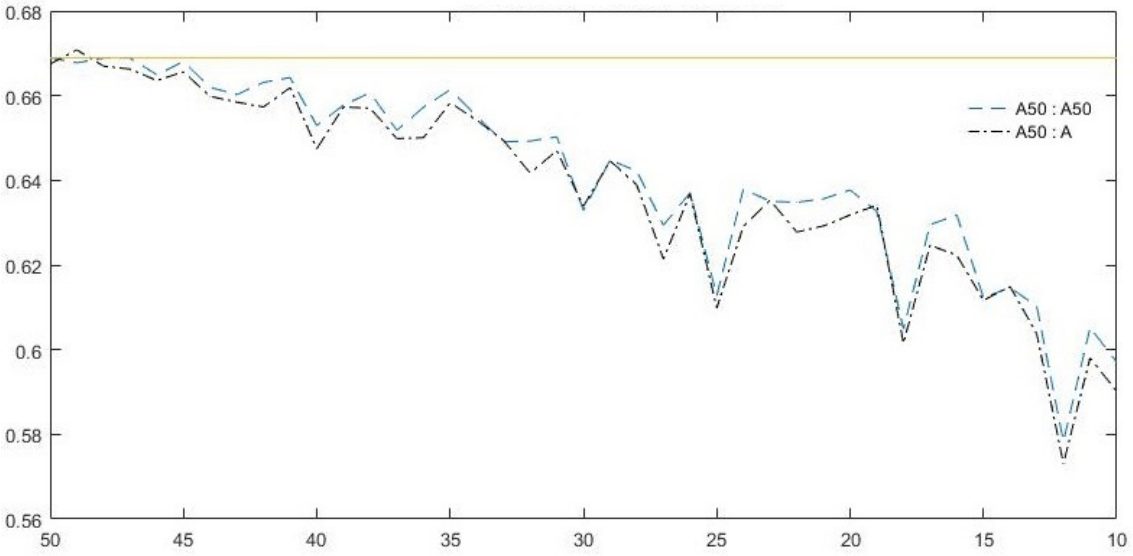


Figure 3.7: Local search on A_{50}, Regression on A_{50}/A

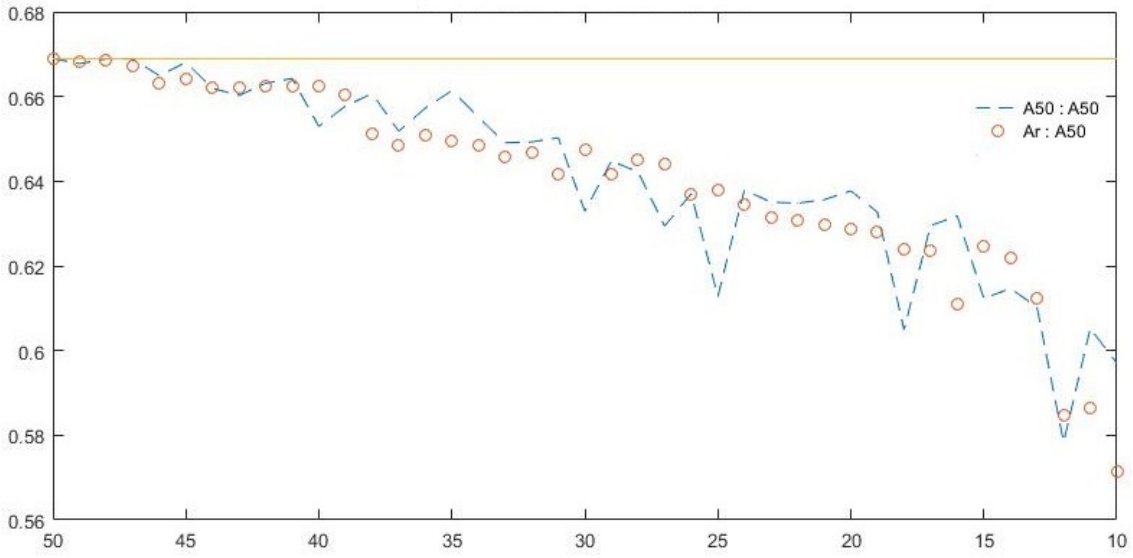


Figure 3.8: Local search on A_{50}/A_r, Regression on A_{50}

Finally, we looked a bit more carefully at the attributes selected by the local searches for $r = 20$, chosen as giving a good level of prediction for a rather low rank. Interesting, there is only agreement on twelve of the twenty attributes selected by the two local searches, while the models have very similar predictive capability. We can see that drawing causal conclusions from the selected attributes would be very dubious.

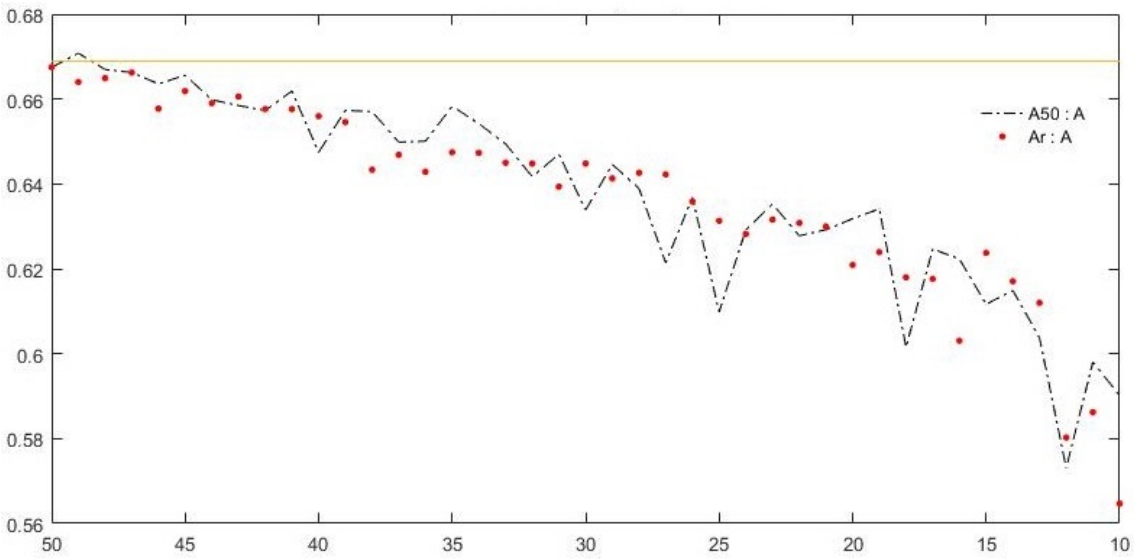


Figure 3.9: Local search on A_{50}/A_r, Regression on A

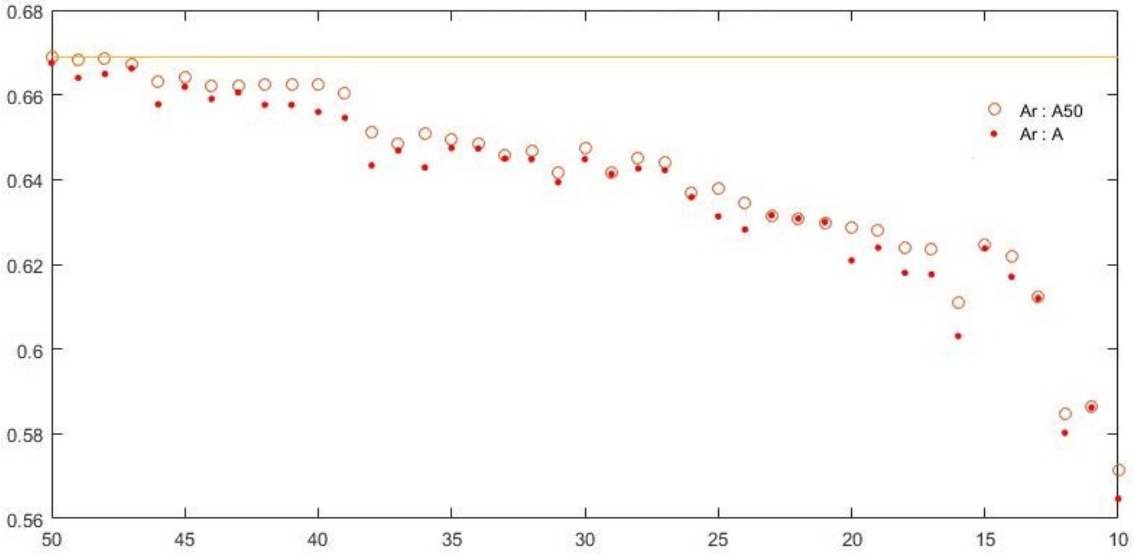


Figure 3.10: Local search on A_r, Regression on A_{50}/A

3.6 Concluding remarks

We have demonstrated that the local-search procedures presented in Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021) can be successfully implemented to construct sparse, block-structured reflexive generalized inverses with different properties. We find that the performance (1-norm achieved) is much better than tight worst-case guarantees. Overall, we find that the search procedures are very robust in terms of many of the algorithmic choices that need to be made. For scaling purposes, we found that it is necessary to be mindful of the numerics and of economizing when seeking local improvements, and calculating initial solutions

efficiently proves to be a surprisingly difficult practical issue.

References Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021) established that the ratios between the norms of the solutions of the local searches and the LP problems P_1, P_{123}, and P_1^{sym} are bounded by r^2, r, and r^2, respectively, when considering generalized inverses, ah-symmetric generalized inverses, and symmetric generalized inverses. We observed in our numerical experiments that the average ratios were much smaller than these worst-case upper bounds, and also that they were smaller for the ah-symmetric case. This can be explained by the fact that the upper bound is smaller for the ah-symmetric generalized inverses (r vs. r^2), but also because in this case, we could include the linearized constraints for property P2 in the LP problem P_{123}, increasing its optimal objective function value.

Comparing the three local-search procedures based on the determinant, we conclude that they converge to solutions of very similar quality. The best improvement approach ('BI(det)') is too expensive and can be discarded. In general, the procedure 'FI$^+$(det)' had slightly better times than 'FI(det)'.

The computational time to solve the LP problems considered is much larger than the times of the local searches, and increases much faster than the times of the local searches when the dimension, the rank, or the density of the matrices increases. So we conclude that LP is not a competitive alternative to the local searches, even if we only cared about running time. An interesting point is that the most costly LP solution is given for problem P_{123}, with more constraints to model ah-symmetric generalized inverses. On the other side, the local-search procedures to construct these matrices are the fastest ones, as the searches are only applied to the columns of the matrices, for a given set of linear independent rows.

The local-search procedures based on the 1-norm were considered with the purpose of determining whether or not the searches based on the determinant could be still improved with respect to the 1-norm of the matrices. For the ah-symmetric case, we saw only a relatively modest improvement in 1-norm. For generalized inverses we saw better improvements in 1-norm, and for symmetric generalized inverses even better. We can conclude that for the ah-symmetric case, 1-norm search is never recommended, and for the others, perhaps they could be considered if one is willing to incur a substantial computational cost.

The running times of the local-search procedures based on the determinants were critically decreased with the use of the results pointed out in our remarks (Remarks 16, 17, 18, 20, 25) which indicate how to efficiently update the determinant of the matrices after the rows and columns swaps at each iteration. A naïve implementation, instead recomputing determinants from scratch, would not allow to scale to large instances.

A significant part of our effort spent in this research was dedicated to developing

a good algorithm to construct an initial solution to our local searches. The computation of an $r \times r$ non-singular submatrix of a rank-r matrix turned out to be a challenge when considering our large, and even medium-sized test instances. The procedure proposed had a very good performance in our numerical experiments and its Matlab implementation is now available through Mathworks.

Chapter 4

Trading off 1-norm and sparsity against rank for linear models using mathematical optimization

This chapter corresponds to the work that has been published as:
Marcia Fampa, Jon Lee, Gabriel Ponte. Trading off 1-norm and sparsity against rank for linear models using mathematical optimization. Open Journal of Mathematical Optimization, 2(4), 14 p, 2021. https://doi.org/10.5802/ojmo.6

4.1 Introduction

A 1-norm minimizing generalized inverse will not generally have minimum rank. A minimum-rank generalized inverse (i.e., a reflexive generalized inverse) will not generally have minimum 1-norm. So we seek generalized inverses that balance these two objectives. In what follows, we investigate algorithmic options and outcomes associated with partially relaxing the reflexive property P2, trading off 1-norm against satisfaction of P2, and in effect also trading off sparsity against rank.

To obtain a 1-norm minimizing ah-symmetric generalized inverse H of a given matrix A, we consider the optimization problem

$$\min\{\|H\|_1 \,:\, P1, P3\}. \tag{4.1}$$

We have the following useful result concerning property P2.

Theorem 26. *(see Rohde (1964)) If H is a generalized inverse of A then* $\operatorname{rank}(H) \geq \operatorname{rank}(A)$*, with equality if and only if H is reflexive.*

To obtain an ah-symmetric generalized inverse with small 1-norm, but also with guaranteed least rank, from Theorem 26, we notice that we should add P2 to the

constraints in (4.1), resulting in

$$\min\{\|H\|_1 \,:\, P1, P2, P3\}. \tag{4.2}$$

The solution of (4.2) is a 1-norm minimizing ah-symmetric reflexive generalized inverse of A. Clearly the price we pay for having the least rank condition on H, is a possible increase in its 1-norm, and we investigate through different approaches, how both measures, the 1-norm and the rank of H, vary as we iteratively add the individual constraints that model P2 to problem (4.1).

We propose algorithmic procedures that compute ah-symmetric generalized inverses of varied ranks and norms for a given $m \times n$ matrix A of rank $r(A) < \min\{m, n\}$. The ah-symmetric generalized inverses constructed vary in a range that goes from matrices with minimum 1-norm to matrices with higher 1-norm and rank $r(A)$, i.e., the minimum possible rank. Through numerical experiments, we investigate if with the procedures proposed, we can see a gradual decrease in the rank as the norm increases. Intermediate solutions obtained by these algorithms can realize a good trade-off between low 1-norm and low rank in applications of the generalized inverses.

We note that in order to formulate (4.2) as a linear-programming (LP) problem. From Proposition 3, we can linearize P2. Therefore, if H satisfies P1 and P3, then P2 can be reformulated as the linear equation

$$HAA^{\dagger} = H. \tag{4.3}$$

Considering (4.3), we formulate (4.2) as the linear program

$$\begin{aligned} (P_{123})\ z_{P_{123}} := \quad \min \quad & \langle J, Z\rangle \ , \\ \text{s.t.:} \quad & Z - H \geq 0 \ , \\ & Z + H \geq 0 \ , \\ & AHA = A \ , \\ & (AH)^{\intercal} = (AH) \ , \\ & HAA^{\dagger} = H \ . \end{aligned}$$

We investigate the trade-off between 1-norm minimization and low rank of H, by iteratively imposing P2 to problem (4.1). We initially omit all constraints that

model P2 in P_{123}, i.e., we solve the linear program

$$\begin{aligned} (P_{13})\ z_{P_{13}} := \quad & \min \quad \langle J, Z \rangle \ , \\ & \text{s.t.:} \quad Z - H \geq 0 \ , \\ & \qquad\ \ Z + H \geq 0 \ , \\ & \qquad\ \ AHA = A \ , \\ & \qquad\ \ (AH)^{\intercal} = (AH) \ . \end{aligned}$$

Then, in different ways, we will gradually impose the (mn individual) constraints (4.3).

In §4.2, we describe our algorithmic approaches. In §4.3, we discuss our numerical results, and in §4.4 we present our conclusions.

4.2 Algorithmic approaches

In this section, we investigate different algorithmic approaches to gradually impose the satisfaction of P2 on the solution of P_{13}. The purpose of our algorithms is to construct a set of diverse ah-symmetric generalized inverses, trading off low 1-norm against low rank. We aim for a sequence of intermediate ah-symmetric generalized inverses, constructed by our algorithms, trending toward lower rank at the expense of increasing 1-norm. We propose five approaches and compare their ability to construct good solutions, where we obtain a gradual decrease in the rank as the 1-norm iteratively increases.

The first four proposed algorithms use an interesting feature that is specific to our problem, of constructing ah-symmetric generalized inverses: the equivalence between satisfaction of P2 and the least-rank condition. While P2 is a nonlinear equation in H (hence beyond the realm of linear programming), Proposition 3 allows us to investigate the trade-off between 1-norm and rank by gradually imposing the linear equations (4.3) on the linear program P_{13}. We propose standard mathematical-programming approaches to gradually enforce these equations, namely, a cutting-plane method, an augmented Lagrangian method, and two penalty methods, where we penalize the 1-norm and the Frobenius norm of the violation, given by the matrix $HAA^{\dagger} - H$, of the equations (4.3). The parameters adopted in these algorithms define both their initial solutions and how fast they converge to the ultimate solutions. When tuning these parameters, our objective is not to have a fast convergence. On the contrary, we aim to construct a nice set of ah-symmetric generalized inverses approximating the Pareto frontier corresponding to our two minimization objectives, 1-norm and rank. We design our algorithms to construct a minimum 1-norm ah-symmetric generalized inverse of a given matrix A at their first iteration, and we

want them to slowly converge to a minimum 1-norm ah-symmetric *reflexive* generalized inverse (which is necessarily a minimum 1-norm ah-symmetric generalized inverse having minimum rank). Along the way, by gradually imposing P2, we aim to get a sequence of ah-symmetric generalized inverses of decreasing rank, with each having low 1-norm relative to other ah-symmetric generalized inverses of its rank.

If not considering the particularity of our problem, given by the equivalence between P2 and the least-rank condition, a standard approach to balance objectives of having low 1-norm and low rank when constructing a matrix, is to solve a convex optimization problem where the nuclear norm is employed as a surrogate for the rank of the matrix, and then the objective is a weighted combination of 1-norm and nuclear norm. This problem can be recast as a semidefinite programming (SDP) problem and has been widely investigated in the literature (see Chandrasekaran, Sanghavi, Parrilo, and Willsky (2011); Mohan and Fazel (2012); Recht, Fazel, and Parrilo (2010), for example). We also apply this general approach to iteratively construct ah-symmetric generalized inverses. However, compared to the four approaches discussed above, this method has the disadvantage of requiring the solution of an SDP problem at each iteration, which generally does not scale well. Furthermore, using the nuclear norm as a surrogate for rank, we lose a nice feature of our other approaches; while the nuclear-norm approach will converge to an ah-symmetric generalized inverse that is reflexive, we lose the guarantee that it has minimum 1-norm among all such ah-symmetric reflexive generalized inverses.

4.2.1 Cutting-plane method for P2

We propose a cutting-plane method, where a linear program is solved at each iteration. We initially solve problem P_{13}, not imposing P2. Then, we consider the nm equations (4.3), i.e., $H_{i\cdot}A(A^\dagger)_{\cdot j} = H_{ij}$, lexically ordered by their indices (j, i), with $i = 1, \ldots, n$ and $j = 1, \ldots, m$. At each iteration of the cutting-plane method, we add the first t equations that are violated by the current solution, to P_{13}. We consider that equation (j, i) is violated, if $|(HAA^\dagger - H)_{ij}| > 10^{-6}$. Aiming at a slow convergence of the algorithm, and therefore at a diverse set of constructed matrices H, we limit t to 1% of the total number of constraints.

4.2.2 Augmented Lagrangian method: dualizing P2

Here, we initially consider a Lagrangian method, where we dualize the constraints (4.3). We solve the Lagrangian-dual problem

$$\begin{array}{ll} \max_{\Lambda} \min_{H,Z} & \langle J, Z\rangle + \left\langle \Lambda, HAA^{\dagger} - H\right\rangle \ , \\ \text{s.t.:} & Z - H \geq 0 \ , \\ & Z + H \geq 0 \ , \\ & AHA = A \ , \\ & (AH)^{\intercal} = (AH) \ , \end{array} \tag{4.4}$$

with a subgradient optimization algorithm. At each iteration, we solve the inner problem in (4.4) fixing the Lagrangian multiplier $\Lambda \in \mathbb{R}^{n\times m}$ at a value Λ_k and obtain its solution (H^k, Z^k). We calculate the subgradient $G_k := H^k AA^{\dagger} - H^k$ and update the Lagrangian multiplier according to $\Lambda_{k+1} := \Lambda_k + \gamma_k G_k$. The step size γ_k is $(z_{P_{123}} - z_k)/\|H^k AA^{\dagger} - H^k\|_F^2$ (the usual Polyak rule), where z_k is the current value of the Lagrangian function, i.e., of the objective function in (4.4).

We note that although we have a theoretical guarantee of convergence of the subgradient algorithm to the optimal value of P_{123} when applying the Lagrangian method, there is no guarantee that a feasible solution for P_{123} is obtained. We actually noticed in preliminary numerical experiments that the rank of H did not converge to the rank of A, i.e., P2 was not satisfied by the solution obtained when the algorithm converged. To overcome this drawback, we have considered the augmented Lagrangian method, adding the convex quadratic term $\frac{\mu_k}{2}\|HAA^{\dagger} - H\|_F^2$ to the objective function of the Lagrangian subproblem. At iteration k of the algorithm, we solve the convex quadratic problem

$$\begin{array}{ll} \min_{H,Z} & \langle J, Z\rangle + \left\langle \Lambda_k, HAA^{\dagger} - H\right\rangle + \frac{\mu_k}{2}\|HAA^{\dagger} - H\|_F^2 \ , \\ \text{s.t.:} & Z - H \geq 0 \ , \\ & Z + H \geq 0 \ , \\ & AHA = A \ , \\ & (AH)^{\intercal} = (AH) \ , \end{array}$$

for fixed μ_k and Λ_k, obtaining its optimal solution (H_k, Z_k). After solving the problem, we update the penalty parameter according to: $\mu_{k+1} = 1.30\mu_k$ and then we update the dual variable according to $\Lambda_{k+1} := \Lambda_i + \mu_{k+1}(H_k AA^{\dagger} - H_k)$. We initialize the algorithm with $\Lambda_0 := 0$ and $\mu_0 := 0.1$.

4.2.3 Penalty method: Frobenius norm of P2 violation

Now we consider penalty methods, penalizing the squared Frobenius norm of $HAA^\dagger - H$. At iteration k of the algorithm, we solve the convex quadratic problem

$$\begin{array}{ll} \min_{H,Z} & \langle J, Z\rangle + \mu_k \|HAA^\dagger - H\|_F^2 \ , \\ \text{s.t.:} & Z - H \geq 0 \ , \\ & Z + H \geq 0 \ , \\ & AHA = A \ , \\ & (AH)^\intercal = (AH) \ , \end{array}$$

for fixed μ_k. After solving the problem, we update the penalty parameter according to: $\mu_{k+1} := 1.30\mu_k$. We initialize the algorithm with $\mu_0 := 0.1$.

4.2.4 Penalty method: 1-norm of P2 violation

We also experimented penalizing the 1-norm of $HAA^\dagger - H$. Besides the modification on the norm with respect to the penalty method described in the previous subsection, in this algorithm, we have tuned the parameters to $\mu_0 := 0.01$ and $\mu_{k+1} := 1.15\mu_k$. We note that the use of the 1-norm instead of the squared Frobenius norm leads to the solution of a linear program at each iteration, instead of a convex quadratic program.

4.2.5 Nuclear-norm method

An standard approach to obtain a balance between the two goals of constructing a sparse matrix and a low-rank matrix, is to minimize a weighted sum of its 0-norm and rank. In our context this would lead to the following problem

$$\min\{\|H\|_0 + \mu \cdot r(H) \ : \ P1, P3\}, \tag{4.5}$$

where μ corresponds to the weight given to the rank of H ($r(H)$) in the minimization. To address a convex optimization problem, it is usual then to employ the 1-norm as a surrogate for the 0-norm, as we have done with the previous approaches proposed, and the nuclear norm as a surrogate for the rank as well. Applying this method to our problem, we solve at iteration k of our last proposed algorithm, the following problem

$$\min\{\|H\|_1 + \mu_k \|H\|_* \ : \ P1, P3\}, \tag{4.6}$$

where μ_k is a fixed real penalty parameter and $\|H\|_* := \sum_k \sigma_k(H)$ (the nuclear norm), where $\sigma(H) := (\sigma_1(H), ..., \sigma_n(H))$ is the vector of singular values of H. It is convenient and usual to assume that the singular values are ordered so that

$\sigma_1(H) \geq \ldots \geq \sigma_n(H)$. After solving the problem, we update the penalty parameter according to $\mu_{k+1} := 1.20\mu_k$. We initialize the algorithm with $\mu_0 := 0.05$.

Problem (4.6) can be recast as an SDP problem. Using the facts that the nuclear norm is dual to the spectral norm and that the spectral norm admits a simple semidefinite characterization, minimizing $\|H\|_*$ can be formulated as

$$\min_{W_1,W_2} \left\{ \frac{1}{2}(\text{trace}(W_1) + \text{trace}(W_2)) \; : \; \begin{pmatrix} W_1 & H \\ H^T & W_2 \end{pmatrix} \succeq 0 \right\}.$$

Therefore, at each iteration k of our algorithm, we solve the SDP problem

$$\begin{array}{l} \min\limits_{H,Z,W_1,W_2} \langle J, Z\rangle + \mu_k \left(\dfrac{1}{2}(\text{trace}(W_1) + \text{trace}(W_2)) \right) \\ \text{subject to:} \\ \begin{pmatrix} W_1 & H \\ H^T & W_2 \end{pmatrix} \succeq 0, \; -Z \leq H \leq Z, \; AHA = A, \; (AH)^\intercal = (AH), \end{array} \tag{4.7}$$

where the matrix variables W_1 and W_2 are of order $n \times n$ and $m \times m$, respectively.

4.3 Numerical results

To analyze the results of the procedures proposed, we randomly generated 20 dense square matrices of size 50×50 and rank $r = 25$. We used the Matlab function *sprand*, which generates a random $m \times n$ dimensional matrix A with singular values given by a nonnegative input vector rc. We selected the r nonzeros of rc as the decreasing vector $M \times (\rho^1, \rho^2, \ldots, \rho^r)$, where $M = 2$, and $\rho = (1/M)^{(2/(r+1))}$. We implemented our algorithms in Python (Spyder 3.3.6), and ran the experiments on a 16-core machine (running Windows Server 2016 Standard): two Intel Xeon CPU E5-2667 v4 processors running at 3.20GHz, with 8 cores each, and 128 GB of memory. We solve all the optimization problems involved in our experiments with Gurobi v.9, except problem (4.7), which we solve with Mosek.

In our numerical analysis, when we refer to the rank and 0-norm of a generalized inverse computed by our algorithms, we mean, respectively, the number of singular values greater than 10^{-5} and the number of elements of the matrix with absolute values greater than 10^{-6}. When we refer to the number of constraints in (4.3) that are satisfied by a given matrix H, we mean the number of elements of the matrix $HAA^\dagger - H$ with absolute value less than 10^{-6}. We denote optimal solutions of P_{13} and P_{123}, respectively by H_{13} and H_{123}. Most of the results presented in this section are average results over our 20 test-instances.

In Table 4.1, we analyze the solutions which the methods proposed in the previous

section converge to, and compare them to the M-P pseudoinverse $A^\dagger$, to H_{13}, and to H_{123}. The stopping criterion for all of our algorithms is the same. They stop when for the constructed matrix H, we have $|(HAA^\dagger - H)_{ij}| \leq 10^{-6}$, for $i = 1, \ldots, n$ and $j = 1, \ldots, m$. We present in the table, the average number of iterations executed by the algorithms, and the average 1-norm, 0-norm and rank of the solutions obtained. We note that all constraints in (4.3) are satisfied by the solutions of the methods proposed, and therefore they all have the minimum rank $r = 25$.

	It	$\|H\|_1$	$\|H\|_0$	$r(H)$
$A^\dagger$	-	157.2	2384.0	25
H_{13}	-	131.7	1218.6	43.8
H_{123}	-	137.9	1581.6	25
Cutting-plane	45.7	137.5	1596.6	25
Aug. Lagrangian	46.4	137.5	1597.0	25
Pen. 1-norm	33.8	137.5	1596.7	25
Pen. Frobenius	60.8	137.5	1597.4	25
Nuclear-Norm	23.4	142.4	1897.7	25

Table 4.1: Comparison of solutions of the methods proposed with $A^\dagger$, H_{13}, and H_{123}

From the results in Table 4.1, we note that the M-P pseudoinverse has significantly greater 1-norm and density (0-norm) than H_{123}, which indicates a significant advantage of using our minimum 1-norm ah-symmetric reflexive generalized inverse in numerical applications. In fact, the M-P pseudoinverse has about 95% nonzero elements, on average, while H_{123} has only about 63%. Comparing H_{13} to H_{123}, we see that when imposing the reflexive property P2 on the ah-symmetric generalized inverse, we obtain an average decrease of 43% in the rank of the matrix at the expense of increasing the 1-norm and the 0-norm by approximately 4.5% and 30%, respectively. In summary, if we insist on minimum rank then we would use H_{123}. But we can alternatively use H_{13}, gaining considerable decreases in 1-norm and density, but with much greater rank. The space between these extreme alternatives is what we aim to explore. We also observe from the results that, except for the Nuclear-norm method, the solutions H obtained by all the methods have, on average, the same 1-norms and only slightly different 0-norms. The 1-norms are slightly less than the 1-norm of H_{123}, because equations (4.3) are only satisfied up to a tolerance. The 0-norms are only slightly greater than the 0-norm of H_{123}. The Nuclear-norm method gains full rank only at the expense of barely weighting the 1-norm.

In the next experiment, we analyze intermediate solutions obtained by the methods with the purpose of seeing their ability to construct solutions with different 1-norms and ranks as P2 is gradually imposed. Our goal now, is to analyze the

trade-off between the 1-norms and 0-norms of the intermediate solutions and their ranks.

In Table 4.2, we present statistics for the first iteration of each method in which the rank of the computed generalized inverse H has sufficiently decreased, more specifically, for the first iteration where the rank of H satisfies

$$r(H) \leq (1-\alpha) \cdot r(H_{13}) + \alpha \cdot r(H_{123}), \tag{4.8}$$

for $\alpha = 0.00, 0.25, 0.50, 0.75, 1.00$. We tabulate the average percentage increase in the 1-norm and 0-norm of H compared to the initial matrix H_{13}, i.e.,

$$\widehat{\|H\|}_i := \frac{\|H\|_i - \|H_{13}\|_i}{\|H_{13}\|_i} \times 100,$$

for $i = 0, 1$. For each α, we also present $r(H)$, given by (4.8), the average percentage of constraints in (4.3) that are satisfied by the intermediate solution H ('P2') and the average iteration in which it is computed ('It').

		Cutting-plane				Aug. Lagrangian			
α	$r(H)$	$\widehat{\|H\|}_1$	$\widehat{\|H\|}_0$	P2	It	$\widehat{\|H\|}_1$	$\widehat{\|H\|}_0$	P2	It
0.00	43.8	0.0	0.0	14.8	0	0.0	0.0	14.9	0
0.25	39.1	2.5	15.1	36.2	18	4.3	29.0	50.2	31
0.50	34.4	3.4	21.5	46.4	27	4.3	29.4	69.1	33
0.75	29.7	4.1	28.1	59.6	36	4.3	29.8	87.5	36
1.00	25.0	4.3	31.0	100.0	51	4.3	30.2	100.0	51

		Pen. 1-norm				Pen. Frobenius				Nuclear-Norm			
α	$r(H)$	$\widehat{\|H\|}_1$	$\widehat{\|H\|}_0$	P2	It	$\widehat{\|H\|}_1$	$\widehat{\|H\|}_0$	P2	It	$\widehat{\|H\|}_1$	$\widehat{\|H\|}_0$	P2	It
0.00	43.8	0.0	0.0	15.4	0	0.0	0.0	14.9	0	0.0	0.0	14.7	0
0.25	39.1	4.3	29.6	68.4	29	4.3	30.4	50.5	50	3.1	24.6	12.7	17
0.50	34.4	4.3	29.9	84.5	30	4.3	30.3	71.3	53	4.8	34.7	12.1	19
0.75	29.7	4.3	30.0	92.4	31	4.3	30.3	85.7	55	6.5	45.3	11.5	21
1.00	25.0	4.3	30.1	100.0	35	4.3	30.2	100.0	61	8.1	53.7	100.0	24

Table 4.2: Trade-off between norms and rank for ah-symmetric generalized inverses

From the results in Table 4.2, we observe that

- the Cutting-plane method presents the best trade-off between norms and rank of H. Approximately 35% of the iterations are executed before the algorithm finds the first matrix H with rank satisfying (4.8) for $\alpha = 0.25$, but matrices with different norms are computed for each $\alpha = 0.0, 0.25, 0.50, 0.75$. Both 1-norm and 0-norm increase as α gradually increase in this interval;

- the methods Augmented Lagrangian, Penalized 1-norm, and Penalized Frobenius have similar behavior. For the values of α considered in Table 4.2, they

only start to decrease the rank of H when the 1-norm reaches its maximum value. Therefore, for the values of α considered in this experiment, the solutions generated by these methods do not show the desired trade-off between 1-norm and rank;

- on average, the 1-norm and the 0-norm of the final solutions of all methods, except Nuclear-norm, are, respectively, 4% and 30% greater than the norms of the initial matrix H_{13}. We can see that the 0-norm increases as the 1-norm increases, demonstrating that the 1-norm works as a good surrogate for the 0-norm in this experiment;

- all methods, except Nuclear-norm, converge to matrices H with similar norms that satisfy all the constraints in (4.3);

- Nuclear-norm converges to worse solutions than the other methods proposed. The solutions have greater norms on average. Unlike we do in the other methods, we do not enforce the constraints (4.3) in this one. Therefore, the number of constraints in (4.3) that are satisfied, is very small during the execution of the method, it increases very fast only in the last iterations, when the rank of H reaches its lowest value. We see that the algorithm converges to a solution H of least rank, but with 1-norm greater than $\|H_{123}\|_1$, which is due to the fact that (4.7) is not a relaxation for P_{123}, in contrast to the problems solved by the other methods. Although we can observe a gradual increase in the norms for this method, because of the higher norms, it is not a good option to generate intermediate ah-symmetric generalized inverses.

In conclusion, from the results in Table 4.2, we see that among the first four methods, which converge to matrices of similar norms, Cutting-plane is the only one that shows a gradual increase in the norms for the values of α selected. The others only start to decrease the rank of H after its 1-norm reaches its greatest value. Nuclear-norm also shows a gradual increase in the norm, however, it converges to solutions of higher norms (on average). Therefore, our experiment points to Cutting-plane as the most suitable among the proposed methods to trade-off 1-norm against rank for ah-symmetric generalized inverses.

In Figures 4.1 to 4.5, we present plots to better show the behavior of the methods proposed as they iterate. The plots depict average values for $\widehat{\|H\|}_1$, $\|HAA^\dagger - H\|_F$, $r(H)$, and the percentage of the constraints in (4.3) that are satisfied.

Comparing the plots for the Cutting-plane method, in Figure 4.1, with the plots for Augmented Lagrangian, Penalized 1-norm and Penalized Frobenius, in Figures 4.2, 4.3, 4.4, we can more clearly observe that during the execution of Cutting-plane the rank starts to decrease when the 1-norm is still increasing and the percentage of

constraints in (4.3) that are satisfied also increases faster from the beginning. The behavior of the solutions of the three other methods is very similar. Nevertheless, Penalized 1-norm shows a slightly better trade-off than the other two. We can observe a small decrease in the rank before the 1-norm achieves its optimal value. In Figure 4.5, we see that Nuclear-norm converges in fewer iterations than the other methods, but in terms of computational effort, it is the most expensive method, owing to the need to solve SDP problems. As guaranteed by the theory, all of the constraints (4.3) are satisfied when the rank is minimum, but only a few of them are satisfied until the last iterations. Despite this fact, the curves show the desired decrease in the rank as the 1-norm increases. However, as observed in Table 4.2 the 1-norm increases significantly more for this method than for the others. We have an increase of 8.1% in the 1-norm, while for the others it is only 4.3%.

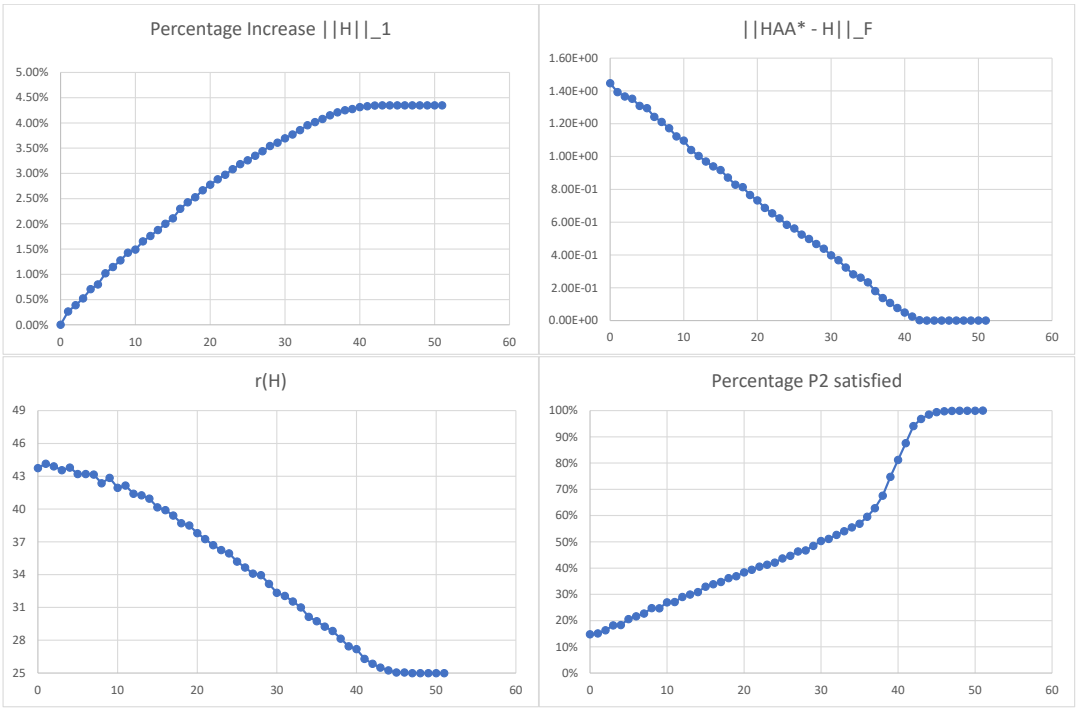


Figure 4.1: Cutting-plane method

As Cutting-plane presented the best trade-off between the rank and the 1-norm of the solutions in the experiments discussed above, we further investigated if other ways of selecting the violated constraints to add to the problem at each iteration could improve even more the results. The best results were obtained when the violated constraints were selected randomly. So, in the following, we include in our analysis, a variation of Cutting-plane called Cutting-plane Random. Recall that for the original Cutting-plane, we consider the nm equations (4.3), i.e., $H_{i\cdot}A(A^{\dagger})_{\cdot j} = H_{ij}$, lexically ordered by their indices (j, i), with $i = 1, \ldots, n$ and $j = 1, \ldots, m$. At each iteration of Cutting-plane, we add the first t equations that are violated by the cur-

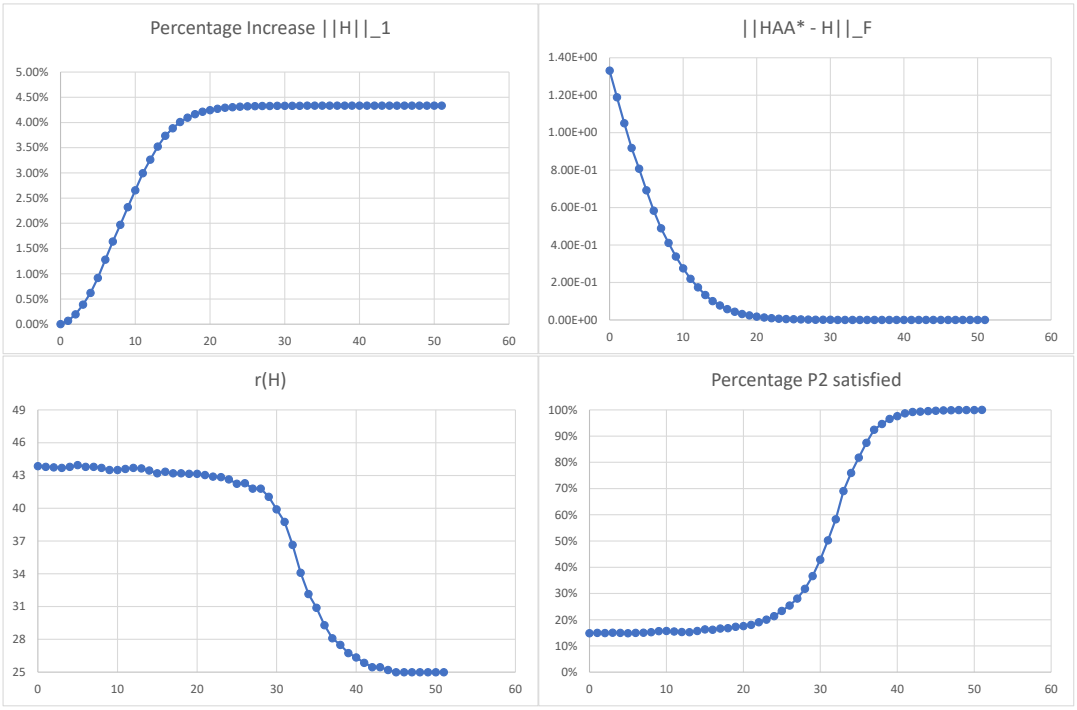


Figure 4.2: Augmented Lagrangian method

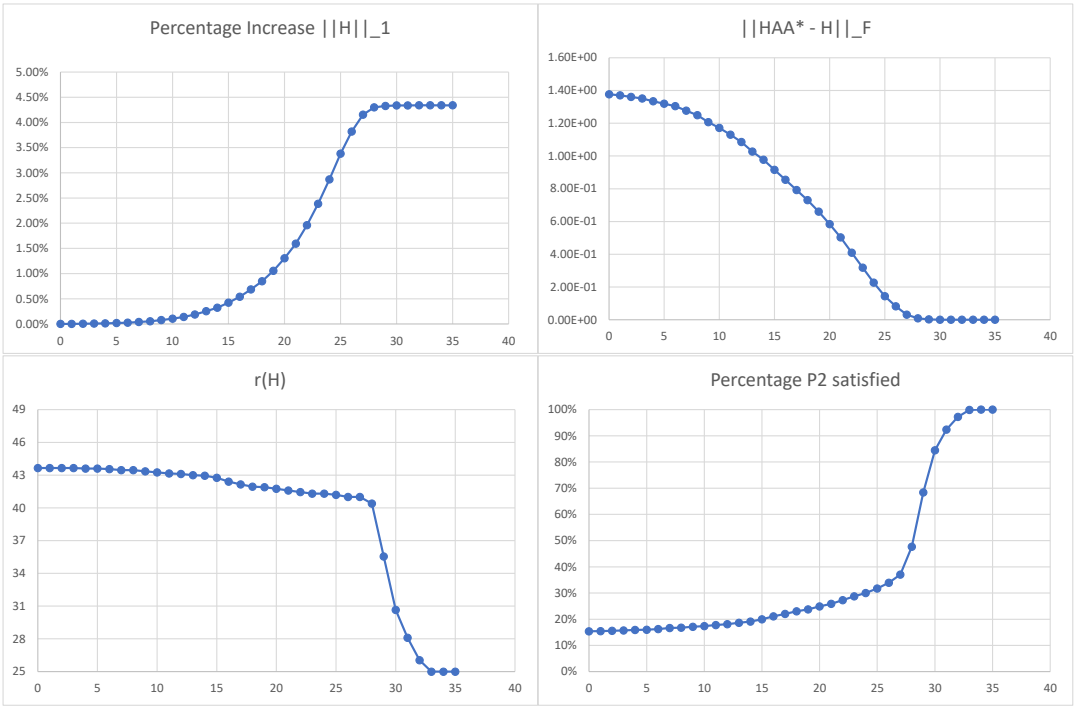


Figure 4.3: Penalized 1-norm method

rent solution, to P_{13}. Alternatively, in Cutting-plane Random, we randomly reorder the indices j or the indices i before executing it. We execute the method twenty times, in the first ten, we randomly reorder j and in the last ten, besides randomly reordering i, we search considering the lexical order of (i, j). The solution of mini-

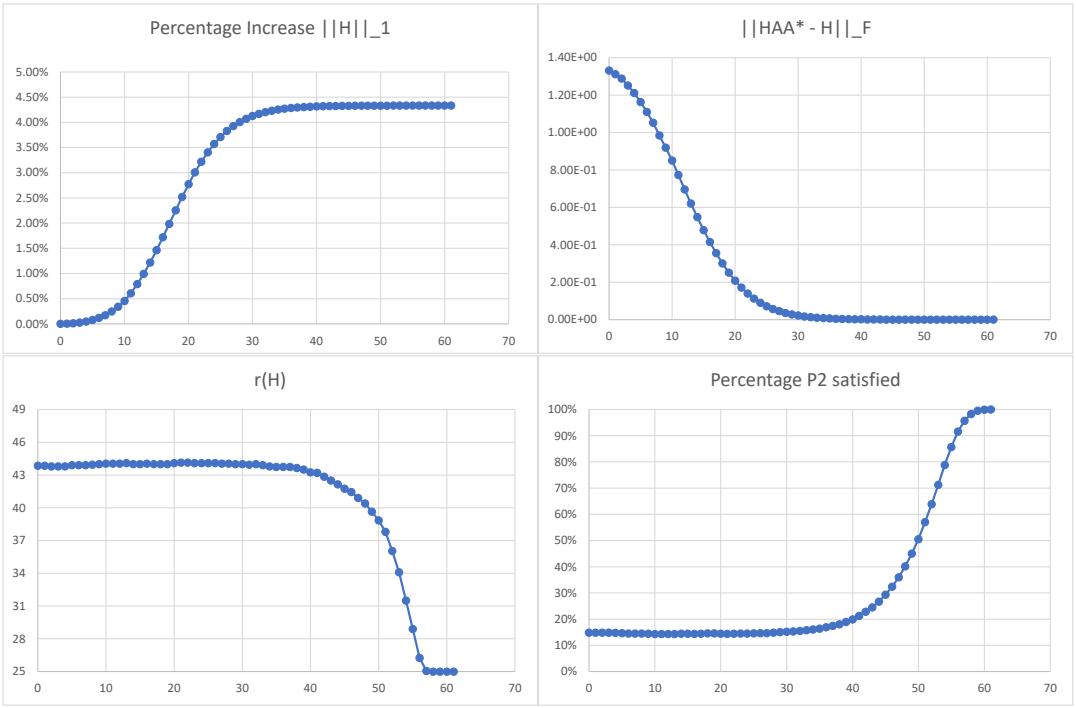


Figure 4.4: Penalized Frobenius method

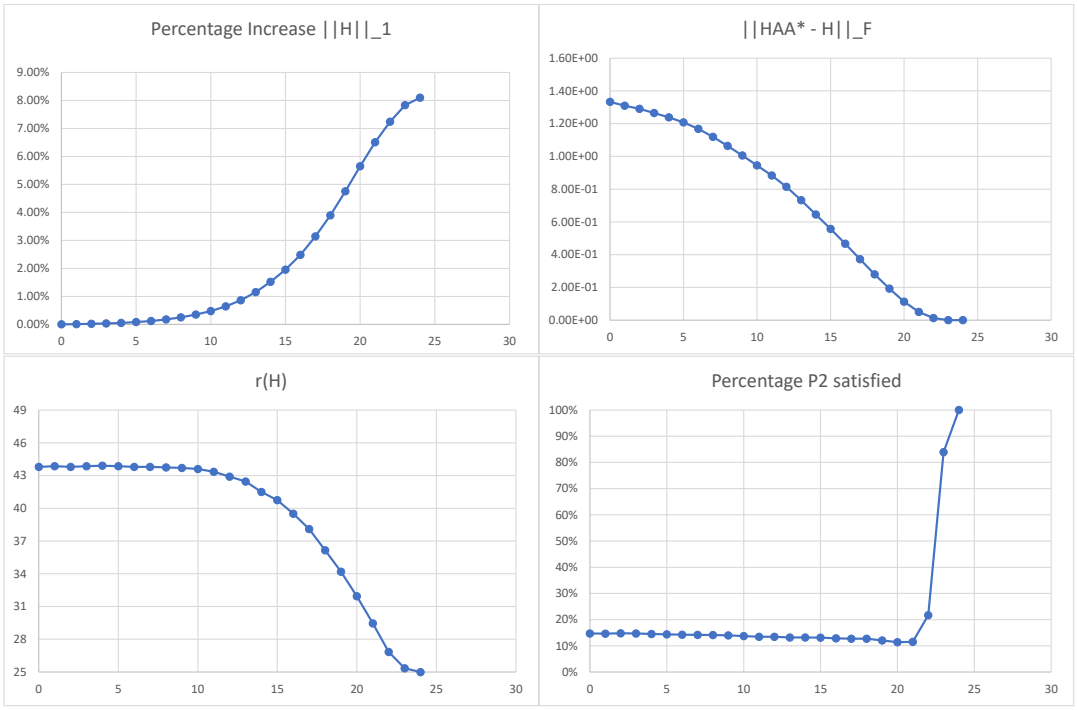


Figure 4.5: Nuclear-norm method

mum 1-norm, for each rank found in the solutions is then presented. We will compare in the following Cutting-plane (CP), Cutting-plane Random (CP.Rand), Penalized 1-norm, and Nuclear-norm. The two other methods, Augmented Lagrangian and Penalized Frobenius, are not considered in the following because their behavior is

very similar to Penalized 1-norm, only a bit worse.

	$\|H\|_1$				$\|H\|_0$			
$r(H)$	Nuclear-norm	Pen.1-norm	CP	CP.Rand	Nuclear-norm	Pen.1-norm	CP	CP.Rand
48	(1, 0)	(1, 0)	(1, 1)	(1, 1)	(1, 0)	(1, 0)	(1, 1)	(1, 1)
47	(3, 0)	(3, 0)	(3, 2)	(3, 3)	(3, 0)	(3, 0)	(3, 2)	(3, 3)
46	(5, 1)	(5, 0)	(5, 2)	(5, 4)	(5, 0)	(5, 0)	(5, 3)	(5, 4)
45	(7, 0)	(5, 1)	(8, 3)	(8, 7)	(7, 0)	(5, 0)	(8, 4)	(8, 7)
44	(11, 0)	(9, 2)	(11, 3)	(11, 9)	(11, 0)	(9, 0)	(11, 5)	(11, 10)
43	(13, 0)	(8, 2)	(15, 6)	(15, 12)	(13, 0)	(8, 0)	(15, 8)	(15, 13)
42	(14, 1)	(11, 1)	(16, 3)	(16, 12)	(14, 0)	(11, 0)	(16, 7)	(16, 12)
41	(15, 0)	(12, 2)	(19, 5)	(19, 15)	(15, 0)	(12, 0)	(19, 7)	(19, 15)
40	(13, 1)	(9, 3)	(19, 3)	(19, 12)	(13, 1)	(9, 1)	(19, 8)	(19, 11)
39	(7, 0)	(8, 0)	(19, 5)	(19, 14)	(7, 0)	(8, 0)	(19, 8)	(19, 12)
38	(8, 0)	(3, 0)	(18, 2)	(19, 17)	(8, 0)	(3, 0)	(18, 4)	(19, 15)
37	(13, 0)	(4, 0)	(20, 5)	(20, 16)	(13, 0)	(4, 0)	(20, 9)	(20, 13)
36	(8, 0)	(5, 0)	(20, 3)	(20, 17)	(8, 0)	(5, 0)	(20, 7)	(20, 13)
35	(10, 0)	(5, 0)	(20, 1)	(20, 19)	(10, 0)	(5, 0)	(20, 7)	(20, 13)
34	(11, 0)	(4, 1)	(20, 1)	(20, 18)	(11, 0)	(4, 0)	(20, 7)	(20, 13)
33	(9, 0)	(5, 1)	(20, 1)	(20, 18)	(9, 0)	(5, 0)	(20, 8)	(20, 13)
32	(8, 0)	(6, 1)	(20, 1)	(20, 18)	(8, 0)	(6, 1)	(20, 8)	(20, 12)
31	(9, 0)	(5, 0)	(20, 1)	(20, 19)	(9, 0)	(5, 0)	(20, 8)	(20, 12)
30	(7, 0)	(5, 0)	(20, 1)	(20, 19)	(7, 0)	(5, 0)	(20, 5)	(20, 16)
29	(9, 0)	(8, 0)	(19, 4)	(20, 16)	(9, 0)	(8, 0)	(19, 11)	(20, 10)
28	(4, 0)	(12, 0)	(20, 3)	(20, 17)	(4, 0)	(12, 0)	(20, 5)	(20, 17)
27	(12, 0)	(6, 0)	(20, 2)	(20, 18)	(12, 0)	(6, 1)	(20, 6)	(20, 14)
26	(12, 0)	(12, 0)	(20, 3)	(20, 18)	(12, 0)	(12, 0)	(20, 6)	(20, 15)
25	(20, 0)	(20, 2)	(20, 6)	(20, 18)	(20, 0)	(20, 19)	(20, 17)	(20, 14)

Table 4.3: Number of instances where the methods find (at least one solution, the solution of minimum norm)

In Table 4.3, each row corresponds to the rank of at least one solution generated by the four methods when applied to our twenty instances. For each method, we show results concerning the 1-norm and the 0-norm of the solutions obtained. Each ordered pair presented in the table, contains the number of instances where the methods find at least one solution for the corresponding rank, and the number of instances where the method finds the least 1-norm (up to a tolerance of 10^{-4}), for that rank, among all of the algorithms. A method for which the first coordinate is large indicates the ability of that method to generate a diverse set of solutions (i.e., with a variety of ranks), when trading-off rank against 1-norm. The second coordinate indicates when a method obtain the best solution among all for that rank, hence giving a nondominated solution, in the Pareto sense, considering the points $(\|H\|, r(H))$. From the results in Table 4.3, we observe:

- On average, the number of solutions of each rank is much smaller for Penalized 1-norm. Both Cutting-plane and Cutting-plane Random generates the same or nearly that same number of solutions of each rank, which are on average, much greater than the numbers for the other two methods.

- Nuclear-norm only obtains solutions of minimum 1-norm for three ranks, and for only one instance for each rank.

- Cutting-plane Random obtains the solution of minimum 1-norm in the vast

majority of the cases, showing much superior results. The second best method is Cutting-plane.

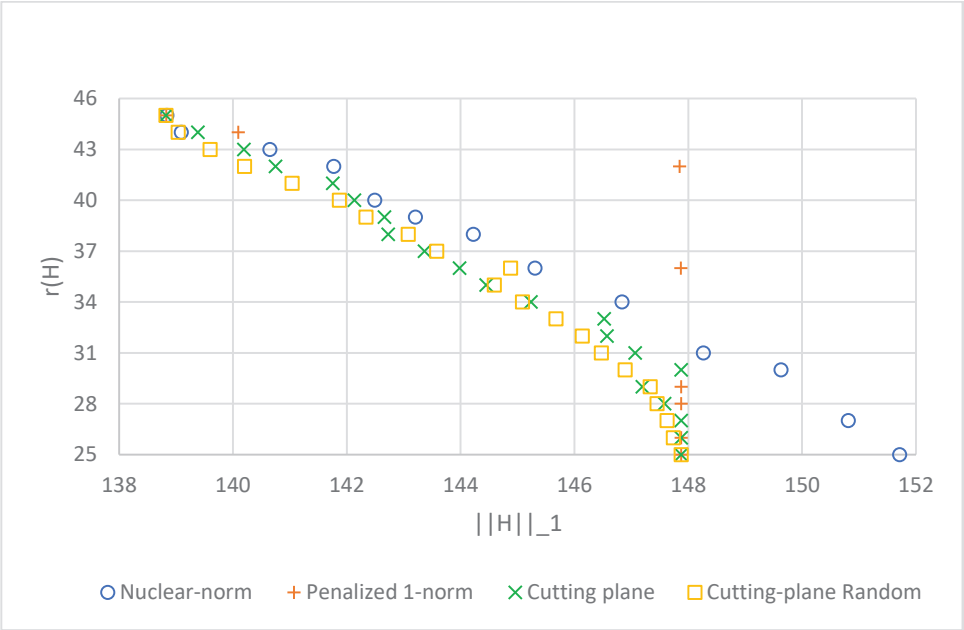
- The results for the 0-norm are very similar to the results for the 1-norm, when analyzing Nuclear-norm and Penalized 1-norm, but when observing the cutting-plane methods, we can see that the results are more balanced when comparing the 0-norms.

	$\|H\|_1$				$\|H\|_0$			
Inst.	Nuclear-norm	Pen.1-norm	CP	CP.Rand	Nuclear-norm	Pen.1-norm	CP	CP.Rand
1	0	0	8	16	0	1	14	10
2	0	0	5	15	0	1	7	11
3	0	0	3	21	0	1	6	14
4	0	4	2	17	0	2	5	14
5	0	0	1	16	0	1	9	9
6	0	1	4	12	0	1	6	9
7	0	0	2	19	0	1	4	14
8	1	0	9	7	1	1	5	8
9	0	1	1	17	0	1	5	14
10	1	1	2	16	0	1	8	10
11	0	2	3	15	0	1	6	15
12	0	0	1	20	0	1	11	15
13	0	3	2	18	0	1	8	9
14	0	0	1	18	0	1	6	14
15	0	3	1	10	0	2	4	7
16	0	0	6	16	0	1	7	17
17	1	0	7	17	0	1	14	10
18	0	0	3	21	0	1	8	15
19	0	1	1	22	0	0	8	13
20	0	0	3	19	0	1	4	18

Table 4.4: Number of solutions nondominated by other methods

Table 4.4 presents results of the same experiments (as presented in Table 4.3), but now the rows indicate "instance" rather than "rank", and we tabulate the number of nondominated solutions generated by instance, in the Pareto sense, considering the points $(\|H\|, r(H))$. We can clearly see that Cutting-plane Random is the best, with Cutting-plane also quite good. Penalized 1-norm generates a few points that are nondominated, and Nuclear norm rarely does.

The plots in Figure 4.6 show the typical iterates generated by the four methods on one of our twenty random instances. These plots expand on the experiments summarized in "Row 16" of Table 4.4. We can see that Cutting-plane Random gives the best approximation of the Pareto curves, trading off 1/0-norms against rank. Nuclear-norm performs quite poorly at generating H with low 1/0-norms for most ranks; it is particularly bad at the lower ranks. Even though Penalized 1-norm is overall quite poor, it can generate H with decent 1/0-norms at very low and very high ranks. Overall the Cutting-plane methods are the clear winners, with Cutting-plane Random offering some improvement over Cutting-plane.



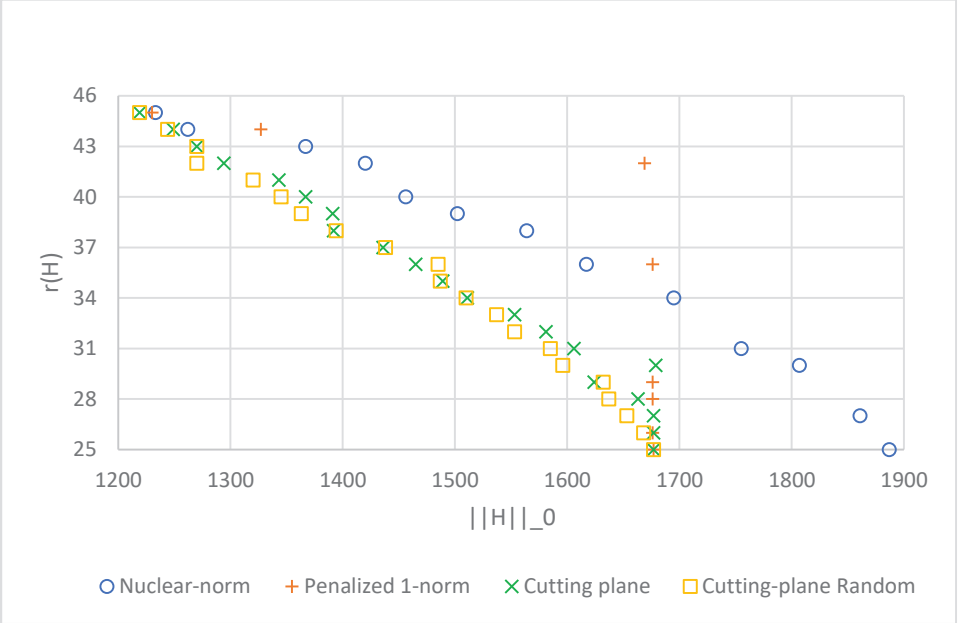


Figure 4.6: Pareto-curve approximations for an instance: four algorithms

4.4 Concluding remarks

There is a trade-off between 1-norm and sparsity against low-rank in the computation of ah-symmetric generalized inverses. These matrices can be applied in the solution of least-square problems, where low 1-norm, sparsity, and low-rank are all desirable features in their numerical applications. In this work, we propose algorithmic approaches to numerically analyze this trade-off, using the fact that a generalized inverse has minimum rank if and only if it satisfies the reflexive property. The algorithms start with a minimum 1-norm ah-symmetric generalized inverse and iteratively impose the reflexive property, gradually increasing 1-norm and decreasing

rank, until the minimum rank is obtained. The intermediate solutions from these algorithms can represent the best trade-off between 1-norm and sparsity against rank in numerical applications. Among the different strategies proposed to gradually impose the reflexive property, the best trade-off between norms and rank is obtained by a cutting-plane method that solves linear-programming problems at each iteration, applying a linear re-formulation of the reflexive property, obtained when combining it with the other properties of ah-symmetric generalized inverses.

Chapter 5

On computing sparse generalized inverses

This chapter corresponds to the work that has been published as:
Gabriel Ponte, Marcia Fampa, Jon Lee, Luze Xu. On computing sparse generalized inverses. Operations Research Letters 52, 2024. https://doi.org/10.1016/j.orl.2023.107058

5.1 Introduction

It is well known that structured sparsity is more useful than sparsity, because of computational efficiency and for explainability. In the context of the least-squares application of an ah-symmetric generalized inverse H, it is desirable to have few rows that have non-zeros, as then the associated linear model is more explainable. Because no generalized inverse of A can have rank less than $\text{rank}(A)$, every generalized inverse of A has at least $\text{rank}(A)$ rows with non-zeros. So a *row-sparse* generalized inverse of A is one having or nearly having $\text{rank}(A)$ rows with non-zeros. Similarly, we can speak of *column-sparse* generalized inverses.

Overview. In §5.2, we establish new structural results concerning generalized inverses of different types. In §5.3 (respectively, §5.4), we investigate minimizing a component-wise increasing function of the vector of 2-norms of the rows (resp., columns) of a generalized inverse. We establish that every such minimizer satisfies P2 and P3 (resp., P4). In §5.5, we show the value of our results in §5.3 by demonstrating how mathematical-optimization formulations related to finding row-sparse generalized inverses can be solved rather efficiently. In the interest of space, we do not do this also for column-sparse generalized inverses. In §5.6, we review the very-fast local-search procedure of Xu, Fampa, Lee, and Ponte (2021) for finding a row-sparse ah-symmetric generalized inverse with low 1-norm. In §5.7, we provide

computational results indicating that the LP-based methods tested in Fampa, Lee, Ponte, and Xu (2021) can be made to scale much better, using our results in §5.5. Nevertheless, we can observe that the local-search procedure of Xu, Fampa, Lee, and Ponte (2021) continues to scale better and maintains other desirable properties. Still, we can see that LP (and even more sophisticated exact optimization models) becomes a more viable option than was previously believed.

5.2 Decomposing generalized inverses

Let $A \in \mathbb{R}^{m\times n}$ with rank r and $A =: U\Sigma V^\intercal$ be the singular-value decomposition of A, where $U \in \mathbb{R}^{m\times m}$, $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices ($U^\intercal U = I_m, V^\intercal V = I_n$), and $\Sigma \in \mathbb{R}^{m\times n}$ with

$$\Sigma =: \begin{bmatrix} \underset{r\times r}{D} & \underset{r\times(n-r)}{0} \\ \underset{(m-r)\times r}{0} & \underset{(m-r)\times(n-r)}{0} \end{bmatrix},$$

with D being a diagonal matrix with rank r. Let $H \in \mathbb{R}^{n\times m}$ and $\Gamma := V^\intercal H U$, where

$$\Gamma =: \begin{bmatrix} \underset{r\times r}{X} & \underset{r\times(m-r)}{Y} \\ \underset{(n-r)\times r}{Z} & \underset{(n-r)\times(m-r)}{W} \end{bmatrix},$$

then $H = I_n H I_m = (VV^\intercal)H(U^\intercal U) = V\Gamma U^\intercal$.

Considering the notation above, we present in the following, results related to the M-P pseudoinverse properties that will be used to prove our main results in the next section. We wish to emphasize that the results that follow refer to the singular-value decomposition of A and associated structural notation for Σ and Γ above.

Lemma 27. (Structure of generalized inverses). *P1 is equivalent to* $X = D^{-1}$.

Proof.

$$
\begin{aligned}
&AHA = A \Leftrightarrow \\
&U\Sigma V^\intercal V \Gamma U^\intercal U \Sigma V^\intercal = U \Sigma V^\intercal \Leftrightarrow \\
&U\Sigma\Gamma\Sigma V^\intercal = U\Sigma V^\intercal \Leftrightarrow \\
&\Sigma\Gamma\Sigma = \Sigma \Leftrightarrow \\
&\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} DX & DY \\ 0 & 0 \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} DXD & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \Leftrightarrow \\
&D^{-1}DXDD^{-1} = D^{-1}DD^{-1} \Leftrightarrow \\
&X = D^{-1}.
\end{aligned}
$$

□

Lemma 28. (Structure of reflexive generalized inverses). *If P1 is satisfied, then P2 is equivalent to* $ZDY = W$.

Proof.

$$
\begin{aligned}
&HAH = H \Leftrightarrow \\
&V\Gamma U^\intercal U \Sigma V^\intercal V \Gamma U^\intercal = V \Gamma U^\intercal \Leftrightarrow \\
&V\Gamma\Sigma\Gamma U^\intercal = V\Gamma U^\intercal \Leftrightarrow \\
&V^\intercal V\Gamma\Sigma\Gamma U^\intercal U = V^\intercal V\Gamma U^\intercal U \Leftrightarrow \\
&\Gamma\Sigma\Gamma = \Gamma \Leftrightarrow \\
&\begin{bmatrix} X & Y \\ Z & W \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} = \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} XD & 0 \\ ZD & 0 \end{bmatrix} \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} = \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} XDX & XDY \\ ZDX & ZDY \end{bmatrix} = \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} X & Y \\ Z & ZDY \end{bmatrix} = \begin{bmatrix} X & Y \\ Z & W \end{bmatrix} \Leftrightarrow \\
&ZDY = W.
\end{aligned}
$$

□

Lemma 29. (Structure of ah-symmetric generalized inverses). *If P1 is satisfied, then P3 is equivalent to $Y = 0$.*

Proof.

$$
\begin{aligned}
&AH = (AH)^\intercal \Leftrightarrow \\
&U\Sigma V^\intercal V\Gamma U^\intercal = (U\Sigma V^\intercal V\Gamma U^\intercal)^\intercal \Leftrightarrow \\
&U\Sigma\Gamma U^\intercal = (U\Sigma\Gamma U^\intercal)^\intercal \Leftrightarrow \\
&U\Sigma\Gamma U^\intercal = U\Gamma^\intercal\Sigma^\intercal U^\intercal \Leftrightarrow \\
&U^\intercal U\Sigma\Gamma U^\intercal U = U^\intercal U\Gamma^\intercal\Sigma^\intercal U^\intercal U \Leftrightarrow \\
&\Sigma\Gamma = \Gamma^\intercal\Sigma^\intercal \Leftrightarrow \\
&\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} X & Y \\ Z & W \end{bmatrix} = \begin{bmatrix} X^\intercal & Z^\intercal \\ Y^\intercal & W^\intercal \end{bmatrix}\begin{bmatrix} D^\intercal & 0 \\ 0 & 0 \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} DX & DY \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} X^\intercal D^\intercal & 0 \\ Y^\intercal D^\intercal & 0 \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} I_r & DY \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} I_r & 0 \\ (DY)^\intercal & 0 \end{bmatrix} \Leftrightarrow \\
&Y = 0.
\end{aligned}
$$

□

Lemma 30. (Structure of ha-symmetric generalized inverses). *If P1 is satisfied, then P4 is equivalent to $Z = 0$.*

Proof.

$$
\begin{aligned}
&HA = (HA)^\intercal \Leftrightarrow \\
&V\Gamma U^\intercal U\Sigma V^\intercal = (V\Gamma U^\intercal U\Sigma V^\intercal)^\intercal \Leftrightarrow \\
&V\Gamma\Sigma V^\intercal = V\Sigma^\intercal\Gamma^\intercal V^\intercal \Leftrightarrow \\
&V^\intercal V\Gamma\Sigma V^\intercal V = V^\intercal V\Sigma^\intercal\Gamma^\intercal V^\intercal V \Leftrightarrow \\
&\Gamma\Sigma = \Sigma^\intercal\Gamma^\intercal \Leftrightarrow \\
&\begin{bmatrix} X & Y \\ Z & W \end{bmatrix}\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} D^\intercal & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} X^\intercal & Z^\intercal \\ Y^\intercal & W^\intercal \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} XD & 0 \\ ZD & 0 \end{bmatrix} = \begin{bmatrix} D^\intercal X^\intercal & D^\intercal Z^\intercal \\ 0 & 0 \end{bmatrix} \Leftrightarrow \\
&\begin{bmatrix} I_r & 0 \\ ZD & 0 \end{bmatrix} = \begin{bmatrix} I_r & (ZD)^\intercal \\ 0 & 0 \end{bmatrix} \Leftrightarrow \\
&Z = 0.
\end{aligned}
$$

□

5.3 Minimizing functions of row 2-norms

We consider the optimization problem

$$\begin{aligned} &\min\ f_{\text{row}}(\|H_{1\cdot}\|_2, \|H_{2\cdot}\|_2, \ldots, \|H_{n\cdot}\|_2) \\ &\text{s.t. } AHA = A, \end{aligned} \tag{$P_1^{f_{\text{row}}}$}$$

where $f_{\text{row}} : \mathbb{R}^n_+ \to \mathbb{R}$ is increasing in each argument.

Theorem 31. *Suppose that H is an optimal solution to $P_1^{f_{\text{row}}}$, then $Y = 0, W = 0$.*

Proof. From Lemma 27, P1 is equivalent to $X = D^{-1}$. Then

$$H_{i\cdot} = \mathbf{e}_i^\intercal V \begin{bmatrix} D^{-1} & Y \\ Z & W \end{bmatrix} U^\intercal ,$$

for $i = 1, \ldots, n$. Note that

$$\begin{aligned} \|H_{i\cdot}\|_2 &= \left\| \mathbf{e}_i^\intercal V \begin{bmatrix} D^{-1} & Y \\ Z & W \end{bmatrix} U^\intercal \right\|_2 \\ &= \left\| \mathbf{e}_i^\intercal V \begin{bmatrix} D^{-1} & Y \\ Z & W \end{bmatrix} \right\|_2 \\ &= \left\| \begin{bmatrix} \mathbf{e}_i^\intercal V \begin{bmatrix} D^{-1} \\ Z \end{bmatrix} & \mathbf{e}_i^\intercal V \begin{bmatrix} Y \\ W \end{bmatrix} \end{bmatrix} \right\|_2 \\ &= \sqrt{\left\| \mathbf{e}_i^\intercal V \begin{bmatrix} D^{-1} \\ Z \end{bmatrix} \right\|_2^2 + \left\| \mathbf{e}_i^\intercal V \begin{bmatrix} Y \\ W \end{bmatrix} \right\|_2^2} . \end{aligned} \tag{5.1}$$

Note that Y and W are variables in $P_1^{f_{\text{row}}}$, and its objective is to minimize f_{row}, which is increasing in each argument $\|H_{i\cdot}\|_2$, for $i = 1, \ldots, n$.

Then, as $\left\| \mathbf{e}_i^\intercal V \begin{bmatrix} Y \\ W \end{bmatrix} \right\|_2^2 \geq 0$, we must have $\mathbf{e}_i^\intercal V \begin{bmatrix} Y \\ W \end{bmatrix} = 0$, for all $i = 1 \ldots, n$, in an optimal solution of $P_1^{f_{\text{row}}}$, i.e., we must have $Y = 0$ and $W = 0$. □

Remark 32. *The proof of Theorem 31 is not valid for other inner norms in the objective of $P_1^{f_{\text{row}}}$, because we can only remove the matrix U from the norm in* (5.1) *if we have the 2-norm.*

Remark 33. *If f_{row} is non-decreasing instead of increasing in each argument, we cannot say that $Y = 0$ and $W = 0$ at every optimal solution of $P_1^{f_{\text{row}}}$. However, we still have that $Y = 0$ and $W = 0$ at some optimal solution.*

Corollary 34. *Suppose that H is an optimal solution to $P_1^{f_{\text{row}}}$. Then, P2 and P3 are satisfied.*

Proof. As H satisfies P1, we have from Lemma 27 that $X = D^{-1}$. From Theorem 31 we show that $Y = 0$, $W = 0$, so from Lemmas 28 and 29 we have that P2 and P3 are satisfied. □

5.4 Minimizing functions of column 2-norms

We consider the optimization problem

$$\begin{aligned} &\min\ f_{\text{col}}(\|H_{\cdot 1}\|_2, \|H_{\cdot 2}\|_2, \ldots, \|H_{\cdot m}\|_2) && (P_1^{f_{\text{col}}}) \\ &\text{s.t.}\quad AHA = A, \end{aligned}$$

where $f_{\text{col}} : \mathbb{R}^m_+ \to \mathbb{R}$ is increasing in each argument.

Theorem 35. *Suppose that H is an optimal solution to $P_1^{f_{\text{col}}}$, then $Z = 0, W = 0$.*

Proof. From Lemma 27, P1 is equivalent to $X = D^{-1}$. Then,

$$H_{\cdot i} = V \begin{bmatrix} D^{-1} & Y \\ Z & W \end{bmatrix} U^\intercal \mathbf{e}_i \ ,$$

for $i = 1, \ldots, m$. Note that

$$\begin{aligned} \|H_{\cdot i}\|_2 &= \left\| V \begin{bmatrix} D^{-1} & Y \\ Z & W \end{bmatrix} U^\intercal \mathbf{e}_i \right\|_2 \\ &= \left\| \begin{bmatrix} D^{-1} & Y \\ Z & W \end{bmatrix} U^\intercal \mathbf{e}_i \right\|_2 \\ &= \left\| \begin{bmatrix} \begin{bmatrix} D^{-1} & Y \end{bmatrix} U^\intercal \mathbf{e}_i \\ \begin{bmatrix} Z & W \end{bmatrix} U^\intercal \mathbf{e}_i \end{bmatrix} \right\|_2 \\ &= \sqrt{\left\| \begin{bmatrix} D^{-1} & Y \end{bmatrix} U^\intercal \mathbf{e}_i \right\|_2^2 + \left\| \begin{bmatrix} Z & W \end{bmatrix} U^\intercal \mathbf{e}_i \right\|_2^2}. \end{aligned}$$

Note that Z and W are variables in $P_1^{f_{\text{col}}}$ and its objective is to minimize f_{col}, which is increasing in each argument $\|H_{\cdot i}\|_2$, for $i = 1, \ldots, m$. Then, as $\left\| \begin{bmatrix} Z & W \end{bmatrix} U^\intercal \mathbf{e}_i \right\|_2^2 \geq 0$, we must have that $\begin{bmatrix} Z & W \end{bmatrix} U^\intercal \mathbf{e}_i = 0$, for all $i = 1, \ldots, m$, in an optimal solution of $P_1^{f_{\text{col}}}$, i.e., we must have $Z = 0$ and $W = 0$. □

Corollary 36. *Suppose that H is an optimal solution to $P_1^{f_{\text{col}}}$. Then P2 and P4 are satisfied.*

Proof. As H satisfies P1, we have from Lemma 27 that $X = D^{-1}$. From Theorem 35 we show that $Z = 0$, $W = 0$, so from Lemmas 28 and 30 we note that P2 and P4 are satisfied. □

5.5 Searching for sparse ah-symmetric reflexive generalized inverses

Next, aiming at sparse and block-structured ah-symmetric reflexive generalized inverses, we consider a special case of $P_1^{f_{\text{row}}}$, where f_{row} is the 1-norm and, therefore, we minimize the 2,1-norm of H.

$$\begin{aligned} z(P_1^{2,1}) := \min \ & \|H\|_{2,1} \\ \text{s.t. } & AHA = A. \end{aligned} \tag{$P_1^{2,1}$}$$

Minimizing the 1-norm of a matrix has been widely applied in the literature as a surrogate to minimizing the 0-norm, or the sparsity. So we also consider the two following alternative formulations to obtain H.

$$\begin{aligned} z((P_1^{2,1})^1) := \min \ & \|H\|_1 \\ \text{s.t. } & AHA = A, \\ & \|H\|_{2,1} \leq z(P_1^{2,1}). \end{aligned} \tag{$(P_1^{2,1})^1$}$$

$$\begin{aligned} z(P_{123}^1) := \min \ & \|H\|_1 \\ \text{s.t. } & AHA = A, \\ & HAH = A, \\ & AH = (AH)^\intercal. \end{aligned} \tag{P_{123}^1}$$

From §5.2, we have that the solution $H \in \mathbb{R}^{n\times m}$ of the three formulations $P_1^{2,1}$, $(P_1^{2,1})^1$ and P_{123}^1, can be written as $H = V\Gamma U^\intercal$, where

$$\Gamma := \begin{bmatrix} D^{-1}_{r\times r} & 0_{r\times(m-r)} \\ Z_{(n-r)\times r} & 0_{(n-r)\times(m-r)} \end{bmatrix}.$$

Let

$$V := \begin{bmatrix} V_1 & V_2 \\ {\scriptstyle n\times r} & {\scriptstyle n\times(n-r)} \end{bmatrix}, \ U := \begin{bmatrix} U_1 & U_2 \\ {\scriptstyle m\times r} & {\scriptstyle m\times(m-r)} \end{bmatrix}, \ G := V_1 D^{-1} U_1^\intercal .$$

Therefore, formulations $P_1^{2,1}$, $(P_1^{2,1})^1$ and P_{123}^1 can be reformulated *much more tractably* as, respectively,

$$z(\mathcal{P}_1^{2,1}) := \min \sum_{i=1}^{n} \left\| \mathbf{e}_i^\top V \begin{bmatrix} D^{-1} \\ Z \end{bmatrix} \right\|_2 \tag{$\mathcal{P}_1^{2,1}$}$$

$$\begin{aligned} z((\mathcal{P}_1^{2,1})^1) := \quad & \min \quad \left\| V \begin{bmatrix} D^{-1} & 0 \\ Z & 0 \end{bmatrix} U^\top \right\|_1 \\ & \text{s.t.} \quad \sum_{i=1}^{n} \left\| \mathbf{e}_i^\top V \begin{bmatrix} D^{-1} \\ Z \end{bmatrix} \right\|_2 \leq z(\mathcal{P}_1^{2,1}). \\ = \quad & \min_{\substack{F \in \mathbb{R}^{n \times m}, \\ Z \in \mathbb{R}^{(n-r) \times r}}} \quad \textstyle\sum_{i=1}^{n} \sum_{j=1}^{m} F_{ij} \\ & \text{s.t.} \quad F - V_2 Z U_1^\top \geq G, \\ & \qquad F + V_2 Z U_1^\top \geq -G, \\ & \qquad \sum_{i=1}^{n} \left\| \mathbf{e}_i^\top V \begin{bmatrix} D^{-1} \\ Z \end{bmatrix} \right\|_2 \leq z(\mathcal{P}_1^{2,1}). \end{aligned} \tag{$(\mathcal{P}_1^{2,1})^1$}$$

$$\begin{aligned} z(\mathcal{P}_{123}^1) := \quad & \min \quad \left\| V \begin{bmatrix} D^{-1} & 0 \\ Z & 0 \end{bmatrix} U^\top \right\|_1 \\ = \quad & \min \quad \|G + V_2 Z U_1^\top\|_1 \\ = \quad & \min_{\substack{F \in \mathbb{R}^{n \times m}, \\ Z \in \mathbb{R}^{(n-r) \times r}}} \quad \textstyle\sum_{i=1}^{n} \sum_{j=1}^{m} F_{ij} \\ & \text{s.t.} \quad F - V_2 Z U_1^\top \geq G, \\ & \qquad F + V_2 Z U_1^\top \geq -G. \end{aligned} \tag{$\mathcal{P}_{123}^1$}$$

5.6 Local-search procedure

Xu, Fampa, Lee, and Ponte (2021) devised a local-search procedure, working with generalized inverses having minimum row sparsity, to find an approximate 1-norm minimizing ah-symmetric generalized inverse. Fampa, Lee, Ponte, and Xu (2021) demonstrated the computational effectiveness of their algorithm, versus the alternative of solving P_{123}^1. In light of our new formulations $\mathcal{P}_1^{2,1}$, $(\mathcal{P}_1^{2,1})^1$ and $\mathcal{P}_{123}^1$, it is worth conducting new experiments, comparing Xu, Fampa, Lee, and Ponte (2021) and our new formulations. In the remainder of this section, we review the local search of Xu, Fampa, Lee, and Ponte (2021), and in the next section, we present our computational results.

Definition 37. *For $A \in \mathbb{R}^{m \times n}$, let $r := \text{rank}(A)$. Let S be any ordered subset of r elements from $\{1, \ldots, m\}$ such that these r rows of A are linearly independent.*

For T an ordered subset of r elements from $\{1,\ldots,n\}$, if $|\det(A[S,T])|$ cannot be increased by swapping an element of T with one from its complement, we say that $A[S,T]$ is a local maximizer for the absolute determinant on the set of $r\times r$ nonsingular submatrices of $A[S,:]$.

Theorem 38 (Xu, Fampa, Lee, and Ponte (2021))**.** *For $A\in\mathbb{R}^{m\times n}$, let $r:=\operatorname{rank}(A)$. For any T, an ordered subset of r elements from $\{1,\ldots,n\}$, let $\hat{A}:=A[:,T]$ be the $m\times r$ submatrix of A formed by columns T. If $\operatorname{rank}(\hat{A})=r$, let $\hat{H}:=\hat{A}^{\dagger}=(\hat{A}^{\intercal}\hat{A})^{-1}\hat{A}^{\intercal}$. The $n\times m$ matrix H with all rows equal to zero, except rows T, which are given by $\hat{H}$, is an ah-symmetric reflexive generalized inverse of A.*

In Xu, Fampa, Lee, and Ponte (2021), a simple local-search procedure was proposed to obtain a matrix $\tilde{A}:=A[S,T]$ which is a local maximizer for the absolute determinant on the set of $r\times r$ non-singular submatrices of $A[S,:]$, for a given ordered subset S of r elements from $\{1,\ldots,m\}$ such that the rows of $A[S,:]$ are linearly independent. Then, an $n\times m$ ah-symmetric reflexive generalized inverse H is constructed over $\hat{A}:=A[:,T]$, by Theorem 38. Xu, Fampa, Lee, and Ponte (2021) described how to make this algorithm run in polynomial time, and demonstrated that the resulting H (which has optimal row-sparsity) has 1-norm within a factor of r of the 1-norm minimizing solution. In fact, Fampa, Lee, Ponte, and Xu (2021) demonstrated that the factor of r is actually very pessimistic, with much better results obtained in practice.

5.7 Numerical experiments

We constructed eleven test instances of varying sizes for our numerical experiments using the Matlab function `sprand` to randomly generate $m\times n$ dense matrices A with rank r, as described in (Fampa, Lee, Ponte, and Xu, 2021, §2.1). We used `Gurobi` to solve P^1_{123} and $\mathcal{P}^1_{123}$ as linear programs, and `Mosek` to solve $\mathcal{P}^{2,1}_1$ and $(\mathcal{P}^{2,1}_1)^1$ as second-order cone programs; see ApS (2019) and Gurobi Optimization, LLC (2023). We ran our experiments on a 16-core machine (running Windows Server 2016 Standard): two Intel Xeon CPU E5-2667 v4 processors running at 3.20GHz, with 8 cores each, and 128 GB of memory.

In Table 5.1 we compare results for problems $\mathcal{P}^{2,1}_1$, $(\mathcal{P}^{2,1}_1)^1$, $\mathcal{P}^1_{123}$, and for the local-search procedure (LS) proposed in Xu, Fampa, Lee, and Ponte (2021) (see §5.6) . We also compare $\mathcal{P}^1_{123}$ to P^1_{123} to show how effective the problem reduction is. We set a time limit of 5 hours for solving each instance. In the first two columns of Table 5.1, we identify, respectively, m,n,r for each instance, and the procedure used to compute H. For each instance/procedure, we show the number of non-zero rows of H (NZR), the zero-norm, the 1-norm, and the 2,1-norm of H ($\|H\|_0$, $\|H\|_1$, and

$\|H\|_{2,1}$), and the elapsed time to solve the problem, or run the local-search procedure, in seconds (Time). The time to calculate the singular-value decomposition, needed for $\mathcal{P}_1^{2,1}$, $(\mathcal{P}_1^{2,1})^1$, $\mathcal{P}_{123}^1$ is trivially small, and is not included. We consider 10^{-5} as the tolerance to distinguish non-zero elements. The symbol '*' in column 'Time', indicates that the problem was not solved to optimality either because the time limit was reached or because we ran out of memory.

We could only solve P_{123}^1 for $n \leq 120$, while $\mathcal{P}_{123}^1$ could be solved for all $n \leq 280$, and when both problems were solved, the latter was solved much faster, showing the impact of reformulating the problem in the reduced format.

We see that the 1-norm works well as a surrogate for sparsity, as the decrease in the 1-norm always leads to a decrease on the zero-norm, when comparing $\mathcal{P}_1^{2,1}$ or $(\mathcal{P}_1^{2,1})^1$ to $\mathcal{P}_{123}^1$ or P_{123}^1.

We see that, although we have distinct solutions for $\mathcal{P}_1^{2,1}$ and $(\mathcal{P}_1^{2,1})^1$, the difference between their 1-norms is small, showing that, for our instances, the minimization of the 2,1-norm in $\mathcal{P}_1^{2,1}$ already leads to solutions with the 1-norm close to minimum.

We see that the 2,1-norm works well as a surrogate to the number of non-zero rows, as the decrease on the 2,1-norm always leads to a decrease on 'NZR', when comparing $\mathcal{P}_{123}^1$ or P_{123}^1 to $\mathcal{P}_1^{2,1}$ or $(\mathcal{P}_1^{2,1})^1$.

If it is important to obtain H with small 2,1-norm or even 1-norm, problem $\mathcal{P}_1^{2,1}$ is a good choice, as it can be solved very efficiently, when compared to the other norm-minimization problems considered. We note that $(\mathcal{P}_1^{2,1})^1$ (an SOCP, solved by `Mosek` using an interior-point algorithm) and $\mathcal{P}_{123}^1$ (a dense LP with many constraints, solved by `Gurobi` using a simplex algorithm) are both quite slow compared to $\mathcal{P}_1^{2,1}$, and neither of them consistently dominates the other on running time.

Finally, as pointed out in Xu, Fampa, Lee, and Ponte (2021), the local-search procedure is very fast compared to the solution of the norm-minimization problems, and leads to solutions with much smaller zero-norms and number of non-zero rows. If sparsity and block structure are the main goals, then we conclude from our experiments, that the local-search procedure is the best option. However, both the 2,1-norm and the 1-norm of the solutions obtained are not close to the minimum possible value. On the other hand, we know from Xu, Fampa, Lee, and Ponte (2021), that the factor between the 1-norm of the local-search solution and the solution of P_{123}^1 is bounded by r. In Xu, Fampa, Lee, and Ponte (2021), we showed that although this bound is achieved in the worst case, for the random instances used in those experiments, the factor was much smaller than the bound. Here, we confirm this result on the bigger instances that we could solve, owing to the reduction obtained on the reformulation of P_{123}^1 as $\mathcal{P}_{123}^1$, the factors computed for all instances with $m \leq 280$, is less than 1.6, while the rank goes up to 70.

m, n, r	Prob	NZR	$\|H\|_0$	$\|H\|_1$	$\|H\|_{2,1}$	Time (sec)
$40, 20, 10$	$\mathcal{P}_1^{2,1}$	15	569	62.409	14.390	0.01
	$(\mathcal{P}_1^{2,1})^1$	15	569	62.389	14.390	0.14
	$\mathcal{P}_{123}^1$	16	542	59.985	14.960	0.27
	P_{123}^1	16	542	59.985	14.960	7.98
	LS	10	380	72.551	16.393	0.01
$80, 40, 20$	$\mathcal{P}_1^{2,1}$	34	2584	181.656	29.324	0.02
	$(\mathcal{P}_1^{2,1})^1$	34	2584	181.606	29.324	2.11
	$\mathcal{P}_{123}^1$	35	2344	174.111	30.537	4.14
	P_{123}^1	35	2344	174.111	30.537	328.51
	LS	20	1520	205.756	33.239	0.02
$120, 60, 30$	$\mathcal{P}_1^{2,1}$	49	5727	280.015	44.738	0.11
	$(\mathcal{P}_1^{2,1})^1$	49	5726	279.921	44.738	10.08
	$\mathcal{P}_{123}^1$	54	5518	256.148	48.145	11.69
	P_{123}^1	54	5518	256.148	48.145	12574.51
	LS	30	3508	368.637	56.173	0.01
$160, 80, 40$	$\mathcal{P}_1^{2,1}$	62	9732	374.365	57.762	0.22
	$(\mathcal{P}_1^{2,1})^1$	62	9729	374.224	57.762	94.18
	$\mathcal{P}_{123}^1$	73	9824	338.068	61.635	75.08
	P_{123}^1	-	-	-	-	*
	LS	40	6289	454.595	66.090	0.02
$200, 100, 50$	$\mathcal{P}_1^{2,1}$	75	14528	561.684	72.490	0.80
	$(\mathcal{P}_1^{2,1})^1$	75	14526	561.530	72.490	221.85
	$\mathcal{P}_{123}^1$	89	14918	516.202	78.013	344.54
	P_{123}^1	-	-	-	-	*
	LS	50	9693	770.098	90.294	0.28
$240, 120, 60$	$\mathcal{P}_1^{2,1}$	102	23812	752.147	90.253	0.97
	$(\mathcal{P}_1^{2,1})^1$	102	23798	751.934	90.253	782.32
	$\mathcal{P}_{123}^1$	114	22591	678.332	97.611	269.50
	P_{123}^1	-	-	-	-	*
	LS	60	14026	1069.745	115.824	0.05
$280, 140, 70$	$\mathcal{P}_1^{2,1}$	115	31108	921.040	104.272	8.57
	$(\mathcal{P}_1^{2,1})^1$	115	31100	920.799	104.272	6385.20
	$\mathcal{P}_{123}^1$	131	30278	837.155	112.825	13401.73
	P_{123}^1	-	-	-	-	*
	LS	70	18959	1246.666	129.702	0.05
$320, 160, 80$	$\mathcal{P}_1^{2,1}$	132	41075	1058.841	119.115	8.50
	$(\mathcal{P}_1^{2,1})^1$	-	-	-	-	*
	$\mathcal{P}_{123}^1$	-	-	-	-	*
	P_{123}^1	-	-	-	-	*
	LS	80	24939	1524.077	152.223	0.08
$1000, 500, 250$	$\mathcal{P}_1^{2,1}$	442	432276	5743.575	382.905	102.89
	$(\mathcal{P}_1^{2,1})^1$	-	-	-	-	*
	$\mathcal{P}_{123}^1$	-	-	-	-	*
	P_{123}^1	-	-	-	-	*
	LS	250	245093	12088.928	626.678	11.41
$2000, 1000, 500$	$\mathcal{P}_1^{2,1}$	913	1801923	14267.101	778.034	1507.02
	$(\mathcal{P}_1^{2,1})^1$	-	-	-	-	*
	$\mathcal{P}_{123}^1$	-	-	-	-	*
	P_{123}^1	-	-	-	-	*
	LS	500	985950	33206.475	1301.181	63.40
$3000, 1500, 750$	$\mathcal{P}_1^{2,1}$	-	-	-	-	*
	$(\mathcal{P}_1^{2,1})^1$	-	-	-	-	*
	$\mathcal{P}_{123}^1$	-	-	-	-	*
	P_{123}^1	-	-	-	-	*
	LS	750	2224075	63486.782	2002.052	98.78

Table 5.1: Comparison between procedures

Chapter 6

Conclusions

Generalized inverses have a wide variety of uses in matrix algebra and its applications. Sparsity of a generalized inverse is highly preferred for efficiency in its use; structured sparsity and low rank (=reflexivity) are both preferred for explainability. (Approximate) 1-norm minimization is useful for keeping entries under control and for inducing sparsity. Ah-symmetric (resp., ha-symmetric) generalized inverses have the key use in solving least-squares (resp., minimum-norm) problems. Reflexive generalized inverses have low rank (same as the input matrix), and this is usually preferred in applications.

We have given a local-search algorithm that efficiently produces reflexive ah-symmetric (ha-symmetric) generalized inverses. Our algorithm produces generalized inverses with guaranteed structured sparsity, with low rank (same as the input matrix), and with entries under control (by approximate 1-norm minimization). To our knowledge, no other method has been proposed in the literature with all these nice properties.

We have demonstrated that the local-search procedures presented in Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021) can be successfully implemented to construct sparse, block-structured reflexive generalized inverses with different properties. We find that the performance (1-norm achieved) is much better than tight worst-case guarantees. Overall, we find that the search procedures are very robust in terms of many of the algorithmic choices that need to be made. For scaling purposes, we found that it is necessary to be mindful of the numerics and of economizing when seeking local improvements, and calculating initial solutions efficiently proves to be a surprisingly difficult practical issue.

We have proposed algorithmic approaches to numerically analyze the trade-off between 1-norm minimization and low rank of generalized inverses, using the fact that a generalized inverse has minimum rank if and only if it satisfies the reflexive property. The algorithms start with a minimum 1-norm ah-symmetric generalized inverse and iteratively impose the reflexive property, gradually increasing 1-norm

and decreasing rank, until the minimum rank is obtained. The intermediate solutions from these algorithms can represent the best trade-off between 1-norm and sparsity against rank in numerical applications.

We show that a 2,1-norm minimizing generalized inverse satisfies two additional M-P pseudoinverse properties, including the one needed for computing least-squares solutions. We present mathematical optimization formulations related to finding row-sparse generalized inverses that can be solved very efficiently, and compare their solutions numerically to generalized inverses constructed by other methodologies, also aiming at sparsity and row-sparse structure.

As future work, we wish extend the work in Fampa and Lee (2018); Xu, Fampa, Lee, and Ponte (2021) and present an approximation algorithm for the 2,1-norm. We also wish to exploit the reformulations presented in Ponte, Fampa, Lee, and Xu (2024) to apply Alternating Direction Method of Multipliers and Linearized Bregman algorithms for fast computation of 1- and 2,1-norm minimizing solutions.

References

APS, M., 2019, *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.* Available at: <http://docs.mosek.com/9.0/toolbox/index.html>.

CAMPBELL, S. L., MEYER, C. D., 2009, *Generalized inverses of linear transformations*. Philadelphia, USA, SIAM.

CHANDRASEKARAN, V., SANGHAVI, S., PARRILO, P. A., WILLSKY, A. S., 2011, "Rank-sparsity incoherence for matrix decomposition", *SIAM Journal on Optimization*, v. 21, n. 2, pp. 572–596.

DOKMANIĆ, I., GRIBONVAL, R., 2017a, "Beyond Moore-Penrose Part I: generalized inverses that minimize matrix norms", http://arxiv.org/abs/1706.08349, a.

DOKMANIĆ, I., GRIBONVAL, R., 2017b, "Beyond Moore-Penrose Part II: the sparse pseudoinverse", https://hal.inria.fr/hal-01547283/file/pseudo-part2.pdf, b.

DOKMANIĆ, I., KOLUNDŽIJA, M., VETTERLI, M., 2013, "Beyond Moore-Penrose: sparse pseudoinverse". In: *ICASSP 2013,* pp. 6526–6530, IEEE.

ELBLE, J. M., SAHINIDIS, N., 2012, "A review of the LU update in the simplex algorithm", *International Journal of Mathematics in Operational Research*, v. 4, pp. 366–399.

FAMPA, M., LEE, J., 2018, "On sparse reflexive generalized inverses", *Operations Research Letters*, v. 46, n. 6, pp. 605–610.

FAMPA, M., LEE, J., PONTE, G., XU, L., 2021, "Experimental analysis of local searches for sparse reflexive generalized inverses", *Journal of Global Optimization*, v. 81, pp. 1057–1093.

FUENTES, V., FAMPA, M., LEE, J., 2020, "Diving for sparse partially-reflexive generalized inverses". In: Le Thi, H.A., et al. (Ed.), *WCGO 2019*, pp. 89–98.

FUENTES, V. K., FAMPA, M., LEE, J., 2016, "Sparse pseudoinverses via LP and SDP relaxations of Moore-Penrose". In: *CLAIO 2016*, pp. 343–350.

GOLUB, G. H., VAN LOAN, C. F., 1996, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA, Johns Hopkins University Press.

GONDZIO, J., 1992, "Stable algorithm for updating dense LU factorization after row or column exchange and row and column addition or deletion", *Optimiz.*, v. 23, pp. 7–26.

GUROBI OPTIMIZATION, LLC, 2023. "Gurobi Optimizer Reference Manual". Available at: <https://www.gurobi.com>.

MEYER, JR, C. D., 1973, "Generalized inversion of modified matrices", *SIAM Journal on Applied Mathematics*, v. 24, n. 3, pp. 315–323.

MOHAN, K., FAZEL, M., 2012, "Iterative reweighted algorithms for matrix rank minimization", *J. Mach. Learn. Res.*, v. 13, pp. 3441–3473.

NATARAJAN, B. K., 1995, "Sparse approximate solutions to linear systems", *SIAM journal on computing*, v. 24, n. 2, pp. 227–234.

PENROSE, R., 1955, "A generalized inverse for matrices", *Mathematical Proceedings of the Cambridge Philosophical Society*, v. 51, pp. 406–413.

PONTE, G., FAMPA, M., LEE, J., XU, L., 2024, "On computing sparse generalized inverses", *Operations Research Letters*, v. 52, pp. 107058.

RECHT, B., FAZEL, M., PARRILO, P. A., 2010, "Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization", *SIAM review*, v. 52, n. 3, pp. 471–501.

ROHDE, C. A., 1964, *Contributions to the theory, computation and application of generalized inverses*. Ph.D. Thesis, North Carolina State University, Raleigh, May. https://www4.stat.ncsu.edu/~boos/library/mimeo.archive/ISMS_1964_392.pdf.

XU, L., FAMPA, M., LEE, J., PONTE, G., 2021, "Approximate 1-norm minimization and minimum-rank structured sparsity for various generalized inverses via local search", *SIAM Journal on Optimization*, v. 31, n. 3, pp. 1722–1747.