

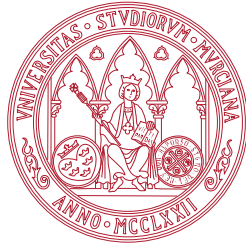
UNIVERSIDAD DE MURCIA

ESCUELA INTERNACIONAL DE DOCTORADO

Contribuciones a la teoría y a las aplicaciones de
los algoritmos de proyección

Contributions to the theory and applications of
projection algorithms

D. Rubén Campoy García
2018



UNIVERSIDAD DE
MURCIA

DOCTORAL THESIS

**CONTRIBUTIONS TO THE THEORY AND
APPLICATIONS OF PROJECTION ALGORITHMS**

Author

Rubén Campoy García

Supervisor

Dr. Francisco J. Aragón Artacho

*A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy (Mathematics) in the*

ESCUELA INTERNACIONAL DE DOCTORADO

UNIVERSIDAD DE MURCIA

Murcia, 2018

Los resultados de esta tesis se enmarcan dentro del proyecto de investigación MTM2014-59179-C2-1-P (*Fundamentos, métodos y aplicaciones de la optimización continua*) del Ministerio de Economía y Competitividad de España (MINECO), cofinanciado por el Fondo Europeo de Desarrollo Regional (FEDER) de la Unión Europea (UE). Dicho proyecto se ha desarrollado en el Departamento de Matemáticas de la Universidad de Alicante. Durante este periodo, el autor ha disfrutado de la ayuda predoctoral BES-2015-073360, dentro del programa “Ayudas para contratos predoctorales para la formación de doctores 2015” del MINECO, cofinanciada por el Fondo Social Europeo (FSE) de la UE.

This dissertation was partially supported by the research project MTM2014-59179-C2-1-P (*Fundamentals, methods and applications of continuous optimization*) by the Ministry of Economy and Competitiveness of Spain (MINECO) and the European Regional Development Fund (ERDF) of the European Union (EU). This project was carried out in the Department of Mathematics at the University of Alicante. The author was additionally supported by MINECO of Spain and the European Social Fund (ESF) of EU (grant BES-2015-073360), under the program “Ayudas para contratos predoctorales para la formación de doctores 2015”.

Agradecimientos

Acknowledgements

Este trabajo no habría sido posible sin la constante ayuda de mi director, Francisco J. Aragón Artacho. Recuerdo cuando, aún siendo estudiante de la licenciatura, asistí a una charla impartida por Fran. Entre otras cosas, nos contó su experiencia académica, desde sus comienzos en la licenciatura hasta su postdoc en Newcastle, pasando por innumerables estancias, viajes y anécdotas. Esa charla me hizo plantearme que ese era el camino que quería seguir. ¿Quién me iba a decir que, unos años más tarde, sería mi director de tesis? Muchas gracias, Fran, por todo el tiempo que me has dedicado, por trasmitirme tu pasión por la investigación, y por todo lo que he me has enseñado. Gracias por la confianza que has tenido en mí, por animarme a aprovechar cualquier oportunidad que se presentara, y por haber sabido valorar mi esfuerzo.

Quiero expresar mi agradecimiento a los profesores Miguel Ángel Goberna y Marco Antonio López por sus valiosos consejos, por toda la ayuda prestada, y por haber confiado en mí animándome a iniciar una carrera investigadora. Extiendo el agradecimiento a todos los compañeros del Departamento de Matemáticas de la Universidad de Alicante por su ayuda en temas docentes y administrativos, y por haberme tratado como a uno más de la plantilla. En particular, al resto de miembros del grupo de optimización, Margarita Rodríguez y Mariló Fajardo, por sus consejos y por todos los buenos momentos compartidos durante los viajes a congresos. Agradecer también a José Vidal y Miguel Ángel Navarro, quienes, no solo han sido compañeros de despacho, sino que han soportado mis agobios y quejas, y me han ayudado en muchas ocasiones. Los meses que estuve solo fueron mucho más aburridos.

Me gustaría agradecer a mi tutor en la Universidad de Murcia, Antonio Pallarés, su interés y su disponibilidad siempre que lo he necesitado.

I would like to express my gratitude to Russell Luke for his kind hospitality during my stay in Göttingen. Many thanks to him and all the members of his research group, for our joint lunch times and those beers after work that I really enjoyed. I want to especially thank Matt Tam, for all his suggestions to make my stay more enjoyable.

I am really thankful to Guoyin Li and Vera Roshchina for having invited me to visit them in Sydney. That was an incredible experience and a great opportunity to meet many colleagues around Australia.

Me siento muy afortunado de contar con grandes personas que me han acompañado a lo largo de esta etapa. Gracias a mis amigas, Lorena y Carolina, por formar parte de mi día a día y por todos los momentos en los que me han ayudado a evadirme del estrés producido por esta tesis. A David, por su fe ciega en mí y sus llamadas constantes para animarme y comprobar que todo va bien. A Pablo, por nuestro café rutinario para empezar la tarde con energía. A Juan Carlos y Andrea, por ser tan especiales y conseguir que cada instante compartido se convierta en un momento inolvidable. A la otra doctoranda (ya doctora) del grupo, Ana, por estar siempre dispuesta a echarme una mano. Y por supuesto, a mis compañeros de fatigas predoctorales, Alba, Elisa, María Jesús y Mañas, por nuestras victorias conjuntas contra LaTeX y esa tradicional inauguración de las vacaciones en Benidorm (siento mucho haberlo estropeado este año; prometo compensaros).

Por último y más importante, quiero dar las gracias a mi familia. A mis padres, que, aunque probablemente aún no entiendan qué he estado haciendo exactamente estos últimos tres años, siempre han apoyado mis decisiones. A mi hermana Iris, que podría considerarse coautora de esta tesis. Gracias por aguantar mis infinitas preguntas, a veces absurdas, sobre gramática inglesa (es lo que tiene contar con una filóloga en casa). También por alimentarme y librarme de preocupaciones durante las últimas semanas que estuve escribiendo esta tesis. Pero sobre todo, gracias por estar siempre a mi lado y ser un pilar fundamental de mi vida. Me gustaría extender este agradecimiento al resto de mi familia por todo su apoyo. Sin duda, lo que más he echado de menos cuando he estado fuera han sido nuestras comidas familiares.

Rubén Campoy García
Alicante, 2018

Contents

Resumen (Spanish)	xi
Abstract	xvii
1 Preliminaries	1
1.1 Convex analysis and monotone operator theory	1
1.1.1 Projection and normal cone mappings	3
1.1.2 Nonexpansive mappings	6
1.1.3 Monotone operators	10
1.2 Matrix analysis and linear algebra	15
1.2.1 The geometry of two subspaces	16
1.2.2 Convergence of power of matrices	18
1.2.3 Complementary sequences and discrete Fourier transform	19
1.3 Closed-form expressions for some selected projectors	22
2 Classical projection methods	25
2.1 Feasibility and best approximation problems	26
2.1.1 Product-space reformulation	27
2.2 Fundamental algorithms	28
2.2.1 The method of Alternating Projections	28
2.2.2 The Douglas–Rachford algorithm	36
2.3 Projection algorithms for best approximation problems	47
2.3.1 Dykstra’s algorithm	47
2.3.2 Haugazeau-like algorithms	49
2.3.3 Halpern’s algorithm	50
2.3.4 Combettes’ method	51

3	The averaged alternating modified reflections method	55
3.1	A new best approximation algorithm	56
3.1.1	The averaged alternating modified reflections operator	57
3.1.2	Iterative scheme for finding the closest point in the intersection . . .	68
3.1.3	Finitely many sets	75
3.1.4	Numerical experiments	79
3.2	Optimal rates of linear convergence for two subspaces	87
3.2.1	Convergence rate analysis	87
3.2.2	Comparison with other projection methods	100
3.2.3	Computational experiments	104
3.3	Extension to monotone operator theory	107
3.3.1	AAMR splitting algorithm for maximally monotone operators . . .	108
3.3.2	Parallel AAMR splitting for the resolvent of a finite sum	113
4	Solving combinatorial problems with the Douglas–Rachford algorithm	119
4.1	Graph coloring problems	120
4.1.1	Introduction	120
4.1.2	A feasibility model based on a binary linear program	131
4.1.3	A feasibility model based on a low-rank constrained matrix	148
4.2	Combinatorial designs of circulant type	167
4.2.1	Introduction	167
4.2.2	Modelling Framework	167
4.2.3	Computational Results	176
	Conclusions and future research	183
	Bibliography	187
	List of Figures	201
	List of Tables	207
	Notation and Symbols	209
	Index of Topics	213

Resumen

Esta tesis contribuye a la familia de los llamados *algoritmos de proyección*. Estos algoritmos son herramientas útiles y de fácil implementación para resolver *problemas de factibilidad*. Dado un espacio de Hilbert \mathcal{H} y dados dos conjuntos $A, B \subseteq \mathcal{H}$, el problema de factibilidad consiste en obtener un punto en la intersección de estos conjuntos, es decir,

$$\text{Encontrar } x \in A \cap B.$$

Cualquier problema de factibilidad definido por más de dos conjuntos se puede reducir al caso anterior gracias a una reformulación en el espacio producto propuesta por Pierra [152].

Muchos problemas reales se pueden modelizar como problemas de factibilidad, aunque obtener una formulación apropiada puede requerir cierta creatividad. En muchas situaciones prácticas, abordar esta intersección en sí misma resulta una tarea complicada, mientras que la proyección sobre cada uno de los conjuntos se puede calcular de forma eficiente. El *operador proyección* o *proyector* sobre un conjunto $C \subseteq \mathcal{H}$ es la correspondencia $P_C : \mathcal{H} \rightrightarrows C$ definida por

$$P_C(x) := \left\{ p \in C : \|x - p\| = d_C(x) := \inf_{c \in C} \|c - x\| \right\}, \quad \text{para todo } x \in \mathcal{H};$$

es decir, el proyector es una aplicación que asigna a cada punto del espacio los puntos del conjunto que están más próximos a éste. Los algoritmos de proyección utilizan estas proyecciones de forma iterativa, definiendo una sucesión que converge a un punto que permite resolver el problema. Desde el punto de vista teórico, estos algoritmos son iteraciones de punto fijo generadas por un operador definido en términos de los proyectores sobre los conjuntos. Cuando los conjuntos son cerrados y convexos, el operador proyección es univariado y satisface ciertas propiedades de no expansividad (este y otros conceptos, así como todos los resultados preliminares que se utilizan a lo largo de la tesis vienen recogidos en el Capítulo 1). Gracias a este hecho, la convergencia de los algoritmos de proyección está garantizada cuando abordamos problemas descritos por conjuntos cerrados y convexos.

El Capítulo 2 proporciona una visión general del tema, recorriendo los diversos algoritmos de proyección existentes en la literatura. Probablemente, el algoritmo de proyección más conocido es el *método de proyecciones alternadas*. Éste debe su origen a John von Neumann [174], quien lo propuso inicialmente para abordar problemas definidos por dos subespacios lineales. Posteriormente, Halperin [113] generalizó el resultado para un número arbitrario de subespacios, y fue finalmente Bregman [62] quien extendió el algoritmo para resolver problemas de factibilidad definidos por conjuntos cerrados y convexos arbitrarios. Es el método más simple e intuitivo ya que, como su propio nombre indica, cada iteración consiste en proyectar de forma alternada sobre los conjuntos que describen el problema. Específicamente, dado cualquier punto inicial $x_0 \in \mathcal{H}$, la sucesión se genera mediante la recurrencia

$$x_{k+1} = P_B P_A(x_k), \quad \text{para } k = 0, 1, 2, \dots$$

Cuando los conjuntos que definen el problema son cerrados y convexos, la sucesión generada converge débilmente a un punto en la intersección.

Otro método de proyección diferente para resolver problemas de factibilidad es comúnmente conocido como el algoritmo de Douglas–Rachford, ya que fue originalmente propuesto por J. Douglas y H.H. Rachford [89] para resolver un sistema de ecuaciones lineales que surge en problemas de transmisión de calor. Sin embargo, fueron realmente Lions y Mercier [140] quienes extendieron el algoritmo para resolver problemas de factibilidad con conjuntos cerrados y convexos arbitrarios. La iteración del algoritmo se define recursivamente mediante

$$x_{k+1} = \frac{1}{2}x_k + \frac{1}{2}(2P_B - \text{Id})(2P_A - \text{Id})(x_k), \quad \text{para } k = 0, 1, 2, \dots,$$

donde Id representa el operador identidad. Dado un conjunto $C \subseteq \mathcal{H}$, el operador $2P_C - \text{Id}$ se conoce como el *operador reflexión* sobre C . El nuevo término en la sucesión generada por el algoritmo de Douglas–Rachford es, por tanto, el punto medio entre la iteración actual y dos reflexiones consecutivas, una sobre cada conjunto. Debido a esta interpretación geométrica, el método es también conocido como el *algoritmo de reflexiones alternadas ponderadas*. De nuevo, si los conjuntos son cerrados y convexos, la sucesión generada por el algoritmo converge débilmente a un punto que permite resolver el problema de factibilidad.

Puede ocurrir que estemos interesados en encontrar, no solo un punto cualquiera en la intersección de los conjuntos, sino el más cercano a un punto dado $z \in \mathcal{H}$. Esta variante

del problema se conoce como el *problema de mejor aproximación* y se define formalmente como

$$\text{Encontrar } w \in A \cap B, \text{ tal que } \|w - z\| = \inf \{\|x - z\| : x \in A \cap B\}.$$

En el caso particular de que los conjuntos que definen el problema sean subespacios afines cerrados, los dos métodos descritos anteriormente no solo encuentran un punto cualquiera en la intersección, sino que dicho punto es, de hecho, el más cercano al punto inicial. De esta forma, permiten resolver problemas de mejor aproximación en este contexto. Sin embargo, esto no ocurre cuando consideramos conjuntos cerrados y convexos arbitrarios. Existen otros métodos de proyección específicos para resolver este problema con conjuntos más generales. Uno de ellos es el *algoritmo de Dykstra* [90, 61], que surge como una modificación apropiada del método de proyecciones alternadas, cuya sucesión generada converge fuertemente a la proyección del punto inicial sobre la intersección. Otras posibilidades son los *algoritmos de tipo Haugazeau* [115], el *algoritmo de Halpern* [114], o el *algoritmo de Combettes* [77]. En la Sección 2.3 resumimos todos estos *algoritmos de mejor aproximación*.

En el Capítulo 3 proponemos una modificación del método de Douglas–Rachford, obteniendo un nuevo algoritmo de proyección que permite resolver problemas de mejor aproximación, en lugar de simplemente problemas de factibilidad. Nuestro enfoque consiste en modificar las reflexiones sobre los conjuntos. Concretamente, cada operador reflexión en el método de Douglas–Rachford se reemplaza por lo que denominamos *operador reflexión modificado*, el cual se define para un conjunto $C \subseteq \mathcal{H}$ como

$$2\beta P_C - \text{Id}, \quad \text{con } \beta \in]0, 1[.$$

Por eso, llamamos al nuevo algoritmo el *método de reflexiones modificadas alternadas ponderadas* (abreviado *AAMR*, del inglés *averaged alternating modified reflections method*). Dado un punto $z \in \mathcal{H}$, fijados dos parámetros $\alpha \in]0, 1]$ y $\beta \in]0, 1[$, y dado cualquier punto inicial $x_0 \in \mathcal{H}$, el nuevo método AAMR se define iterativamente mediante la recurrencia

$$x_{k+1} := (1 - \alpha)x_k + \alpha(2\beta P_{B-z} - \text{Id})(2\beta P_{A-z} - \text{Id})(x_k), \quad \text{para } k = 0, 1, 2, \dots$$

Si A y B son conjuntos cerrados y convexos tal que $A \cap B \neq \emptyset$, bajo la *cualificación de restricciones*

$$z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z)),$$

donde N_A y N_B denotan los conos normales a los conjuntos A y B , respectivamente, probamos que la sucesión generada por el algoritmo cuando $\alpha < 1$, $(x_k)_{k=0}^{\infty}$, converge débilmente a un punto x^* tal que

$$P_A(z + x^*) = P_{A \cap B}(z).$$

Además, la sucesión $(P_A(z + x_k))_{k=0}^{\infty}$ converge fuertemente a $P_{A \cap B}(z)$, incluso cuando $\alpha = 1$, resolviendo así el problema de mejor aproximación. Una condición suficiente para garantizar la cualificación de restricciones en el punto de interés es la propiedad conocida como *strong CHIP* (del inglés *strong conical hull intersection property*), que establece que

$$N_A + N_B = N_{A \cap B}.$$

De hecho, esta propiedad caracteriza la convergencia del método para todo punto en el espacio.

Comparamos el nuevo AAMR con otros métodos de proyección en problemas de mejor aproximación definidos por subespacios lineales de dimensión finita. Esta comparación se lleva a cabo primero de forma numérica, mediante varios experimentos computacionales en la Sección 3.1.4, y luego analíticamente en la Sección 3.2, estudiando la tasa de convergencia lineal del método en este contexto. Este análisis teórico de la tasa se realiza mediante el estudio de la serie de potencias de matrices definida por el algoritmo, gracias a la representación matricial de los proyectores sobre subespacios lineales de dimensión finita. La tasa de convergencia lineal del método depende del *ángulo de Friedrichs* que forman los subespacios, así como de los parámetros que definen el esquema. Cuando estos parámetros son elegidos de forma óptima, la tasa obtenida resulta ser la mejor entre las tasas conocidas de otros algoritmos de proyección clásicos. Incluimos, además, nuevos experimentos numéricos que ilustran los resultados teóricos obtenidos.

El nuevo algoritmo AAMR se presenta como una modificación del método de Douglas–Rachford. Este último puede aplicarse de forma más general para encontrar un cero en la suma de dos operadores monótonos maximales. Dicha extensión se obtiene reemplazando proyectores sobre conjuntos cerrados y convexos por resolventes de operadores monótonos maximales. De la misma forma, también extendemos el algoritmo AAMR a este contexto más general en la Sección 3.3. Esto da lugar a un nuevo método de descomposición, que permite calcular el resolvente de la suma de dos operadores monótonos maximales, utilizando evaluaciones individuales de los resolventes de cada operador. Gracias al enfoque con el que se desarrolla esta extensión, junto con la reformulación en el espacio producto,

podemos derivar dos versiones diferentes del algoritmo paralelizado para calcular el resolvente de una suma finita de operadores monótonos maximales.

En los últimos años el algoritmo de Douglas–Rachford ha despertado un gran interés, debido en parte a su buen comportamiento en escenarios no convexos. A pesar de la escasez de resultados teóricos que lo justifiquen, el algoritmo ha sido empleado con éxito en una amplia lista de problemas combinatorios. Algunas de estas aplicaciones incluyen problemas de completación de matrices [8], reconstrucción de la estructura de proteínas [58], recuperación de fase (*phase retrieval*) [39, 93], ecuaciones diferenciales [133], Sudokus [10] o problemas de satisfacibilidad [94], entre otros. Sin embargo, la teoría es mucho más limitada: existen muy pocos resultados que expliquen por qué el algoritmo funciona en problemas no convexos, y aún menos, justificando su buen comportamiento global. Por ejemplo, la convergencia global del método está garantizada para el caso de una esfera y un subespacio [54], o para el caso de un semiespacio y un conjunto cerrado no convexo verificando cierta propiedad [9]. Otros resultados existentes establecen convergencia de tipo local [49, 116, 151].

En el Capítulo 4 presentamos dos nuevos problemas combinatorios que pueden ser abordados de forma satisfactoria con el algoritmo de Douglas–Rachford: el *problema de coloreado de grafos* y la construcción de *diseños combinatorios de tipo circular*. El objetivo de este capítulo es doble. Por una parte, mostramos que el algoritmo de Douglas–Rachford puede utilizarse como un heurístico eficaz para resolver los problemas considerados, algo que puede ser de interés por sí mismo. Por otra parte, adquirimos un mejor conocimiento del comportamiento del algoritmo en ciertos escenarios no convexos. Esto puede proporcionar nuevas perspectivas y jugar un papel fundamental en el desarrollo de nuevos resultados de convergencia.

Consideramos primero, en la Sección 4.1, el problema de coloreado de grafos. Un grafo consiste en un conjunto de vértices o nodos, que están conectados por ciertos enlaces. Dado un grafo y un conjunto de colores, el problema consiste en asignar un color a cada vértice, de forma que cualquier par de vértices conectados no compartan el mismo color. El coloreado de grafos tiene aplicaciones en la planificación de horarios [135], el reparto de frecuencias de radio [112], la asignación de registros en computación [70] o en pruebas de placas de circuitos impresos [106], entre otros campos. Se sabe que se trata de un problema NP-completo [127], por lo que sería razonable pensar que no existe un algoritmo exacto que resuelva el problema en tiempo polinómico. Por este motivo, se han desarrollado una gran variedad de heurísticos y algoritmos aproximados para abordar el problema. En esta tesis, mostramos que el algoritmo de Douglas–Rachford se puede

utilizar satisfactoriamente de forma heurística para colorear grafos, cuando reformulamos el problema como un problema de factibilidad. De hecho, presentamos dos modelos de naturaleza distinta que reformulan el problema. El primero de ellos, introducido en la Sección 4.1.2, se basa en la representación del problema mediante programación entera, haciendo uso de variables binarias. Posteriormente, en la Sección 4.1.3, presentamos un enfoque alternativo basado en programación semidefinida, cuyo problema de factibilidad derivado consiste en reconstruir una matriz simétrica, semidefinida positiva y con rango acotado. Mediante varios experimentos numéricos, mostramos el buen funcionamiento del algoritmo y analizamos las diferencias cuando se aplica a cada una de estas formulaciones.

Por último, en la Sección 4.2, estudiamos la construcción de diseños combinatorios de tipo circular. Diseños de este tipo pueden describirse mediante matrices circulares, con entradas en un conjunto finito, cuyas filas o columnas verifican ciertas propiedades en términos de correlación. Entre otros campos, los diseños combinatorios tienen aplicaciones a la teoría de códigos [19, 158], a la computación cuántica [101, 172], y a la comunicación sin cables, criptografía y radares [109]. Para construir de forma explícita diseños combinatorios de gran orden es necesario explotar la estructura subyacente de los mismos. Algunas posibilidades para ello son el estudio de estructuras de grupo, o una representación eficiente del problema donde los algoritmos de búsqueda, como los metaheurísticos, puedan ser implementados. En esta tesis, proponemos una formulación genérica mediante un problema de factibilidad que permite modelar diferentes clases de estos diseños. La característica fundamental del problema, que permite una implementación eficaz del algoritmo de Douglas–Rachford, es que la función autocorrelación da lugar a un operador proyección que puede calcularse de forma explícita y eficiente. La aplicabilidad de esta formulación se ilustra con la construcción de *matrices circulares de ponderación*, *matrices D -óptimas* y *matrices de Hadamard con dos núcleos circulares*. Asimismo, construimos de forma explícita dos matrices circulares de ponderación cuya existencia estaba previamente indicada como un problema abierto.

Abstract

This thesis focuses on the family of the so-called *projection algorithms*. These algorithms are employed for solving *feasibility problems*, whose goal is to find a point in the intersection of a collection of sets in a Hilbert space. Often, the intersection set is hard to deal with, while the projection mappings onto the individual sets are readily computable. Projection algorithms iterate by making use of these projections, to generate a convergent sequence whose limit solves the problem.

Two of the most well-known algorithms within this class are *the method of alternating projections* and the *Douglas–Rachford algorithm*. The weak convergence of these algorithms to a point in the intersection is guaranteed, provided that the involved sets are convex. In the special case where these sets are closed affine subspaces, the aforementioned methods not only find a point in the intersection, but they converge to the closest one to the initial point. However, this is not the case for arbitrary convex sets. The problem of finding the closest point in the intersection to any given point in the space is known as the *best approximation problem*. There are specific projection methods for solving this type of problems. For instance, *Dykstra’s algorithm* is an appropriate modification of the method of alternating projections that forces the strong convergence to the projection of the initial point onto the intersection.

In this dissertation, we propose a modification of the Douglas–Rachford method that allow us to solve best approximation problems, rather than just feasibility ones. Due to the geometry of the iteration, the Douglas–Rachford algorithm is also referred to as the *averaged alternating reflections (AAR) method*. Our approach consists in altering the reflection steps at each iterate. For this reason, the new iterative projection method is termed *AAMR* for *averaged alternating modified reflections method*. Under a constraint qualification at the point of interest, we show strong convergence of the method. We compare the performance of AAMR against other projection methods for finding the closest point in the intersection of pairs of finite dimensional subspaces, first numerically, and then analytically by studying the rate of linear convergence of the algorithm within this context. When the parameters defining the scheme are optimally selected, the obtained rate becomes the best among all known rates of other projection algorithms. We also

extend the algorithm so that it can deal with operators instead of sets. This gives rise to a new splitting algorithm for computing the resolvent of the sum of maximally monotone operators.

In the last years, the Douglas–Rachford algorithm has received much attention, due in part to the good performance of the method in nonconvex scenarios. Despite a lack of convergence results, the algorithm has been successfully employed in a wide list of combinatorial problems. In this thesis we extend that list of applications, by incorporating the *graph coloring problem* and the construction of *combinatorial designs of circulant type*. For the former, we present two feasibility problems of different nature which model the problem. The good performance of the algorithm when it is applied to these formulations is demonstrated through various computational experiments. For the case of combinatorial designs, we propose a general feasibility problem which models different classes of them. We illustrate the applicability of the formulation to the construction of *circulant weighing matrices*, *D-optimal matrices*, and *Hadamard matrices with two circulant cores*. Furthermore, we derive explicit constructions of two circulant weighing matrices whose existence was previously marked as an open question.

The structure of this dissertation is as follows. In Chapter 1 we provide basic definitions and preliminary results that are needed. Chapter 2 contains an overview of various classical projection algorithms existing in the literature. We develop the new AAMR algorithm in Chapter 3. Finally, in Chapter 4, we present some combinatorial problems which can be successfully tackled with the Douglas–Rachford algorithm.

Chapter 1

Preliminaries

The aim of this introductory chapter is to fix the notation and provide some basic notions and preliminary results that shall be needed throughout this thesis. Most of them are taken from the specialized literature in their mathematical areas. The chapter is structured as follows. In Section 1.1 we introduce the essentials of convex analysis and monotone operator theory. The content in that section is mainly based on the fundamental book by Bauschke and Combettes [35]. Then, in Section 1.2, a concise overview on linear algebra and matrix analysis is given. That section is written in a more direct way in order to provide just those results, collected from different resources, that shall be used in our subsequent developments. For further information, the reader is encouraged to check classical references in matrix analysis as [119, 144]. Finally, in Section 1.3 we present various examples of projectors onto different convex and nonconvex sets.

1.1 Convex analysis and monotone operator theory

Unless otherwise is stated, throughout this thesis

\mathcal{H} is a *real Hilbert space*,

equipped with the inner product $\langle \cdot, \cdot \rangle$ and the induced norm $\| \cdot \|$. We abbreviate *norm convergence* of sequences in \mathcal{H} with \rightarrow and we use \rightharpoonup for *weak convergence*. We denote by $\mathbb{B}(x, \rho)$ the open ball with radius ρ centered at x , i.e, $\mathbb{B}(x, \rho) := \{x \in X \mid \|x - y\| < \rho\}$.

For a subset $C \subseteq \mathcal{H}$, we denote by $\text{int } C$ and \overline{C} the *interior* and the *closure* of C , respectively. We recall that a set C is said to be *convex* if for any $x, y \in C$,

$$(1 - \lambda)x + \lambda y \in C, \quad \forall \lambda \in [0, 1];$$

and we say C is a *cone* if for all $x \in C$,

$$\lambda x \in C, \quad \forall \lambda \geq 0.$$

The *span* of C , the *affine hull* of C , the *convex hull* of C and the *cone generated* by C are denoted, respectively, by $\text{span } C$, $\text{aff } C$, $\text{conv } C$ and $\text{cone } C$; i.e.,

$$\begin{aligned} \text{span } C &:= \{\lambda x + \mu y \mid x, y \in C, \lambda, \mu \in \mathbb{R}\}, \\ \text{aff } C &:= \{(1 - \lambda)x + \lambda y \mid x, y \in C, \lambda \in \mathbb{R}\}, \\ \text{conv } C &:= \{(1 - \lambda)x + \lambda y \mid x, y \in C, \lambda \in [0, 1]\}, \\ \text{cone } C &:= \{\lambda x \mid x \in C, \lambda \geq 0\}. \end{aligned}$$

The *orthogonal complement* of C , denoted by C^\perp , is the set

$$C^\perp := \{x \in \mathcal{H} \mid \langle c, x \rangle = 0, \forall c \in C\}.$$

For a convex set $C \subseteq \mathcal{H}$, we denote by $\text{ri } C$, $\text{sri } C$ and $\text{core } C$, the *relative interior*, the *strong relative interior* and the *algebraic interior* of C , respectively; i.e.,

$$\begin{aligned} \text{ri } C &:= \{x \in C \mid \text{cone}(C - x) = \text{span}(C - x)\}, \\ \text{sri } C &:= \{x \in C \mid \text{cone}(C - x) = \overline{\text{span}}(C - x)\}, \\ \text{core } C &:= \{x \in C \mid \text{cone}(C - x) = \mathcal{H}\}. \end{aligned}$$

Given a non-empty set $D \subseteq \mathcal{H}$, we denote by $T : D \rightrightarrows \mathcal{H}$ a *set-valued operator* that maps any point from D to a subset of \mathcal{H} , i.e., $T(x) \subseteq \mathcal{H}$ for all $x \in D$. In the case when T always maps to singletons, i.e., $T(x) = \{u\}$, for all $x \in D$, T is said to be a *single-valued mapping* and it is denoted by $T : D \rightarrow \mathcal{H}$. In an abuse of notation, we may write $T(x) = u$ when $T(x) = \{u\}$. The *domain*, the *range*, the *graph*, the set of *fixed points* and the set of *zeros* of T , are denoted, respectively, by $\text{dom } T$, $\text{ran } T$, $\text{gra } T$, $\text{Fix } T$ and $\text{zer } T$; i.e.,

$$\begin{aligned} \text{dom } T &:= \{x \in \mathcal{H} \mid T(x) \neq \emptyset\}, \quad \text{ran } T := \{u \in \mathcal{H} \mid \exists x \in \mathcal{H} : u \in T(x)\}, \\ \text{gra } T &:= \{(x, u) \in \mathcal{H} \times \mathcal{H} \mid u \in T(x)\}, \quad \text{Fix } T := \{x \in \mathcal{H} \mid x \in T(x)\}, \\ \text{and } \text{zer } T &:= \{x \in \mathcal{H} \mid 0 \in T(x)\}. \end{aligned}$$

The *identity operator* is the mapping $\text{Id} : \mathcal{H} \rightarrow \mathcal{H}$ that maps every point to itself. The *inverse operator* of T , denoted by T^{-1} , is defined through $x \in T^{-1}(u) \Leftrightarrow u \in T(x)$.

Given an extended real-valued function $f : \mathcal{H} \rightarrow \mathbb{R} \cup \{\pm\infty\}$, the *domain* and the *epigraph* of f are denoted, respectively, by $\text{dom } f$ and $\text{epi } f$; i.e.,

$$\text{dom } f := \{x \in \mathcal{H} \mid f(x) < +\infty\} \quad \text{and} \quad \text{epi } f := \{(x, \xi) \in \mathcal{H} \times \mathbb{R} \mid f(x) \leq \xi\}.$$

A function f is *proper* if $\text{dom } f \neq \emptyset$ and $f(x) \neq -\infty$ for all $x \in \mathcal{H}$. We say that f is *convex* if its epigraph is a convex set, or equivalently, when

$$f((1 - \lambda)x + \lambda y) \leq (1 - \lambda)f(x) + \lambda f(y), \quad \forall x, y \in \text{dom } f, \forall \lambda \in [0, 1].$$

A function f is said to be *lower semicontinuous (lsc)* at $\bar{x} \in \mathcal{H}$ if

$$f(\bar{x}) \leq \liminf_{x \rightarrow \bar{x}} f(x),$$

and we just say f is lower semicontinuous if it is so at every point in \mathcal{H} . The *subdifferential* of a proper function $f : \mathcal{H} \rightarrow]\infty, +\infty]$ is the operator $\partial f : \mathcal{H} \rightrightarrows \mathcal{H}$ defined by

$$\partial f(x) := \{u \in \mathcal{H} \mid \langle y - x, u \rangle + f(x) \leq f(y), \quad \forall y \in \mathcal{H}\}, \quad \text{for all } x \in \mathcal{H}.$$

The *proximity operator* of f is the mapping $\text{prox}_f : \mathcal{H} \rightarrow \mathcal{H}$ given by

$$\text{prox}_f(x) := \underset{u \in \mathcal{H}}{\text{argmin}} \left(f(u) + \frac{1}{2} \|x - u\|^2 \right), \quad \text{for all } x \in \mathcal{H}.$$

Given a nonempty subset C of \mathcal{H} , the *distance function* to C , the *support function* of C and the *indicator function* of C , denoted by d_C , σ_C and ι_C , respectively, are the extended real-valued functions defined at each $x \in \mathcal{H}$ by

$$d_C(x) := \inf_{c \in C} \|c - x\|, \quad \sigma_C(x) := \sup_{c \in C} \langle c, x \rangle \quad \text{and} \quad \iota_C(x) := \begin{cases} 0, & \text{if } x \in C; \\ +\infty, & \text{if } x \notin C. \end{cases}$$

1.1.1 Projection and normal cone mappings

In this section we introduce the concept of *best approximation* to a set, which leads to define the *projection mapping*. We also discuss some results regarding existence, uniqueness and characterizations of the projection when dealing with closed and convex sets, as well as its relation with the concept of *normal cone* to a set. For a basic reference on best approximation see [81].

Definition 1.1 (Projection mapping). Given a nonempty subset $C \subseteq \mathcal{H}$, the projection mapping (or projector) onto C is the possibly set-valued operator, $P_C : \mathcal{H} \rightrightarrows C$, defined at each $x \in \mathcal{H}$ by

$$P_C(x) := \left\{ p \in C : \|x - p\| = d_C(x) := \inf_{c \in C} \|c - x\| \right\}.$$

Any point $p \in P_C(x)$ is said to be a *best approximation* to x from C (or a *projection* of x onto C). If a best approximation in C exists for every point in \mathcal{H} , then C is *proximal*. If every point $x \in \mathcal{H}$ has exactly one best approximation from C , then C is *Chebyshev*. In this case, the projector P_C is a single-valued mapping that maps every $x \in \mathcal{H}$ to its unique projection onto C , that is $P_C(x) = p$. The projector is also referred to as *projection operator*, *metric projection*, *nearest point mapping* or *best approximation operator*, among other names. The *reflector* with respect to C is the operator $R_C : \mathcal{H} \rightarrow \mathcal{H}$ given by

$$R_C := 2P_C - \text{Id},$$

where each element $r \in R_C(x)$ is called a *reflection* of x with respect to C . It is clear that the reflector is single-valued if and only if the projector is so (see Figure 1.1). The following proposition characterizes the projection mapping onto closed and convex sets, as well as presents some geometric properties.

Proposition 1.2. Let $C \subseteq \mathcal{H}$ be nonempty, closed and convex. Then the following hold.

(i) C is Chebyshev.

(ii) For every $x \in \mathcal{H}$,

$$p = P_C(x) \quad \Leftrightarrow \quad p \in C \text{ and } \langle c - p, x - p \rangle \leq 0 \text{ for all } c \in C.$$

(iii) For every $x \in \mathcal{H}$ and $\lambda \geq 0$,

$$P_C(P_C(x) + \lambda(x - P_C(x))) = P_C(x).$$

(iv) For every $y \in \mathcal{H}$, $P_{y+C}(x) = y + P_C(x - y)$.

(v) For every $\lambda \in \mathbb{R}$, $P_{\lambda C}(\lambda x) = \lambda P_C(x)$.

Proof. See, e.g., [35, Theorem 3.16 and Propositions 3.19 and 3.21] and [82, page 2.7]. \square

Any nonempty, closed and convex set is Chebyshev by Proposition 1.2(i). The question about whether the converse implication is true is known as the *Chebyshev problem*. The answer is affirmative in finite-dimensional spaces, while it still remains open in general. Nevertheless, it is proved that any nonempty weakly closed set is Chebyshev if and only if it is convex (see, e.g., [7, Theorem 14]).



(a) A closed and convex set C (Chebyshev). The projection and reflection of the point x are unique

(b) A closed but nonconvex set C . The projector and reflector are multi-valued at x , with $P_C(x) = \{p_1, p_2\}$ and $R_C(x) = \{r_1, r_2\}$

FIGURE 1.1: Examples of projectors and reflectors onto convex and nonconvex sets.

We define next the *normal cone* to a convex set, which arises in convex analysis as a tool to study the local geometry of the set.

Definition 1.3 (Normal cone). Let $C \subseteq \mathcal{H}$ be a nonempty convex set and let $x \in \mathcal{H}$. The normal cone mapping to C is the operator $N_C : \mathcal{H} \rightrightarrows \mathcal{H}$ given by

$$N_C(x) := \begin{cases} \{u \in \mathcal{H} \mid \langle u, c - x \rangle \leq 0, \quad \forall c \in C\}, & \text{if } x \in C; \\ \emptyset, & \text{otherwise.} \end{cases}$$

The projection onto closed and convex sets can be characterized by the normal cone.

Proposition 1.4. Let $C \subseteq \mathcal{H}$ be a nonempty closed and convex set, and let $x, p \in \mathcal{H}$. Then,

$$p = P_C(x) \quad \Leftrightarrow \quad x - p \in N_C(p).$$

Proof. See, e.g., [35, Proposition 6.47]. □

Given two convex sets $A, B \subseteq \mathcal{H}$, it always holds that $N_A + N_B \subseteq N_{A \cap B}$. The reverse inclusion, which leads to the equality, has been widely studied in the literature [72, 73, 84, 81]. We use the following name coined by Chui, Deutsch and Ward in [72].

Definition 1.5 (Strong CHIP). *Let C and D be two closed and convex subsets of \mathcal{H} . The pair of sets $\{C, D\}$ is said to have the strong conical hull intersection property (or the strong CHIP) at $x \in C \cap D$ if*

$$N_{C \cap D}(x) = N_C(x) + N_D(x).$$

We say $\{C, D\}$ has the strong CHIP if it has the strong CHIP at each $x \in C \cap D$.

For relationships between strong CHIP and the so-called *bounded linear regularity* property in Euclidean spaces, which plays an important role in the rate of convergence of projection algorithms, see [31, 132].

The next result collects some sufficient conditions for the strong CHIP to hold.

Proposition 1.6. *Let C and D be two closed and convex subsets of \mathcal{H} . Then $\{C, D\}$ has the strong CHIP if one of the following conditions holds.*

- (i) *The set $\text{epi } \sigma_C + \text{epi } \sigma_D$ is weakly closed (which holds, e.g., if $(\text{int } D) \cap C \neq \emptyset$, $0 \in \text{core}(C - D)$ or $\text{cone}(C - D)$ is a closed subspace).*
- (ii) *\mathcal{H} is finite dimensional and $(\text{ri } C) \cap (\text{ri } D) \neq \emptyset$.*

Proof. (i) See [65, Theorem 3.1 and Proposition 3.1]. (ii) See [157, Corollary 23.8.1]. \square

1.1.2 Nonexpansive mappings

Many problems in applied mathematics can be reduced to finding a fixed point of a given operator. A natural approach to addressing it consists in iteratively applying the mapping, and expect the generated sequence to reach such a fixed point in the limit. Specifically, given a mapping $T : \mathcal{H} \rightarrow \mathcal{H}$ and any $x_0 \in \mathcal{H}$, we refer to the scheme generated by

$$x_{k+1} = T(x_k), \quad \text{for } k = 0, 1, 2, \dots, \tag{1.1}$$

as the *fixed point iteration* or *Banach–Picard iteration* defined by T . In this section we present several notions of *nonexpasiveness* for single-valued operators. They represent a key feature to provide sufficient conditions for the convergence of fixed point iterations.

Definition 1.7 (Notions of nonexpansiveness). *Let D be a nonempty subset of \mathcal{H} and let $T : D \rightarrow \mathcal{H}$. The operator T is said to be*

(i) nonexpansive if

$$\|T(x) - T(y)\| \leq \|x - y\|, \quad \forall x, y \in D;$$

(ii) firmly nonexpansive if

$$\|T(x) - T(y)\|^2 + \|(\text{Id} - T)(x) - (\text{Id} - T)(y)\|^2 \leq \|x - y\|^2, \quad \forall x, y \in D,$$

or, equivalently,

$$\langle x - y, T(x) - T(y) \rangle \geq \|T(x) - T(y)\|^2, \quad \forall x, y \in D;$$

(iii) μ -cocoercive for $\mu > 0$ if μT is firmly nonexpansive, i.e.,

$$\langle x - y, T(x) - T(y) \rangle \geq \mu \|T(x) - T(y)\|^2, \quad \forall x, y \in D;$$

(iv) contractive if there exists some constant $0 \leq \kappa < 1$ such that

$$\|T(x) - T(y)\| \leq \kappa \|x - y\|, \quad \forall x, y \in D;$$

(v) quasi-nonexpansive if

$$\|T(x) - y\| \leq \|x - y\|, \quad \forall x \in D, \forall y \in \text{Fix } T;$$

(vi) strictly quasi-nonexpansive if

$$\|T(x) - y\| < \|x - y\|, \quad \forall x \in D \setminus \text{Fix } T, \forall y \in \text{Fix } T;$$

(vii) α -averaged for $\alpha \in]0, 1[$, if there exists a nonexpansive operator $R : D \rightarrow \mathcal{H}$ such that

$$T = (1 - \alpha)\text{Id} + \alpha R.$$

Firm nonexpansiveness implies nonexpansiveness, which itself implies quasi-nonexpansiveness. The converses are not true. Other additional implications are shown in the following proposition. For more, see [35, Chapter 4].

Proposition 1.8. *Let $D \subseteq \mathcal{H}$ be nonempty and let $T : D \rightarrow \mathcal{H}$. The following hold:*

(i) T is firmly nonexpansive $\Leftrightarrow 2T - \text{Id}$ is nonexpansive.

(ii) If T is α -averaged, then T is nonexpansive and strictly quasi-nonexpansive. In addition, if $\alpha \in]0, \frac{1}{2}]$ then T is firmly nonexpansive.

Proof. See, e.g., [35, Proposition 4.2, Remark 4.34, Remark 4.36 and Remark 4.37]. \square

Nonexpansive operators can be easily turned into averaged ones by considering their *relaxation*, a concept which is generally defined as follows.

Definition 1.9 (Relaxation). Given a mapping $T : D \rightarrow \mathcal{H}$ and $\lambda \in [0, 2]$, the operator $T_\lambda : D \rightarrow \mathcal{H}$ defined by

$$T_\lambda := (1 - \lambda) \text{Id} + \lambda T, \quad (1.2)$$

is called a λ -relaxation or, shortly, relaxation of the operator T . We call λ a relaxation parameter. A relaxation is said to be strict if $\lambda \in]0, 2[$. More specifically, T_λ is called

(i) an under-relaxation of T if $\lambda \in]0, 1[$;

(ii) an over-relaxation of T if $\lambda \in]1, 2[$.

If $\lambda = 2$ then T_λ is called the reflection of T .

REMARK 1.10. Note that the set of fixed points is not affected by relaxations, i.e.,

$$\text{Fix } T_\lambda = \text{Fix } T, \quad \text{for all } \lambda \in]0, 2].$$

If T is nonexpansive and $\lambda \in]0, 1[$, it directly follows from the definition that T_λ is λ -averaged. Furthermore, observe that (1.2) can be expressed as

$$T_\lambda = (1 - \lambda) \text{Id} + \lambda T = \left(1 - \frac{\lambda}{2}\right) \text{Id} + \frac{\lambda}{2}(2T - \text{Id}).$$

Hence, according to the previous equality together with Proposition 1.8(i), if $\lambda \in]0, 2[$ we get that T_λ is $\frac{\lambda}{2}$ -averaged, for every firmly-nonexpansive operator T .

The following fundamental result is responsible for the well behavior of projection algorithms in the convex setting.

Proposition 1.11. Let $C \subseteq \mathcal{H}$ be nonempty, closed and convex. Then the projector operator P_C is firmly nonexpansive. Moreover, if C is a closed subspace, then P_C is a linear mapping.

Proof. See, e.g., [35, Proposition 4.16] and [81, Theorem 5.13]. \square

As a direct consequence of Propositions 1.8 and 1.11(i), if C is a nonempty, closed and convex subset of \mathcal{H} , the reflector R_C is a nonexpansive mapping. Moreover observe that $\text{Fix } R_C = C$, and hence $\text{Fix } R_C$ is a closed and convex set. As shown in the next proposition, the latter assertion can be generalized to the set of fixed points of any nonexpansive mapping defined over a closed and convex domain.

Proposition 1.12. *Let $D \subseteq \mathcal{H}$ be nonempty, closed and convex, and let $T : D \rightarrow \mathcal{H}$ be nonexpansive. Then $\text{Fix } T$ is a closed and convex set.*

Proof. See, e.g., [35, Corollary 4.24]. □

Even when the operator T defining (1.1) is nonexpansive, the Banach–Picard iteration may fail to converge. For a simple example, consider $T = -\text{Id}$ and any $x_0 \neq 0$. Therefore, stronger conditions must be required. Indeed, the so-called *asymptotic regularity* property, which imposes that

$$T(x_k) - x_k \rightarrow 0,$$

becomes critical for guaranteeing such desirable convergence (see, e.g., [35, Theorem 5.14]). In the following result we show how this property can be achieved by taking relaxations of the mapping, and thus the convergence is obtained.

Theorem 1.13 (Krasnosel’skiĭ–Mann iteration). *Let D be a nonempty, closed and convex subset of \mathcal{H} , let $T : D \rightarrow D$ be a nonexpansive operator such that $\text{Fix } T \neq \emptyset$ and let $(\lambda_k)_{k=0}^{\infty}$ be a sequence in $[0, 1]$ such that $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$. Given $x_0 \in D$, set*

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k T(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Then the following hold.

- (i) $(T(x_k) - x_k)_{k=0}^{\infty}$ converges strongly to 0.
- (ii) $(x_k)_{k=0}^{\infty}$ converges weakly to a point in $\text{Fix } T$.

Proof. See, e.g., [35, Theorem 5.15]. □

Particularly, Theorem 1.13 asserts that the Banach–Picard iteration in (1.1) will converge weakly to a fixed point if we require T to be averaged. The next proposition shows how this result can be sharpened if T is additionally a linear operator, since in that case we obtain strong convergence to the closest fixed point.

Proposition 1.14. *Let $T : \mathcal{H} \rightarrow \mathcal{H}$ be a nonexpansive linear operator and let $x_0 \in \mathcal{H}$. Set $x_{k+1} = T(x_k)$, for $k = 0, 1, 2, \dots$. Then*

$$x_k \rightarrow P_{\text{Fix}T}(x_0) \iff x_k - x_{k+1} \rightarrow 0.$$

Proof. See, e.g., [35, Proposition 5.28]. □

Clearly, a Banach–Picard iteration (1.1) can only converge whenever the operator T has at least one fixed point. When $\text{Fix}T = \emptyset$, there exist some results provided by Pazy [149] in 1971, and by Baillon, Bruck and Reich [24] in 1978, regarding the asymptotic behavior of the iteration for an average mapping. These are summarized in the following theorem.

Theorem 1.15 (Asymptotic behavior of averaged iterations). *Given $\alpha \in]0, 1[$, let $T : \mathcal{H} \rightarrow \mathcal{H}$ be an α -averaged operator. For any $x \in \mathcal{H}$, the following hold:*

(i) $(T^k(x) - T^{k+1}(x))_{k=0}^\infty$ converges in norm to the unique element of minimum norm in $\overline{\text{ran}}(\text{Id} - T)$;

(ii) $\text{Fix}T = \emptyset \iff \|T^k(x)\| \rightarrow \infty$.

Proof. (i) See [24, Corollary 2.3], [149, Corollary 2]. (ii) See [24, Corollary 2.2]. □

1.1.3 Monotone operators

In this section we give an overview of the fundamentals of the theory of monotone operators. A problem of great interest is finding zeros of a given set-valued operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$; that is, solving the inclusion

$$0 \in A(x). \tag{1.3}$$

Many problems in optimization and variational analysis can be modeled as (1.3). For instance: convex minimization, systems of nonlinear equations, complementarity problems and variational inequalities, among others. The problem has been widely studied assuming certain *monotonicity* properties of the operator, which are defined next.

Definition 1.16 (Notions of monotonicity). *An operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$ is said to be*

(i) *monotone if*

$$\langle x - y, u - v \rangle \geq 0, \quad \forall (x, u), (y, v) \in \text{gra} A;$$

(ii) maximally monotone if it is monotone and there exists no other monotone operator $B : \mathcal{H} \rightrightarrows \mathcal{H}$ such that $\text{gra } B$ properly contains $\text{gra } A$; i.e., for every $(x, u) \in \mathcal{H} \times \mathcal{H}$,

$$(x, u) \in \text{gra } A \quad \Leftrightarrow \quad \langle x - y, u - v \rangle \geq 0, \quad \forall (y, v) \in \text{gra } A;$$

(iii) μ -strongly monotone for $\mu > 0$, if $A - \mu \text{Id}$ is monotone; i.e.,

$$\langle x - y, u - v \rangle \geq \mu \|x - y\|^2, \quad \forall (x, u), (y, v) \in \text{gra } A.$$

REMARK 1.17. Given any monotone operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$ there always exists a maximally monotone extension, i.e., a maximally monotone operator $\tilde{A} : \mathcal{H} \rightrightarrows \mathcal{H}$ with $\text{gra } A \subset \text{gra } \tilde{A}$ (see, e.g., [35, Theorem 20.21]).

Two well-known examples of maximally monotone operators are given next.

Example 1.18 (The subdifferential and the normal cone operators).

(i) Let $f : \mathcal{H} \rightarrow]-\infty, +\infty]$ be a proper lsc convex function. The subdifferential of f , ∂f , is a maximally monotone operator (see, e.g., [35, Theorem 20.48]).

(ii) Let C be a nonempty closed and convex subset of \mathcal{H} . The normal cone to C , N_C , is a maximally monotone operator (see, e.g., [35, Example 20.26]).

The following lemma shows the preservation of (maximal) monotonicity under affine transformations.

Lemma 1.19. Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be a (maximally) monotone operator, let $w, z \in \mathcal{H}$ and let $\gamma, \lambda \in \mathbb{R}$ such that $\gamma\lambda > 0$. Then the operator $\tilde{A} : \mathcal{H} \rightrightarrows \mathcal{H}$ defined for any $x \in \mathcal{H}$ by

$$\tilde{A}(x) := w + \gamma A(\lambda x + z),$$

is (maximally) monotone.

Proof. Let $(x, u), (y, v)$ be two any arbitrary pairs in $\text{gra } \tilde{A}$. Then

$$\left(\lambda x + z, \frac{u - w}{\gamma} \right), \left(\lambda y + z, \frac{v - w}{\gamma} \right) \in \text{gra } A.$$

Assuming that A is monotone we obtain that

$$\left\langle (\lambda x + z) - (\lambda y + z), \frac{u - w}{\gamma} - \frac{v - w}{\gamma} \right\rangle \geq 0,$$

and equivalently,

$$\frac{\lambda}{\gamma} \langle x - y, u - v \rangle \geq 0.$$

Since $\frac{\gamma}{\lambda} > 0$, it can be removed from the last inequality to deduce that \tilde{A} is monotone. The case of maximal monotonicity can be analogously reasoned so we decide to omit the proof. \square

A very useful characterization of maximal monotonicity is provided by the following fundamental result established by Minty [145] in 1967.

Theorem 1.20 (Minty). *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be monotone. Then*

$$A \text{ is maximally monotone} \quad \Leftrightarrow \quad \text{ran}(\text{Id} + A) = \mathcal{H}.$$

Proof. See, e.g., [35, Theorem 21.1]. \square

Next we recall the definition of the *resolvent* of an operator, which is an important tool in the theory of monotone operators.

Definition 1.21 (Resolvent). *Given an operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$, the resolvent of A is the operator defined by*

$$J_A := (\text{Id} + A)^{-1}.$$

The reflected resolvent of A is defined by $R_A := 2J_A - \text{Id}$.

Clearly, $\text{dom } J_A = \text{ran}(\text{Id} + A)$, and thus Minty's theorem (Theorem 1.20) guarantees that the resolvent has full domain precisely when A is maximally monotone. In the next result we collect some additional properties regarding the single-valuedness and nonexpansiveness of the resolvent and the reflected resolvent of maximally monotone operators.

Proposition 1.22. *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be a maximally monotone operator. Then*

- (i) $J_A : \mathcal{H} \rightarrow \mathcal{H}$ is firmly nonexpansive;
- (ii) $R_A : \mathcal{H} \rightarrow \mathcal{H}$ is nonexpansive.

Proof. See, e.g., [35, Corollary 23.11]. \square

The previous proposition establishes that the resolvents of maximally monotone operators are firmly nonexpansive. In fact, this turns out to be a characterization of the firm nonexpansiveness of any given mapping.

Proposition 1.23. *Let $T : \mathcal{H} \rightarrow \mathcal{H}$. Then T is firmly nonexpansive if and only if it is the resolvent of a maximally monotone operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$.*

Proof. See, e.g., [35, Corollary 23.9]. □

Furthermore, the strong monotonicity of an operator is related with the cocoercivity of its resolvent.

Proposition 1.24. *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be monotone and let $\mu > 0$. Then A is μ -strongly monotone if and only if J_A is $(\mu + 1)$ -cocoercive.*

Proof. See, e.g., [35, Proposition 23.13]. □

The resolvents of the maximally monotone operators considered in Example 1.18 are also some well-known mappings.

Example 1.25 (The proximity and the projection operators). *According to Proposition 1.22, the resolvents of the operators considered in Example 1.18 are single-valued and firmly nonexpansive with full domain.*

(i) ([35, Example 23.3]) *Let $f : \mathcal{H} \rightarrow] - \infty, +\infty]$ be a proper lower semicontinuous convex function. The resolvent of its subdifferential is the proximity operator of f , i.e.,*

$$J_{\partial f} = \text{prox}_f.$$

(ii) ([35, Example 23.4]) *Let $C \subseteq \mathcal{H}$ be a nonempty, closed and convex set. Then the resolvent of the normal cone to C is the projection mapping onto C , i.e.,*

$$J_{N_C} = P_C.$$

We recall next the concept of perturbation of an operator, which is denoted as in [48].

Definition 1.26 (Inner perturbation). *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ and let $w \in \mathcal{H}$. The corresponding inner w -perturbation of A is the operator $A_w : \mathcal{H} \rightrightarrows \mathcal{H}$ defined by*

$$A_w(x) := A(x - w), \quad \text{for all } x \in \mathcal{H}.$$

Lemma 1.27. *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ and let $w \in \mathcal{H}$. Then*

$$J_{A_w} = (J_A)_w + w.$$

Proof. Observe that, for any $x \in \mathcal{H}$,

$$p \in J_{A_w}(x) \Leftrightarrow x \in p + A(p - w) \Leftrightarrow x - w \in p - w + A(p - w) \Leftrightarrow p - w \in J_A(x - w),$$

which proves the result. \square

We now turn our attention to the set of zeros of monotone operators.

Proposition 1.28. *Let $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ be two monotone operators and let $\gamma > 0$. Then,*

$$(i) \text{ zer } A = \text{Fix } J_{\gamma A};$$

$$(ii) \text{ zer}(A + B) = J_{\gamma A}(\text{Fix } R_{\gamma B} R_{\gamma A}).$$

Proof. (i) See, e.g., [35, Proposition 23.38]. (ii) See, e.g., [35, Proposition 26.1(iii)(c)]. \square

Proposition 1.29. *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be strongly monotone. Then $\text{zer } A$ is at most a singleton. If in addition, A is also maximally monotone then $\text{zer } A$ is exactly a singleton.*

Proof. See, e.g., [35, Proposition 23.35 and Corollary 23.37]. \square

Some of the previous results give us some insight into the role of resolvents for solving (1.3). Specifically, Propositions 1.22(i) and 1.28(i) can be combined with Theorem 1.13 to construct a fixed point iteration which will be weakly convergent to a zero of a maximally monotone operator. Closely related to this idea, a finer algorithm was proposed in 1976 by Rockafellar [156].

Theorem 1.30 (Rockafellar's proximal-point algorithm). *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be a maximally monotone operator such that $\text{zer } A \neq \emptyset$, and let $(\gamma_k)_{k=0}^{\infty}$ be a sequence in $]0, +\infty[$ verifying that $\sum_{k=0}^{\infty} \gamma_k = +\infty$. Given $x_0 \in \mathcal{H}$, set*

$$x_{k+1} = J_{\gamma_k A}(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Then the following hold.

(i) $(x_k)_{k=0}^{\infty}$ converges weakly to a point in $\text{zer } A$.

(ii) Suppose that A is strongly monotone. Then $(x_k)_{k=0}^{\infty}$ converges strongly to the unique point in $\text{zer } A$.

Proof. See, e.g., [35, Theorem 23.41]. \square

1.2 Matrix analysis and linear algebra

Consider $\mathbb{C}^{n \times m}$ ($\mathbb{R}^{n \times m}$), the vector space of $n \times m$ complex (real) matrices. We denote by I_n , 0_n and $0_{n \times m}$, the $n \times n$ identity matrix, the $n \times n$ zero matrix, and the $n \times m$ zero matrix, respectively. For simplicity, we shall omit the subindices when the size can be deduced. The transpose of a matrix $A \in \mathbb{C}^{n \times m}$ is denoted by A^T , while the conjugate transpose is denoted by A^* . The *kernel*, the *range*, the *rank* and the set of *fixed points* of A are denoted, respectively, by $\ker A$, $\text{ran } A$, $\text{rank } A$ and $\text{Fix } A$; i.e.,

$$\begin{aligned} \ker A &:= \{x \in \mathbb{C}^n \mid Ax = 0_n\}, & \text{ran } A &:= \{y \in \mathbb{C}^n \mid \exists x \in \mathbb{C}^m : Ax = y\}, \\ \text{rank } A &:= \dim(\text{ran } A), & \text{and } \text{Fix } A &:= \ker(A - I); \end{aligned}$$

where $\dim U$ denotes the dimension of a linear subspace $U \subseteq \mathcal{H}$. Observe that some of the previous concepts are inherited from the operator theory since we can identify any matrix $A \in \mathbb{C}^{n \times m}$ with the linear mapping $A : \mathbb{C}^m \rightarrow \mathbb{C}^n$, defined for any $x \in \mathbb{C}^m$ by $A(x) := Ax$. Thus, operator properties can also be defined among matrices. For instance, we say that A is *nonexpansive* if

$$\|Ax - Ay\| \leq \|x - y\|, \quad \text{for all } x, y \in \mathbb{C}^m.$$

The *trace* of a matrix $A \in \mathbb{C}^{n \times m}$ is denoted by $\text{tr } A$. The *determinant* and the *inverse* of a square matrix $A \in \mathbb{C}^{n \times n}$ are denoted by $\det A$ and A^{-1} , respectively. Given $a \in \mathbb{C}^n$, we denote by $\text{diag}(a)$ and $\text{circ}(a)$, respectively, the *diagonal* $n \times n$ matrix whose diagonal entries are the elements of a and the square *circulant* matrix whose rows are cyclic permutations of a (offset by their row index). A square matrix $A \in \mathbb{C}^{n \times n}$ ($A \in \mathbb{R}^{n \times n}$) is said to be a *Hermitian* (*symmetric*) matrix if $A^* = A$, and it is said to be *unitary* (*orthogonal*) if $A^*A = I_n$. Further, a real symmetric matrix $A \in \mathbb{R}^{n \times n}$ is said to be *positive semidefinite* if

$$x^T Ax \geq 0, \quad \text{for all } x \in \mathbb{R}^n.$$

We denote by \mathcal{S}^n the subspace of symmetric matrices in $\mathbb{R}^{n \times n}$, while \mathcal{S}_+^n stands for the cone of positive semidefinite matrices.

Given $v_1, v_2, \dots, v_r \in \mathbb{R}^n$, the associated *Gram matrix* $G \in \mathcal{S}^r$ is constructed componentwise as $G = [\langle v_i, v_j \rangle]$. The Gram matrix G verifies (see, e.g. [119, Theorem 7.2.10]):

$$G \in \mathcal{S}_+^r \quad \text{and} \quad \text{rank } G = \dim \text{span}\{v_1, v_2, \dots, v_r\}. \quad (1.4)$$

For any matrix $A \in \mathbb{C}^{n \times n}$, the set of all its eigenvalues is called the *spectrum* of A , and is denoted by $\sigma(A)$. An eigenvalue $\lambda \in \sigma(A)$ is said to be *semisimple* if its algebraic multiplicity coincides with its geometric multiplicity (cf. [144, p. 510]), or, equivalently, if $\ker(A - \lambda I) = \ker((A - \lambda I)^2)$ (see [26, Fact 2.3]). The *spectral radius* of A is defined as

$$\rho(A) := \max\{|\lambda| : \lambda \in \sigma(A)\}.$$

An eigenvalue $\lambda \in \sigma(A)$ is said to be *subdominant* if $|\lambda| = \gamma(A)$, where

$$\gamma(A) := \max\{|\lambda| : \lambda \in \{0\} \cup \sigma(A), |\lambda| < \rho(A)\}.$$

The vector space of matrices $\mathbb{C}^{n \times m}$ (or $\mathbb{R}^{n \times m}$) can be endowed with different norms. For instance, we can always derive an *induced norm*, which is inherited from a norm in \mathbb{C}^m (\mathbb{R}^m) by the expression

$$\|A\| := \max_{\|x\| \leq 1} \|Ax\|.$$

The norm induced by the euclidean vector norm becomes the *matrix 2-norm* given by

$$\|A\|_2 := \sqrt{\lambda_{\max}}, \quad (1.5)$$

where λ_{\max} denotes the largest eigenvalue of A^*A . Trivially, induced matrix norms verify the useful inequality $\|Ax\| \leq \|A\|\|x\|$, for all $x \in \mathbb{C}^m$ (\mathbb{R}^m). However, if we need the space $\mathbb{C}^{n \times m}$ ($\mathbb{R}^{n \times m}$) to be a Hilbert space, then we have to turn to the *Frobenius norm*, defined for any matrix $A = [a_{ij}]$ by

$$\|A\|_F := \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}, \quad (1.6)$$

which is induced by the inner product $\langle A, B \rangle := \text{tr}(A^T B)$ (see, e.g., [35, Example 2.4]).

1.2.1 The geometry of two subspaces

The notion of angle between two subspaces permits to analyze the geometry described by them. For one-dimensional subspaces (lines), it is defined as the angle formed by their direction vectors. However, the generalization of the concept to higher dimensions is not straightforward. In fact, different notions of angles have been defined for general pairs of subspaces. We shall employ the one proposed by Friedrichs [104] in 1937.

Definition 1.31 (Friedrichs angle). Let U and V be two closed subspaces in \mathcal{H} . The Friedrichs angle between U and V is the angle in $[0, \frac{\pi}{2}]$ whose cosine is

$$c_F(U, V) := \sup \{ |\langle u, v \rangle| : u \in U \cap (U \cap V)^\perp, v \in V \cap (U \cap V)^\perp, \|u\| \leq 1, \|v\| \leq 1 \}.$$

The fulfillment of the strong CHIP for two closed subspaces can be identified by the Friedrichs angle between them. To proceed, we first provide a characterization of the normal cone to a closed subspace.

Proposition 1.32. Let $U, V \subseteq \mathcal{H}$ be closed subspaces. Then the following hold.

(i) For all $x \in U$, one has $N_U(x) = U^\perp$.

(ii) $(U \cap V)^\perp = \overline{U^\perp + V^\perp}$.

Proof. See, e.g., [81, Theorem 4.5 and Theorem 4.6]. □

Proposition 1.33. Let U, V be two closed subspaces in \mathcal{H} . Then $\{U, V\}$ has the strong CHIP if and only if any of the following equivalent statements hold:

(i) $U + V$ is closed;

(ii) $U^\perp + V^\perp$ is closed;

(iii) $c_F(U, V) < 1$.

In particular, the latter holds if U or V has finite dimension or finite codimension.

Proof. See, e.g., [81, Theorem 9.35 and Corollary 9.37] and Proposition 1.32. □

For the remainder of the section we shall assume that the underlying Hilbert space is finite-dimensional.

Definition 1.34 (Principal angles). Let U and V be two subspaces of a Euclidean space and consider $p := \min\{\dim U, \dim V\}$. The principal angles between U and V are the angles $0 \leq \theta_1 \leq \theta_2 \leq \dots \leq \theta_p \leq \frac{\pi}{2}$ whose cosines are recursively defined by

$$\cos \theta_i := \langle u_i, v_i \rangle = \max \left\{ \langle u, v \rangle : \begin{array}{l} u \in U, v \in V, \quad \|u\| = \|v\| = 1, \\ \langle u, u_j \rangle = \langle v, v_j \rangle = 0 \text{ for } j = 1, \dots, i-1 \end{array} \right\},$$

with $u_0 = v_0 := 0$.

The Friedrichs angle and the principal angles are related in the following sense.

Proposition 1.35. *Let $\theta_1, \theta_2, \dots, \theta_p$ be the principal angles between U and V , and let $s := \dim(U \cap V)$. Then we have*

$$\theta_i = 0, \quad \text{for all } i = 1, \dots, s.$$

Furthermore, if $s < p$ then

$$\theta_{s+1} = \theta_F > 0,$$

where θ_F denotes the Friedrichs angle between U and V .

Proof. See [26, Proposition 3.3]. □

The projection operator onto subspaces is known to be a linear mapping. The following result provides a matrix representation of the projectors onto a pair of subspaces in terms of their principal angles.

Proposition 1.36. *Let U and V be two subspaces of \mathbb{R}^n with $p := \dim U$ and $q := \dim V$. If $p < q$ and $p + q < n$, we may find an orthogonal matrix $D \in \mathbb{R}^{n \times n}$ such that*

$$P_U = D \begin{pmatrix} I_p & 0 & 0 & 0 \\ 0 & 0_p & 0 & 0 \\ 0 & 0 & 0_{q-p} & 0 \\ 0 & 0 & 0 & 0_{n-p-q} \end{pmatrix} D^* \quad \text{and} \quad P_V = D \begin{pmatrix} C^2 & CS & 0 & 0 \\ CS & S^2 & 0 & 0 \\ 0 & 0 & I_{q-p} & 0 \\ 0 & 0 & 0 & 0_{n-p-q} \end{pmatrix} D^*, \quad (1.7)$$

where C and S are two $p \times p$ diagonal matrices defined by

$$C := \text{diag}(\cos \theta_1, \dots, \cos \theta_p) \quad \text{and} \quad S := \text{diag}(\sin \theta_1, \dots, \sin \theta_p),$$

with $\theta_1, \dots, \theta_p$ being the principal angles between U and V .

Proof. See [26, Proposition 3.4]. □

1.2.2 Convergence of power of matrices

Throughout this section the vector space of square complex matrices $\mathbb{C}^{n \times n}$ is equipped with the induced matrix 2-norm (1.5).

Definition 1.37 (Convergent matrix). A matrix $A \in \mathbb{C}^{n \times n}$ is said to be convergent to $A^\infty \in \mathbb{C}^{n \times n}$ if and only if

$$\lim_{k \rightarrow \infty} \|A^k - A^\infty\| = 0.$$

We say A is linearly convergent to A^∞ with rate $\mu \in [0, 1[$ if there exist a positive integer k_0 and some $M > 0$ such that

$$\|A^k - A^\infty\| \leq M\mu^k, \quad \text{for all } k \geq k_0.$$

In this case, μ is called a linear convergence rate of A . When the infimum of all the convergence rates is also a convergence rate, we say this minimum is the optimal linear convergence rate.

The convergence of any matrix is fully determined by its spectrum as follows.

Proposition 1.38. $A \in \mathbb{C}^{n \times n}$ is convergent if and only if one of the following holds:

- (i) $\rho(A) < 1$;
- (ii) $\rho(A) = 1$ and $\lambda = 1$ is semisimple and is the only eigenvalue on the unit circle.

When this happens, A is linearly convergent with any rate $\mu \in]\gamma(A), 1[$, and if A is linearly convergent with rate $\mu \in [0, 1[$, then $\mu \in [\gamma(A), 1[$. Further, $\gamma(A)$ is the optimal linear convergence rate of A if and only if all the subdominant eigenvalues are semisimple. Moreover, if A is convergent and nonexpansive, then $\lim_{k \rightarrow \infty} A^k = P_{\text{Fix } A}$.

Proof. See [144, pp. 617–618 and 630] and [26, Theorem 2.12, Theorem 2.15 and Corollary 2.7(ii)]. □

1.2.3 Complementary sequences and discrete Fourier transform

In this section we provide the definitions of the *periodic correlation operator* and *complementary sequences*, as well as one result concerning the properties of the *discrete Fourier transform*. Before this, we start by recalling the *Jacobi–Trudi identity*.

Consider a vector of n variables denoted by $x = (x_1, x_2, \dots, x_n)$. A polynomial is *symmetric* if it is invariant under every permutation of its variables. The k -th *elementary symmetric polynomial* of x , denoted σ_k , is defined by

$$\sigma_k(x) := \sum_{1 \leq j_1 < j_2 < \dots < j_k \leq n} \left(\prod_{l=1}^k x_{j_l} \right).$$

Every symmetric polynomial can be uniquely written as a polynomial in the elementary symmetric polynomials (see [166, Theorem 1.1.1]). The k -th power polynomial of x , denoted p_k , is defined by

$$p_k(x) := \sum_{j=1}^n x_j^k. \quad (1.8)$$

The relationship between the latter two objects is provided by the *Jacobi-Trudi* identity presented in the following result.

Theorem 1.39 (Jacobi–Trudi identity). *For each $k = 1, 2, \dots, n$, it holds that*

$$\sigma_k = \frac{1}{k!} \det \begin{pmatrix} p_1 & 1 & 0 & \dots & 0 \\ p_2 & p_1 & 2 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ p_{k-1} & p_{k-2} & \dots & p_1 & k-1 \\ p_k & p_{k-1} & \dots & \dots & p_1 \end{pmatrix}.$$

Proof. See, e.g., [166, p. 7]. □

The most important case of this identity for our purposes arises when $k = 2$, in which case it yields

$$2\sigma_2 = \det \begin{pmatrix} p_1 & 1 \\ p_2 & p_1 \end{pmatrix} = p_1^2 - p_2.$$

In fact, the case $k = 2$ has an elementary proof, since

$$\sigma_2(x) = \sum_{1 \leq j_1 < j_2 \leq n} x_{j_1} x_{j_2} = \frac{1}{2} \left(\sum_{j=1}^n x_j \right)^2 - \frac{1}{2} \sum_{j=1}^n x_j^2.$$

Let $\star: \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$ denote the *periodic correlation operator* whose s -th entry is defined according to

$$(a \star b)_s := \sum_{l=0}^{n-1} a_l b_{l+s}, \quad s = 0, 1, \dots, n-1; \quad (1.9)$$

where $a = (a_0, a_1, \dots, a_{n-1}) \in \mathbb{R}^n$ and $b = (b_0, b_1, \dots, b_{n-1}) \in \mathbb{R}^n$ are n -dimensional real vectors, and the indices in (1.9) are understood modulo n .

Definition 1.40 ((Real) complementary sequences). Let $a^0, a^1, \dots, a^{m-1} \in \mathbb{R}^n$. We say that the collection of sequences $\{a^j\}_{j=0}^{m-1}$ is (real) complementary if there exist some constants ν_0 and ν_1 such that

$$\sum_{j=0}^{m-1} a^j \star a^j = (\nu_0, \nu_1, \dots, \nu_1).$$

We note that the previous definition appears in [85, Definition 2] for sequences which are potentially complex-valued.

Using the Jacobi–Trudi identity, we are able to deduce the following necessary condition for complementary sequences.

Proposition 1.41. Suppose that the collection of sequences $\{a^j\}_{j=0}^{m-1} \subset \mathbb{R}^n$ is complementary with constants ν_0 and ν_1 , i.e.,

$$\sum_{j=0}^{m-1} a^j \star a^j = (\nu_0, \nu_1, \dots, \nu_1),$$

Then $\{p_1(a^j)\}_{j=0}^{m-1} \subset \mathbb{R}$ satisfy the equation

$$\sum_{j=0}^{m-1} p_1^2(a^j) = \nu_1(n-1) + \nu_0,$$

where p_1 is given by (1.8).

Proof. Applying the Jacobi–Trudi identity (Theorem 1.39), we deduce that

$$\sum_{s=1}^{n-1} (a^j \star a^j)_s = 2\sigma_2(a^j) = \det \begin{pmatrix} p_1(a^j) & 1 \\ p_2(a^j) & p_1(a^j) \end{pmatrix} = p_1^2(a^j) - p_2(a^j),$$

for all $j \in \{0, 1, \dots, m-1\}$. Consequently,

$$\begin{aligned} \nu_1(n-1) &= \sum_{s=1}^{n-1} \sum_{j=0}^{m-1} (a^j \star a^j)_s = \sum_{j=0}^{m-1} \sum_{s=1}^{n-1} (a^j \star a^j)_s \\ &= \sum_{j=0}^{m-1} p_1^2(a^j) - \sum_{j=0}^{m-1} p_2(a^j) = \sum_{j=0}^{m-1} p_1^2(a^j) - \nu_0. \end{aligned}$$

The claimed result follows by a routine rearrangement, thus completing the proof. \square

Let \mathcal{F} denote the (unitary) discrete Fourier transform (DFT), that is, the linear mapping $\mathcal{F} : \mathbb{C}^n \rightarrow \mathbb{C}^n$ defined for any $a \in \mathbb{C}^n$ by

$$\mathcal{F}(a) := \frac{1}{\sqrt{n}} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega^{1 \cdot 1} & \cdots & \omega^{1 \cdot (n-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(n-1) \cdot 1} & \cdots & \omega^{(n-1) \cdot (n-1)} \end{pmatrix} a, \quad (1.10)$$

where $\omega := e^{2\pi i/n}$ is a primitive n -th root of unity. In the following result, both $|\cdot|$ and $(\cdot)^2$ are understood in the pointwise sense, and $(\cdot)^*$ denotes the (complex) conjugate of a complex number.

Proposition 1.42 (Properties of the DFT). *Let $a = (a_0, a_1, \dots, a_{n-1}) \in \mathbb{C}^n$.*

(i) (Conjugate symmetry) $a \in \mathbb{R}^n$ if and only if $\mathcal{F}(a)$ is conjugate symmetric, that is,

$$\mathcal{F}(a)_0 \in \mathbb{R} \quad \text{and} \quad \mathcal{F}(a)_s = (\mathcal{F}(a)_{n-s})^*, \quad \forall s = 1, 2, \dots, n-1.$$

(ii) (Correlation theorem) $\mathcal{F}(a \star a) = |\mathcal{F}(a)|^2$.

(iii) \mathcal{F} is a linear isometry on \mathbb{C}^n .

Proof. (i) See, e.g., [64, pp. 76–77]. (ii) See, e.g., [64, p. 83]. (iii) This follows from the fact that \mathcal{F} is unitary. \square

1.3 Closed-form expressions for some selected projectors

This section contains a variety of results that characterize the projector onto certain sets. We begin with two simple and well-known sets in any arbitrary Hilbert space.

Proposition 1.43 (Projector onto a hyperplane). *Given $u \in \mathcal{H} \setminus \{0\}$ and $\nu \in \mathbb{R}$, consider the hyperplane*

$$H := \{x \in \mathcal{H} : \langle x, u \rangle = \nu\}.$$

The projection of any $x \in \mathcal{H}$ onto H can be computed as

$$P_H(x) = x + \frac{\nu - \langle x, u \rangle}{\|u\|^2} u.$$

Proof. See, e.g., [35, Example 3.23]. \square

Proposition 1.44 (Projector onto a sphere). *Let S be the unit sphere in \mathcal{H} , i.e.,*

$$S := \{x \in \mathcal{H} : \|x\| = 1\}.$$

The projector onto S at any $x \in \mathcal{H}$ is given by

$$P_S(x) = \begin{cases} S, & \text{if } x = 0; \\ \left\{ \frac{x}{\|x\|} \right\}, & \text{otherwise.} \end{cases}$$

Proof. See, e.g., [81, pp. 39 Exercise 6]. □

The following two projectors become very useful in combinatorial problems in \mathbb{R}^n .

Proposition 1.45 (Projector onto the standard basis). *Let e_1, \dots, e_n denote the unit vectors of the standard basis of \mathbb{R}^n , and consider*

$$C := \{e_1, \dots, e_n\}.$$

Then, for any $x = (x_1, \dots, x_n) \in \mathbb{R}^n$,

$$P_C(x) = \{e_i : x_i = \max\{x_1, \dots, x_n\}\}.$$

Proof. See, e.g., [10, Remark 5.1]. □

Proposition 1.46 (Projector onto the set of nonzero binary vectors). *Consider the set of vectors in \mathbb{R}^n*

$$C := \left\{ z \in \{0, 1\}^n : \sum_{i=1}^n z_i \geq 1 \right\}.$$

Then, for any $x = (x_1, \dots, x_n) \in \mathbb{R}^n$, the projector onto C is described by

$$P_C(x) = \begin{cases} P_{\{0,1\}^n}(x) \setminus \{0_n\}, & \text{if } P_{\{0,1\}^n}(x) \neq \{0_n\}, \\ \{e_i : x_i = \max\{x_1, \dots, x_n\}\}, & \text{otherwise;} \end{cases}$$

where e_1, \dots, e_n denote the unit vectors of the standard basis of \mathbb{R}^n .

Proof. First note that $C = \{0, 1\}^n \setminus \{0_n\}$. If $P_{\{0,1\}^n}(x) \neq \{0_n\}$, we trivially have that $P_C(x) = P_{\{0,1\}^n}(x) \setminus \{0_n\}$. Otherwise, if $P_{\{0,1\}^n}(x) = \{0_n\}$ then it necessarily holds that $x_i < 0.5$ for all $i = 1, \dots, n$. Then, any projection in $P_C(x)$ will contain exactly one nonzero value, and the result follows from Proposition 1.45. □

In the following two last results the underlying Hilbert spaces are the vector space of real matrices $\mathbb{R}^{n \times m}$ and the one of the real square matrices $\mathbb{R}^{n \times n}$, respectively, endowed with the Frobenius norm (1.6).

Proposition 1.47 (Projector onto the null space). *Let $A \in \mathbb{R}^{l \times n}$ be a full row rank matrix and consider*

$$C := \{Z \in \mathbb{R}^{n \times m} : AZ = 0\}.$$

Then, for any $X \in \mathbb{R}^{n \times m}$, one has

$$P_C(X) = \left(I_n - A^T (AA^T)^{-1} A \right) X.$$

Proof. See, e.g., [35, Proposition 3.30(iii)] combined with [35, Example 3.29]. \square

REMARK 1.48. Observe that Proposition 1.47 with $l = m = 1$ covers the setting of Proposition 1.43 with $\mathcal{H} = \mathbb{R}^n$ as a particular case.

Proposition 1.49 (Projector onto the set of low-rank nonnegative matrices). *Let $0 \leq k \leq n$ be a non-negative integer and let C be the set of positive semidefinite matrices in $\mathbb{R}^{n \times n}$ whose rank is bounded by k , i.e.,*

$$C := \{X \in \mathcal{S}_+^n : \text{rank}(X) \leq k\}.$$

Given any $X \in \mathcal{S}^n$, consider the set

$$\Sigma(X) := \left\{ (\Lambda, Q) \in (\mathbb{R}^{n \times n})^2 : \begin{array}{l} X = Q\Lambda Q^T \text{ is a spectral decomposition of } X \text{ with} \\ \Lambda = \text{diag}(\lambda_1, \dots, \lambda_n) \text{ and } \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \end{array} \right\},$$

and for a given $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$, define

$$\Lambda_k^+ = \text{diag}(\max\{0, \lambda_1\}, \dots, \max\{0, \lambda_k\}, 0, \dots, 0).$$

Then, the projector onto C at X is given by

$$P_C(X) = \bigcup_{(\Lambda, Q) \in \Sigma(X)} \{Q\Lambda_k^+ Q^T\}.$$

Proof. See, e.g., [168, Proposition 3.11]. \square

Chapter 2

Classical projection methods

The family of the so-called *projection methods* is a wide class of algorithms which are successfully used for finding common points of a collection of sets. Projection methods iterate by finding best approximations to the sets at each step, under the philosophy that these projections onto the individual sets can be efficiently computed, rather than dealing with the intersection itself. They have received a great attention for the last few decades due to their easy implementation and their broad applicability in many areas of mathematics, engineering and physical sciences.

From a theoretical perspective, projection algorithms rely on Banach–Picard iterations defined by an operator whose fixed points solve the problem of interest. Thus, their convergence is well understood when the involved sets are convex by making use of the nonexpansiveness properties of the projection mapping. In the nonconvex setting, the analysis is more delicate, and the results available are either local or only apply to specific types of nonconvex sets. Despite a lack of a general theoretical foundation, some projection algorithms have been successfully employed in many applications involving nonconvex sets.

In order to place the contributions of this thesis in context, this chapter is intended to be a condensed survey of the several projection methods that have been developed in the literature. Further general reviews on projection algorithms can be found in [35, 66, 68, 69, 81]. The structure of the chapter is the following. We start by introducing in Section 2.1 the concepts of *feasibility problem* and *best approximation problem* in a Hilbert space. In Section 2.2 we focus on the probably two most well-known projection methods for solving feasibility problems, namely, the *method of alternating projections* and the *Douglas–Rachford method*. We finally discuss some other projection algorithms for tackling best approximation problems in Section 2.3. Those include *Dijkstra’s algorithm*, *Haugazeau’s algorithm*, *Halpern’s algorithm* and *Combettes’ algorithm*.

2.1 Feasibility and best approximation problems

We begin by stating the concept of *feasibility problem*, which basically consists in finding a common point of a collection of sets.

Definition 2.1 (Feasibility problem). *Given a finite family of sets $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$, the corresponding feasibility problem becomes*

$$\text{Find } x \in \bigcap_{i=1}^r C_i. \quad (2.1)$$

Many problems can be modeled as feasibility problems. However, devising an appropriate formulation may not be a trivial task, and some creativity is required. In many practical situations, the projector onto each of these sets can be easily computed, while directly finding a point in the intersection of the sets might be intricate. As commented before, this is in fact the spirit of projection methods, so they are employed to solve (2.1) in such cases.

A feasibility problem is said to be *consistent* when it has solution, that is, if $\bigcap_{i=1}^r C_i \neq \emptyset$. Otherwise we refer to it as an *inconsistent* feasibility problem. In the inconsistent case, we may look for an alternative or generalized solution. For instance, when dealing with two disjoint sets $C_1, C_2 \subseteq \mathcal{H}$, it might be desirable to find a *best approximation pair*, which is a pair of points $(c_1, c_2) \in C_1 \times C_2$ such that

$$\|c_1 - c_2\| = d(C_1, C_2) := \inf \{\|x - y\| : (x, y) \in C_1 \times C_2\}.$$

It may occur that we are interested in finding, not only a point in the intersection of the sets, but the closest one to any given point in the space. This problem is known as the *best approximation problem*, and is formally stated as follows.

Definition 2.2 (Best approximation problem). *Given $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$ and given $z \in \mathcal{H}$, the corresponding best approximation problem is defined as*

$$\text{Find } w \in \bigcap_{i=1}^r C_i, \text{ such that } \|w - z\| = \inf \{\|x - z\| : x \in \bigcap_{i=1}^r C_i\}. \quad (2.2)$$

In other words, the task is to find the closest point to z in the intersection set $\bigcap_{i=1}^r C_i$. A projection algorithm for solving (2.2) shall exploit the individual projectors onto the constraints to generate a projection onto the intersection.

2.1.1 Product-space reformulation

When tackling a feasibility problem or a best approximation problem as in (2.1)–(2.2) via projection methods, we need an algorithm which is able to handle with an arbitrary number of constraints. However, some projection algorithms are usually designed to deal with only two sets. Thus, in order to address a general many sets problem, one needs to find an appropriate extension of the algorithm, or alternatively, to reduce the problem to the two-set case. The latter can always be achieved thanks to the following well-known *product-space reformulation* originally stated by Pierra [152] in 1984. Consider the Hilbert product space

$$\mathcal{H} := \mathcal{H}^r = \mathcal{H} \times \cdots \times \mathcal{H}, \quad (2.3)$$

endowed with the inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle := \sum_{i=1}^r \langle x_i, y_i \rangle, \quad \text{for all } \mathbf{x} = (x_i)_{i=1}^r, \mathbf{y} = (y_i)_{i=1}^r \in \mathcal{H};$$

and define the sets

$$\mathbf{C} := C_1 \times C_2 \times \cdots \times C_r \quad \text{and} \quad \mathbf{D} := \{(x, x, \dots, x) \in \mathcal{H} : x \in \mathcal{H}\}. \quad (2.4)$$

While the set \mathbf{D} , sometimes called the *diagonal*, is always a closed subspace, the properties of \mathbf{C} are largely inherited. For instance, \mathbf{C} is nonempty, closed and convex whenever the sets C_1, \dots, C_r are so. We denote by $\mathbf{j} : \mathcal{H} \rightarrow \mathbf{D}$ the canonical embedding that maps any $x \in \mathcal{H}$ to $\mathbf{j}(x) = (x, x, \dots, x) \in \mathbf{D}$. The reformulation of (2.1) as a two-set problem in the product space relies on the equivalence

$$x \in \bigcap_{i=1}^r C_i \subseteq \mathcal{H} \quad \Leftrightarrow \quad \mathbf{j}(x) \in \mathbf{C} \cap \mathbf{D} \subseteq \mathcal{H}; \quad (2.5)$$

and the same applies to the best approximation problem (2.2), since

$$w \in P_{\bigcap_{i=1}^r C_i}(z) \quad \Leftrightarrow \quad \mathbf{j}(w) \in P_{\mathbf{C} \cap \mathbf{D}}(\mathbf{j}(z)). \quad (2.6)$$

In order to employ any projection method with the reformulated problems, it is necessary that the projectors onto the sets \mathbf{C} and \mathbf{D} are effectively computable. As we summarize in the following proposition, this is indeed the case whenever the projectors onto the underlying constraint sets in the original problem, C_1, \dots, C_r , can be efficiently computed.

Proposition 2.3 (Product-space projectors). *Let $\mathbf{x} = (x_1, \dots, x_r) \in \mathcal{H}$. The projectors onto the sets \mathbf{C} and \mathbf{D} in (2.4) at \mathbf{x} are given by*

$$P_{\mathbf{C}}(\mathbf{x}) = P_{C_1}(x_1) \times P_{C_2}(x_2) \times \cdots \times P_{C_r}(x_r) \quad \text{and} \quad P_{\mathbf{D}}(\mathbf{x}) = \mathbf{j} \left(\frac{1}{r} \sum_{i=1}^r x_i \right).$$

Proof. See [152, Lemma 1.1]. □

2.2 Fundamental algorithms

The *method of alternating projections* and the *Douglas–Rachford method* are essential schemes in the family of projection algorithms. Each of them can be seen, not only as a plain method, but as an entire subfamily of methods due to the numerous generalizations, variants and extensions that have been developed. In this section we give an overview of both of them.

2.2.1 The method of Alternating Projections

The *method of alternating projections (AP)* is undoubtedly the most intuitive and well-known projection method. It dates back to 1933, when John von Neumann [174] originally introduced the algorithm for solving the best approximation problem for two closed linear subspaces. For this reason, the method is sometimes called the *von Neumann’s alternating projections algorithm*. He proved the following fundamental result, which sparked off the development of the method.

Theorem 2.4 (von Neumann). *Let $U, V \subseteq \mathcal{H}$ be closed subspaces. Then*

$$\lim_{k \rightarrow \infty} (P_V P_U)^k(x) = P_{U \cap V}(x), \quad \text{for each } x \in \mathcal{H}.$$

Proof. See, e.g. [98, Theorem 3.3] □

The von Neumann’s theorem was generalized in 1962 by Halperin [113] for an arbitrary number of closed subspaces, and it was Bregman [62] who extended the method in 1965 for solving feasibility problems with arbitrary closed and convex sets. The alternating projections method has been widely studied and generalized by many authors; see, e.g., [29, 55, 82, 113, 129, 138, 137]. Specific references will be provided throughout the section. For a basic monograph on the alternating projections method, we recommend [98].

2.2.1.1 Alternating projections for two closed and convex sets

Given two nonempty, closed and convex subsets $A, B \subseteq \mathcal{H}$, the alternating projections algorithm iterates by successively projecting onto each of the sets. That is, given any initial point $x_0 \in \mathcal{H}$, the sequence is generated by the recurrence

$$x_{k+1} = P_B P_A(x_k). \quad (2.7)$$

Note that (2.7) is nothing more than the Banach–Picard iteration defined by the operator $T = P_B P_A$. Taking advantage of the firm nonexpansiveness of the projectors and the suitable geometry of the set of fixed points, the scheme permits to successfully address any convex feasibility problem. The following result reviews the main behavior of the algorithm.

Theorem 2.5 (Alternating projections method). *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets and let $v := P_{\overline{B-A}}(0)$. Given any $x_0 \in \mathcal{H}$, consider the alternating projections iteration generated by*

$$x_{k+1} = P_B P_A(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Then $x_k - P_A(x_k) \rightarrow v$ and exactly one of the following holds:

- (i) $v \in B - A$ and $(x_k)_{k=0}^{\infty}$ converges weakly to some point $x^* \in (A + v) \cap B$;
- (ii) $v \notin B - A$ and $\|x_k\| \rightarrow +\infty$.

Proof. See, e.g., [29, Theorem 4.8]. □

REMARK 2.6. The vector $v := P_{\overline{B-A}}(0)$ defined in Theorem 2.5 is known as the *displacement vector*. One can easily check that v measures the gap between the sets, since

$$\|v\| = d(A, B).$$

Furthermore, $v \in B - A$ if and only if the distance $d(A, B)$ is attained, which means that there exists a best approximation pair. If this happens, such a pair is given by $(P_A(x^*), x^*)$, being x^* the weak limit point in Theorem 2.5(i). In addition, if the problem is consistent, we trivially have that $v = 0 \in B - A$, and thus $x^* \in A \cap B$ is a solution to the feasibility problem. Otherwise, when $d(A, B)$ is not attained, the generated sequence will be unbounded according to Theorem 2.5(ii). These three scenarios are illustrated in Figure 2.1.

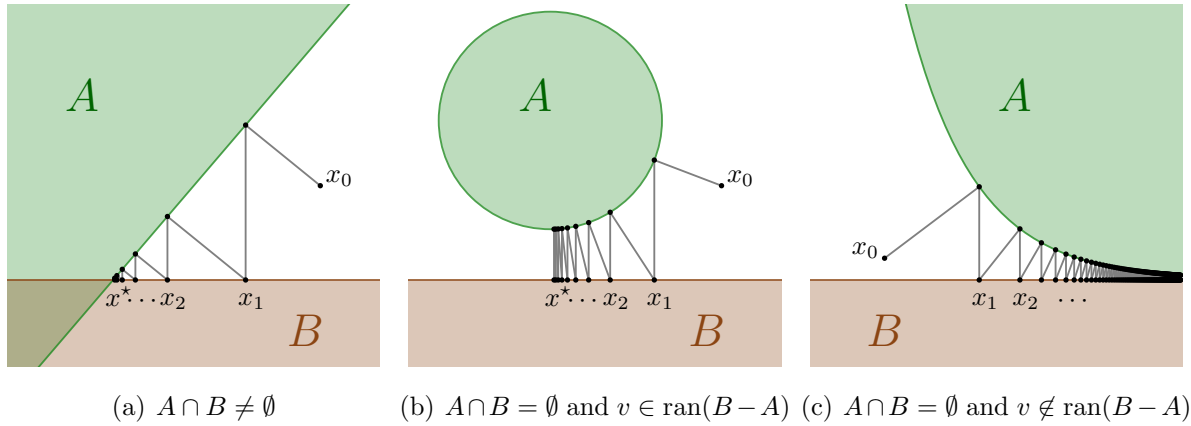


FIGURE 2.1: Behavior of the alternating projections method in three possible scenarios.

The von Neumann’s theorem (Theorem 2.4) ensures strong convergence of AP to the solution of the best approximation problem for the case of closed subspaces. It remains true for closed affine subspaces. Note, however, that weaker assertions are claimed in Theorem 2.5 for more general sets. On the one hand, the convergence of the method is only established to be weak. The question about whether strong convergence could still hold for arbitrary closed and convex sets, remained open for a long time. It was Hundal who discovered in 2004 the first known counter-example of an alternating projections iteration which does not converge in norm [120] (see also [143]). On the other hand, the method provides a point which is a solution to the feasibility problem, when consistent. Nonetheless, this point will not solve the best approximation problem in general (see Figure 2.2).

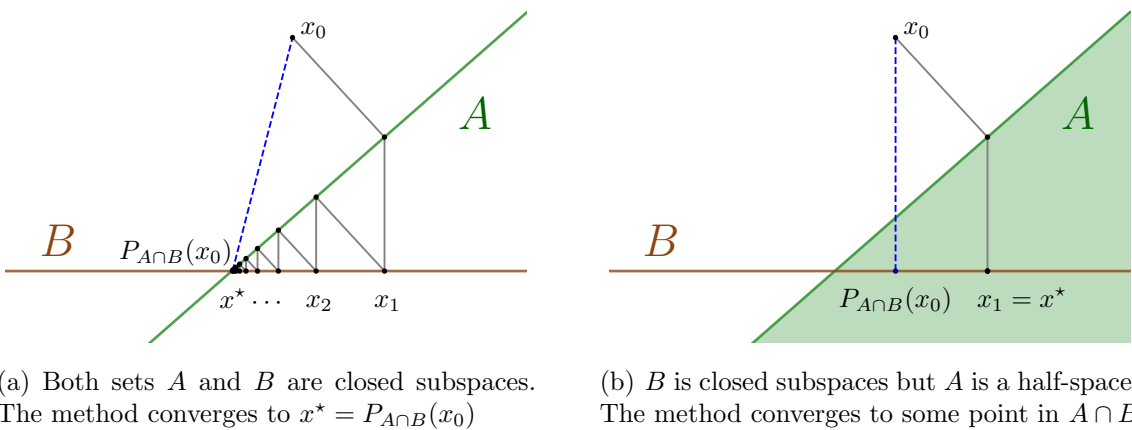


FIGURE 2.2: Failure of the method of alternating projections for solving the best approximation problem for arbitrary convex sets.

2.2.1.2 Some extensions for finitely many sets

There are different possible generalizations of the method of alternating projection for dealing with more than two sets. Suppose that $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$ are nonempty closed and convex sets.

Cyclic alternating projections The most evident approach is the iteration defined by

$$x_{k+1} = P_{C_r} P_{C_{r-1}} \cdots P_{C_1}(x_k). \quad (2.8)$$

The algorithm generated by (2.8) is known as the *cyclic projections method* and it is proved to converge weakly to a common point of the sets (see, e.g. [35, Corollary 5.26]). In fact, this cyclic scheme for finitely many sets is the one originally proposed by Bregman [62]. Halperin [113] had previously proved that, as it happens in the two-set case, when we consider closed affine subspaces the method converges strongly to the closest point in the intersection (see, e.g., [35, Corollary 5.30]). An exhaustive analysis of the method in the inconsistent case can be found in [30]. It is worth mentioning that the cyclic projections method was independently introduced by Kaczmarz [125] in 1937, for solving a consistent system of linear equations in a Euclidean space.

Averaged alternating projections A different scheme, which is commonly known as the *averaged projections method*, is given by

$$x_{k+1} = \frac{1}{r} \sum_{i=1}^r P_{C_i}(x_k). \quad (2.9)$$

Note that (2.9) is just the result of implementing the classical alternating projections in the product space, as explained in Section 2.1.1. Therefore, the convergence of the algorithm can be straightforwardly derived from Theorem 2.5 (or Theorem 2.4 for closed subspaces). Unlike in the cyclic version, the projections onto the sets in (2.9) can be simultaneously computed at each iteration, which makes the method highly parallelizable. For this reason the algorithm is also known as the *method of simultaneous projections (SP)*, see e.g. [134, 153]. The algorithm is usually accredited to Cimmino [74] due to the similarity with the so-called *Cimmino's method* that he proposed in 1938.

In Figure 2.3 we depict the iterations generated by the two previous methods.

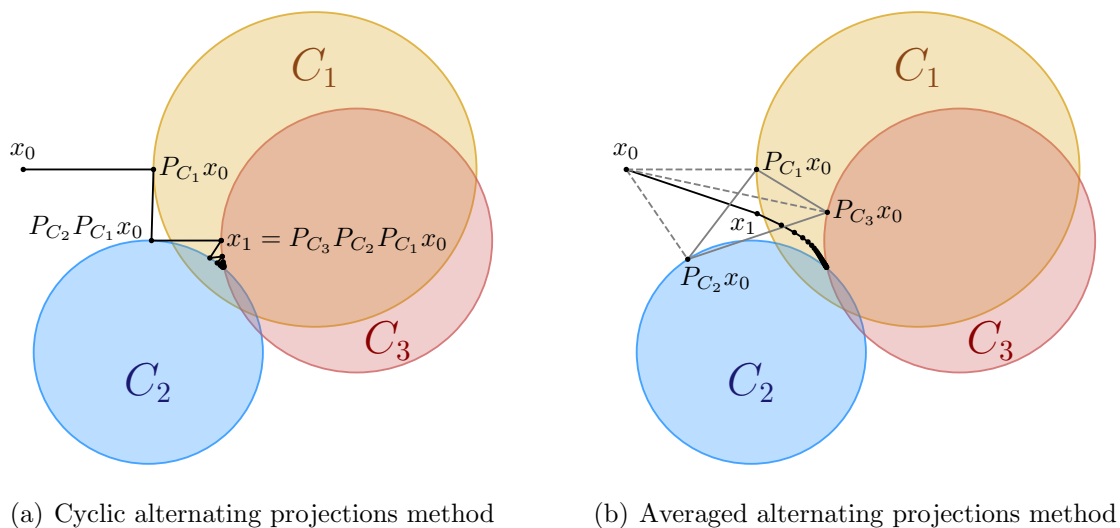


FIGURE 2.3: Illustration of two possible generalization of AP for finitely many sets.

2.2.1.3 Relaxed versions

The classical method of alternating projections often presents a very slow convergence. For this reason, some extrapolated versions have been developed. Such variants are typically constructed by relaxing some of the involved operators, so that consequently, some parameters are incorporated into the scheme. Depending on the geometry of the problem, these parameters can be optimally tuned, and thus an acceleration of the algorithm is achieved. This shall be further discussed in Section 2.2.1.4. The first relaxation of AP traces back to the method of Agmon [4] and Motzkin and Schoenberg [146], proposed in 1954, where relaxed projections were considered in a cyclic alternating projections method for a system of linear inequalities. The method was later extended for arbitrary closed and convex sets by Gubin, Polyak and Raik [110] in 1967. We present here three different relaxed versions of the method of alternating projections for two sets. We would like to emphasize that the terminology among them is not deep-rooted in the literature, so each name might refer to a different variant depending on the consulted reference.

Relaxed alternating projections The first natural approach is known as the *relaxed alternating projections method (RAP)*, and consist in a strict relaxation of the classical alternating projections. The iterative scheme is then given by

$$x_{k+1} = (1 - \alpha)x_k + \alpha P_B P_A(x_k), \quad \text{with } \alpha \in]0, 2[. \quad (2.10)$$

Partial relaxed alternating projections When only the first projection is relaxed we obtain the *partial relaxed alternating projections method (PRAP)*, whose iteration becomes

$$x_{k+1} = P_B((1 - \alpha)x_k + \alpha P_A(x_k)), \quad \text{with } \alpha \in]0, 2[. \quad (2.11)$$

Generalized alternating projections The most complete relaxed version of AP is the *generalized alternating projections method (GAP)*. It relaxes both individual projectors, as well as the whole operator. Thus the iterative scheme is defined by

$$x_{k+1} = (1 - \alpha)x_k + \alpha((1 - \alpha_2)\text{Id} + \alpha_2 P_B)((1 - \alpha_1)\text{Id} + \alpha_1 P_A)(x_k), \quad (2.12)$$

for $\alpha \in]0, 1]$ and $\alpha_1, \alpha_2 \in]0, 2[$.

The iterations of the RAP, PRAP and GAP methods are illustrated in Figure 2.4.

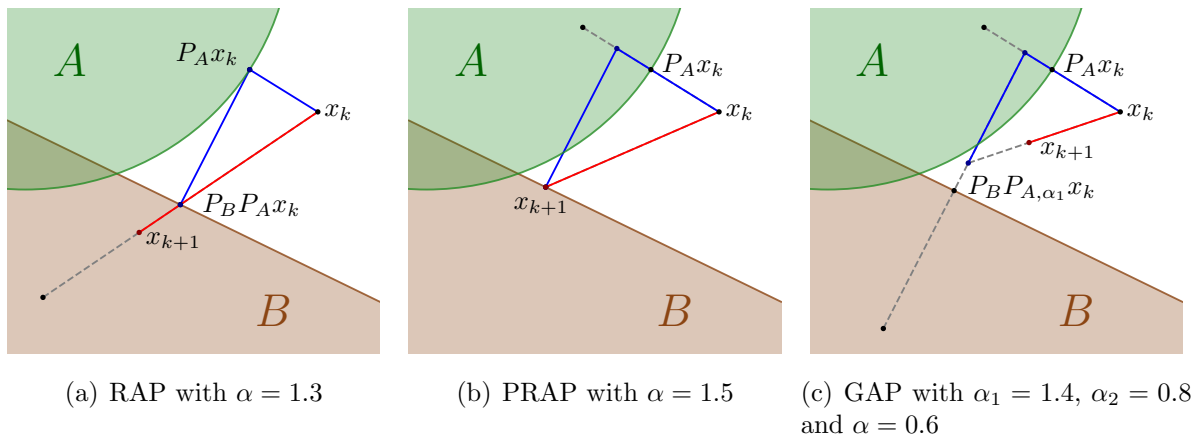


FIGURE 2.4: One iteration of RAP, PRAP and GAP methods.

As it happens with the classical alternating projections algorithm, if the involved sets in the feasibility problem are closed subspaces, RAP, PRAP and GAP also converge strongly to the closest point in the intersection.

Note that the relaxation parameters in the methods described above have been assumed to be fixed. Notwithstanding, we may let them to vary along the iterations, building up a sequence $\{\alpha_k\}_{k=0}^{\infty}$. Under some conditions on these sequences, the algorithms can be still proved to converge. Furthermore, some geometrical information of the problem can be used when updating the relaxation parameter at each step, which leads to the development of the so-called *acceleration techniques* (see, e.g., [36, 42, 67, 80, 99, 108]).

2.2.1.4 Rate of convergence for subspaces

The rate of convergence of the method of alternating projections has been widely studied in the context of closed subspaces. As shown in the following result, the rate for two subspaces turns out to depend on the Friedrichs angle between them.

Theorem 2.7 (Rate of convergence of AP for two closed subspaces). *Let U and V be two closed subspaces of \mathcal{H} and let $c_F := c_F(U, V)$ be the cosine of the Friedrichs angle between them. Then, for any $x_0 \in \mathcal{H}$,*

$$\|(P_U P_V)^k(x_0) - P_{U \cap V}(x_0)\| \leq c_F^{2k-1} \|x_0 - P_{U \cap V}(x_0)\|, \quad \text{for all } k = 0, 1, 2, \dots \quad (2.13)$$

Proof. See, e.g., [66, Theorem 5.1.7]. □

According to Theorem 2.7, the alternating projections method is linear convergent with rate the squared cosine of the Friedrichs angle between the two subspaces. The upper bound in terms of c_F^{2k-1} given in (2.13) was first estimated by Aronszajn [23] in 1950. Different bounds were later obtained, independently, by Smith, Solmon and Wagner [163] in 1977, by Deutsch [90] in 1983, and by Franchetti and Light [103] in 1986. However these were not as tight as (2.13). In fact, Kayalar and Weinert [128] proved in 1988 that

$$\|(P_U P_V)^k - P_{U \cap V}\| = c_F^{2k-1}, \quad \text{for all } k = 1, 2, \dots;$$

that is, Aronszajn's bound is the sharpest possible.

For the case of more than two subspaces, upper bounds for the rate of convergence of the cyclic projections method have been provided in [83, 128, 163]. The rate of linear convergence of the method of simultaneous projections has been recently obtained in [155]. For the case of two subspaces it becomes $\frac{1}{2} + \frac{1}{2}c_F$, so in this framework, the classical alternating projections would always be faster than its parallelized version.

Clearly, small angles between the subspaces will entail a slow convergence of the alternating projections method. As pointed out in Section 2.2.1.3, the method can be sped up if we rather consider relaxed variants. Thanks to the linearity of the projection operator onto subspaces, when these lay in a finite-dimensional (Euclidean) space, projection methods reduce to matrix iterations. Taking advantage of this fact, optimal convergence rates for RAP (2.10) and PRAP (2.11) have been obtained in [26]. By following an analogous matrix analysis, the rate of convergence with optimal parameters for GAP (2.12) has been recently given in [100]. The results for the three methods are gathered next.

Theorem 2.8 (Optimal parameters and rates for RAP, PRAP and GAP). *Let U and V be two linear subspaces in the Euclidean space \mathbb{R}^n and let θ_F and θ_p be the Friedrichs angle and the largest principal angle between U and V , respectively. Consider the relaxed, the partial relaxed, and the generalized alternating projections methods (2.10), (2.11) and (2.12), respectively, applied to U and V . Then the following hold.*

(i) *The RAP method attains its smallest optimal rate of linear convergence*

$$\gamma^*(RAP) = \frac{1 - \sin^2 \theta_F}{1 + \sin^2 \theta_F}, \quad \text{at } \alpha^* = \frac{2}{1 + \sin^2 \theta_F}.$$

(ii) *The PRAP method attains its smallest optimal rate of linear convergence*

$$\gamma^*(PRAP) = \frac{\sin^2 \theta_p - \sin^2 \theta_F}{\sin^2 \theta_F + \sin^2 \theta_p}, \quad \text{at } \alpha^* = \frac{2}{\sin^2 \theta_F + \sin^2 \theta_p}.$$

(iii) *The infimum of the linear convergence rates of the GAP method attains its smallest value*

$$\gamma^*(GAP) = \frac{1 - \sin \theta_F}{1 + \sin \theta_F}, \quad \text{at } \alpha^* = 1 \text{ and } \alpha_1^* = \alpha_2^* = \frac{2}{1 + \sin \theta_F}.$$

Proof. (i) See [26, Theorem 3.6]. (ii) See [26, Theorem 3.7]. (iii) See [100, Theorem 2]. \square

2.2.1.5 Alternating projections in the nonconvex setting

The method of alternating projections is also popular in nonconvex settings. Local linear convergence of the method is usually analyzed by assuming some regularity property of the individual sets (*prox-regularity*, *super-regularity* or *Clarke-regularity*, among others) and of their intersection (*transversality* or *subtransversality*). See [132] for a unifying discussion on different notions of regularity of sets and collection of sets.

The first results achieving local linear convergence of alternating projections for nonconvex sets appeared in [137, 138]. This analysis has been later extended by requiring weaker or different regularity properties (see, e.g., [45, 46, 51, 116, 147]). Specific applications include, for instance, inverse problems such as image recovery [136] and systems of linear equations with sparsity constraints [44, 117]. For further applications of alternating projections method for convex and nonconvex problems we refer the reader to [98] and the references therein.

2.2.2 The Douglas–Rachford algorithm

We now turn our attention to a different projection method. It is commonly known as the *Douglas–Rachford (DR) algorithm*, since it was originally proposed in 1956 by J. Douglas and H.H. Rachford [89] for solving a system of linear equations arising in heat conduction problems. However, Lions and Mercier [140] were the ones who successfully extended the algorithm for solving feasibility problems with arbitrary closed and convex sets in 1979. Actually, such an extension provided an splitting algorithm for finding a zero of the sum of two maximally monotone operators, a more general framework that covers the convergence of the scheme in the convex feasibility setting. We recommend [47, Appendix] to the reader interested in the connection between the original algorithm and the extension of Lions and Mercier. The method was more generally studied in 1992 by Eckstein and Bertsekas [91]. Further significant contributions are due to Bauschke, Combettes and Luke [38] in 2004, who deeply investigated the convex feasibility framework, to Svaiter [167] in 2011, who established the convergence of the sequence of interest, and to Bauschke and Moursi [48], who have recently provided in 2017 an exhaustive analysis of the inconsistent case.

2.2.2.1 The Douglas–Rachford algorithm for two closed and convex sets

Given two nonempty, closed and convex sets $A, B \subseteq \mathcal{H}$, the DR algorithm is the fixed point iteration generated by the *Douglas–Rachford operator* defined by

$$T_{A,B} := \frac{\text{Id} + R_B R_A}{2}. \quad (2.14)$$

The algorithm iterates by computing an average between the current point and two consecutive reflections (see Figure 2.5). For this reason, the DR algorithm is also known as the *averaged alternating reflections (AAR) method*.

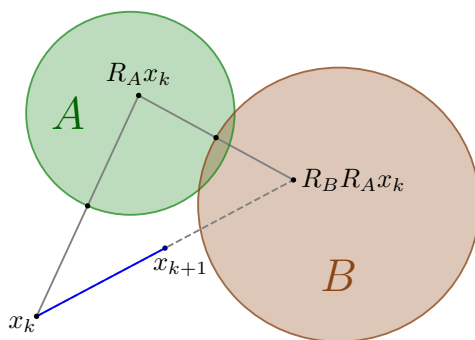


FIGURE 2.5: Geometric interpretation of the Douglas–Rachford iteration.

Since reflectors with respect to convex sets are nonexpansive, the Douglas–Rachford operator is $\frac{1}{2}$ -averaged (in fact, firmly nonexpansive according to Proposition 1.8(ii)). Therefore, the convergence of the algorithm only depends on the nonemptiness of the set of fixed points of the operator. Furthermore, when they exist, these fixed points need to be useful for solving the feasibility problem. Let us show that this is indeed the case. Observe that

$$A \cap B \subseteq \text{Fix}(R_B R_A) = \text{Fix} T_{A,B}.$$

Moreover, $x \in \text{Fix}(R_B R_A)$ if and only if

$$x = (2P_B - \text{Id})(2P_A - \text{Id})(x) = 2P_B(2P_A(x) - x) - 2P_A(x) + x;$$

that is,

$$x \in \text{Fix} T_{A,B} \quad \Leftrightarrow \quad P_B(2\beta P_A(x) - x) = P_A(x).$$

Consequently, we have $P_A(\text{Fix} T_{A,B}) = A \cap B$ and thus the algorithm will be convergent provided that $A \cap B \neq \emptyset$. The set of fixed points was completely characterized in [38]. Thanks to the results in that work, together with the ones in [48, 167], the behavior of the algorithm is fully determined in the convex feasibility setting. The next theorem collects the main results regarding the behavior of the algorithm.

Theorem 2.9 (Douglas–Rachford method). *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets and let $v := P_{A-B}(0)$. Given any $x_0 \in \mathcal{H}$, consider the Douglas–Rachford iteration generated by*

$$x_{k+1} = \frac{x_k + R_B R_A(x_k)}{2}, \quad \text{for } k = 0, 1, 2, \dots \quad (2.15)$$

Then the following hold.

(i) *If $A \cap B \neq \emptyset$, then $(x_k)_{n=0}^\infty$ is weakly convergent to a point $x^* \in \text{Fix}(R_B R_A)$ and*

$$(P_A(x_k))_{k=0}^\infty \rightarrow P_A(x^*) \in A \cap B.$$

(ii) *If $A \cap B = \emptyset$, then $\|x_k\| \rightarrow +\infty$. Further, if $v \in A - B$ then*

$$(P_A(x_k))_{k=0}^\infty \rightarrow a^* \in A \cap (v + B).$$

Proof. (i) See [38, Theorem 3.13 and Corollary 3.9] and [167, Theorem 1]. (ii) See [24, Corollary 2.2] and [48, Theorem 4.5]. \square

REMARK 2.10. The fixed point given in Theorem 2.9(i) may already be laying in the intersection, i.e., $x^* \in A \cap B$, which would solve the problem (see Figure 2.6(a)). However, this will not be the case in general, and we shall need to compute $P_A(x^*)$ to generate a solution (see Figure 2.6(b)). Hence, the sequence of interest is not $(x_k)_{k=0}^\infty$, but $(P_A(x_k))_{k=0}^\infty$, which is known as the *shadow sequence*.

Although the DR sequence is unbounded when the sets do not intersect, according to Theorem 2.9(ii) the shadow sequence remains convergent to a point a^* if the distance between the sets $d(A, B)$ is attained. In this case, $(a^*, P_B(a^*))$ would be a best approximation pair (see Figure 2.6(c)).

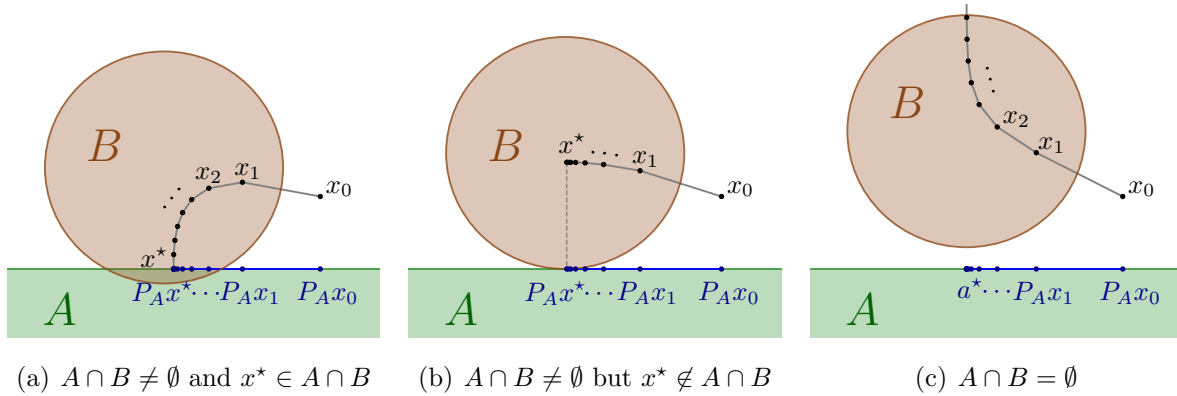


FIGURE 2.6: Behavior of the Douglas–Rachford algorithm in three possible scenarios.

The DR algorithm also provides a solution to the best approximation problem when it is applied to closed (affine) subspaces (see, e.g., [27, Corollaries 4.4 and 4.5]). However this is not the case for arbitrary convex sets (see Figure 2.7).

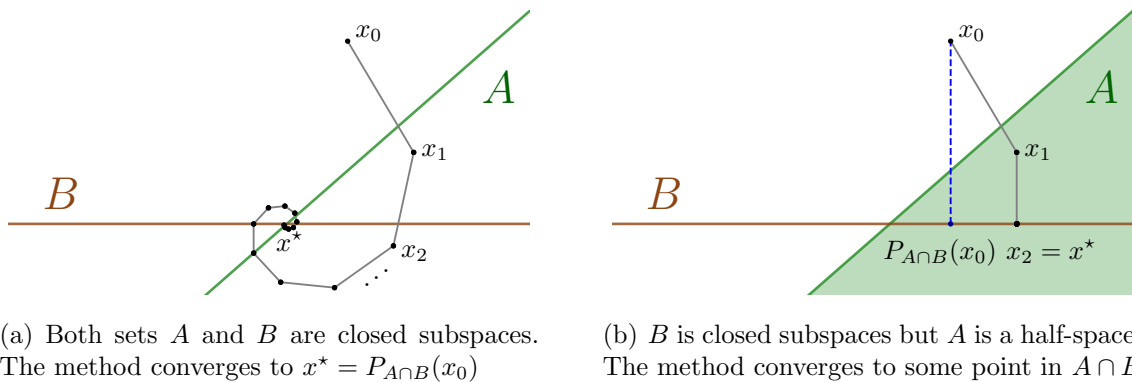


FIGURE 2.7: Failure of the Douglas–Rachford method for solving the best approximation problem for arbitrary convex sets.

2.2.2.2 Extensions for finitely many sets

We discuss now the range of possibilities for the Douglas–Rachford scheme to be applied to more than two sets. Let us consider first the case of three sets $A, B, C \subseteq \mathcal{H}$. An obvious approach would be considering the fixed point iteration generated by

$$T_{A,B,C} := \frac{\text{Id} + R_C R_B R_A}{2}. \quad (2.16)$$

This scheme will remain convergent since the operator in (2.16) is still firmly nonexpansive and has nonempty set of fixed points provided that the three sets intersect. However, the reached fixed point may fail to produce a point in the intersection (see Figure 2.8).

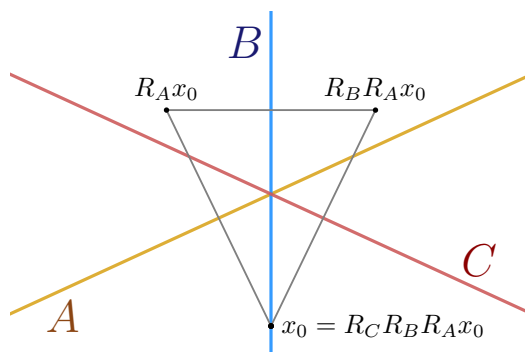


FIGURE 2.8: Failure of the 3-sets Douglas–Rachford iteration.

Therefore, alternative approaches must be addressed, some of which are presented next. Assume that $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$ are closed and convex sets.

Douglas–Rachford in the product space As pointed out in Section 2.1.1, we can always turn to the product space reformulation. In this context, given r arbitrary starting points $x_{1,0}, x_{2,0}, \dots, x_{r,0} \in \mathcal{H}$, the iterative scheme can be expressed as

$$\begin{aligned} & \text{for } k = 0, 1, 2, \dots : \\ & \left[\begin{array}{l} p_k = \frac{1}{r} \sum_{i=1}^r x_{i,k}, \\ \text{for } i = 1, 2, \dots, r : \\ \quad \left[\begin{array}{l} x_{i,k+1} = \frac{1}{2} x_{i,k} + \frac{1}{2} R_{C_i}(2p_k - x_{i,k}). \end{array} \right. \end{array} \right. \end{array} \quad (2.17)$$

Hence Theorem 2.9 guarantees the convergence of (2.17). We chose here to compute first the reflection with respect to the diagonal \mathbf{D} , so that the shadow sequence, $(P_{\mathbf{D}}(x_k))_{k=0}^{\infty}$, can be identified with $(p_k)_{k=0}^{\infty}$. Thus, the latter sequence will converge weakly to a common

point of the sets, whenever it exists (see Figure 2.9(a)). Observe that the projections onto the sets can be parallelized and simultaneously computed. However, as many points as number of sets need to be stored at each step to compute the next iteration. This makes the algorithm intractable when the number of constraints is large.

Cyclic Douglas–Rachford Instead of the operator in (2.16), Borwein and Tam [57] introduced the *cyclic Douglas–Rachford operator* defined as

$$T_{[C_1, C_2, \dots, C_r]} := T_{C_r, C_1} T_{C_{r-1}, C_r} \cdots T_{C_1, C_2}. \quad (2.18)$$

The fixex point iteration generated by this operator, known as the *cyclic Douglas–Rachford method*, does allow to address a convex feasibility problem defined by an arbitrary number of sets, without having to turn to the product space reformulation (see [57, Theorem 3.2]). The algorithm iterates by cyclically applying the classical Douglas–Rachford method to pairs of sets (see Figure 2.9(b)). The analysis of the method in the inconsistent case was developed in [59], and a further extension of the algorithm has recently been proposed in [17]. Observe that

$$T_{[A, B]} := T_{B, A} T_{A, B} \neq T_{A, B},$$

that is, (2.18) does not coincide with the classical DR operator for $r = 2$. Alternatively, Bauschke, Noll and Phan [50] proposed the *cyclically anchored Douglas–Rachford method*, which does turn into the original DR when dealing with two sets.

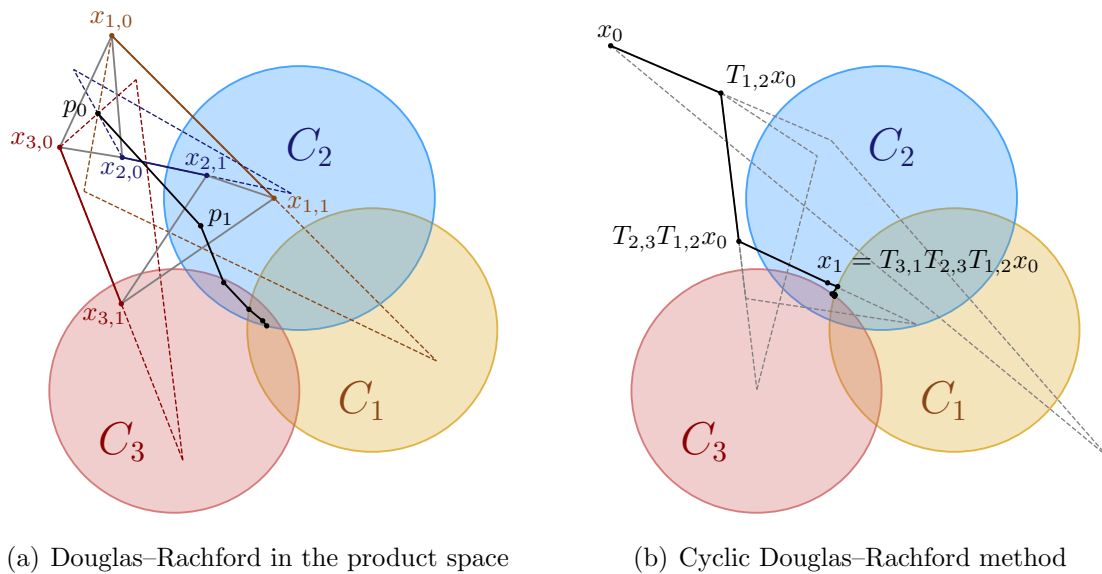


FIGURE 2.9: Illustration of two versions of DR for finitely many sets.

2.2.2.3 Some modified and relaxed versions

Some relaxations or modifications can be considered in the classical Douglas–Rachford algorithm, leading to different variants of the method. Three of them are presented next, whose iterations are illustrated in Figure 2.10.

Generalized Douglas–Rachford The *generalized Douglas–Rachford (GDR) method* is the most evident generalization, which is in fact the method studied by Eckstein and Bertsekas in [91]. The algorithm relies on iteratively applying a general α -averaged version of the DR operator, rather than a $\frac{1}{2}$ -averaged; i.e.,

$$x_{k+1} = T_{A,B,\alpha}(x_k) := (1 - \alpha)x_k + \alpha R_B R_A(x_k), \quad \text{with } \alpha \in]0, 1[. \quad (2.19)$$

Note that for $\alpha = \frac{1}{2}$ it becomes the classical Douglas–Rachford.

Relaxed Averaged Alternating Reflections Luke proposed in [142] the *relaxed averaged alternating reflections (RAAR) method*, whose iteration is defined as an average between the classical DR and the projection onto the first set; i.e.,

$$x_{k+1} = (1 - \beta)P_A(x_k) + \beta T_{A,B}(x_k), \quad \text{with } \beta \in]0, 1[.$$

This scheme is proved to converge even for inconsistent feasibility problems, without turning to the shadow sequence, as long as the distance between the sets is attained. In that case, the method provides a best approximation pair. It is worth to mention that a more general relaxed version of DR has been recently proposed by Thao [171].

Circumcentered Douglas–Rachford Motivated by the “spiraling dynamics” shown by DR when applied to subspaces (see Figure 2.7(a)), Behling, Bello Cruz and Santos considered an accelerated version of the algorithm in [52]. It is called the *circumcentered Douglas–Rachford (CDR) method*, since the new point at each step is computed as the circumcenter of the triangle implicitly generated by the classical DR iteration; that is,

$$x_{k+1} = C_T[x_k, R_A(x_k), R_B R_A(x_k)], \quad (2.20)$$

where $C_T[a, b, c]$ denotes the circumcenter of the triangle of vertices a , b and c . The method requires the sets to be closed subspaces, otherwise the circumcenter may not be defined. For one-dimensional subspaces the algorithm always converges in one iteration.

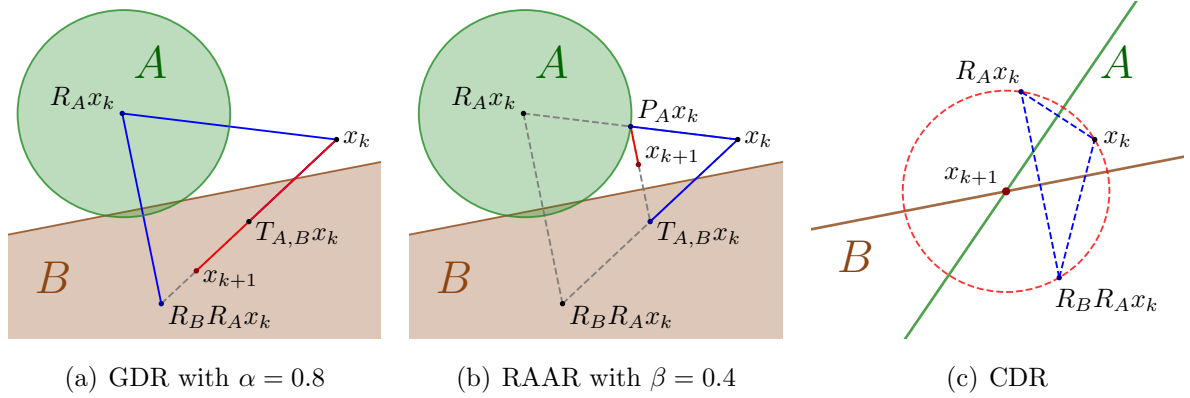


FIGURE 2.10: One iteration of GDR, RAAR and CDR methods.

2.2.2.4 Rate of convergence for subspaces

The rate of convergence of the Douglas–Rachford method when applied to two closed subspaces in a Hilbert space was analyzed in [27]. In that setting, the method converges strongly to the projection of the starting point onto the intersection. Further, the rate of convergence turns out to be the cosine of the Friedrichs angle between the subspaces.

Theorem 2.11 (Rate of convergence of DR for two closed subspaces). *Let U and V be two closed subspaces of \mathcal{H} and let $c_F := c_F(U, V)$ be the cosine of the Friedrichs angle between them. Then, for any $x_0 \in \mathcal{H}$,*

$$\|P_U T_{U,V}^k(x_0) - P_{U \cap V}(x_0)\| \leq c_F^k \|x_0\|, \quad \text{for all } k = 0, 1, 2, \dots \quad (2.21)$$

Proof. See, e.g., [27, Theorem 4.3]. □

The rate c_F in (2.21) is in fact the tightest possible since, as shown in [27, Theorem 4.3(i)], it holds that

$$\|P_U T_{U,V}^k - P_{U \cap V}\| = c_F^k, \quad \text{for all } k = 1, 2, \dots$$

REMARK 2.12 (Comparison with AP). Taking into account Theorems 2.7 and 2.11, the DR method proves to be twice slower than AP, when applied to a pair of closed subspaces. In practice, however, the shadow sequence of DR presents an oscillatory behavior with non-monotone progress (see [27, Figure 1]). This causes some unexpected situations where, at a certain iteration k_0 , it might hold that

$$\|P_U T_{U,V}^{k_0} x_0 - P_{U \cap V} x_0\| < \|(P_U P_V)^{k_0} x_0 - P_{U \cap V} x_0\|.$$

One may think that an appropriate tuning of the relaxation parameter involved in the generalized Douglas–Rachford algorithm (2.19) could speed up the method. However, the following result, which summarizes the analysis of the rate of convergence of GDR in finite-dimensional spaces given in [26], proves that the classical Douglas–Rachford algorithm is always faster.

Theorem 2.13 (Optimal parameters and rates for GDR). *Let U and V be two linear subspaces in the Euclidean space \mathbb{R}^n and let θ_F be the Friedrichs angle between them. Then the optimal rate of linear convergence of the generalized Douglas–Rachford method (2.19) applied to U and V is*

$$\gamma_\alpha(\text{GDR}) = \sqrt{\alpha(2 - \alpha) \cos^2 \theta_F + (1 - \alpha^2)}.$$

Hence, the method attains its smallest optimal rate of convergence

$$\gamma^*(\text{GDR}) = \cos \theta_F, \quad \text{at } \alpha^* = \frac{1}{2}.$$

Proof. See [26, Theorem 3.10]. □

The cosine of the Friedrichs angle is also a rate of linear convergence for the Circumcentered Douglas–Rachford method (2.20) (see [52, Theorem 1]). Then CDR will converge at least as fast as the classical Douglas–Rachford algorithm does. However it is not known how sharp this rate is. The numerical experiments in [52] suggest that CDR has a better performance than DR.

2.2.2.5 Generalization for monotone operators

The Douglas–Rachford scheme can be more generally applied to monotone operators [140]. In this context, the DR algorithm can be used to solve problems of the form

$$\text{Find } x \in \text{zer}(A + B) = \{x \in \mathcal{H} \mid 0 \in Ax + Bx\}, \quad (2.22)$$

where $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ are maximally monotone operators. The general structure of the iteration is the same as in the feasibility context, but replacing the projectors onto the sets with the resolvents J_A and J_B of the operators, i.e.,

$$x_{k+1} := \frac{1}{2}x_k + \frac{1}{2}(2J_B - \text{Id})(2J_A - \text{Id})(x_k). \quad (2.23)$$

In fact, a convex feasibility problem of the form

$$\text{Find } x \in C_1 \cap C_2,$$

can be written in the form (2.22) by taking $A = N_{C_1}$ and $B = N_{C_2}$. Indeed, we have that

$$x \in C_1 \cap C_2 \Leftrightarrow 0 \in N_{C_1}(x) \cap N_{C_2}(x) \Leftrightarrow 0 \in N_{C_1}(x) + N_{C_2}(x). \quad (2.24)$$

Moreover, as shown in Example 1.25(ii), the resolvent of a normal cone to a convex set coincides with the projector onto the set, and thus (2.23) becomes the classical DR iteration for solving feasibility problems (2.15).

Observe that in the case when the operator $A + B$ is also maximally monotone, problem (2.22) could be addressed with the proximal-point algorithm given in Theorem 1.30. However, this approach would only be viable when the resolvent of the sum J_{A+B} was readily computable, something rather unusual. By contrast, the Douglas–Rachford splitting algorithm exploits Proposition 1.28(i) and permits to find a zero of the sum of two maximally monotone operators, with no further assumptions on the operators and only involving individual evaluations of their resolvents. Some of the main convergence properties of the algorithm, stated in its most general form, are collected in the following result.

Theorem 2.14 (Douglas–Rachford splitting algorithm). *Let $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ be maximally monotone operators such that $\text{zer}(A + B) \neq \emptyset$, let $\gamma > 0$ and let $(\lambda_k)_{k=0}^\infty$ be a sequence in $[0, 1]$ such that $\sum_{k=0}^\infty \lambda_k(1 - \lambda_k) = +\infty$. Given any $x_0 \in \mathcal{H}$, set*

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k R_{\gamma B} R_{\gamma A}(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Then there exists $x^ \in \text{Fix}(R_{\gamma B} R_{\gamma A})$ such that following assertions hold.*

- (i) $(x_{k+1} - x_k)_{k=0}^\infty$ converges strongly to 0.
- (ii) $(x_k)_{k=0}^\infty$ converges weakly to x^* , and $J_{\gamma A}(x^*) \in \text{zer}(A + B)$.
- (iii) $(J_{\gamma A}(x_k))_{k=0}^\infty$ converges weakly to $J_{\gamma A}(x^*)$.
- (iv) Suppose that either A or B is μ -strongly monotone for some constant $\mu > 0$. Then $(J_{\gamma A}(x_k))_{k=0}^\infty$ converges strongly to the unique point in $\text{zer}(A + B)$.

Proof. See, e.g., [35, Theorem 26.11]. □

The assertion in Theorem 2.14(iv) can be still guaranteed even when $\lambda_k = 1$, for each $k = 0, 1, \dots$. In fact, this limiting case corresponds to the well known *Peaceman–Rachford algorithm* [150].

Theorem 2.15 (Peaceman–Rachford splitting algorithm). *Let $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ be maximally monotone operators such that $\text{zer}(A+B) \neq \emptyset$ with A being μ -strongly monotone for some constant $\mu > 0$, and let $\gamma > 0$. Given any $x_0 \in \mathcal{H}$, set*

$$x_{k+1} = R_{\gamma B} R_{\gamma A}(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Then $(J_{\gamma A}(x_k))_{k=0}^{\infty}$ converges strongly to the unique point in $\text{zer}(A+B)$.

Proof. See, e.g., [35, Proposition 26.13]. □

The problem of finding a zero in a finite sum of maximally monotone operators can be tackled again by an appropriate extension of the Pierra’s product space formulation exhibited in Section 2.1.1. The following result collects the fundamentals of the reformulation within this context.

Proposition 2.16 (Product space reformulation for operators). *Given a finite family of operators $A_i : \mathcal{H} \rightrightarrows \mathcal{H}$, $i = 1, 2, \dots, r$, consider the product Hilbert space \mathcal{H} as in (2.3) and the diagonal set \mathbf{D} as in (2.4). Define the operator $\mathbf{A} : \mathcal{H} \rightrightarrows \mathcal{H}$ by*

$$\mathbf{A}(\mathbf{x}) := A_1(x_1) \times A_2(x_2) \times \cdots \times A_r(x_r), \quad \forall \mathbf{x} = (x_1, x_2, \dots, x_r) \in \mathcal{H}. \quad (2.25)$$

The following hold.

(i) *The resolvent of \mathbf{A} can be computed as*

$$J_{\mathbf{A}}(\mathbf{x}) = J_{A_1}(x_1) \times J_{A_2}(x_2) \times \cdots \times J_{A_r}(x_r), \quad \forall \mathbf{x} = (x_1, x_2, \dots, x_r) \in \mathcal{H}.$$

Further, the operator \mathbf{A} is (maximally) monotone whenever A_1, A_2, \dots, A_r are so.

(ii) *The normal cone to \mathbf{D} is a maximally monotone operator described by*

$$N_{\mathbf{D}}(\mathbf{x}) = \begin{cases} \{\mathbf{u} = (u_1, u_2, \dots, u_r) \in \mathcal{H} \mid \sum_{i=1}^r u_i = 0\}, & \text{if } \mathbf{x} \in \mathbf{D}, \\ \emptyset, & \text{otherwise.} \end{cases}$$

(iii) $\text{zer}(\mathbf{A} + N_{\mathbf{D}}) = \mathbf{j}(\text{zer}(\sum_{i=1}^r A_i))$.

Proof. See, e.g., [35, Proposition 26.4]. □

2.2.2.6 The Douglas–Rachford algorithm in nonconvex settings

The Douglas–Rachford algorithm has recently gained much popularity, in part thanks to its good behavior in nonconvex settings. In this framework, observe that the DR operator may be multivalued due to the fact that the projection onto nonconvex sets is not necessarily unique. Therefore, the equality in (2.15) must be replaced by an inclusion, and the iteration takes the form

$$x_{k+1} \in T_{A,B}(x_k) := \{x_k + b_k - a_k \in \mathcal{H} : a_k \in P_A(x_k), b_k \in P_B(2a_k - x_k)\}. \quad (2.26)$$

Despite that the convergence of the algorithm is only guaranteed for convex sets, the method has been successfully employed for solving many different nonconvex optimization problems, specially those of combinatorial nature. Some of these applications include, among others, matrix completion [8], protein conformation determination [58], phase and bit retrieval [39, 93, 95], differential equations [133], and a wide variety of NP-hard problems such as Sudoku, 3-Satisfiability and graph coloring [10, 94].

However, the theory is much more limited. There are very few results explaining why the algorithm works in nonconvex settings, and even less justifying its good global performance. The first nonconvex scenario was considered by Borwein and Sims [60]. They proved local convergence of the Douglas–Rachford iteration near each of the intersection points of a line and a sphere in a Euclidean space. Specific regions of convergence were later provided by Aragón Artacho and Borwein [6]. It was finally Benoist [54] who, via the construction of a Lyapunov function, established the convergence of the algorithm for every starting point not lying on the hyperplane of symmetry. Lyapunov functions are powerful tools in difference inclusions whose existence guarantees the convergence of the iteration. By using this approach, Dao and Tam [79] proved global convergence of the Douglas–Rachford algorithm for finding a zero of a function, with applications to several nonconvex feasibility problems.

From a different perspective, Aragón Artacho, Borwein and Tam [9] proved global convergence for the case of a halfspace and a potentially nonconvex set (possibly finite). We make use of this scenario to illustrate in Figure 2.11 the difference between the behavior of Douglas–Rachford and alternating projections when addressing combinatorial problems. While AP usually gets stuck in those points which are close to be solutions (as it finds a local best approximation pair), DR is satisfactorily capable to escape from them (thanks to Theorem 2.9(ii)). Even so, the Douglas–Rachford algorithm does not break free from getting caught by cycles in other nonconvex settings. Although this is something that

does not seem to happen very often, it may be hard to detect. It is worth to mention that the cycling of the algorithm for a simple inconsistent nonconvex feasibility problem, specifically, a hyperplane and a doubleton, has been recently analyzed in [41].

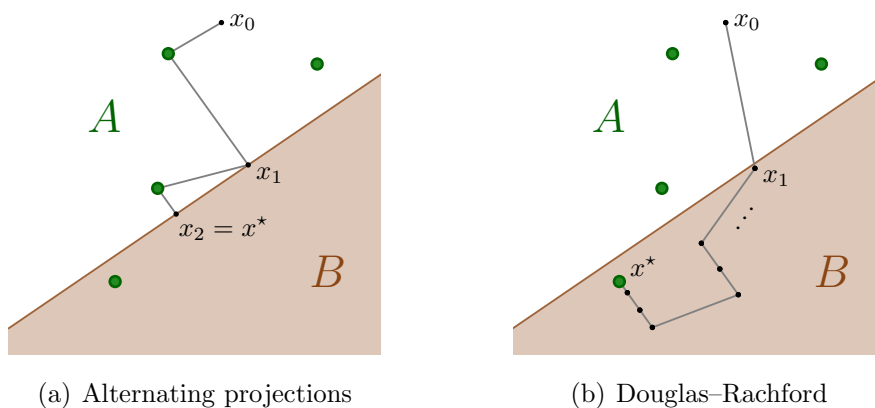


FIGURE 2.11: AP and DR algorithms applied to a finite set and a halfspace.

Local convergence of the algorithm in nonconvex settings has been established, for instance, for the case of a line and an ellipse or a p -sphere [56], and for union of convex sets [49]. Other results regarding local convergence are usually obtained by requiring regularity properties of the sets and/or of their intersection, see e.g. [40, 116, 117, 151].

2.3 Projection algorithms for best approximation problems

The two methods described in the previous section solve best approximation problems when the involved sets are closed affine subspaces. However, they only solve feasibility problems for more general sets. There are other approaches based on individual projections onto the sets to solve best approximation problems. These are sometimes called *best approximation methods*. A good variety of them has been recently collected in [43, Section 4.2], see also [35, Chapter 30]. In this section we shall focus on *Dykstra's algorithm*, *Haugazeau's algorithm*, *Halpern's algorithm* and *Combettes' algorithm*.

2.3.1 Dykstra's algorithm

Dykstra's algorithm is probably the most well-known best approximation algorithm. It arose as a suitable modification of the method of alternating projections that forces strong convergence to the solution of the best approximation problem. The method was first

proposed by Dykstra [90] in 1983 for closed and convex cones in Euclidean spaces, and then extended by Boyle and Dykstra [61] in 1986 for arbitrary closed and convex sets in a Hilbert space. An in-depth discussion on the algorithm can be found in [28].

Theorem 2.17 (Dykstra's algorithm). *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets and let $v := P_{B-A}(0)$. Given any $x_0 \in \mathcal{H}$, define $p_0 := q_0 := 0$ and set*

$$\text{for } k = 0, 1, 2, \dots : \quad \begin{cases} a_{k+1} = P_A(x_k + p_k), & p_{k+1} = x_k + p_k - a_{k+1}, \\ x_{k+1} = P_B(a_{k+1} + q_k), & q_{k+1} = a_{k+1} + q_k - x_{k+1}. \end{cases} \quad (2.27)$$

Then $x_k - a_k \rightarrow v$ and exactly one of the following holds:

- (i) $v \in B - A$ and $(x_k)_{k=0}^{\infty}$ converges strongly to $P_{(A+v) \cap B}(x_0)$;
- (ii) $v \notin B - A$ and $\|x_k\| \rightarrow +\infty$.

Proof. See, e.g., [28, Theorem 3.8]. □

REMARK 2.18. Dykstra's iteration (2.27) can be seen as a modified version of AP (2.7) perturbed by the sequences of increments $(p_k)_{k=0}^{\infty}$ and $(q_k)_{k=0}^{\infty}$ (see Figure 2.12). Thank to this, the convergence of the algorithm becomes strong and the limit point is not only a point in the intersection but the closest to the starting point. It can be proved that the sequences of increments have no effect when the method is applied to affine subspaces, so in this context, Dykstra's algorithm coincides with AP (see, e.g., [81, pp. 215–216]).

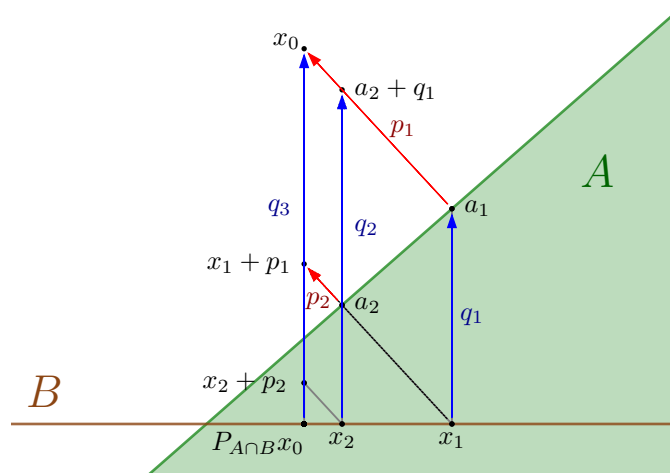


FIGURE 2.12: Illustration of Dykstra's algorithm.

REMARK 2.19. Theorem 2.17 states a complete characterization of Dykstra's iteration for non-necessarily consistent problems. Note that the algorithm behaves similarly to AP in the three possible scenarios considered in Figure 2.1, but Dykstra's algorithm always provides the nearest best approximation pair to the initial point, if it exists (particularly, the closest point in the intersection when the problem is consistent).

Although we have presented the algorithm to deal with two sets, Boyle and Dykstra [61] directly provided a version for finitely many sets, which is usually known as the *cyclic Dykstra's algorithm* (see, e.g., [35, Theorem 30.7]). A parallelized version can be constructed by turning to the product space reformulation. It is worth to mention that such a variant of the algorithm was originally introduced by Gaffke and Mathar [105] in 1989.

2.3.2 Haugazeau-like algorithms

The *Haugazeau-like algorithms* introduce a projector onto the intersection of two half-spaces, which can be explicitly computed, combined in a suitable manner with another projection algorithm. This combination ensures the strong convergence of the algorithm to the closest point in the set of fixed points. In the following result we show Haugazeau's algorithm on its basic form, which was first proposed by Y. Haugazeau [115] in 1968.

Theorem 2.20 (Haugazeau's algorithm). *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets such that $A \cap B \neq \emptyset$. Consider the operator $Q : \mathcal{H} \times \mathcal{H} \times \mathcal{H} \rightarrow \mathcal{H}$ defined by*

$$Q(x, y, z) := \begin{cases} z, & \text{if } \rho = 0 \text{ and } \chi \geq 0, \\ x + \left(1 + \frac{\chi}{\nu}\right)(z - y), & \text{if } \rho > 0 \text{ and } \chi\nu \geq \rho, \\ y + \frac{\nu}{\rho}(\chi(x - y) + \mu(z - y)), & \text{if } \rho > 0 \text{ and } \chi\nu < \rho; \end{cases}$$

where

$$\chi = \langle x - y, y - z \rangle, \quad \mu = \|x - y\|^2, \quad \nu = \|y - z\|^2 \quad \text{and} \quad \rho = \mu\nu - \chi^2.$$

Given any $x_0 \in \mathcal{H}$, set

for $k = 0, 1, 2, \dots$:

$$\begin{cases} y_k = Q(x_0, x_k, P_A(x_k)), \\ x_{k+1} = Q(x_0, y_k, P_B(y_k)). \end{cases} \quad (2.28)$$

Then $(x_k)_{k=0}^{\infty}$ converges strongly to $P_{A \cap B}(x_0)$.

Proof. See, e.g., [35, Corollary 30.15]. □

We illustrate the iteration of Haugazeau's algorithm in Figure 2.13.

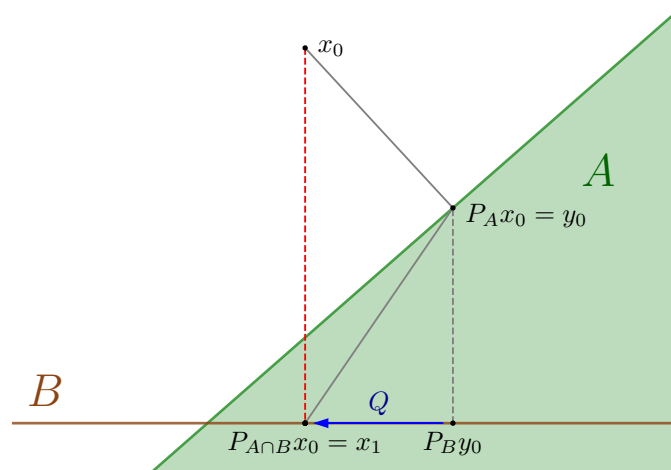


FIGURE 2.13: Illustration of Haugazeau's algorithm.

Thanks to the *weak-to-strong convergence principle* given in [34], different modifications of the method have been obtained. For instance, a *Haugazeau-like averaged alternating reflections (HAAR) method* was constructed in [37].

2.3.3 Halpern's algorithm

Another scheme is the one originally proposed by Halpern [114] in 1967. This method can be seen as a modified version of the Krasnosel'skiĭ–Mann iteration given in Theorem 1.13, whose strong convergence to the closest fixed point has been proved by different authors, under different conditions for the parameters. The main contributions are due to Lions [141] in 1977, Wittmann [175] in 1992, and Bauschke [25] in 1996. As a result, this algorithm is sometimes called the *Halpern–Lions–Wittmann–Bauschke (HLWB) method*. The scheme in the context of best approximation is presented in the following theorem.

Theorem 2.21 (Halpern's algorithm). *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets such that $A \cap B \neq \emptyset$, let $z \in \mathcal{H}$ and let $(\lambda_k)_{k=0}^{\infty}$ be a sequence in $]0, 1[$ verifying*

$$\lambda_k \rightarrow 0, \quad \sum_{k=0}^{\infty} \lambda_k = +\infty \quad \text{and} \quad \sum_{k=0}^{\infty} |\lambda_{k+1} - \lambda_k| < +\infty. \quad (2.29)$$

Given any $x_0 \in \mathcal{H}$, set

$$x_{k+1} = \lambda_k z + (1 - \lambda_k) P_B P_A(x_k), \quad \text{for } k = 0, 1, 2, \dots \quad (2.30)$$

Then $(x_k)_{k=0}^\infty$ converges strongly to $P_{A \cap B}(z)$.

Proof. See, e.g., [35, Theorem 30.1]. \square

REMARK 2.22. Observe that, unlike in previously mentioned fixed point iterations, the sequence of parameters $(\lambda_k)_{k=0}^\infty$ involved in Theorem 2.21 is not allowed to be constant. The simplest choice for that sequence is then

$$\lambda_k := \frac{1}{2+k}, \quad \text{for } k = 0, 1, 2, \dots \quad (2.31)$$

One can easily check that this sequence verifies the assumptions in (2.29).

The iteration generated by Halpern's algorithm using the sequence in (2.31) is shown in Figure 2.14.

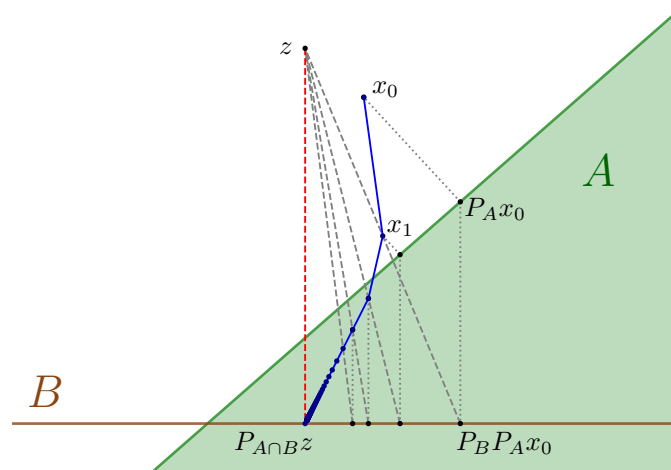


FIGURE 2.14: Illustration of Halpern's algorithm.

2.3.4 Combettes' method

It is also worth to mention the work of Combettes [77], where a Douglas–Rachford-like strongly convergent algorithm is proposed to compute the resolvent of the sum of maximally monotone operators. Particularly, under the strong CHIP property, the scheme can be applied for solving best approximation problems. The algorithm was originally introduced for dealing with finitely many operators by relying in the product space. The corresponding version for solving best approximation problems is stated in the next result.

Theorem 2.23 (Combettes' algorithm). Let $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$ be nonempty closed and convex sets such that $\bigcap_{i=1}^r C_i \neq \emptyset$ and $N_{\bigcap_{i=1}^r C_i} = N_{C_1} + N_{C_2} + \dots + N_{C_r}$. Let $z \in \mathcal{H}$, let $\gamma > 0$ and let $(\lambda_k)_{k=0}^{\infty}$ be a sequence in $]0, 2]$ such that $\inf_{k \geq 0} \lambda_k > 0$. Given r arbitrary initial points $w_{1,0}, w_{2,0}, \dots, w_{r,0} \in \mathcal{H}$, set

$$\begin{aligned} & \text{for } k = 0, 1, 2, \dots : \\ & \left[\begin{array}{l} p_k = \frac{1}{r} \sum_{i=1}^r w_{i,k}, \\ \text{for } i = 1, 2, \dots, r : \\ \quad \left[\begin{array}{l} y_{i,k} = P_{C_i} \left(\frac{w_{i,k} + \gamma z}{\gamma + 1} \right); \\ x_k = \frac{1}{r} \sum_{i=1}^r y_{i,k}, \\ \text{for } i = 1, 2, \dots, r : \\ \quad \left[\begin{array}{l} w_{i,k+1} = w_{i,k} + \lambda_k (2x_k - p_k - y_{i,k}). \end{array} \right. \end{array} \right. \end{array} \right. \end{array} \quad (2.32)$$

Then $(x_k)_{k=0}^{\infty}$ converges strongly to $P_{\bigcap_{i=1}^r C_i}(z)$.

Proof. See [77, Theorem 2.8]. □

In Figure 2.15 we illustrate the iterative scheme of Combettes' method with parameters $\gamma = 0.5$ and $\lambda_k = 0.5$, for projecting the point $z = 0$ onto the intersection of three sets.

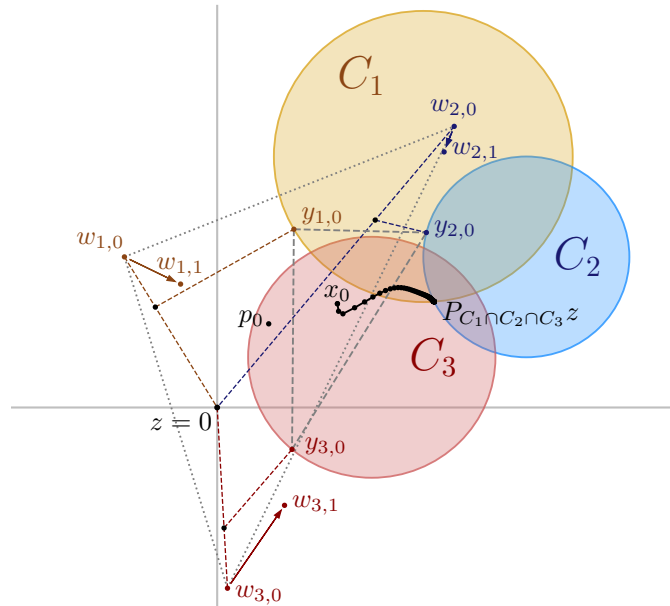


FIGURE 2.15: Illustration of Combettes' algorithm.

REMARK 2.24 (Splitting algorithms for computing the resolvent of the sum). Best approximation problems can be seen as particular instances of the problem consisting in computing the resolvent of a sum of maximally monotone operators (under the strong CHIP). Given a point z in the domain of $J_{\sum_{i=1}^r A_i}$ (i.e., in the range of $\text{Id} + \sum_{i=1}^r A_i$), the generalized version of the best approximation problem (2.2) can be stated as,

$$\text{Find } w = J_{\sum_{i=1}^r A_i}(z), \quad (2.33)$$

for some maximally monotone operators $A_1, A_2, \dots, A_r : \mathcal{H} \rightrightarrows \mathcal{H}$. This is indeed a generalization of the best approximation problem (2.2). Note that, if the sets defining the problem have the strong CHIP, by Example 1.25(ii), we have that

$$P_{\bigcap_{i=1}^r C_i}(z) = J_{N_{\bigcap_{i=1}^r C_i}}(z) = J_{N_{C_1 + N_{C_2} + \dots + N_{C_r}}}(z),$$

and thus (2.33) becomes (2.2).

- (i) Combettes' method was, in fact, originally proposed in [77] within this context, as follows. Given $A_1, A_2, \dots, A_r : \mathcal{H} \rightrightarrows \mathcal{H}$ maximally monotone operators, and given $(\omega_i)_{i=1}^r \subset]0, 1[$ such that $\sum_{i=1}^r \omega_i = 1$, consider the operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$ defined by

$$A := \omega_1 A_1 + \omega_2 A_2 + \dots + \omega_r A_r.$$

If we consider the iteration in (2.32) but replacing, for each $i = 1, 2, \dots, r$, the projector P_{C_i} by the resolvent $J_{\frac{\gamma}{\gamma+1} A_i}$, then

$$x_k \rightarrow J_A(z).$$

- (ii) Another iterative approach can be found in [33], where Dykstra's algorithm was extended to monotone operators.
- (iii) In [78], or the more recent work [3], the particular case of proximity mappings (see Example 1.25(i)) is tackled.

Chapter 3

The averaged alternating modified reflections method

In this chapter we present a new iterative projection method for finding the closest point in the intersection of convex sets to any arbitrary point in a Hilbert space. This method can be viewed as an adequate modification of the Douglas–Rachford algorithm. Precisely, each reflector in the DR operator (2.14) is replaced by what we call a *modified reflector*, which is defined for a given set $C \subseteq \mathcal{H}$ as

$$2\beta P_C - \text{Id}, \quad \text{with } \beta \in]0, 1[.$$

For this reason, the new algorithm is termed the *averaged alternating modified reflections (AAMR) method*. Surprisingly, the slight modification in the reflector operators completely changes the dynamics of the sequence generated by the scheme. It permits to find, not only a point in the intersection of convex sets, but the closest point in the intersection to any arbitrary point in the space. Moreover, it forces the strong convergence of the shadow sequence, which, like in the DR scheme, is the sequence of interest as it is the one that converges to the solution of the problem. The convergence of the method is conditioned to a constraint qualification to be held at the point of interest. In fact, it is the strong CHIP property what fully characterizes the convergence of the AAMR method for every point in the space.

We report some promising numerical experiments where we compare the performance of AAMR against other projection methods for finding the closest point in the intersection of pairs of finite-dimensional subspaces. Motivated by this, we obtain the rate of linear convergence of the AAMR method in terms of the Friedrichs angle between the subspaces and the parameters defining the scheme, by studying the linear convergence rates of the powers of matrices. We further optimize the value of these parameters in order

to get the minimal convergence rate, which turns out to be better than the one of other projection methods. We also provide some others numerical experiments that demonstrate the theoretical results.

The averaged alternating modified reflections algorithm belongs to the class of best approximation methods introduced in Section 2.3. Recall that some of them have been successfully extended to the monotone operator context (see Remark 2.24). This is also the case for the AAMR: the scheme can be generalized so that it can be used to compute the resolvent of the sum of two maximally monotone operators. This gives rise to a new splitting method, and the standard product space reformulation permits to apply it for computing the resolvent of a finite sum of maximally monotone operators. Based on this, we propose two variants of such parallel splitting method.

The remainder of the chapter is structured as follows. In Section 3.1 we introduce the AAMR operator and analyze its main properties, as well as we establish some convergence results for the new projection scheme. The rate of linear convergence of the method for the case of two subspaces in a finite-dimensional space is addressed in Section 3.2. Finally, in Section 3.3, we extend the algorithm so that it can deal with maximally monotone operators to compute the resolvent of their sum.

3.1 A new best approximation algorithm

Given two nonempty closed and convex subsets A, B of a Hilbert space \mathcal{H} and any point $z \in \mathcal{H}$, we are interested in solving the best approximation problem of finding the closest point to z in $A \cap B$; i.e.,

$$\text{Find } w \in A \cap B \text{ such that } \|w - z\| = \inf_{x \in A \cap B} \|x - z\|. \quad (3.1)$$

For any pair of parameters $\alpha \in]0, 1]$ and $\beta \in]0, 1[$, we introduce the *averaged alternating modified reflections operator* (AAMR operator), which is the operator $T_{A,B,\alpha,\beta} : \mathcal{H} \mapsto \mathcal{H}$ given by

$$T_{A,B,\alpha,\beta} := (1 - \alpha) \text{Id} + \alpha(2\beta P_B - \text{Id})(2\beta P_A - \text{Id}). \quad (3.2)$$

Given any initial point $x_0 \in \mathcal{H}$, we define a new projection method termed *averaged alternating modified reflections (AAMR) method*, which is iteratively defined by

$$\boxed{x_{k+1} := T_{A-z, B-z, \alpha, \beta}(x_k), \quad \text{for } k = 0, 1, 2, \dots} \quad (3.3)$$

If $A \cap B \neq \emptyset$, under the *constraint qualification*

$$z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z)), \quad (3.4)$$

we shall show (Theorem 3.17) that if $\alpha < 1$, the sequence generated by (3.3) weakly converges to a point x^* such that

$$P_A(z + x^*) = P_{A \cap B}(z),$$

and the *shadow sequence* $(P_A(z + x_k))_{k=0}^{\infty}$ is strongly convergent to $P_{A \cap B}(z)$ (even when $\alpha = 1$), and thus solves problem (3.1). Although we show that the strong CHIP of $\{A, B\}$ at the point $P_{A \cap B}(z)$ is sufficient but not necessary for the convergence of AAMR (see Example 3.21), the strong CHIP turns out to be the precise condition to be required for the convergence of the method *for every point* $z \in \mathcal{H}$ (see Theorem 3.17 and Proposition 3.22).

The Douglas–Rachford algorithm or, more precisely, its generalized version GDR (2.19), can be seen as a limiting case of AAMR when $\beta = 1$ and $z = 0$ in (3.3). Nonetheless, the behavior is remarkably different since, in general, GDR would only provide a point in the intersection of the sets whereas AAMR would find the closest one to z .

Observe that some of the methods discussed in Section 2.3, and also AP and DR for affine subspaces, produce an iterative sequence that converges to the projection of the initial point onto the intersection of the sets. As it happens to Halpern’s algorithm and Combettes’ method, the initial point of AAMR can be arbitrarily chosen in the space.

Furthermore, it is important to point out that, in general, the set of fixed points of the operator $T_{A,B,\alpha,\beta}$ is not equal to the intersection of the sets of fixed points of the operators $2\beta P_B - \text{Id}$ and $2\beta P_A - \text{Id}$. Therefore, the operator $T_{A,B,\alpha,\beta}$ does not belong to the broad family of operators studied in [154], see Remark 3.7(ii) for additional details.

3.1.1 The averaged alternating modified reflections operator

We begin this section with the following simple result that motivates the definition of what we call a *modified reflection*.

Proposition 3.1. *Let D be a nonempty subset of \mathcal{H} and let $T : D \mapsto \mathcal{H}$. If T is firmly nonexpansive, then $2\beta T - \text{Id}$ is nonexpansive for any $\beta \in]0, 1]$.*

Proof. Since T is firmly nonexpansive, the operator βT is firmly nonexpansive for any $\beta \in]0, 1]$. The result follows from Proposition 1.8(i). \square

Definition 3.2 (Modified reflector). Let $C \subseteq \mathcal{H}$ be a nonempty closed and convex set. Given any $\beta \in]0, 1]$, the operator $2\beta P_C - \text{Id}$ is called a modified reflector operator.

REMARK 3.3. Observe that for the case $\beta = 1$ it coincides with the classical reflector R_C . In general, for any $\beta \in]0, 1]$ and any $x \in \mathcal{H}$, one has

$$(2\beta P_C - \text{Id})(x) = \beta R_C(x) + \beta x - x = \beta R_C(x) + (1 - \beta)(-x);$$

that is, $(2\beta P_C - \text{Id})(x)$ is a convex combination of $R_C(x)$ and $-x$ (see Figure 3.1).

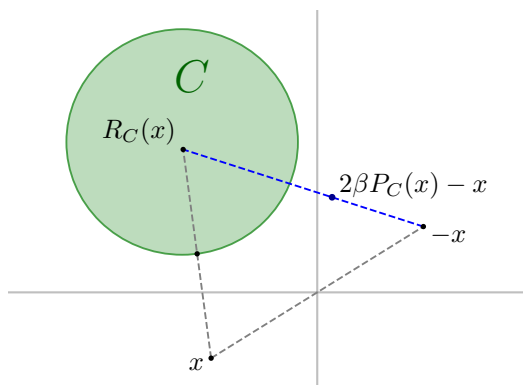


FIGURE 3.1: Geometric interpretation of the modified reflector.

The next result shows that the modified reflector operators have a unique fixed point.

Proposition 3.4. Let $C \subseteq \mathcal{H}$ be nonempty, closed and convex, and let $\beta \in]0, 1[$. Then

$$\text{Fix}(2\beta P_C - \text{Id}) = \{\beta P_C(0)\}.$$

Proof. First, notice that $y \in \text{Fix}(2\beta P_C - \text{Id})$ if and only if $\beta P_C(y) = y$; that is,

$$\text{Fix}(2\beta P_C - \text{Id}) = \text{Fix}(\beta P_C).$$

Since $\beta P_C(0) = P_C(0) + (1 - \beta)(0 - P_C(0))$, with $1 - \beta > 0$, according to Proposition 1.2(iii) we deduce that $P_C(\beta P_C(0)) = P_C(0)$; and thus

$$\beta P_C(0) \in \text{Fix}(2\beta P_C - \text{Id}).$$

The uniqueness directly follows from the fact that βP_C is a contraction, since P_C is firmly nonexpansive (particularly nonexpansive) by Proposition 1.11. \square

Definition 3.5 (AAMR operator). Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets. Given $\alpha \in]0, 1]$ and $\beta \in]0, 1[$, we define the averaged alternating modified reflections (AAMR) operator $T_{A,B,\alpha,\beta} : \mathcal{H} \mapsto \mathcal{H}$ as

$$T_{A,B,\alpha,\beta} := (1 - \alpha) \text{Id} + \alpha(2\beta P_B - \text{Id})(2\beta P_A - \text{Id}). \quad (3.5)$$

Where there is no ambiguity, we will abbreviate the notation $T_{A,B,\alpha,\beta}$ by $T_{\alpha,\beta}$.

Proposition 3.6. If $A, B \subseteq \mathcal{H}$ are nonempty, closed and convex sets, then the AAMR operator $T_{\alpha,\beta}$ is nonexpansive for all $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Moreover, if $\alpha \in]0, 1[$, then $T_{\alpha,\beta}$ is α -averaged, and thus strictly quasi-nonexpansive.

Proof. It is straightforward, in view of Proposition 3.1 and Proposition 1.8(ii). \square

A geometric interpretation of the AAMR operator $T_{A,B,\alpha,\beta}$ is shown in Figure 3.2.

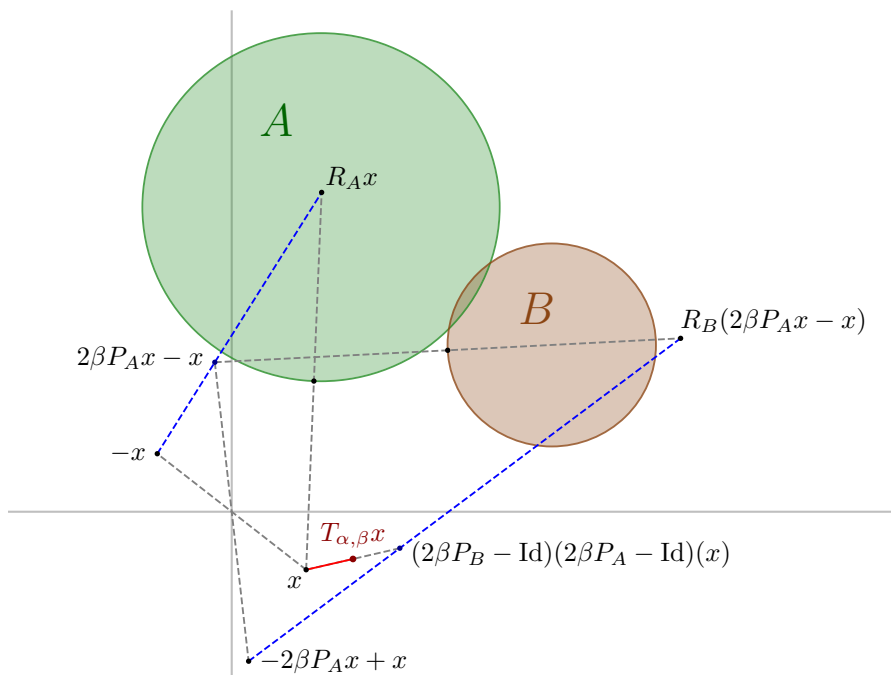


FIGURE 3.2: Geometric interpretation of the AAMR operator.

It is important to emphasize that we require $\beta < 1$ in the definition of the AAMR operator. The case $\beta = 1$ in (3.5) corresponds with the (generalized) Douglas–Rachford operator $DR_{A,B,\alpha}$ given in (2.19), whose behavior is significantly different. In particular, one has $P_A(\text{Fix } DR_{A,B,\alpha}) = A \cap B$, while $P_A(\text{Fix } T_{\alpha,\beta}) \subsetneq A \cap B$, as we show in the next remark.

REMARK 3.7 (On the set of fixed points of the AAMR operator).

- (i) Observe that $\text{Fix } T_{\alpha,\beta} = \text{Fix}((2\beta P_B - \text{Id})(2\beta P_A - \text{Id}))$ for all $\alpha \in]0, 1[$. In fact, $x \in \text{Fix } T_{\alpha,\beta}$ if and only if

$$\begin{aligned} x &= T_{\alpha,\beta}(x) = (1 - \alpha)x + \alpha(2\beta P_B - \text{Id})(2\beta P_A(x) - x) \\ &= (1 - \alpha)x + 2\alpha\beta P_B(2\beta P_A(x) - x) - \alpha(2\beta P_A(x) - x) \\ &= x + 2\alpha\beta P_B(2\beta P_A(x) - x) - 2\alpha\beta P_A(x); \end{aligned}$$

that is,

$$x \in \text{Fix } T_{\alpha,\beta} \quad \Leftrightarrow \quad P_B(2\beta P_A(x) - x) = P_A(x). \quad (3.6)$$

As a consequence, we have $P_A(\text{Fix } T_{\alpha,\beta}) \subset A \cap B$. In general, though,

$$P_A(\text{Fix } T_{\alpha,\beta}) \neq A \cap B.$$

For a simple example, choose any $\beta \in]0, 1[$ and consider the sets

$$A := \mathbb{R}^2 \quad \text{and} \quad B := \{(x_1, x_2) \in \mathbb{R}^2 \mid x_1 = 1\}.$$

Then, it can be easily checked using (3.6) that $\text{Fix } T_{\alpha,\beta} = \{(1, 0)\}$ and

$$P_A(\text{Fix } T_{\alpha,\beta}) = P_A(\{(1, 0)\}) = \{(1, 0)\} \neq A \cap B = B.$$

- (ii) As a consequence of Proposition 3.4, $\text{Fix}(2\beta P_A - \text{Id}) \cap \text{Fix}(2\beta P_B - \text{Id}) = \emptyset$ whenever $P_A(0) \neq P_B(0)$. Hence, in general,

$$\text{Fix } T_{\alpha,\beta} = \text{Fix}((2\beta P_B - \text{Id})(2\beta P_A - \text{Id})) \neq \text{Fix}(2\beta P_A - \text{Id}) \cap \text{Fix}(2\beta P_B - \text{Id}).$$

For instance, consider the same example as in (i). By Proposition 3.4, we have $\text{Fix}(2\beta P_A - \text{Id}) = \{(0, 0)\}$ and $\text{Fix}(2\beta P_B - \text{Id}) = \{(\beta, 0)\}$, while $\text{Fix } T_{\alpha,\beta} = \{(1, 0)\}$.

Therefore, the operator $T_{A,B,\alpha,\beta}$ does not belong to the broad family of operators studied by Reich and Zalas in [154], which covers many projection algorithms, because they consider the general problem of finding $x \in \bigcap_{i=1}^n \text{Fix } U_i \neq \emptyset$ for some quasi-nonexpansive operators $U_i : \mathcal{H} \rightarrow \mathcal{H}$.

The following result shows that, in fact, the fixed points of the AAMR operator $T_{\alpha,\beta}$ are very special.

Proposition 3.8. *Let A and B be two nonempty, closed and convex subsets of \mathcal{H} , and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. If $x \in \text{Fix} T_{\alpha, \beta}$, then $A \cap B \neq \emptyset$ and*

$$P_A(x) = P_{A \cap B}(0).$$

Proof. If $x \in \text{Fix} T_{\alpha, \beta}$, we know by (3.6) that

$$P_A(x) = P_B(2\beta P_A(x) - x),$$

which implies $P_A(x) \in A \cap B$. Using twice the characterization of the projections given in Proposition 1.2(ii), we obtain

$$\langle y - P_A(x), x - P_A(x) \rangle \leq 0, \quad \forall y \in A, \quad (3.7)$$

and

$$\langle y - P_A(x), 2\beta P_A(x) - x - P_A(x) \rangle \leq 0, \quad \forall y \in B. \quad (3.8)$$

Inequalities (3.7) and (3.8) hold simultaneously for any $y \in A \cap B$. Then, by adding them, we deduce

$$\langle y - P_A(x), -2(1 - \beta)P_A(x) \rangle \leq 0, \quad \forall y \in A \cap B.$$

As $\beta < 1$, the factor $2(1 - \beta)$ is strictly positive and can be removed. Therefore,

$$\langle y - P_A(x), -P_A(x) \rangle \leq 0, \quad \forall y \in A \cap B.$$

By Proposition 1.2(ii), we conclude that $P_A(x) = P_{A \cap B}(0)$. □

In the next theorem we present a *constraint qualification* that characterizes the nonemptiness of the set of fixed points of the AAMR operator.

Theorem 3.9. *Let $A, B \subseteq \mathcal{H}$ be nonempty closed and convex sets, and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Then,*

$$\text{Fix} T_{\alpha, \beta} \neq \emptyset \quad \Leftrightarrow \quad A \cap B \neq \emptyset \quad \text{and} \quad -P_{A \cap B}(0) \in (N_A + N_B)(P_{A \cap B}(0)).$$

Proof. To prove the direct implication, pick any $x \in \text{Fix} T_{\alpha, \beta}$. Then, by Proposition 3.8, we have $A \cap B \neq \emptyset$ and

$$P_{A \cap B}(0) = P_A(x) = P_B(2\beta P_A(x) - x).$$

Thus, by Proposition 1.4, we deduce that $x - P_{A \cap B}(0) \in N_A(P_{A \cap B}(0))$ and

$$(2\beta - 1)P_{A \cap B}(0) - x = 2\beta P_A(x) - x - P_{A \cap B}(0) \in N_B(P_{A \cap B}(0)).$$

By taking $d_A := \frac{1}{2(1-\beta)}(x - P_{A \cap B}(0))$ and $d_B := \frac{1}{2(1-\beta)}((2\beta - 1)P_{A \cap B}(0) - x)$, we get

$$-P_{A \cap B}(0) = d_A + d_B,$$

with $d_A \in N_A(P_{A \cap B}(0))$ and $d_B \in N_B(P_{A \cap B}(0))$, as claimed.

To prove the converse implication, assume that $A \cap B \neq \emptyset$, and let $d_A \in N_A(P_{A \cap B}(0))$ and $d_B \in N_B(P_{A \cap B}(0))$ be such that

$$-P_{A \cap B}(0) = d_A + d_B. \quad (3.9)$$

Take

$$x := P_{A \cap B}(0) + 2(1 - \beta)d_A. \quad (3.10)$$

As $\beta < 1$, we have $2(1 - \beta)d_A \in N_A(P_{A \cap B}(0))$. Then, by Proposition 1.4, we get

$$P_A(x) = P_A(P_{A \cap B}(0) + 2(1 - \beta)d_A) = P_{A \cap B}(0). \quad (3.11)$$

Hence,

$$\begin{aligned} 2\beta P_A(x) - x &= 2\beta P_{A \cap B}(0) - P_{A \cap B}(0) - 2(1 - \beta)d_A \\ &= P_{A \cap B}(0) + 2(1 - \beta)(-P_{A \cap B}(0) - d_A). \end{aligned} \quad (3.12)$$

Now, by combining (3.9) and (3.12), we have

$$2\beta P_A(x) - x = P_{A \cap B}(0) + 2(1 - \beta)d_B. \quad (3.13)$$

Then, we use again Proposition 1.4 in (3.13) to obtain

$$P_B(2\beta P_A(x) - x) = P_{A \cap B}(0). \quad (3.14)$$

Finally, from (3.11) and (3.14), we deduce

$$P_A(x) = P_B(2\beta P_A(x) - x), \quad (3.15)$$

which implies $x \in \text{Fix } T_{\alpha, \beta}$, by (3.6). \square

The following corollary is a direct consequence of Theorem 3.9 and characterizes the nonemptiness of the set of fixed points of the AAMR operator defining our iterative method (3.2).

Corollary 3.10. *Let $A, B \subseteq \mathcal{H}$ be nonempty closed and convex sets, and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Then for any $z \in \mathcal{H}$,*

$$\text{Fix } T_{A-z, B-z, \alpha, \beta} \neq \emptyset \quad \Leftrightarrow \quad A \cap B \neq \emptyset \text{ and } z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z)).$$

Proof. According to Theorem 3.9, $\text{Fix } T_{A-z, B-z, \alpha, \beta} \neq \emptyset$ if and only if

$$(A - z) \cap (B - z) \neq \emptyset \quad \text{and} \quad -P_{(A-z) \cap (B-z)}(0) \in (N_{A-z} + N_{B-z})(P_{(A-z) \cap (B-z)}(0)).$$

Observe that $(A - z) \cap (B - z) = (A \cap B) - z$. Then $(A - z) \cap (B - z) \neq \emptyset$ if and only if $A \cap B \neq \emptyset$. Now, by Proposition 1.2(iv), we have

$$-P_{(A \cap B) - z}(0) = -P_{A \cap B}(z) + z,$$

and then

$$\begin{aligned} (N_{A-z} + N_{B-z})(P_{(A \cap B) - z}(0)) &= (N_A + N_B)(P_{(A \cap B) - z}(0) + z) \\ &= (N_A + N_B)(P_{A \cap B}(z)), \end{aligned}$$

which completes the proof. □

The next result shows that, when $P_A(0) = P_B(0)$, any point in the segment with end points $P_A(0)$ and $(2\beta - 1)P_A(0)$ is a fixed point of the mapping $T_{\alpha, \beta}$. Thus, if $P_A(0) = P_B(0) \neq 0$, the mapping $T_{\alpha, \beta}$ has multiple fixed points.

Proposition 3.11. *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex, and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. The following hold.*

- (i) *If $P_A(0) \in B$, then $(2\beta - 1)P_A(0) \in \text{Fix } T_{\alpha, \beta}$.*
- (ii) *If $P_B(0) \in A$, then $P_B(0) \in \text{Fix } T_{\alpha, \beta}$.*
- (iii) *If $P_A(0) = P_B(0)$, then*

$$(1 - 2\lambda(1 - \beta))P_A(0) \in \text{Fix } T_{\alpha, \beta} \quad \text{for all } \lambda \in [0, 1].$$

Proof. The assertion in (i) can be deduced from the second part of the proof of Theorem 3.9. Indeed, if $P_A(0) \in B$, then $P_{A \cap B}(0) = P_A(0)$. Since $-P_A(0) \in N_A(P_A(0))$, we may take $d_A := -P_A(0)$ and $d_B := 0$ and (3.9) holds. Thus, taking x as in (3.10), we have

$$x = P_A(0) - 2(1 - \beta)P_A(0) = (2\beta - 1)P_A(0),$$

which is a fixed point of $T_{\alpha, \beta}$ by (3.15). Item (ii) is analogous to the previous one. Finally, to prove (iii), use (i) and (ii) together with Proposition 1.12. \square

Next, we show some results regarding the range of the operator $\text{Id} - T_{A, B, \alpha, \beta}$, which will be useful later having in mind Theorem 1.15.

Lemma 3.12. *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets, and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. The following hold:*

$$(i) \quad x - T_{A, B, \alpha, \beta}(x) = 2\alpha\beta (P_A(x) - P_B(2\beta P_A(x) - x)), \quad \forall x \in \mathcal{H};$$

$$(ii) \quad \text{ran}(\text{Id} - T_{A, B, \alpha, \beta}) = \text{ran}(\text{Id} - T_{A+z, B+(2\beta-1)z, \alpha, \beta}) + 4\alpha\beta(\beta - 1)z, \quad \forall z \in \mathcal{H}.$$

Proof. Assertion (i) is straightforward from the definition of $T_{A, B, \alpha, \beta}$. Indeed, for any $x \in \mathcal{H}$ we have

$$\begin{aligned} x - T_{A, B, \alpha, \beta}(x) &= x - (1 - \alpha)x - \alpha(2\beta P_B - \text{Id})(2\beta P_A - \text{Id})(x) \\ &= \alpha(x - (2\beta P_B(2\beta P_A(x) - x) + 2\beta P_A(x) - x)) \\ &= 2\alpha\beta (P_A(x) - P_B(2\beta P_A(x) - x)). \end{aligned}$$

To prove (ii), pick any $x, z \in \mathcal{H}$. By using the translation formula for projections given in Proposition 1.2(iv), we obtain

$$\begin{aligned} &P_A(x) - P_B(2\beta P_A(x) - x) \\ &= P_{A+z}(x+z) - z - P_B(2\beta P_{A+z}(x+z) - 2\beta z - x) \\ &= P_{A+z}(x+z) - z - P_{B+(2\beta-1)z}(2\beta P_{A+z}(x+z) - x - z) + (2\beta - 1)z \\ &= P_{A+z}(x+z) - P_{B+(2\beta-1)z}(2\beta P_{A+z}(x+z) - (x+z)) + 2(\beta - 1)z. \end{aligned}$$

Therefore, by assertion (i), we get

$$(\text{Id} - T_{A, B, \alpha, \beta})(x) = (\text{Id} - T_{A+z, B+(2\beta-1)z, \alpha, \beta})(x+z) + 4\alpha\beta(\beta - 1)z, \quad \forall z \in \mathcal{H},$$

and we are done. \square

Theorem 3.13. *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets, and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Suppose that one of the following holds:*

- (i) \mathcal{H} is finite-dimensional;
- (ii) $\text{int } A \neq \emptyset$ or $\text{int } B \neq \emptyset$.

Then the unique element of minimum norm in $\overline{\text{ran}}(\text{Id} - T_{\alpha, \beta})$ is $2\alpha\beta v$, where $v := P_{\overline{A-B}}(0)$.

Proof. Let w be the unique element of minimum norm in $\overline{\text{ran}}(\text{Id} - T_{\alpha, \beta})$. By Lemma 3.12(i), we have

$$\text{ran}(\text{Id} - T_{\alpha, \beta}) \subseteq 2\alpha\beta(A - B),$$

which implies that $w \in (2\alpha\beta)\overline{A - B}$.

Suppose that (i) holds. Pick any $a \in \text{ri } A$, $b \in \text{ri } B$ and set $z_{a,b} := \frac{a-b}{2(\beta-1)}$. Then, by Lemma 3.12(ii), we have

$$\text{ran}(\text{Id} - T_{A, B, \alpha, \beta}) = \text{ran}(\text{Id} - T_{A+z_{a,b}, B+(2\beta-1)z_{a,b}, \alpha, \beta}) + 2\alpha\beta(a - b), \quad (3.16)$$

with

$$\begin{aligned} b + (2\beta - 1)z_{a,b} &= b + \frac{2\beta - 1}{2(\beta - 1)}(a - b) = b + \left(1 + \frac{1}{2(\beta - 1)}\right)(a - b) \\ &= a + \frac{a - b}{2(\beta - 1)} = a + z_{a,b}; \end{aligned}$$

i.e., we have obtained that

$$a + z_{a,b} = b + (2\beta - 1)z_{a,b} \in \text{ri}(A + z_{a,b}) \cap \text{ri}(B + (2\beta - 1)z_{a,b}).$$

Hence, the mapping $T_{A+z_{a,b}, B+(2\beta-1)z_{a,b}, \alpha, \beta}$ has a fixed point according to Proposition 1.6(ii) and Theorem 3.9, and therefore we deduce that 0 is the unique element of minimum norm in $\text{ran}(\text{Id} - T_{A+z_{a,b}, B+(2\beta-1)z_{a,b}, \alpha, \beta})$. Then, by (3.16), we get

$$\begin{aligned} \|w\| &= \inf \{ \|u\| : u \in \overline{\text{ran}}(\text{Id} - T_{A+z_{a,b}, B+(2\beta-1)z_{a,b}, \alpha, \beta}) + 2\alpha\beta(a - b) \} \\ &\leq 2\alpha\beta\|a - b\| + \inf \{ \|u\| : u \in \overline{\text{ran}}(\text{Id} - T_{A+z_{a,b}, B+(2\beta-1)z_{a,b}, \alpha, \beta}) \} \\ &= 2\alpha\beta\|a - b\|, \end{aligned} \quad (3.17)$$

and this holds for every $a \in \text{ri } A$ and every $b \in \text{ri } B$.

Now, choose any $a \in A, b \in B$. Then, there exist a pair of sequences $\{a_k\} \subset \text{ri } A, \{b_k\} \subset \text{ri } B$ such that $a_k \rightarrow a$ and $b_k \rightarrow b$, and by (3.17), we get

$$2\alpha\beta\|a - b\| = 2\alpha\beta \left\| \lim_{k \rightarrow \infty} (a_k - b_k) \right\| = 2\alpha\beta \lim_{k \rightarrow \infty} \|a_k - b_k\| \geq \|w\|.$$

Thus, since $(2\alpha\beta)^{-1}w \in \overline{A - B}$ and

$$\|(2\alpha\beta)^{-1}w\| \leq \|a - b\|, \quad \text{for all } a \in A, b \in B;$$

it must be that $(2\alpha\beta)^{-1}w = P_{\overline{A-B}}(0)$, which proves the result assuming (i).

To prove the result when (ii) holds, if $\text{int } A \neq \emptyset$ ($\text{int } B \neq \emptyset$), then take any $a \in \text{int } A$ and $b \in B$ ($a \in A$ and $b \in \text{int } B$) and repeat the idea of the proof of the previous case but using Proposition 1.6(i) instead of Proposition 1.6(ii). \square

Next we present some translation formulas for the AAMR operator in the special case when both sets are closed affine subspaces.

Proposition 3.14. *Let $U, V \subseteq \mathcal{H}$ be closed affine subspaces with nonempty intersection. Let $y \in U \cap V$ and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Then, for any $x \in \mathcal{H}$,*

$$T_{U,V,\alpha,\beta}(x) = T_{U-y,V-y,\alpha,\beta}(x) + T_{U,V,\alpha,\beta}(0), \quad (3.18)$$

and

$$T_{U,V,\alpha,\beta}(x) = T_{U-y,V-y,\alpha,\beta}(x - x^*) + x^*, \quad \forall x^* \in \text{Fix } T_{U,V,\alpha,\beta}. \quad (3.19)$$

Furthermore, one has

$$\text{Fix } T_{U,V,\alpha,\beta} = x^* + \text{Fix } T_{U-y,V-y,\alpha,\beta}, \quad \forall x^* \in \text{Fix } T_{U,V,\alpha,\beta}. \quad (3.20)$$

Proof. Because $U - y$ and $V - y$ are closed linear subspaces of \mathcal{H} , then P_{U-y} and P_{V-y} are linear mappings by Proposition 1.11. Denote the modified reflector operator onto any set $C \subseteq \mathcal{H}$ by $Q_{\beta,C} := 2\beta P_C - \text{Id}$. Then, the mappings $Q_{\beta,U-y}$ and $Q_{\beta,V-y}$ are also linear. Further, for any $x \in \mathcal{H}$, by Proposition 1.2(iv), we have

$$\begin{aligned} Q_{\beta,U}(x) &= 2\beta P_U(x) - x = 2\beta P_{U-y}(x - y) + 2\beta y - x \\ &= 2\beta P_{U-y}(x) - x + 2\beta(P_{U-y}(-y) + y) \\ &= Q_{\beta,U-y}(x) + 2\beta P_U(0) = Q_{\beta,U-y}(x) + Q_{\beta,U}(0). \end{aligned}$$

Similarly, we get $Q_{\beta,V}(x) = Q_{\beta,V-y}(x) + Q_{\beta,V}(0)$. Combining these equalities together and using the linearity of $Q_{\beta,V-y}$, we obtain

$$\begin{aligned} Q_{\beta,V}Q_{\beta,U}(x) &= Q_{\beta,V}(Q_{\beta,U-y}(x) + Q_{\beta,U}(0)) \\ &= Q_{\beta,V-y}Q_{\beta,U-y}(x) + Q_{\beta,V-y}Q_{\beta,U}(0) + Q_{\beta,V}(0) \\ &= Q_{\beta,V-y}Q_{\beta,U-y}(x) + Q_{\beta,V}Q_{\beta,U}(0), \end{aligned}$$

which implies (3.18).

Now take any $x^* \in \text{Fix } T_{U,V,\alpha,\beta}$. Then, by (3.18), we have

$$x^* = T_{U,V,\alpha,\beta}(x^*) = T_{U-y,V-y,\alpha,\beta}(x^*) + T_{U,V,\alpha,\beta}(0).$$

By replacing $T_{U,V,\alpha,\beta}(0) = -T_{U-y,V-y,\alpha,\beta}(x^*) + x^*$ in (3.18) and using the linearity of the operator $T_{U-y,V-y,\alpha,\beta}$, we obtain (3.19).

The last assertion easily follows from (3.19). Indeed, for any $x^* \in \text{Fix } T_{U,V,\alpha,\beta}$, one has

$$\begin{aligned} w^* \in \text{Fix } T_{U,V,\alpha,\beta} &\Leftrightarrow T_{U,V,\alpha,\beta}(w^*) = w^* \Leftrightarrow T_{U-y,V-y,\alpha,\beta}(w^* - x^*) + x^* = w^* \\ &\Leftrightarrow w^* - x^* \in \text{Fix } T_{U-y,V-y,\alpha,\beta}, \end{aligned}$$

which implies (3.20). \square

We conclude this section by providing the following characterization of the set of fixed points of the AAMR operator for closed linear subspaces.

Proposition 3.15. *Let $U, V \subseteq \mathcal{H}$ be closed subspaces and let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Then*

$$\text{Fix } T_{U,V,\alpha,\beta} = U^\perp \cap V^\perp.$$

Proof. Observe that by (3.6) and Proposition 3.8, it holds that

$$x \in \text{Fix } T_{U,V,\alpha,\beta} \Leftrightarrow P_V(2\beta P_U(x) - x) = P_U(x) = P_{U \cap V}(0) = 0.$$

Therefore, $P_U(x) = P_V(x) = 0$, which implies $x \in U^\perp \cap V^\perp$.

To prove the converse implication, pick any $x \in U^\perp \cap V^\perp$. Then $P_U(x) = P_V(x) = 0$, so we trivially have that

$$P_V(2\beta P_U(x) - x) = P_U(x),$$

and thus $x \in \text{Fix } T_{U,V,\alpha,\beta}$ by (3.6). \square

3.1.2 Iterative scheme for finding the closest point in the intersection

In the main result of this section we show that the iterative method defined by the AAMR operator in (3.3) is weakly convergent to a fixed point of the operators, and the *shadow sequence* $(P_A(z + x_k))_{k=0}^{\infty}$ is strongly convergent to the solution of problem (3.1). To proceed, we first provide the following lemma which contains a finer version of the Krasnosel'skiĭ–Mann algorithm in Theorem 1.13, for a Douglas–Rachford-like iteration.

Lemma 3.16. *Let $T_1, T_2 : \mathcal{H} \mapsto \mathcal{H}$ be firmly nonexpansive operators and let $(\lambda_k)_{k=0}^{\infty}$ be a sequence in $[0, 1]$. Consider $T := (2T_2 - \text{Id})(2T_1 - \text{Id})$ and suppose $\text{Fix } T \neq \emptyset$. Given any $x_0 \in \mathcal{H}$, set*

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k T(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Then the following hold.

(a) *If $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$, then*

(i) *$(x_{k+1} - x_k)_{k=0}^{\infty}$ converges strongly to 0;*

(ii) *$(x_k)_{k=0}^{\infty}$ converges weakly to a point $x^* \in \text{Fix } T$.*

(b) *Assume that T_1 is μ -cocoercive for some $\mu > 1$, and suppose that either*

$$\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty \quad \text{or} \quad \lambda_k = 1, \quad \text{for all } k = 0, 1, 2, \dots$$

Then $(T_1(x_k))_{k=0}^{\infty}$ converges strongly to the unique point in $T_1(\text{Fix } T)$.

Proof. Since T_1 and T_2 are firmly nonexpansive, by Proposition 1.23, there exist two maximally monotone operators $A_1, A_2 : \mathcal{H} \rightrightarrows \mathcal{H}$ such that

$$T_i = J_{A_i}, \quad \text{for } i = 1, 2.$$

By assumption and Proposition 1.28(ii), we have that

$$\text{zer}(A_1 + A_2) = J_{A_1}(\text{Fix}((2J_{A_2} - \text{Id})(2J_{A_1} - \text{Id}))) = T_1(\text{Fix } T) \neq \emptyset.$$

Then, (a) follows from applying Theorem 2.14(i)–(ii) to A_1 and A_2 .

Assume now that T_1 is μ -cocoercive for some $\mu > 1$. Then, by Proposition 1.24, we know that A_1 is $(\mu - 1)$ -strongly monotone. Hence, assertion (b) is a direct consequence of Theorem 2.14(iv), when $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$; or Theorem 2.15, when $\lambda_k = 1$. \square

Theorem 3.17 (AAMR iterative scheme). *Let $A, B \subseteq \mathcal{H}$ be nonempty closed and convex sets. Fix any $\alpha \in]0, 1]$ and any $\beta \in]0, 1[$. Given $z \in \mathcal{H}$, choose any $x_0 \in \mathcal{H}$ and consider the sequence defined by*

$$x_{k+1} = T_{A-z, B-z, \alpha, \beta}(x_k), \quad \text{for } k = 0, 1, 2, \dots \quad (3.21)$$

If $A \cap B \neq \emptyset$ and $z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z))$, then the following assertions hold.

(a) *If $\alpha < 1$, then*

(i) *$(x_{k+1} - x_k)_{k=0}^\infty$ is strongly convergent to 0;*

(ii) *$(x_k)_{k=0}^\infty$ is weakly convergent to a point $x^* \in \text{Fix } T_{A-z, B-z, \alpha, \beta}$ such that*

$$P_A(z + x^*) = P_{A \cap B}(z). \quad (3.22)$$

(b) *The shadow sequence $(P_A(z + x_k))_{k=0}^\infty$ is strongly convergent to $P_{A \cap B}(z)$.*

Otherwise, if $\alpha < 1$ then $\|x_k\| \rightarrow \infty$.

Proof. First note that the projector operators P_{A-z} and P_{B-z} are firmly nonexpansive by Proposition 1.11. Then, as $\beta \in]0, 1[$, the operators

$$T_1 := \beta P_{A-z} \quad \text{and} \quad T_2 := \beta P_{B-z}$$

are also firmly nonexpansive, and moreover $\frac{1}{\beta}$ -cocoercive (with $\frac{1}{\beta} > 1$). Observe that

$$T_{A-z, B-z, \alpha, \beta} = (1 - \alpha) \text{Id} + \alpha(2T_2 - \text{Id})(2T_1 - \text{Id}),$$

and then by Corollary 3.10, we get that

$$\text{Fix}((2T_2 - \text{Id})(2T_1 - \text{Id})) = \text{Fix } T_{A-z, B-z, \alpha, \beta} \neq \emptyset.$$

Hence, we can use Lemma 3.16(a) with $\lambda_k := \alpha$ to show that the operator $T_{A-z, B-z, \alpha, \beta}$ has a fixed point x^* such that the sequence defined by (3.21) satisfies

$$x_k \rightharpoonup x^* \quad \text{and} \quad x_{k+1} - x_k \rightarrow 0, \quad \text{provided that } \alpha < 1;$$

and

$$\beta P_{A-z}(x_k) \rightarrow \beta P_{A-z}(x^*). \quad (3.23)$$

Moreover, by Proposition 3.8, we have

$$P_{A-q}(x^*) = P_{(A-q) \cap (B-q)}(0) = P_{A \cap B-z}(0), \quad (3.24)$$

and, according to Proposition 1.2(iv), it holds that

$$P_{A \cap B-z}(0) = P_{A \cap B}(z) - z \quad \text{and} \quad P_{A-z}(x) = P_A(x+z) - z, \quad \forall x \in \mathcal{H}. \quad (3.25)$$

Thus, in view of (3.25), we get from (3.24) that $P_A(z+x^*) = P_{A \cap B}(z)$, and then (3.23) implies that $P_A(z+x_k) \rightarrow P_{A \cap B}(z)$. This concludes the proof of statements (a) and (b).

The remaining case easily follows from Theorem 1.15(ii) together with the facts that $\text{Fix } T_{A-z, B-z, \alpha, \beta} = \emptyset$, by Corollary 3.10, and that $T_{A-z, B-z, \alpha, \beta}$ is α -averaged for $\alpha \in]0, 1[$, according to Proposition 3.6. \square

In Figure 3.3 we illustrate how the AAMR iterative scheme permits to solve a best approximation problem according to the assertion in Theorem 3.17(a)(ii).

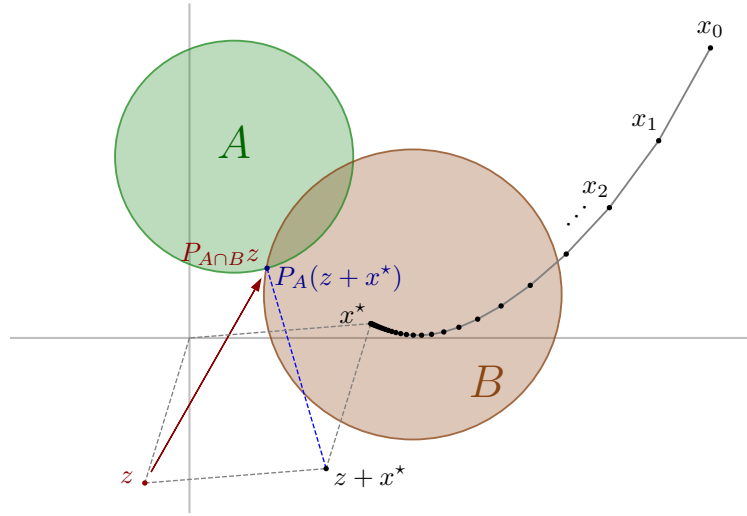


FIGURE 3.3: Illustration of the AAMR iterative scheme for two sets.

Observe that, once the limit point x^* is reached, one needs to compute $P_A(z+x^*)$ in order to solve the problem. Hence, as it happens with Douglas–Rachford (see Remark 2.10), the sequence of interest is the one of the shadows $(P_A(z+x_k))_{k=0}^{\infty}$. In fact, the shadow sequence for the AAMR is strongly convergent to the solution point, by Theorem 3.17(b). The dynamics of the AAMR iteration in the three possible scenarios covered by Theorem 3.17 is illustrated in Figure 3.4, where for the sake of clarity we chose $z = 0$.

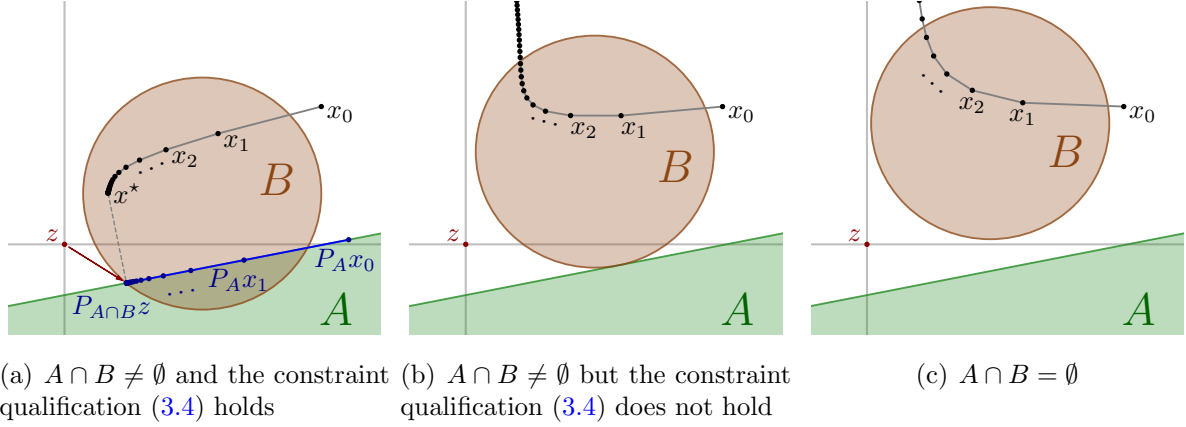


FIGURE 3.4: Behavior of the AAMR algorithm in three possible scenarios.

REMARK 3.18 (Finer versions of the algorithm).

- (i) It is straightforward to construct a version of the AAMR algorithm where the value of the parameter α may vary across the iterations (see Corollary 3.41). Specifically, Theorem 3.17(a)–(b) still holds if one replaces (3.21) by the iterative method

$$x_{k+1} = (1 - \alpha_k)x_k + \alpha_k(2\beta P_{B-z} - \text{Id})(2\beta P_{A-z} - \text{Id})(x_k), \quad k = 0, 1, 2, \dots,$$

where $\{\alpha_k\}_{n=0}^\infty \subset [0, 1]$ satisfies the appropriate conditions in Lemma 3.16.

- (ii) Similarly, it is easy to include errors in the AAMR scheme, by using in the proof of Theorem 3.17, a version of Lemma 3.16 derived from a refined Douglas–Rachford splitting algorithm given in [77, Theorem 2.1].

Observe that, unlike for the shadow sequence, only weak convergence is asserted for $(x_k)_{k=0}^\infty$ in Theorem 3.17. As we show next, it becomes strong for closed affine subspaces.

Theorem 3.19 (Strong convergence of AAMR for affine subspaces). *Let U and V be closed affine subspaces of \mathcal{H} with nonempty intersection. Let $\alpha, \beta \in]0, 1[$ and let $z \in \mathcal{H}$ such that $z - P_{U \cap V}(z) \in (U - U)^\perp + (V - V)^\perp$. Then, for any $x_0 \in \mathcal{H}$,*

$$T_{U-z, V-z, \alpha, \beta}^k(x_0) \rightarrow P_{\text{Fix} T_{U-z, V-z, \alpha, \beta}}(x_0).$$

Proof. Since U is a closed affine subspace, by Proposition 1.32(i), we have

$$N_U(x) = N_{U-x}(0) = (U - x)^\perp = (U - U)^\perp, \quad \text{for all } x \in U.$$

Likewise, $N_V(x) = (V - V)^\perp$ for all $x \in V$. Therefore, we have

$$z - P_{U \cap V}(z) \in (N_U + N_V)(P_{U \cap V}(z)),$$

and this implies, according to Corollary 3.10, that $\text{Fix } T_{U-z, V-z, \alpha, \beta} \neq \emptyset$. Thus, taking any $y \in U \cap V$ and any $x^* \in \text{Fix } T_{U-z, V-z, \alpha, \beta}$, since $y - z \in (U - z) \cap (V - z)$, we can apply Proposition 3.14 recursively to get

$$T_{U-z, V-z, \alpha, \beta}^k(x_0) = T_{U-y, V-y, \alpha, \beta}^k(x_0 - x^*) + x^*,$$

and, moreover, $\text{Fix } T_{U-y, V-y, \alpha, \beta} \neq \emptyset$. Thus, using Theorem 3.17(a)(i), we obtain that

$$T_{U-y, V-y, \alpha, \beta}^{k+1}(x_0 - x^*) - T_{U-y, V-y, \alpha, \beta}^k(x_0 - x^*) \rightarrow 0.$$

Since $T_{U-y, V-y, \alpha, \beta}$ is linear, by Proposition 1.14, we deduce

$$T_{U-y, V-y, \alpha, \beta}^k(x_0 - x^*) \rightarrow P_{\text{Fix } T_{U-y, V-y, \alpha, \beta}}(x_0 - x^*).$$

Consequently,

$$T_{U-z, V-z, \alpha, \beta}^k(x_0) \rightarrow P_{\text{Fix } T_{U-y, V-y, \alpha, \beta}}(x_0 - x^*) + x^* = P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0),$$

where the last equality holds by Proposition 1.2(iv) and Proposition 3.14. \square

REMARK 3.20 (On the constraint qualification). If any of the conditions given in Proposition 1.6 (or in Proposition 1.33 for affine subspaces) holds, then the *constraint qualification*

$$z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z))$$

holds at every point $z \in \mathcal{H}$, according to Proposition 1.4, and thus the convergence of the sequence defined by (3.21) is guaranteed.

In [84, Theorem 3.2] and [81, Theorem 10.13], the existence of a point x^* satisfying (3.22) for every $z \in \mathcal{H}$ is proved to be equivalent to the strong CHIP, for the particular case where $B = L^{-1}(b)$, for some bounded linear operator L from \mathcal{H} into a finite-dimensional Hilbert space Y and $b \in Y$. In addition, Deutsch and Ward proposed in [84] a steepest descent method with line search for finding the point x^* , whose linear convergence is proved under the additional assumption that C is polyhedral.

If the sets A and B , with nonempty intersection, have the strong CHIP at the point $P_{A \cap B}(z)$, then trivially $z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z))$ and the scheme converges. However, in the following example we show that the strong CHIP is not a necessary condition for the AAMR method to converge for a particular point $z \in \mathcal{H}$.

Example 3.21. Consider the Hilbert space $\mathcal{H} = \mathbb{R}^2$ and define the sets

$$A := (1, 1) + \mathbb{B}(0, 1) \quad \text{and} \quad B := (-1, 1) + \mathbb{B}(0, 1).$$

The pair $\{A, B\}$ does not have strong CHIP at any point, since $A \cap B = \{(0, 1)\}$ and

$$(N_A + N_B)((0, 1)) = \mathbb{R} \times \{0\} \neq \mathbb{R}^2 = N_{A \cap B}((0, 1)).$$

If we take any $z \in \mathbb{R} \times \{1\}$, then

$$z - P_{A \cap B}(z) = z - (0, 1) \in \mathbb{R} \times \{0\} = (N_A + N_B)(P_{A \cap B}(z)),$$

and by Theorem 3.17, the AAMR method defined by (3.21) will generate a sequence that converges to a point x^* such that $P_A(x^* + z) = \{(0, 1)\}$. On the other hand, if $z \notin \mathbb{R} \times \{1\}$, then

$$z - P_{A \cap B}(z) \notin (N_A + N_B)(P_{A \cap B}(z)),$$

and the sequence $(x_k)_{k=0}^{\infty}$ generated by (3.21) will be unbounded, i.e., $\|x_k\| \rightarrow \infty$. The two possibilities are illustrated in Figure 3.5.

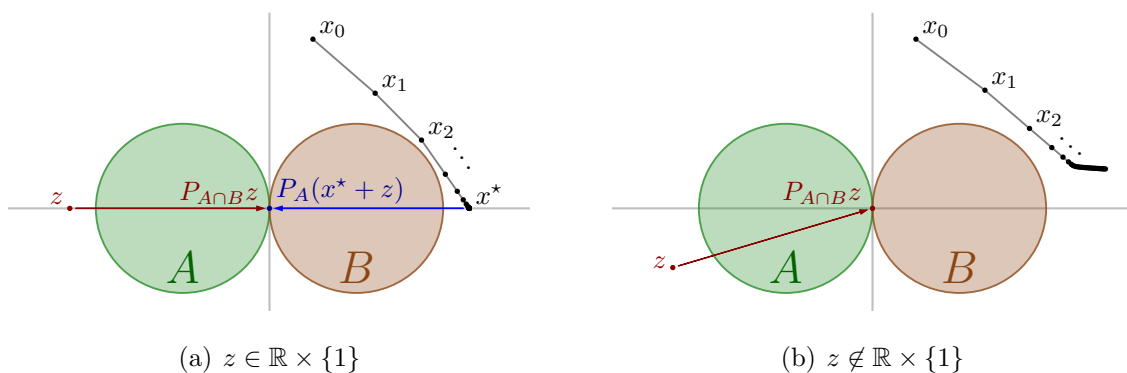


FIGURE 3.5: Illustration of Example 3.21.

We have seen that even when the strong CHIP does not hold, the method can converge for a particular point $z \in \mathcal{H}$. However, the following result establishes that if we want the method to converge for every point in \mathcal{H} , the strong CHIP will have to be required.

Proposition 3.22. *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets with nonempty intersection. Then the following assertions are equivalent:*

(i) $\{A, B\}$ has the strong CHIP;

(ii) for all $z \in \mathcal{H}$,

$$z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z)).$$

Proof. Assume (ii). Take $x \in A \cap B$ and let $y \in N_{A \cap B}(x)$. By Proposition 1.4,

$$P_{A \cap B}(x + y) = x.$$

Hence, by (ii), we have

$$y = x + y - P_{A \cap B}(x + y) \in (N_A + N_B)(P_{A \cap B}(x + y)) = (N_A + N_B)(x).$$

Therefore $N_{A \cap B}(x) \subseteq (N_A + N_B)(x)$. Since x is arbitrary in $A \cap B$ and the reverse inclusion always holds, then $\{A, B\}$ has the strong CHIP, which proves that (ii) \Rightarrow (i). The opposite implication clearly holds by Proposition 1.4. \square

We finish this section with the following result which, under some conditions, provides the asymptotic behavior of the AAMR iteration for those situations where it does not converge.

Corollary 3.23 (Asymptotic behavior of the AAMR iteration). *Let $A, B \subseteq \mathcal{H}$ be nonempty closed and convex, and define $v := P_{\overline{A-B}}(0)$. Let $\alpha, \beta \in]0, 1[$ and let $z \in \mathcal{H}$. For any $x_0 \in \mathcal{H}$, consider the iterated sequence defined by*

$$x_{k+1} = T_{A-z, B-z, \alpha, \beta}(x_k), \quad \text{for } k = 0, 1, 2, \dots$$

Suppose that one of the following holds:

(i) \mathcal{H} is finite-dimensional;

(ii) $\text{int } A \neq \emptyset$ or $\text{int } B \neq \emptyset$.

Then, the sequence $(x_k - x_{k+1})_{k=0}^{\infty}$ converges in norm to $2\alpha\beta v$.

Proof. Since $T_{A-z, B-z, \alpha, \beta}$ is α -averaged according to Proposition 3.6, the result follows from applying Theorem 1.15(i) together with Theorem 3.13. \square

REMARK 3.24. Observe that if the problem is consistent but the required constraint qualification does not hold, i.e.,

$$A \cap B \neq \emptyset \quad \text{but} \quad z - P_{A \cap B}(z) \notin (N_A + N_B)(P_{A \cap B}(z)),$$

then Theorem 3.17(a)–(b) cannot be applied. Nonetheless, Corollary 3.23 asserts that if any of conditions (i)–(ii) therein is satisfied, then Theorem 3.17(a)(i) remains valid. This can be observed in Figures 3.4(b) and 3.5(b). Furthermore, if the sets are disjoint, then the difference sequence allows us to measure the gap between them, since

$$\|x_k - x_{k+1}\| \rightarrow 2\alpha\beta d(A, B).$$

The latter scenario is illustrated in Figure 3.6, where the AAMR algorithm is applied to two disjoint balls.

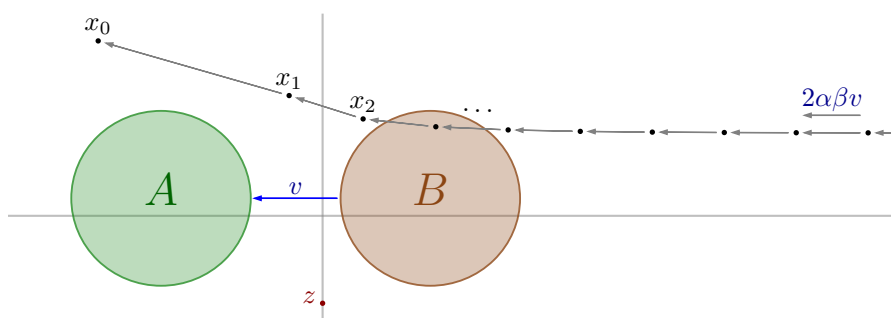


FIGURE 3.6: Asymptotic behavior of the AAMR iteration.

3.1.3 Finitely many sets

In this section we show how to apply the AAMR method to best approximation problems defined by an arbitrary finite number of sets, as in (2.2). Therefore, given r nonempty, closed and convex sets $C_1, \dots, C_r \subset \mathcal{H}$, and given any $z \in \mathcal{H}$, we are interested in solving the problem

$$\text{Find } w = P_{\bigcap_{i=1}^r C_i}(z).$$

In order to derive a version of the AAMR algorithm for finitely many sets, we turn to the product space trick described in Section 2.1.1. To proceed, we need first to present the following characterization, which permits to rewrite the constraint qualification (3.4) in the product space.

Lemma 3.25. *Let $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$ be nonempty closed and convex sets. Let \mathcal{H} be the product Hilbert space as in (2.3) and consider the sets \mathbf{C} and \mathbf{D} defined in (2.4). For every $\mathbf{x} = \mathbf{j}(x) \in \mathbf{C} \cap \mathbf{D}$, it holds that*

$$(N_{\mathbf{C}}(\mathbf{x}) + N_{\mathbf{D}}(\mathbf{x})) \cap \mathbf{D} = \mathbf{j} \left(\sum_{i=1}^r N_{C_i}(x) \right),$$

where $\mathbf{j}(x) := (x, x, \dots, x) \in \mathbf{D}$ for all $x \in \mathcal{H}$.

Proof. Let $\mathbf{x} = \mathbf{j}(x) \in \mathbf{C} \cap \mathbf{D}$. First note that, since \mathbf{C} is the product of r sets, then

$$N_{\mathbf{C}}(\mathbf{x}) = N_{C_1}(x) \times N_{C_2}(x) \times \dots \times N_{C_r}(x);$$

and moreover, by Proposition 2.16(ii), we know that

$$N_{\mathbf{D}}(\mathbf{x}) = \mathbf{D}^\perp = \left\{ \mathbf{u} = (u_1, \dots, u_r) \in \mathcal{H} : \sum_{i=1}^r u_i = 0 \right\}.$$

To prove the direct inclusion, pick any $\mathbf{y} \in (N_{\mathbf{C}}(\mathbf{x}) + N_{\mathbf{D}}(\mathbf{x})) \cap \mathbf{D}$. Then $\mathbf{y} = \mathbf{j}(y)$ for some $y \in \mathcal{H}$ verifying

$$y \in N_{C_i}(x) + u_i, \quad \text{for } i = 1, \dots, r,$$

with $\sum_{i=1}^r u_i = 0$. Thus, as $N_{C_i}(x)$ are all cones, we have $y \in \sum_{i=1}^r N_{C_i}(x)$, which yields

$$\mathbf{y} \in \mathbf{j} \left(\sum_{i=1}^r N_{C_i}(x) \right).$$

To prove the reverse inclusion, pick $\mathbf{y} = \mathbf{j}(y)$ for any $y \in \sum_{i=1}^r N_{C_i}(x)$. Then, there exists $d_i \in N_{C_i}(x)$, for each $i = 1, \dots, r$, such that $y = \sum_{i=1}^r d_i$. Define

$$\mathbf{d} := (d_1, d_2, \dots, d_r) \in N_{\mathbf{C}}(\mathbf{x}) \quad \text{and} \quad \mathbf{u} := (u_1, u_2, \dots, u_r) \in \mathcal{H},$$

where $u_i := y - rd_i$, for each $i = 1, \dots, r$. Since

$$\sum_{i=1}^r u_i = \sum_{i=1}^r (y - rd_i) = ry - r \sum_{i=1}^r d_i = 0,$$

we have $\mathbf{u} \in \mathbf{D}^\perp$. Hence, $\mathbf{y} = r\mathbf{d} + \mathbf{u} \in N_{\mathbf{C}}(\mathbf{x}) + N_{\mathbf{D}}(\mathbf{x})$, and the result is proved. \square

We are now ready to prove the convergence of the AAMR method for finitely many sets.

Theorem 3.26 (AAMR scheme for finitely many sets). *Let $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$ be nonempty closed and convex sets. Let $z \in \mathcal{H}$ and fix any $\alpha, \beta \in]0, 1[$. Given r arbitrary starting points $x_{1,0}, x_{2,0}, \dots, x_{r,0} \in \mathcal{H}$, set*

$$\begin{aligned} & \text{for } k = 0, 1, 2, \dots : \\ & \left[\begin{array}{l} p_k = \frac{1}{r} \sum_{i=1}^r x_{i,k}, \\ \text{for } i = 1, 2, \dots, r : \\ \quad \left[\begin{array}{l} x_{i,k+1} = (1 - \alpha)x_{i,k} + \alpha(2\beta P_{C_i - z} - \text{Id})(2\beta p_k - x_{i,k}). \end{array} \right. \end{array} \right. \end{aligned} \quad (3.26)$$

If $\bigcap_{i=1}^r C_i \neq \emptyset$ and $z - P_{\bigcap_{i=1}^r C_i}(z) \in \sum_{i=1}^r N_{C_i}(P_{\bigcap_{i=1}^r C_i}(z))$, then the following hold:

(i) $(x_{i,k})_{k=0}^\infty$ is weakly convergent to a point x_i^* , for each $i = 1, 2, \dots, r$, and

$$z + \frac{1}{r} \sum_{i=1}^r x_i^* = P_{\bigcap_{i=1}^r C_i}(z);$$

(ii) the sequence $(z + p_k)_{k=0}^\infty$ is strongly convergent to $P_{\bigcap_{i=1}^r C_i}(z)$.

Otherwise, at least one of the sequences $(x_{1,k})_{k=0}^\infty, (x_{2,k})_{k=0}^\infty, \dots, (x_{r,k})_{k=0}^\infty$ is unbounded.

Proof. Let \mathcal{H} be the product Hilbert space as in (2.3) and consider the sets \mathbf{C} and \mathbf{D} defined in (2.4). Define $\mathbf{z} := \mathbf{j}(z)$ and for each $k = 0, 1, 2, \dots$, set

$$\mathbf{x}_k := (x_{1,k}, \dots, x_{r,k}) \quad \text{and} \quad \mathbf{p}_k := \mathbf{j}(p_k).$$

In view of the expressions of $P_{\mathbf{C}}$ and $P_{\mathbf{D}}$ in Proposition 2.3, the iterative scheme in (3.26) reduces to

$$\mathbf{x}_{k+1} = T_{\mathbf{D}, \mathbf{C} - \mathbf{z}, \alpha, \beta}(\mathbf{x}_k), \quad k = 0, 1, 2, \dots \quad (3.27)$$

Observe that $\mathbf{D} - \mathbf{z} = \mathbf{D}$, since \mathbf{D} is a subspace containing \mathbf{z} . Therefore, the operator defining the iteration (3.27) is simply $T_{\mathbf{D} - \mathbf{z}, \mathbf{C} - \mathbf{z}, \alpha, \beta}$. Observe also that $\bigcap_{i=1}^r C_i \neq \emptyset$ if and only if $\mathbf{C} \cap \mathbf{D} \neq \emptyset$ by (2.5). Moreover, $\mathbf{j}(P_{\bigcap_{i=1}^r C_i}(z)) = P_{\mathbf{C} \cap \mathbf{D}}(\mathbf{z})$ by (2.6), and then by Lemma 3.25, we have

$$\mathbf{z} - P_{\mathbf{C} \cap \mathbf{D}}(\mathbf{z}) \in N_{\mathbf{C}} + N_{\mathbf{D}}(P_{\mathbf{C} \cap \mathbf{D}}(\mathbf{z})) \quad \Leftrightarrow \quad \mathbf{z} - P_{\bigcap_{i=1}^r C_i}(z) \in \sum_{i=1}^r N_{C_i}(P_{\bigcap_{i=1}^r C_i}(z)).$$

The result thus follows from Theorem 3.17. \square

The iteration generated by (3.26) in Theorem 3.26 is illustrated in Figure 3.7.

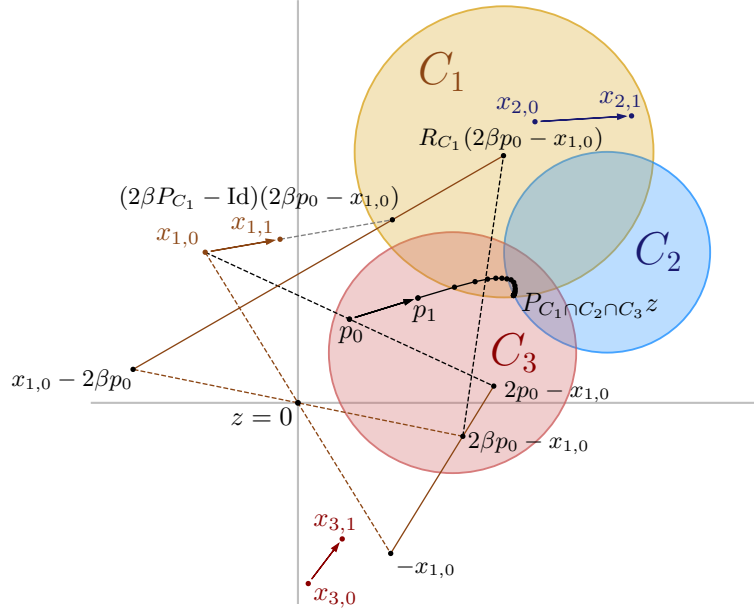


FIGURE 3.7: Illustration of the AAMR iterative scheme for many sets in Theorem 3.26.

REMARK 3.27. Some comments concerning Theorem 3.26 are given next.

- (i) For simplicity, we have assumed that $\alpha \in]0, 1[$. Nonetheless, Theorem 3.26(ii) still holds if we set $\alpha = 1$ in (3.26), according to Theorem 3.17(b).
- (ii) If the sets C_1, C_2, \dots, C_r are closed affine subspaces of \mathcal{H} , then we can use Theorem 3.19 instead of Theorem 3.17 in the proof, and therefore the convergence in Theorem 3.26(i) becomes strong.
- (iii) The order of action of the projections onto D and C in (3.27) makes the shadow sequence $P_D(z + x_k)$ to lay in the diagonal. In this way, it can be identified with the sequence in the original space $(z + p_k)_{k=0}^{\infty} \subset \mathcal{H}$, which can be monitored.
- (iv) Thanks to Lemma 3.25, it is straightforward to prove an analogous result to Proposition 3.22, showing that the strong CHIP of $\{C_1, \dots, C_r\}$ characterizes the convergence of the iterative method (3.26) for every point $z \in \mathcal{H}$.

Comparison with Combettes' method Let us now show some similarities (and differences) between AAMR and the method by Combettes presented in Section 2.3.4. Combettes' algorithm relies on the product space \mathcal{H}^r and we can then express the recurrence

in (2.32) as

$$\mathbf{w}_{k+1} = \left(1 - \frac{\lambda_k}{2}\right) \mathbf{w}_n + \frac{\lambda_k}{2} R_{\mathcal{D}} \left(2P_{\mathcal{C}} \left(\frac{\mathbf{w}_k + \gamma \mathbf{z}}{\gamma + 1}\right) - \mathbf{w}_k\right), \quad (3.28)$$

$$\mathbf{x}_{k+1} = P_{\mathcal{D}} P_{\mathcal{C}}(\mathbf{w}_{k+1}), \quad (3.29)$$

for $\gamma > 0$, and $(\lambda_k)_{k=0}^{\infty} \in]0, 2]$ such that $\inf_{k \geq 0} \lambda_k > 0$. Observe that, by the dilatation formula in Proposition 1.2(v) and the linearity of $R_{\mathcal{D}}$ given by Proposition 1.11, we have

$$\begin{aligned} R_{\mathcal{D}} \left(2P_{\mathcal{C}} \left(\frac{\mathbf{w}_k + \gamma \mathbf{z}}{\gamma + 1}\right) - \mathbf{w}_k\right) &= R_{\mathcal{D}} \left(2\frac{1}{\gamma + 1} (P_{(\gamma+1)\mathcal{C}-\gamma\mathbf{z}}(\mathbf{w}_k) + \gamma \mathbf{z}) - \mathbf{w}_k\right) \\ &= R_{\mathcal{D}} \left(2\frac{1}{\gamma + 1} P_{(\gamma+1)\mathcal{C}-\gamma\mathbf{z}}(\mathbf{w}_k) - \mathbf{w}_k\right) + R_{\mathcal{D}} \left(2\frac{\gamma}{\gamma + 1} \mathbf{z}\right) \\ &= R_{\mathcal{D}} \left(2\frac{1}{\gamma + 1} P_{(\gamma+1)\mathcal{C}-\gamma\mathbf{z}}(\mathbf{w}_k) - \mathbf{w}_k\right) + 2\frac{\gamma}{\gamma + 1} \mathbf{z}. \end{aligned}$$

Thus, setting $\beta := \frac{1}{1+\gamma}$ and $\alpha_k := \frac{\lambda_k}{2}$, the recurrence in (3.28) can be expressed as

$$\mathbf{w}_{k+1} = (1 - \alpha_k) \mathbf{w}_n + \alpha_k R_{\mathcal{D}} \left(2\beta P_{\frac{1}{\beta}\mathcal{C}-\frac{1-\beta}{\beta}\mathbf{z}}(\mathbf{w}_k) - \mathbf{w}_k\right) + 2\alpha_k(1 - \beta)\mathbf{z}; \quad (3.30)$$

or equivalently, in terms of the AAMR operator (3.5),

$$\mathbf{w}_{k+1} = T_{\frac{1}{\beta}\mathcal{C}-\frac{1-\beta}{\beta}\mathbf{z}, \mathcal{D}, \alpha_k, \beta}(\mathbf{w}_k) + 2(1 - \beta)\alpha_k P_{\mathcal{D}} \left(2\beta P_{\frac{1}{\beta}\mathcal{C}-\frac{1-\beta}{\beta}\mathbf{z}}(\mathbf{w}_k) - \mathbf{w}_k + \mathbf{z}\right),$$

with $\beta \in]0, 1[$ and $\alpha_k \in]0, 1]$. The latter scheme clearly differs from the AAMR iteration in (3.27), even when $\mathbf{z} = \mathbf{0}$ (compare also Figure 3.7 with Figure 2.15).

3.1.4 Numerical experiments

In this section we show the results of five different numerical experiments with the common setting of

Find $P_{U \cap V}(x_0)$, where $U, V \subset \mathbb{R}^{50}$ are closed subspaces such that $U \cap V \neq \{0\}$.

We compare the new AAMR method with Combettes' method (CM) given by (3.29)–(3.30), the method of alternating projections (AP) (2.7), the Douglas–Rachford (DR) method (2.15) (or GDR (2.19)) and Haugazeau's method in its basic form (2.28); as well as we test the influence of the parameters α and β in the behavior of the AAMR method

to see which values give better results. We have also tested the HLWB method (2.30) with the sequence in (2.31), but the obtained results are only shown in Figure 3.13, as the method was clearly outperformed by all the other algorithms in our experiments.

Recall that within this context, the rates of linear convergence of AP and DR depend on the Friedrichs angle between the subspaces (see Theorems 2.7 and 2.11). It was then compulsory to take this angle into consideration in our numerical experiments. In our tests we computed Friedrichs angles from principal angles, via Proposition 1.35.

Observe that, for DR and AAMR, the sequences of interest to be monitored are, respectively,

$$\left(P_U(DR_{U,V}^k(x_0))\right)_{k=0}^{\infty} \quad \text{and} \quad \left(P_U(T_{U-x_0, V-x_0, \alpha, \beta}^k(x_0) + x_0)\right)_{k=0}^{\infty},$$

as these are the sequences that converge to the desired point $P_{U \cap V}(x_0)$; while for AP, CM and Haugazeau's method, the sequence $(x_k)_{k=0}^{\infty}$ given by the respective algorithm is directly the sequence of interest. We used a stopping criterion based on the true error; that is, we terminated the algorithms when the current iterate of the monitored sequence $(w_k)_{k=0}^{\infty}$ satisfies

$$\|w_k - P_{U \cap V}(x_0)\| < \varepsilon$$

for the first time (in real situations this information is not usually available). As in the numerical experiments in [27], the tolerance was set to $\varepsilon := 10^{-3}$.

3.1.4.1 Influence of the parameter α

The purpose of our first experiment was to find out which value of α is optimal for AAMR when it is applied to subspaces. To this aim, we randomly generated 1000 pairs of subspaces and run the AAMR algorithm with a random starting point for each value of $\alpha \in \{0.01, 0.02, \dots, 0.99, 1\}$ and $\beta \in \{0.6, 0.7, 0.8, 0.9\}$. In Figure 3.8 we have plotted the best value of α (the one for which AAMR was faster) against the Friedrichs angle. For a fair comparison in the subsequent tests, we performed the same experiment with CM.

In Figure 3.9 we show prototypical examples of the number of iterations required by AAMR (and GDR as the limit case $\beta = 1$) to find a solution for four different angles. It can be clearly seen that the optimal value of α for GDR is 0.5, as it was expected by Theorem 2.13. For this reason, we use the classical DR ($\alpha = 0.5$) in our subsequent experiments, and set the value of α to 0.9 for AAMR, based on the results shown in Figures 3.8 and 3.9. Although it is not so clear, the same choice appears to be sensible for CM.

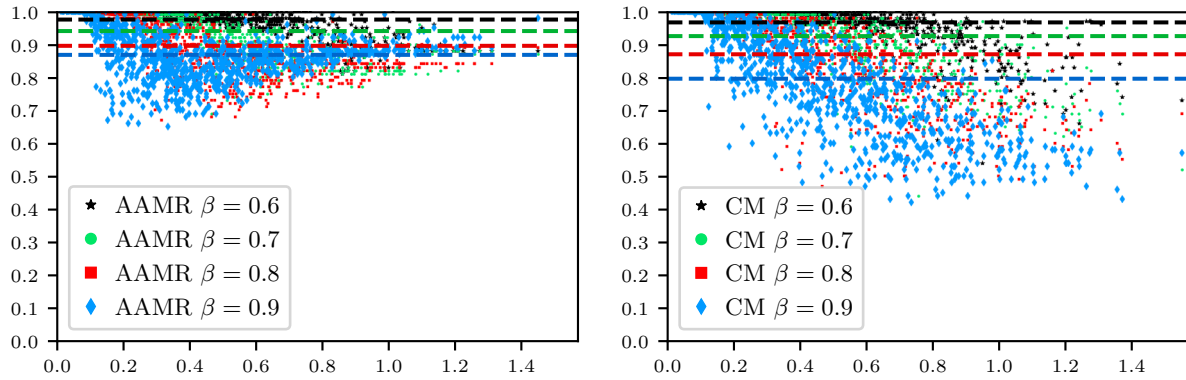


FIGURE 3.8: Best value of α with respect to the Friedrichs angle for 1000 pairs of random subspaces for AAMR (left) and CM (right). For each β , the average value of the best α is represented by a dashed line.

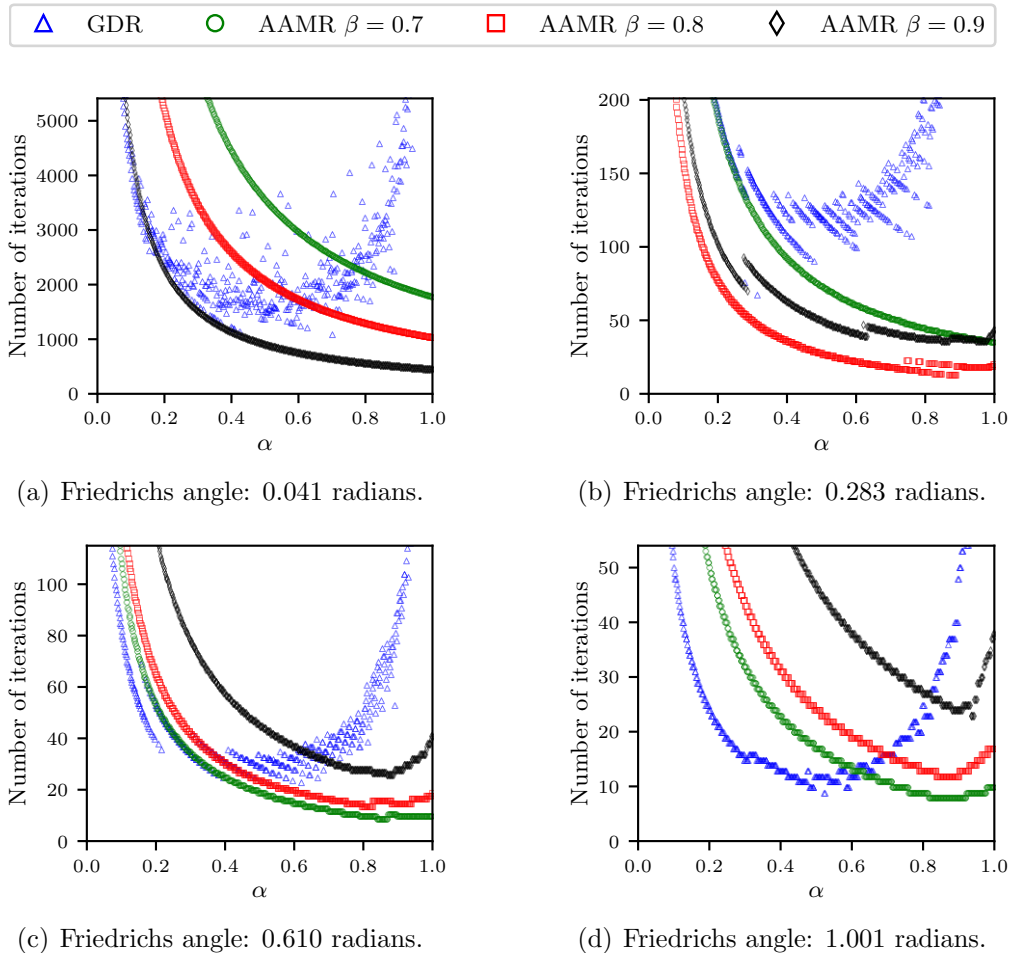


FIGURE 3.9: Number of required iterations with respect to the value of α of GDR and AAMR for three different values of the parameter β .

3.1.4.2 Comparison with other projection methods

Our second experiment consisted in replicating some of the tests performed in [27] to compare DR and AP, adding this time the results of CM, Haugazeau’s method and the new AAMR. We randomly generated 100 pairs of subspaces U and V in \mathbb{R}^{50} . For each pair of subspaces, 10 random starting points (with Euclidean norm 10) were chosen and each of the five methods were applied. We realized that the parameter β has a big influence in the behavior of the AAMR scheme, as can be observed in Figure 3.9. Thus, we computed the sequences generated by the AAMR method for six different values of β (these values were 0.5, 0.6, 0.7, 0.8, 0.9 and 0.99), and we did the same with CM. Although there is a freedom of choice for the initial point in AAMR and CM, we took it as the point to be projected, as this is the starting point that needs to be used by DR, AP and Haugazeau’s method. The results are shown in Figures 3.11 and 3.12. For each pair of subspaces, the horizontal axis represents the Friedrichs angle, and the vertical axis represents the median (Fig. 3.11) or the standard deviation (Fig. 3.12) of the number of iterations required to converge for the 10 random initializations.

On one hand, we can deduce from Figure 3.11 that the rate of convergence of the AAMR method depends on both the angle and the parameter β . For values of β above 0.7, there exists an interval of small angles for which AAMR is the fastest method. For large angles, AP and Haugazeau’s method clearly outperforms DR and AAMR. A simple example showing this behavior is depicted in Figure 3.10.

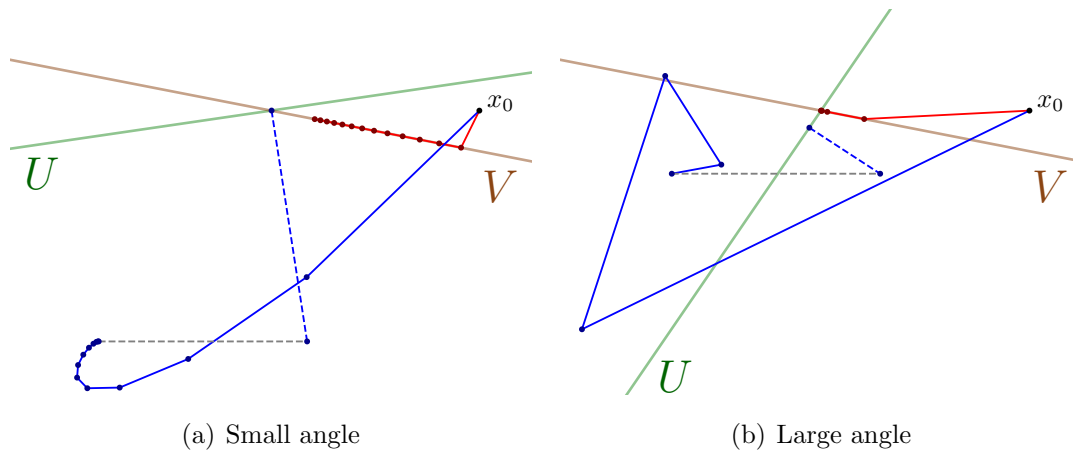


FIGURE 3.10: Behavior of the AAMR (in blue) and alternating projections (in red) algorithms when applied to two lines in \mathbb{R}^2 for two different Friedrichs angles. We see that AAMR outperforms AP for small angles, while AP is faster for large angles.

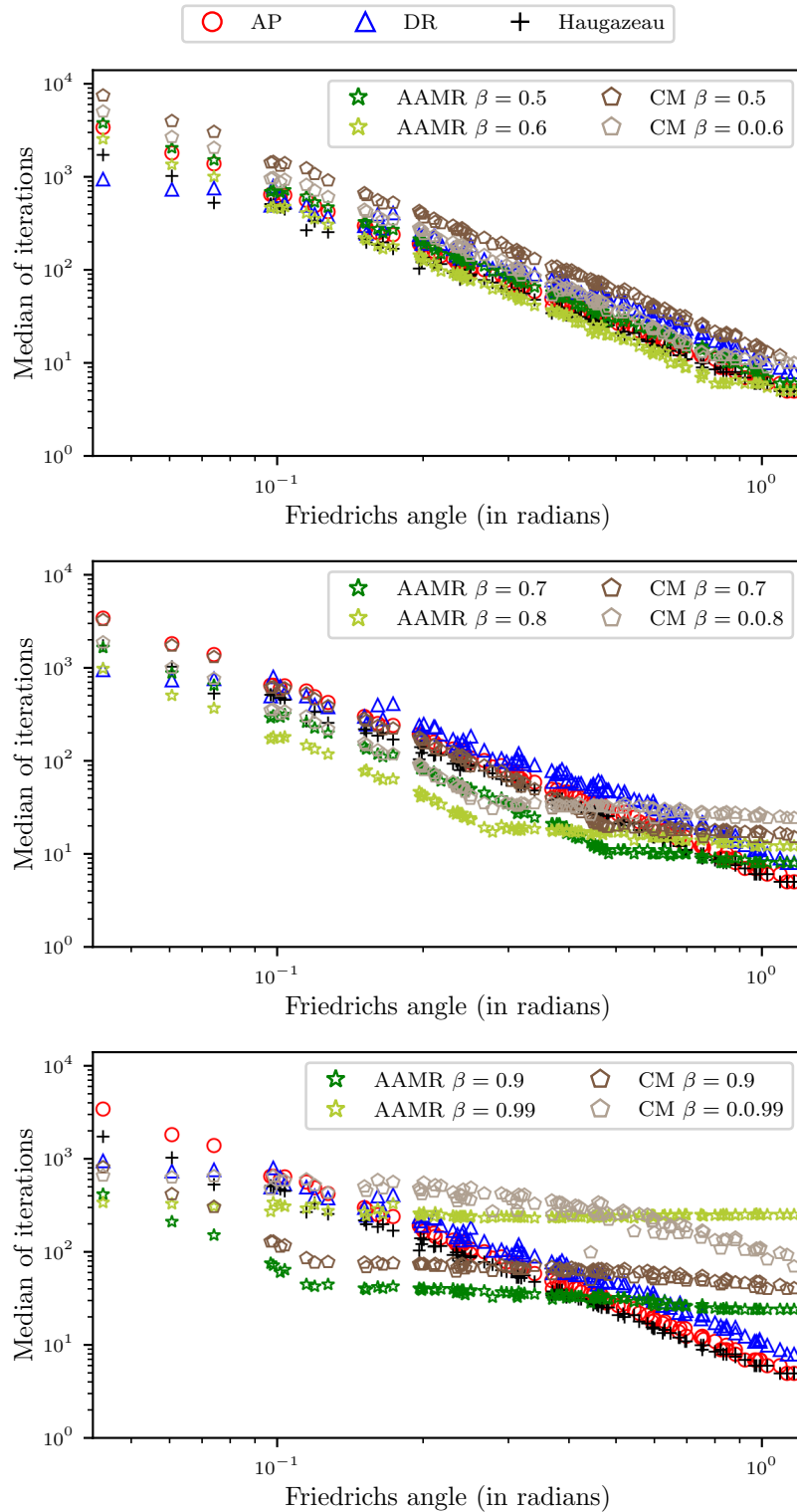


FIGURE 3.11: Median of the required number of iterations with respect to the Friedrichs angle of AP, DR, Haugazeau’s method, CM and AAMR for six different values of β .

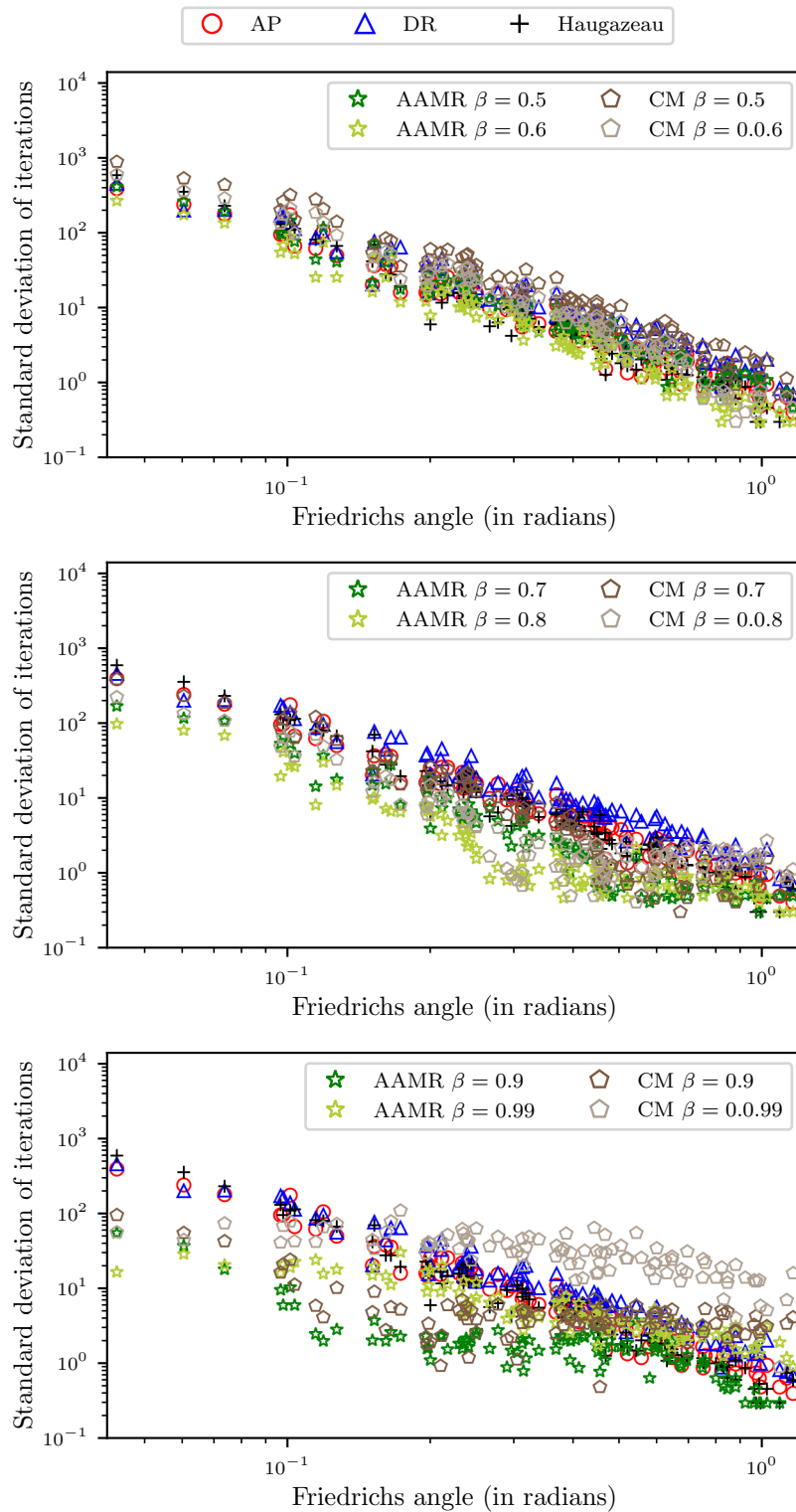


FIGURE 3.12: Standard deviation of the required number of iterations with respect to the Friedrichs angle of AP, DR, Haugazeau's method, CM and AAMR for six different values of β .

We observe that AP, DR and Haugazeau's method satisfy a decrease in the number of iterations when the angle increases. Unfortunately, while the number of iterations in these three methods keep on decreasing for large values of the angle, the AAMR method and CM seem to have an asymptotic behavior around a horizontal line. That is, they need a minimum number of iterations to converge whatever the angle is (although this number is not very big). On the other hand, Figure 3.12 shows that the AAMR method is *more robust* in terms of the standard deviation of the number of iterations. In fact, it seems that the larger the value of β is, the more robust it becomes.

For additionally comparing the rate of convergence of the methods, we computed the distance of the first 100 iterates of the sequences to be monitored to the real solution. We show the results of four instances with well-differentiated Friedrichs angles in Figure 3.13.

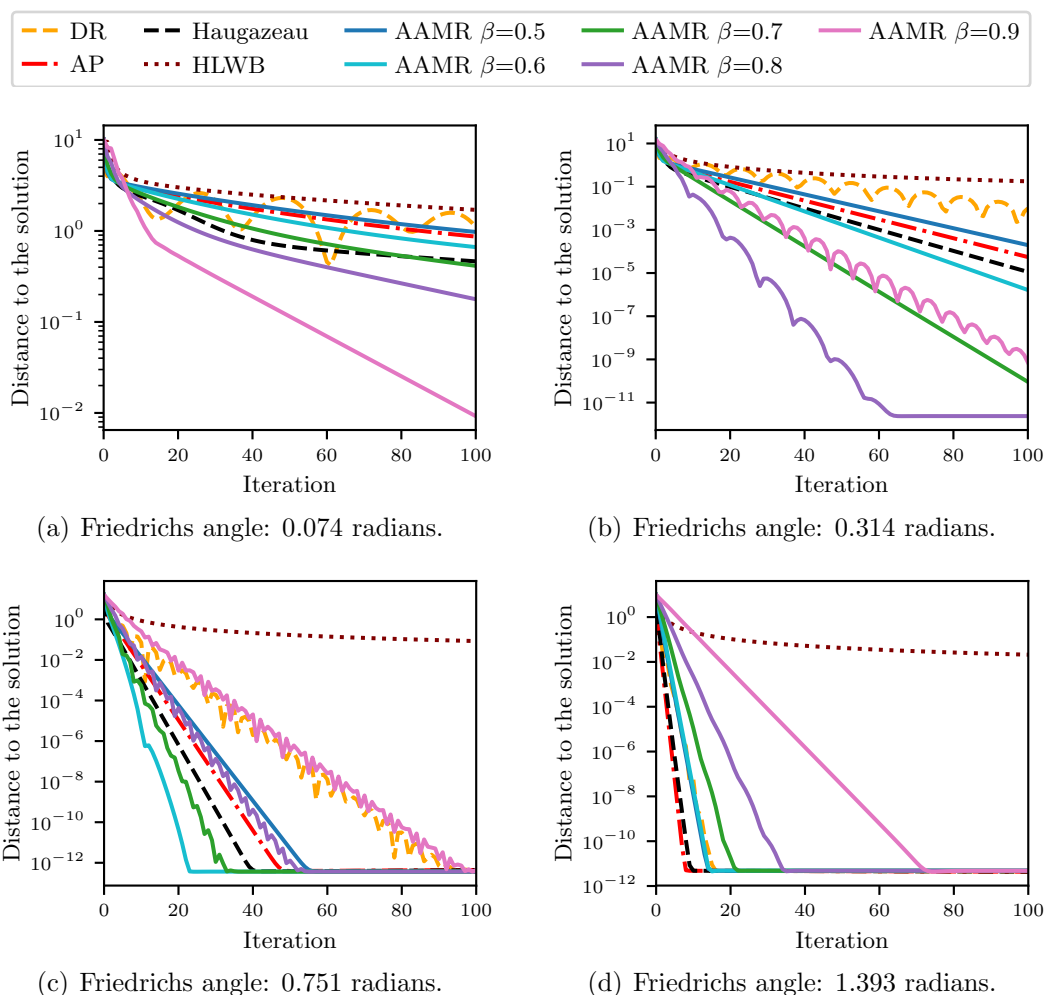


FIGURE 3.13: Distance to the solution of the 100 first iterations of the monitored sequences of AP, DR, HLWB, Haugazeau's method and AAMR for five different values of the parameter β .

Note that we have not included the results of CM in Figure 3.13 to improve the clarity and comprehensibility of the plots, as it was outperformed by AAMR. One might expect the AAMR method to inherit the “rippling” behavior of the DR, specially when β is large, which is when the definition of the iterations of both methods are more similar. This is not entirely truth: although the AAMR method indeed shows these “waves” in Figure 3.13, this behavior depends on both the Friedrichs angle and the parameter β . Additionally, we see in this figure that the AAMR method with a large parameter β clearly outperforms the other schemes when the Friedrichs angle is small. The larger the angle becomes, the better AP and Haugazeau’s method behave. As pointed out in Remark 2.12, it is expected that AP performs better than DR, specially when the Friedrichs angle is large. Finally, we clearly observe in Figure 3.13 that HLWB is the slowest algorithm for solving this problem.

3.1.4.3 Influence of the parameter β

In our third numerical experiment, we continued investigating how the parameter β affects the number of iterations depending on the angle. In this experiment, 1000 pairs of subspaces were generated. Then, for 10 random starting points, we ran the AAMR method for every value of β in $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.99\}$. The results are shown in Figure 3.14. One can see that values of $\beta \leq 0.4$ are an inefficient choice, since $\beta = 0.5$ appears to dominate them for every angle. Larger values of β work better for small angles, but then the performance of the method becomes worse for large angles.

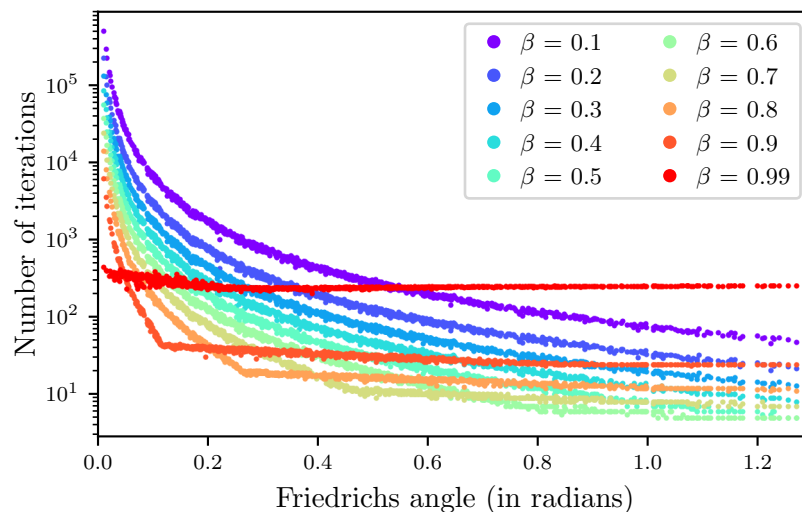


FIGURE 3.14: Median number of iterations for 10 random starting points required by the AAMR method for different values of the parameter β with respect to the Friedrichs angle.

3.2 Optimal rates of linear convergence for two subspaces

The goal of this section is to provide the theoretical results that substantiate the behavior of the algorithm that has been numerically observed in Section 3.1.4. Therefore, throughout this section, we shall assume without loss of generality that U and V are two subspaces of the Euclidean space \mathbb{R}^n such that

$$1 \leq p := \dim U \leq \dim V =: q \leq n - 1, \quad \text{with } U \neq U \cap V \text{ and } U \cap V \neq \{0\}; \quad (3.31)$$

otherwise, the best approximation problem defined by U and V would be trivial.

By following the same approach as in [26], we analyze the linear rate of convergence of the AAMR method by studying the convergence rates of powers of matrices (see Section 1.2.2). The rate obtained depends on both the Friedrichs angle and the parameters defining the algorithm. In addition, we also analyze the optimal selection of the parameters according to the Friedrichs angle, so that the rate of convergence is minimized. This rate coincides with the one for GAP, which is the best among all the known rates with optimal parameters of the classical projection algorithms discussed in Section 2.2. This is not just by chance: the shadow sequences of GAP and AAMR coincide for linear subspaces under some conditions (see Theorem 3.34 and Figure 3.18). The theoretical results are also demonstrated through various numerical experiments.

3.2.1 Convergence rate analysis

We begin this section with the following theorem that establishes the rate of convergence of the AAMR algorithm in terms of α , β and the Friedrichs angle between the subspaces. We denote the positive part of $x \in \mathbb{R}$ by $x^+ := \max\{0, x\}$.

Theorem 3.28 (Rate of linear convergence of AAMR for two subspaces). *Let $U, V \subset \mathbb{R}^n$ be two subspaces verifying (3.31), and fix $\alpha \in]0, 1[$ and $\beta \in]0, 1[$. Then, the AAMR operator*

$$T_{\alpha, \beta} := (1 - \alpha)I_n + \alpha(2\beta P_V - I_n)(2\beta P_U - I_n)$$

is linearly convergent to $P_{U^\perp \cap V^\perp}$ with any rate $\mu \in]\gamma(T_{\alpha, \beta}), 1[$, where

$$\gamma(T_{\alpha, \beta}) = \begin{cases} 1 - 4\alpha\beta(1 - \beta), & \text{if } 0 \leq c_F < c(\alpha, \beta); \\ \sqrt{4(1 - \alpha)\alpha\beta^2 c_F^2 + (1 - 2\alpha\beta)^2}, & \text{if } c(\alpha, \beta) \leq c_F < \hat{c}_\beta; \\ 1 + 2\alpha\beta \left(\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1} \right), & \text{if } \hat{c}_\beta \leq c_F < 1; \end{cases} \quad (3.32)$$

with $c_F := \cos \theta_F$ and θ_F being the Friedrichs angle between U and V ,

$$\widehat{c}_\beta := \frac{\sqrt{(2\beta - 1)^+}}{\beta} \quad \text{and} \quad c(\alpha, \beta) := \begin{cases} \sqrt{\frac{((1-4\alpha\beta(1-\beta))^2 - (1-2\alpha\beta)^2)^+}{4(1-\alpha)\alpha\beta^2}}, & \text{if } \alpha < 1; \\ 0, & \text{if } \alpha = 1. \end{cases} \quad (3.33)$$

Furthermore, $\gamma(T_{\alpha,\beta})$ is the optimal linear convergence rate if and only if

$$\beta \neq \frac{1}{1 + \sin \theta_F} \quad \text{or} \quad \theta_F = \frac{\pi}{2}.$$

Proof. To prove the result, we consider two main cases.

Case 1: $p + q < n$. By Proposition 1.36, we can find an orthogonal matrix $D \in \mathbb{R}^{n \times n}$ such that (1.7) holds. After some calculations, we obtain

$$T_{\alpha,\beta} = D \begin{pmatrix} M_{\alpha,\beta} & 0 & 0 \\ 0 & (1 - 2\alpha\beta)I_{q-p} & 0 \\ 0 & 0 & I_{n-p-q} \end{pmatrix} D^*, \quad (3.34)$$

where

$$M_{\alpha,\beta} := \begin{pmatrix} 2\alpha\beta(2\beta - 1)C^2 + (1 - 2\alpha\beta)I_p & -2\alpha\beta CS \\ 2\alpha\beta(2\beta - 1)CS & 2\alpha\beta C^2 + (1 - 2\alpha\beta)I_p \end{pmatrix}.$$

Let $s := \dim(U \cap V)$ and let $1 = c_1 = \dots = c_s > c_{s+1} = c_F \geq c_{s+2} \geq \dots \geq c_p \geq 0$ be the cosine of the principal angles $0 = \theta_1 = \dots = \theta_s < \theta_{s+1} = \theta_F \leq \theta_{s+2} \leq \dots \leq \theta_p \leq \frac{\pi}{2}$ between U and V (see Proposition 1.35). By using the block determinant formula (see, e.g., [119, (0.8.5.13)]), we deduce after some algebraic manipulation that the spectrum of the matrix $T_{\alpha,\beta}$ is

$$\sigma(T_{\alpha,\beta}) = \begin{cases} \bigcup_{i=1}^p \left\{ 1 + 2\alpha\beta \left(\beta c_i^2 - 1 \pm c_i \sqrt{\beta^2 c_i^2 - 2\beta + 1} \right) \right\} \cup \{1\}, & \text{if } q = p; \\ \bigcup_{i=1}^p \left\{ 1 + 2\alpha\beta \left(\beta c_i^2 - 1 \pm c_i \sqrt{\beta^2 c_i^2 - 2\beta + 1} \right) \right\} \cup \{1, 1 - 2\alpha\beta\}, & \text{if } q > p. \end{cases}$$

Then $\lambda_{i,r} := 1 + 2\alpha\beta \left(\beta c_i^2 - 1 + (-1)^r c_i \sqrt{\beta^2 c_i^2 - 2\beta + 1} \right)$ are eigenvalues of $T_{\alpha,\beta}$, with $i = 1, \dots, p$ and $r = 1, 2$. Observe that the real-valuedness of these eigenvalues depends on the sign of $\beta^2 c_i^2 - 2\beta + 1$. Hence we have that $\lambda_{i,r} \in \mathbb{R}$ if $c_i \geq \beta^{-1} \sqrt{(2\beta - 1)^+} =: \widehat{c}_\beta$, while $\lambda_{i,r} \in \mathbb{C}$ otherwise. In order to study the modulus of the eigenvalues $\lambda_{i,r}$, consider

the function $f_{\alpha,\beta,r} : [0, 1] \rightarrow \mathbb{R}$ given by

$$f_{\alpha,\beta,r}(c) := \begin{cases} (1 + 2\alpha\beta(\beta c^2 - 1))^2 - 4\alpha^2\beta^2c^2(\beta^2c^2 - 2\beta + 1), & \text{if } c < \widehat{c}_\beta; \\ \left(1 + 2\alpha\beta\left(\beta c^2 - 1 + (-1)^r c\sqrt{\beta^2c^2 - 2\beta + 1}\right)\right)^2, & \text{if } c \geq \widehat{c}_\beta. \end{cases}$$

Hence, one has $|\lambda_{i,r}|^2 = f_{\alpha,\beta,r}(c_i)$.

Let us analyze some properties of the function $f_{\alpha,\beta,r}$. When $\widehat{c}_\beta > 0$, observe that $f_{\alpha,\beta,r}$ is continuous at \widehat{c}_β , since

$$\lim_{c \rightarrow \widehat{c}_\beta^-} f_{\alpha,\beta,r}(c) = \lim_{c \rightarrow \widehat{c}_\beta^+} f_{\alpha,\beta,r}(c) = (1 - 2\alpha(1 - \beta))^2.$$

To simplify the presentation, define the auxiliary function

$$g_{\beta,r}(c) := \beta c^2 - 1 + (-1)^r c\sqrt{\beta^2c^2 - 2\beta + 1}, \quad \text{for } c \geq \widehat{c}_\beta.$$

Then,

$$f_{\alpha,\beta,r}(c) = \begin{cases} 4(1 - \alpha)\alpha\beta^2c^2 + (1 - 2\alpha\beta)^2, & \text{if } c < \widehat{c}_\beta; \\ (1 + 2\alpha\beta g_{\beta,r}(c))^2, & \text{if } c \geq \widehat{c}_\beta. \end{cases}$$

The derivative of $f_{\alpha,\beta,r}$ is given for $c \neq \widehat{c}_\beta$ by

$$f'_{\alpha,\beta,r}(c) = \begin{cases} 8(1 - \alpha)\alpha\beta^2c, & \text{if } c < \widehat{c}_\beta; \\ 4\alpha\beta(1 + 2\alpha\beta g_{\beta,r}(c))(-1)^r \frac{(\sqrt{\beta^2c^2 - 2\beta + 1} + (-1)^r \beta c)^2}{\sqrt{\beta^2c^2 - 2\beta + 1}}, & \text{if } c > \widehat{c}_\beta. \end{cases}$$

Further, we claim that $1 + 2\alpha\beta g_{\beta,2}(c) > 1 + 2\alpha\beta g_{\beta,1}(c) \geq 0$ for all $c > \widehat{c}_\beta$. Indeed, since

$$(2\beta^2c^2 - 2\beta + 1)^2 = 4\beta^2c^2(\beta^2c^2 - 2\beta + 1) + (2\beta - 1)^2 \geq (2\beta c)^2(\beta^2c^2 - 2\beta + 1),$$

we deduce, after taking square roots and reordering, that

$$-1 \leq 2\beta \left(\beta c^2 - 1 - c\sqrt{\beta^2c^2 - 2\beta + 1} \right) = 2\beta g_{\beta,1}(c) < 2\beta g_{\beta,2}(c),$$

from where the assertion easily follows.

All the above properties of the function $f_{\alpha,\beta,r}$ can be summarized as follows:

(i) for all $0 \leq c < d \leq \widehat{c}_\beta$,

$$F_{\alpha,\beta}^0 := (1 - 2\alpha\beta)^2 \leq f_{\alpha,\beta,r}(c) \leq f_{\alpha,\beta,r}(d) \leq (1 - 2\alpha(1 - \beta))^2;$$

(ii) for all $\widehat{c}_\beta \leq c < d \leq 1$,

$$\begin{aligned} F_{\alpha,\beta}^1 &:= (1 - 4\alpha\beta(1 - \beta))^2 \leq f_{\alpha,\beta,1}(d) \leq f_{\alpha,\beta,1}(c) \\ &\leq (1 - 2\alpha(1 - \beta))^2 \leq f_{\alpha,\beta,2}(c) < f_{\alpha,\beta,2}(d) \leq 1. \end{aligned}$$

In view of Proposition 1.38, we have to show that the eigenvalue $\lambda = 1$ is semisimple and the only eigenvalue in the unit circle. According to the monotonicity properties of $f_{\alpha,\beta,r}$ in (i)–(ii), we have that

$$|\lambda_{i,r}| \leq 1, \quad \text{for all } i = 1, 2, \dots, p \text{ and } r = 1, 2;$$

and further,

$$|\lambda_{i,r}| = 1 \quad \Leftrightarrow \quad i \in \{1, 2, \dots, s\} \text{ and } r = 2,$$

in which case $\lambda_{i,2} = 1$. Thus, we have shown that $\rho(T_{\alpha,\beta}) = 1$ and $\lambda = 1$ is the only eigenvalue in the unit circle.

Let us show now that $\lambda = 1$ is semisimple. First observe that, for any $\lambda \in \mathbb{C}$, given the block diagonal structure of $T_{\alpha,\beta}$, one has

$$\ker(T_{\alpha,\beta} - \lambda I) = \ker((T_{\alpha,\beta} - \lambda I)^2) \quad \Leftrightarrow \quad \ker(M_{\alpha,\beta} - \lambda I) = \ker((M_{\alpha,\beta} - \lambda I)^2).$$

Then, we can compute

$$M_{\alpha,\beta} - I = 2\alpha\beta \begin{pmatrix} (2\beta - 1)C^2 - I_p & -CS \\ (2\beta - 1)CS & -S^2 \end{pmatrix}.$$

Observe that the matrices C and S can be decomposed as

$$C = \begin{pmatrix} I_s & 0 \\ 0 & \widetilde{C} \end{pmatrix} \quad \text{and} \quad S = \begin{pmatrix} 0_s & 0 \\ 0 & \widetilde{S} \end{pmatrix}, \quad (3.35)$$

where both \widetilde{C} and \widetilde{S} are diagonal matrices and \widetilde{S} has strictly positive entries. Hence,

$$M_{\alpha,\beta} - I = 2\alpha\beta \begin{pmatrix} -2(1 - \beta)I_s & 0 & 0 & 0 \\ 0 & (2\beta - 1)\widetilde{C}^2 - I_{p-s} & 0 & -\widetilde{C}\widetilde{S} \\ 0 & 0 & 0 & 0 \\ 0 & (2\beta - 1)\widetilde{C}\widetilde{S} & 0 & -\widetilde{S}^2 \end{pmatrix},$$

and one has that $\ker(M_{\alpha,\beta} - I) = \ker((M_{\alpha,\beta} - I)^2)$ if and only if $\ker(M_0) = \ker(M_0^2)$, where

$$M_0 := \begin{pmatrix} (2\beta - 1)\tilde{C}^2 - I_{p-s} & -\tilde{C}\tilde{S} \\ (2\beta - 1)\tilde{C}\tilde{S} & -\tilde{S}^2 \end{pmatrix}.$$

Since $\det(M_0) = \det(\tilde{S}^2) \neq 0$ (again, by the block determinant formula), we conclude that $\lambda = 1$ is a semisimple eigenvalue. Then, since $T_{\alpha,\beta}$ is nonexpansive by Proposition 3.6, we have by Proposition 1.38 that $T_{\alpha,\beta}$ is linearly convergent to $P_{\text{Fix}T_{\alpha,\beta}}$ with any rate $\mu \in]\gamma(T_{\alpha,\beta}), 1[$, and $\text{Fix}T_{\alpha,\beta} = U^\perp \cap V^\perp$ by Proposition 3.15.

Furthermore, we can also deduce from the properties (i)–(ii) that the subdominant eigenvalues of $T_{\alpha,\beta}$ are determined by

$$\begin{aligned} \gamma(T_{\alpha,\beta}) &= \max\{|\lambda_{s+1,2}|, |\lambda_{1,1}|\} \\ &= \max\left\{\left|1 + 2\alpha\beta \left(\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1}\right)\right|, 1 - 4\alpha\beta(1 - \beta)\right\}. \end{aligned}$$

To prove (3.32), let us compute the value of $\gamma(T_{\alpha,\beta})$. If $c_F > \hat{c}_\beta$, then $|\lambda_{1,1}| < |\lambda_{s+1,2}|$. Otherwise,

$$|\lambda_{s+1,2}| \leq |\lambda_{1,1}| \quad \Leftrightarrow \quad f_{\alpha,\beta,2}(c_F) \leq f_{\alpha,\beta,1}(1) \quad \Leftrightarrow \quad 4(1 - \alpha)\alpha\beta^2 c_F^2 \leq F_{\alpha,\beta}^1 - F_{\alpha,\beta}^0.$$

Consequently, if we define

$$c(\alpha, \beta) := \begin{cases} \sqrt{\frac{F_{\alpha,\beta}^1 - F_{\alpha,\beta}^0}{4(1 - \alpha)\alpha\beta^2}}, & \text{if } F_{\alpha,\beta}^1 > F_{\alpha,\beta}^0; \\ 0, & \text{otherwise;} \end{cases}$$

which is equivalent to the expression in (3.33), we obtain (3.32). Three possible scenarios for $f_{\alpha,\beta,r}$ and the constants $c(\alpha, \beta)$ and \hat{c}_β depending on the values of α and β are shown in Figure 3.15.

To conclude the proof, let us show that the subdominant eigenvalues are semisimple if and only if $\beta^2 c_F^2 - 2\beta + 1 \neq 0$ or $c_F = 0$. The candidate eigenvalues to be subdominant are $\lambda_{1,1}$ and $\lambda_{s+1,2}$, possibly simultaneously.

Consider first the case where $\lambda_{1,1} = 1 - 4\alpha\beta(1 - \beta)$ is subdominant, and compute

$$M_{\alpha,\beta} - \lambda_{1,1}I = 2\alpha\beta \begin{pmatrix} -(2\beta - 1)S^2 & -CS \\ (2\beta - 1)CS & C^2 - (2\beta - 1)I_p \end{pmatrix}.$$

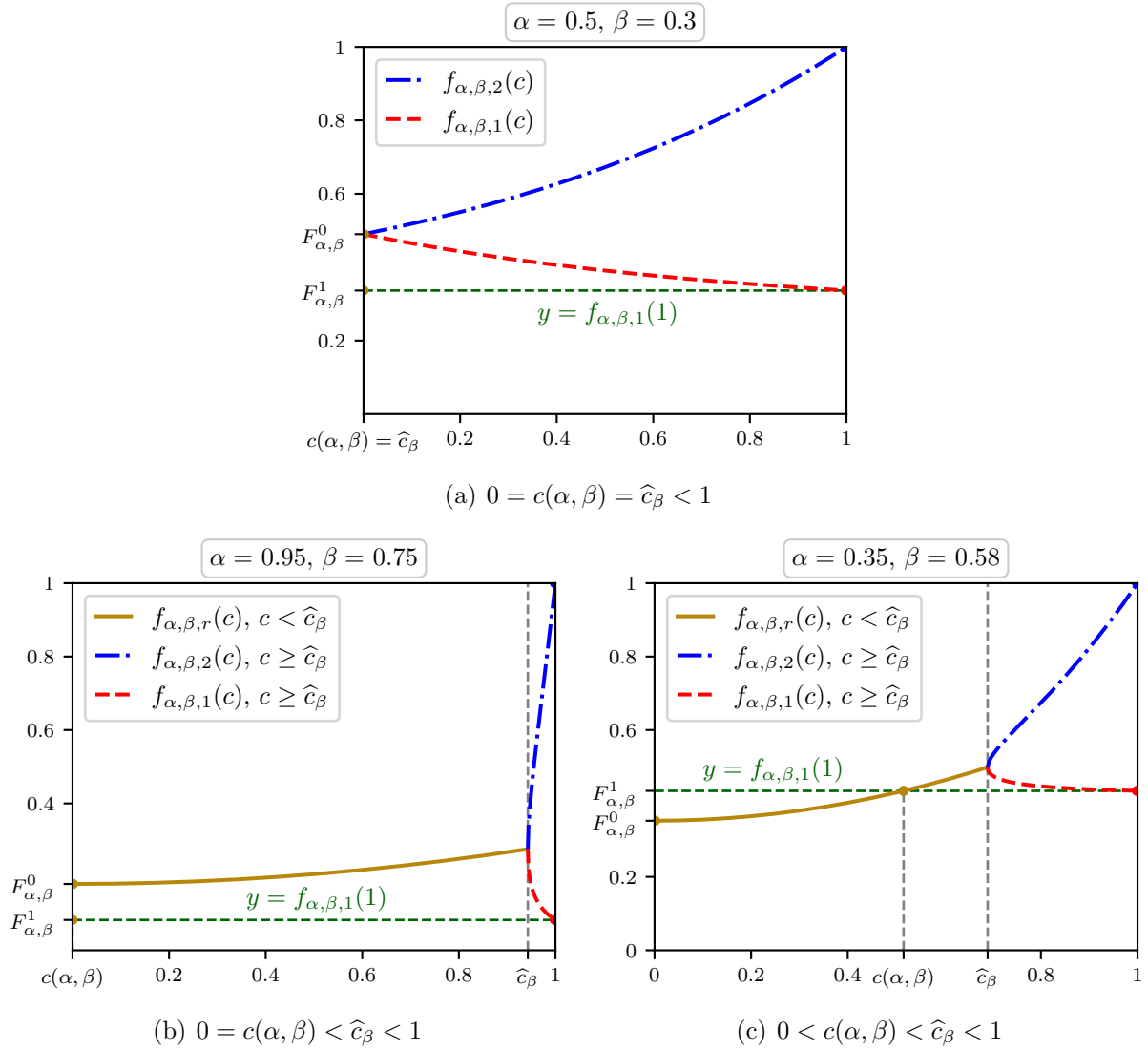


FIGURE 3.15: The three possible scenarios for the function $f_{\alpha,\beta,r}(c)$.

Using the decomposition of C and S given in (3.35), we get

$$M_{\alpha,\beta} - \lambda_{1,1}I = 2\alpha\beta \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & -(2\beta - 1)\widetilde{S}^2 & 0 & -\widetilde{C}\widetilde{S} \\ 0 & 0 & 2(1 - \beta)I_s & 0 \\ 0 & (2\beta - 1)\widetilde{C}\widetilde{S} & 0 & \widetilde{C}^2 - (2\beta - 1)I_{p-s} \end{pmatrix},$$

and one has that $\ker(M_{\alpha,\beta} - \lambda_{1,1}I) = \ker((M_{\alpha,\beta} - \lambda_{1,1}I)^2)$ if and only if

$$\ker(M_1) = \ker(M_1^2), \quad \text{where } M_1 := \begin{pmatrix} -(2\beta - 1)\widetilde{S}^2 & -\widetilde{C}\widetilde{S} \\ (2\beta - 1)\widetilde{C}\widetilde{S} & \widetilde{C}^2 - (2\beta - 1)I_{p-s} \end{pmatrix}.$$

Since we are assuming that $\lambda_{1,1}$ is subdominant, it necessarily holds that $\frac{1}{2} < \beta < 1$. Hence,

$$\det(M_1) = \det\left((2\beta - 1)^2 \tilde{S}^2\right) \neq 0,$$

and one trivially has that $\ker(M_1) = \ker(M_1^2)$, which proves that $\lambda_{1,1}$ is semisimple.

Consider now the case where $\lambda_{s+1,2} = 1 + 2\alpha\beta \left(\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1}\right)$ is a subdominant eigenvalue. Denote $\Delta_F := \sqrt{\beta^2 c_F^2 - 2\beta + 1}$ and compute

$$M_{\alpha,\beta} - \lambda_{s+1,2}I = 2\alpha\beta \begin{pmatrix} (2\beta - 1)C^2 - c_F(\beta c_F + \Delta_F)I_p & -CS \\ (2\beta - 1)CS & C^2 - c_F(\beta c_F + \Delta_F)I_p \end{pmatrix}.$$

Let $t \in \{1, \dots, p - s\}$ be such that $c_F = c_{s+1} = c_{s+2} = \dots = c_{s+t} > c_{s+t+1}$. Then

$$C = \begin{pmatrix} I_s & 0 & 0 \\ 0 & c_F I_t & 0 \\ 0 & 0 & \tilde{C} \end{pmatrix} \quad \text{and} \quad S = \begin{pmatrix} 0_s & 0 & 0 \\ 0 & s_F I_t & 0 \\ 0 & 0 & \tilde{S} \end{pmatrix},$$

where both \tilde{C} and \tilde{S} are diagonal matrices and \tilde{C} has entries strictly smaller than c_F . Hence, one has

$$M_F := M_{\alpha,\beta} - \lambda_{s+1,2}I = 2\alpha\beta \begin{pmatrix} m_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 & m_{25} & 0 \\ 0 & 0 & m_3 & 0 & 0 & m_{36} \\ 0 & 0 & 0 & m_4 & 0 & 0 \\ 0 & m_{52} & 0 & 0 & m_5 & 0 \\ 0 & 0 & m_{63} & 0 & 0 & m_6 \end{pmatrix},$$

where

$$\begin{aligned} m_1 &:= (2\beta - 1 - c_F(\beta c_F + \Delta_F))I_s, & m_2 &:= -c_F(\Delta_F + (1 - \beta)c_F)I_t, \\ m_3 &:= (2\beta - 1)\tilde{C}^2 - c_F(\beta c_F + \Delta_F)I_{p-t-s}, & m_4 &:= (1 - c_F(\beta c_F + \Delta_F))I_s, \\ m_5 &:= -c_F(\Delta_F - (1 - \beta)c_F)I_t, & m_6 &:= \tilde{C}^2 - c_F(\beta c_F + \Delta_F)I_{p-t-s}, \\ m_{25} &:= -c_F s_F I_t, & m_{36} &:= -\tilde{C}\tilde{S}, & m_{52} &:= (2\beta - 1)c_F s_F I_t & \text{and} & m_{63} &:= (2\beta - 1)\tilde{C}\tilde{S}. \end{aligned}$$

Thus, if we denote

$$M_{\{2,5\}} := \begin{pmatrix} m_2 & m_{25} \\ m_{52} & m_5 \end{pmatrix} \quad \text{and} \quad M_{\{3,6\}} := \begin{pmatrix} m_3 & m_{36} \\ m_{63} & m_6 \end{pmatrix},$$

we get that

$$\ker(M_F) = \ker(M_F^2) \Leftrightarrow \ker(M_{\{2,5\}}) = \ker(M_{\{2,5\}}^2) \text{ and } \ker(M_{\{3,6\}}) = \ker(M_{\{3,6\}}^2).$$

On the one hand, by the block determinant formula we have that

$$\begin{aligned} \det(M_{\{3,6\}}) &= \det(m_3m_6 - m_{63}m_{36}) \\ &= \det\left((2\beta - 1)\tilde{C}^4 + c_F^2(\beta c_F + \Delta_F)^2 I_{p-t-s} - 2\beta c_F(\beta c_F + \Delta_F)\tilde{C}^2 + (2\beta - 1)\tilde{C}^2\tilde{S}^2\right) \\ &= \det\left((2\beta - 1 - 2\beta c_F(\beta c_F + \Delta_F))\tilde{C}^2 + c_F^2(\beta c_F + \Delta_F)^2 I_{p-t-s}\right) \\ &= \det\left((-\beta^2 c_F^2 - 2\beta + 1) - \beta^2 c_F^2 - 2\beta c_F \Delta_F\right)\tilde{C}^2 + c_F^2(\beta c_F + \Delta_F)^2 I_{p-t-s} \\ &= \det\left(-(\beta c_F + \Delta_F)^2 \left(\tilde{C}^2 - c_F^2 I_{p-t-s}\right)\right). \end{aligned}$$

Observe that $\beta c_F + \Delta_F = 0$ if and only if $\beta = \frac{1}{2}$ and $c_F = 0$, what implies that $M_{\{3,6\}} = 0_{2p-2t-2s}$. If $\beta c_F + \Delta_F \neq 0$, then $\det(M_{\{3,6\}}) \neq 0$. Thus, in either case, we get $\ker(M_{\{3,6\}}) = \ker(M_{\{3,6\}}^2)$.

On the other hand, we can rewrite

$$M_{\{2,5\}} = -c_F \begin{pmatrix} (\Delta_F + (1 - \beta)c_F)I_t & s_F I_t \\ -(2\beta - 1)s_F I_t & (\Delta_F - (1 - \beta)c_F)I_t \end{pmatrix},$$

and one has

$$M_{\{2,5\}}^2 = c_F^2 \begin{pmatrix} ((\Delta_F + (1 - \beta)c_F)^2 - (2\beta - 1)s_F^2) I_t & 2\Delta_F s_F I_t \\ -2\Delta_F(2\beta - 1)s_F I_t & ((\Delta_F - (1 - \beta)c_F)^2 - (2\beta - 1)s_F^2) I_t \end{pmatrix}.$$

Observing that

$$\begin{aligned} (\Delta_F - (1 - \beta)c_F)^2 - (2\beta - 1)s_F^2 &= 2\Delta_F(\Delta_F - (1 - \beta)c_F), \\ (\Delta_F + (1 - \beta)c_F)^2 - (2\beta - 1)s_F^2 &= 2\Delta_F(\Delta_F + (1 - \beta)c_F), \end{aligned}$$

we deduce that $M_{\{2,5\}}^2 = -2\Delta_F c_F M_{\{2,5\}}$. If $c_F = 0$, then $M_{\{2,5\}} = 0_{2t}$. Therefore, it holds that $\ker(M_{\{2,5\}}) = \ker(M_{\{2,5\}}^2)$ if and only if $\Delta_F \neq 0$ or $c_F = 0$.

Summarizing the discussion above, we have shown that

$$\ker(M_F) = \ker(M_F^2) \Leftrightarrow \Delta_F \neq 0 \text{ or } c_F = 0.$$

Finally, observe that $\Delta_F = 0$ if and only if $c_F = \widehat{c}_\beta$, in which case $\lambda_{s+1,1}$ is a subdominant eigenvalue, and this proves the last assertion in the statement.

Case 2: $p + q \geq n$. We can take some $t \geq 1$ such that $n' := n + t > p + q$, and consider the subspaces $U' := U \times \{0_{t \times 1}\} \subset \mathbb{R}^{n'}$, $V' := V \times \{0_{t \times 1}\} \subset \mathbb{R}^{n'}$, and denote by $T'_{\alpha,\beta} := T_{U',V',\alpha,\beta} = (1 - \alpha)I + \alpha(2\beta P_{V'} - I)(2\beta P_{U'} - I)$. Since

$$P_{U'} = \begin{pmatrix} P_U & 0 \\ 0 & 0_t \end{pmatrix} \quad \text{and} \quad P_{V'} = \begin{pmatrix} P_V & 0 \\ 0 & 0_t \end{pmatrix},$$

it holds that

$$T'_{\alpha,\beta} = \begin{pmatrix} T_{\alpha,\beta} & 0 \\ 0 & I_t \end{pmatrix}.$$

Therefore, $\sigma(T_{\alpha,\beta}) \cup \{1\} = \sigma(T'_{\alpha,\beta})$ and $\gamma(T_{\alpha,\beta}) = \gamma(T'_{\alpha,\beta})$. Note that the principal angles between U' and V' are the same that the ones between U and V . Hence, the result follows from applying Case 1 to $T'_{\alpha,\beta}$. \square

REMARK 3.29. For simplicity, we have assumed that $\dim U = p \leq q = \dim V$. If this is not the case and $q < p$, observe that one has to exchange the matrix decomposition of P_U and P_V given in (1.7). In this case, one can check that the matrix $T_{\alpha,\beta}$ obtained corresponds to the transpose of the one given in (3.34). Hence, the spectrum of $T_{\alpha,\beta}$ remains the same and thus all the results in Theorem 3.28 also hold.

REMARK 3.30. The expression in (3.32) corroborates what it was numerically observed in Figure 3.11: there are values of α and β for which the rate of convergence of AAMR does not depend on the value of the Friedrichs angle for all angles larger than $\arccos c(\alpha, \beta)$.

Corollary 3.31. *Let $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Given $z \in \mathbb{R}^n$, choose any $x_0 \in \mathbb{R}^n$ and consider the sequence generated, for $k = 0, 1, 2, \dots$, by*

$$x_{k+1} = T_{U-z, V-z, \alpha, \beta}(x_k) = (1 - \alpha)x_k + \alpha(2\beta P_{V-z} - I)(2\beta P_{U-z} - I)(x_k).$$

Let $\gamma(T_{\alpha,\beta})$ be given by (3.32). Then, for every $\mu \in]\gamma(T_{\alpha,\beta}), 1[$, the sequence $(x_k)_{k=0}^\infty$ and the shadow sequence $(P_U(z + x_k))_{k=0}^\infty$ are R -linearly convergent to $P_{\text{Fix} T_{U-z, V-z, \alpha, \beta}}(x_0)$ and to $P_{U \cap V}(z)$, respectively, both with rate μ . That is, there exists a positive integer k_0 such that

$$\|P_U(z + x_k) - P_{U \cap V}(z)\| \leq \|x_k - P_{\text{Fix} T_{U-z, V-z, \alpha, \beta}}(x_0)\| \leq \mu^k, \quad \text{for all } k \geq k_0. \quad (3.36)$$

Proof. Since U and V are finite-dimensional subspaces they have the strong CHIP by Proposition 1.33. Then, Corollary 3.10 yields $\text{Fix } T_{U-z, V-z, \alpha, \beta} \neq \emptyset$ and by Proposition 3.14 we have that

$$x_{k+1} = T_{U-z, V-z, \alpha, \beta}(x_k) = T_{U, V, \alpha, \beta}(x_k - x^*) + x^*,$$

for $x^* := P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0)$. Hence, one has

$$\|x_k - x^*\| = \|T_{U, V, \alpha, \beta}(x_{k-1} - x^*)\| = \cdots = \|T_{U, V, \alpha, \beta}^k(x_0 - x^*)\|.$$

Again by Proposition 3.14, together with Proposition 3.15, one has

$$\text{Fix } T_{U-z, V-z, \alpha, \beta} = x^* + U^\perp \cap V^\perp.$$

By using the translation formula for projections in Proposition 1.2(iv), we get that

$$x^* = P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0) = P_{x^* + U^\perp \cap V^\perp}(x_0) = P_{U^\perp \cap V^\perp}(x_0 - x^*) + x^*,$$

which implies $P_{U^\perp \cap V^\perp}(x_0 - x^*) = 0$, and therefore,

$$\|x_k - x^*\| = \|T_{U, V, \alpha, \beta}^k(x_0 - x^*)\| = \|(T_{U, V, \alpha, \beta}^k - P_{U^\perp \cap V^\perp})(x_0 - x^*)\|.$$

Let $\nu \in]\gamma(T_{\alpha, \beta}), \mu[$. Since $\nu > \gamma(T_{\alpha, \beta})$, by Theorem 3.28, there exists a positive integer k_1 and some $M > 0$ such that

$$\|T_{U, V, \alpha, \beta}^k - P_{U^\perp \cap V^\perp}\| \leq M\nu^k, \quad \text{for all } k \geq k_1.$$

Let $k_0 \geq k_1$ be a positive integer such that

$$\left(\frac{\mu}{\nu}\right)^k \geq M\|x_0 - x^*\|, \quad \text{for all } k \geq k_0.$$

Then, we deduce that

$$\|x_k - x^*\| \leq \|T_{U, V, \alpha, \beta}^k - P_{U^\perp \cap V^\perp}\| \|x_0 - x^*\| \leq M\nu^k \|x_0 - x^*\| \leq \mu^k,$$

for all $k \geq k_0$, which proves the second inequality in (3.36).

By Proposition 3.8 and the translation formula for projections in Proposition 1.2(iv), we can deduce that

$$P_U(z + P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0)) = P_{U \cap V}(z).$$

Thus, the first inequality in (3.36) is a consequence of this, and the linearity and nonexpansiveness of P_U . Indeed

$$\begin{aligned} \|P_U(z + x_k) - P_{U \cap V}(z)\| &= \|P_U(z + x_k) - P_U(z + P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0))\| \\ &= \|P_U(x_k - P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0))\| \leq \|x_k - P_{\text{Fix } T_{U-z, V-z, \alpha, \beta}}(x_0)\|, \end{aligned}$$

which completes the proof. \square

REMARK 3.32. The convergence of AAMR for arbitrary closed and convex sets was only guaranteed for the shadow sequence when $\alpha = 1$ (see Theorem 3.17). Observe that Corollary 3.31 extends this result for the case of finite-dimensional spaces since it proves the convergence of both the original and the shadow sequence even when $\alpha = 1$.

We now look for the values of the parameters α and β , in order to minimize the rate of convergence of the AAMR method obtained in Theorem 3.28.

Theorem 3.33. *The infimum of the linear convergence rates of the AAMR operator $T_{\alpha, \beta}$ attains its smallest value at $\alpha^* = 1$ and $\beta^* = \frac{1}{1 + \sin \theta_F}$, where θ_F is the Friedrichs angle between U and V ; i.e., it holds*

$$\frac{1 - \sin \theta_F}{1 + \sin \theta_F} = \gamma(T_{1, \beta^*}) \leq \gamma(T_{\alpha, \beta}) \quad \text{for all } (\alpha, \beta) \in]0, 1[\times]0, 1[.$$

Furthermore, $\gamma(T_{1, \beta^*})$ is an optimal linear convergence rate if and only if $\theta_F = \frac{\pi}{2}$.

Proof. Let us look for the values of parameters α and β that minimize the rate $\gamma(T_{\alpha, \beta})$ given by (3.32). Define the sets $D :=]0, 1[\times]0, 1[$,

$$\begin{aligned} D_1 &:= \left\{ (\alpha, \beta) \in D : \beta < \frac{1}{1 + s_F} \right\}, \\ D_2 &:= \left\{ (\alpha, \beta) \in D : \frac{1}{1 + s_F} \leq \beta \leq \frac{1}{1 + s_F^2} \text{ or } \alpha \geq \frac{1 - \beta(1 + s_F^2)}{\beta(4(1 - \beta)^2 - s_F^2)}, \beta > \frac{1}{1 + s_F^2} \right\}, \\ D_3 &:= \left\{ (\alpha, \beta) \in D : \alpha < \frac{1 - \beta(1 + s_F^2)}{\beta(4(1 - \beta)^2 - s_F^2)}, \beta > \frac{1}{1 + s_F^2} \right\}, \end{aligned}$$

and the functions

$$\begin{aligned} \Gamma_1(\alpha, \beta) &:= 1 + 2\alpha\beta \left(\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1} \right), \quad \text{for } (\alpha, \beta) \in D_1, \\ \Gamma_2(\alpha, \beta) &:= \sqrt{4(1 - \alpha)\alpha\beta^2 c_F^2 + (1 - 2\alpha\beta)^2}, \quad \text{for } (\alpha, \beta) \in D, \\ \Gamma_3(\alpha, \beta) &:= 1 - 4\alpha\beta(1 - \beta), \quad \text{for } (\alpha, \beta) \in D, \end{aligned}$$

having $D = D_1 \cup D_2 \cup D_3$. Hence, we can define the convergence rate in terms of the parameters α and β through the function

$$\Gamma(\alpha, \beta) := \gamma(T_{\alpha, \beta}) = \begin{cases} \Gamma_1(\alpha, \beta), & \text{if } (\alpha, \beta) \in D_1, \\ \Gamma_2(\alpha, \beta), & \text{if } (\alpha, \beta) \in D_2, \\ \Gamma_3(\alpha, \beta), & \text{if } (\alpha, \beta) \in D_3, \end{cases}$$

see Figure 3.16.

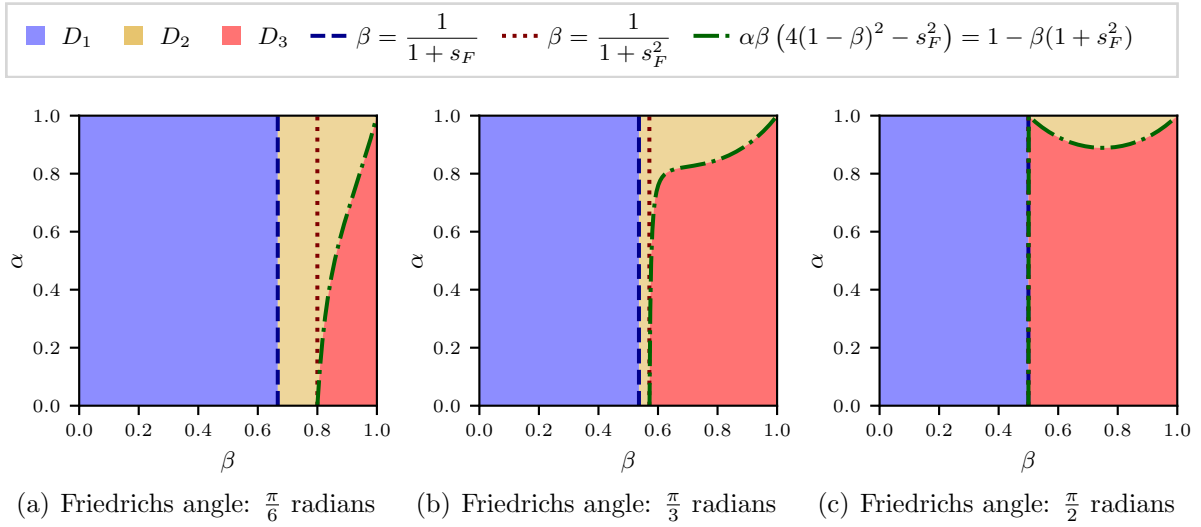


FIGURE 3.16: Piecewise domain of the function $\Gamma(\alpha, \beta)$ for three different values of the angle.

The function Γ is piecewise defined, continuous and differentiable on the interior of each of the three regions D_1 , D_2 and D_3 , but is not differentiable on the boundaries. Let us analyze the three problems of minimizing the function Γ over the closure of each of the three pieces. The gradient of the functions Γ_1 , Γ_2 and Γ_3 are given by

$$\nabla\Gamma_1(\alpha, \beta) = \begin{pmatrix} 2\beta \left(\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1} \right) \\ 2\alpha \left(\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1} \right) \left(\frac{\beta c_F + \sqrt{\beta^2 c_F^2 - 2\beta + 1}}{\sqrt{\beta^2 c_F^2 - 2\beta + 1}} \right) \end{pmatrix},$$

$$\nabla\Gamma_2(\alpha, \beta) = \frac{1}{\sqrt{4(1-\alpha)\alpha\beta^2 c_F^2 + (1-2\alpha\beta)^2}} \begin{pmatrix} 2\beta (\beta c_F^2 - 1 + 2\alpha\beta(1 - c_F^2)) \\ 2\alpha (2\beta c_F^2 - 1 + 2\alpha\beta(1 - c_F^2)) \end{pmatrix},$$

$$\nabla\Gamma_3(\alpha, \beta) = \begin{pmatrix} -4\beta(1-\beta) \\ -4\alpha(1-2\beta) \end{pmatrix}.$$

To minimize Γ over $\overline{D_1}$, first observe that for any $(\alpha, \beta) \in D_1$ we have that $\beta < \frac{1}{1+s_F}$ and thus $\beta^2 c_F^2 - 2\beta + 1 > 0$. Then Γ_1 is smooth in D_1 , and we further assert that

$$\frac{\partial \Gamma_1}{\partial \alpha}(\alpha, \beta) < 0, \quad \frac{\partial \Gamma_1}{\partial \beta}(\alpha, \beta) < 0, \quad \text{for all } (\alpha, \beta) \in D_1. \quad (3.37)$$

Indeed, on the one hand, since $c_F^2 < 1$, then $\beta c_F^2 - 1 + c_F \sqrt{\beta^2 c_F^2 - 2\beta + 1} < 0$. On the other hand, one has $\beta c_F + \sqrt{\beta^2 c_F^2 - 2\beta + 1} \geq 0$, with equality if and only if $c_F = 0$ and $\beta = \frac{1}{2}$. In this case, the point has the form $(\alpha, \frac{1}{2}) \notin D_1$ for $\alpha \in]0, 1]$. We have then shown that (3.37) holds, and thus $(1, \frac{1}{1+s_F})$ becomes the unique minimum of Γ over $\overline{D_1}$.

Let us consider now the problem of minimizing Γ over $\overline{D_2}$. To address this problem, we consider two cases. Suppose first that $c_F = 0$ and observe that

$$\Gamma_2(\alpha, \beta) = \sqrt{(1 - 2\alpha\beta)^2} \geq 0, \quad \text{for all } (\alpha, \beta) \in \overline{D_2},$$

having $\Gamma_2(\alpha, \beta) = 0$ if and only if $2\alpha\beta = 1$. Since $(1, \frac{1}{2})$ is the only point in $\overline{D_2}$ satisfying this equation, we deduce that it is the unique minimum.

Suppose now that $c_F > 0$. In this case, we claim that Γ_2 attains its minimum over the region $\overline{D_2} \cup D_3 = [0, 1] \times [\frac{1}{1+s_F}, 1]$ at the point $(1, \frac{1}{1+s_F}) \in D_2$, and so does Γ over $\overline{D_2}$. Indeed, observe that Γ_2 is smooth on the interior of the set $D_2 \cup D_3$. Moreover, $\nabla \Gamma_2$ only vanishes in D at the point $(0, 0)$. Therefore, the minimum has to be attained at some point in the boundary. Note that, for all $\beta \in [\frac{1}{1+s_F}, 1]$, the following hold.

- (i) $\Gamma_2(0, \beta) = 1$.
- (ii) $\Gamma_2(1, \beta) = 2\beta - 1$, which attains its minimum at $\beta = \frac{1}{1+s_F}$.
- (iii) The function $\alpha \mapsto \Gamma_2(\alpha, \beta)$ is the square root of a positive non-degenerated convex parabola,

$$\alpha \mapsto \Gamma_2(\alpha, \beta) = \sqrt{4\beta^2 s_F^2 \alpha^2 - 4\beta(1 - \beta c_F^2)\alpha + 1};$$

which attains its minimum at $\alpha^*(\beta) := \frac{1 - \beta c_F^2}{2\beta s_F^2}$. Since $\alpha^*(\frac{1}{1+s_F}) = \frac{1+s_F}{2s_F} \geq 1$, we have

$$\Gamma_2\left(1, \frac{1}{1+s_F}\right) < \Gamma_2\left(\alpha, \frac{1}{1+s_F}\right), \quad \text{for all } \alpha \in [0, 1].$$

On the other hand, $\alpha^*(1) = \frac{1}{2}$, which implies

$$\Gamma_2\left(\frac{1}{2}, 1\right) < \Gamma_2(\alpha, 1), \quad \text{for all } \alpha \in [0, 1].$$

Then, noting that

$$\Gamma_2 \left(1, \frac{1}{1+s_F} \right) = \frac{1-s_F}{1+s_F} < c_F = \Gamma_2 \left(\frac{1}{2}, 1 \right),$$

we have shown by (i)–(iii) that Γ_2 attains its minimum over $\overline{D_2 \cup D_3}$ at $\left(1, \frac{1}{1+s_F} \right) \in D_2$, as claimed.

Finally, observe that if $(\alpha, \beta) \in D_3$, it holds that $\frac{1}{2} < \beta < 1$. Then

$$\frac{\partial \Gamma_3}{\partial \alpha}(\alpha, \beta) < 0, \quad \frac{\partial \Gamma_3}{\partial \beta}(\alpha, \beta) > 0, \quad \text{for all } (\alpha, \beta) \in D_3.$$

Thus, there exists some point $(\alpha_3^*, \beta_3^*) \in \overline{D_3}$ with $\alpha_3^* \beta_3^* (4(1 - \beta_3^*)^2 - s_F^2) = 1 - \beta_3^* (1 + s_F^2)$ such that

$$\Gamma_3(\alpha_3^*, \beta_3^*) < \Gamma_3(\alpha, \beta), \quad \text{for all } (\alpha, \beta) \in D_3.$$

Note that (α_3^*, β_3^*) lies on the boundary curve between D_2 and D_3 . Since Γ is continuous on D , it holds that $\Gamma_2(\alpha_3^*, \beta_3^*) = \Gamma_3(\alpha_3^*, \beta_3^*)$ and hence,

$$\Gamma \left(1, \frac{1}{1+s_F} \right) \leq \Gamma(\alpha_3^*, \beta_3^*) < \Gamma(\alpha, \beta), \quad \text{for all } (\alpha, \beta) \in D_3.$$

Hence, all the reasoning above proves that

$$\operatorname{argmin}_{(\alpha, \beta) \in D} \Gamma(\alpha, \beta) = \left(1, \frac{1}{1+s_F} \right),$$

with $\Gamma \left(1, \frac{1}{1+s_F} \right) = \frac{1-s_F}{1+s_F}$. Finally, by the last assertion in Theorem 3.28, $\gamma \left(T_{1, \frac{1}{1+s_F}} \right)$ is an optimal linear convergence rate if and only if $c_F = 0$, as claimed. \square

3.2.2 Comparison with other projection methods

We compare now the rate of AAMR with optimal parameters obtained in Theorem 3.33 with the rates of AP, SP and DR, as well as the ones of their relaxed variants in Theorems 2.8 and 2.13. We summarize the key features of these schemes in Table 3.1, where we recall the operator defining the iteration of each method, as well as the optimal parameters and rates of convergence when these schemes are applied to linear subspaces. Note that all these rates only depend on the Friedrichs angle θ_F between the subspaces (this is not the case for PRAP, see Theorem 2.8(ii), so it has not been taken into consideration).

Method	Optimal parameter(s)	Rate
Alternating Projections $AP = P_V P_U$	–	$\cos^2 \theta_F$
Simultaneous Projections $SP = \frac{1}{2}(P_V + P_U)$	–	$\frac{1}{2} + \frac{1}{2} \cos \theta_F$
Relaxed Alternating Projections $RAP = (1 - \alpha)I + \alpha P_V P_U$	$\alpha^* = \frac{2}{1 + \sin^2 \theta_F}$	$\frac{1 - \sin^2 \theta_F}{1 + \sin^2 \theta_F}$
Generalized Alternating Projections $GAP = (1 - \alpha)I + \alpha (\alpha_2 P_V + (1 - \alpha_2)I) (\alpha_1 P_U + (1 - \alpha_1)I)$	$\alpha^* = 1$ $\alpha_1^* = \alpha_2^* = \frac{2}{1 + \sin \theta_F}$	$\frac{1 - \sin \theta_F}{1 + \sin \theta_F}$
Generalized Douglas–Rachford $GDR = (1 - \alpha)I + \alpha(2P_V - I)(2P_U - I)$	$\alpha^* = \frac{1}{2} (DR)$	$\cos \theta_F$
Averaged Alternating Modified Reflections $AAMR = (1 - \alpha)I + \alpha(2\beta P_V - I)(2\beta P_U - I)$	$\alpha^* = 1, \beta^* = \frac{1}{1 + \sin \theta_F}$	$\frac{1 - \sin \theta_F}{1 + \sin \theta_F}$

TABLE 3.1: Rates of convergence with optimal parameters of AP, SP, RAP, GAP, GDR and AAMR when they are applied to two subspaces.

On the one hand, we observe that the rates with optimal parameters for AAMR and GAP coincide. The reason for this is discussed in Section 3.2.2.1, where under some conditions, we show that the shadow sequences of these methods coincide for linear subspaces (Theorem 3.34). On the other hand, we note that the rate for AAMR/GAP is considerably smaller than the one of other methods, see Figure 3.17.

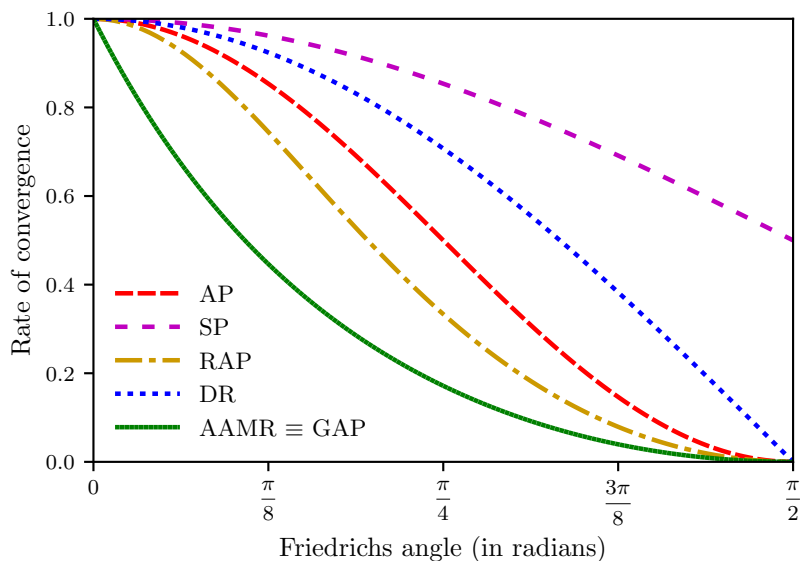


FIGURE 3.17: Comparison of the rates of linear convergence with optimal parameters of AP, SP, RAP, DR, AAMR and GAP.

3.2.2.1 Relationship between AAMR and GAP for subspaces

Observe in Table 3.1 that, apart from the fact that the rates of GAP and AAMR coincide, their optimal parameters are closely related, in the sense that

$$\alpha_{AAMR}^* = \alpha_{GAP}^* \quad \text{and} \quad \alpha_{1,GAP}^* = \alpha_{2,GAP}^* = 2\beta_{AAMR}^*. \quad (3.38)$$

In general, if the parameters defining the AAMR and GAP schemes are chosen so that they satisfy (3.38), then both methods turn out to be strongly connected when applied to two subspaces. This is formally stated in the following result.

Theorem 3.34. *Let $U, V \subseteq \mathbb{R}^n$ be two closed subspaces and let $z \in \mathbb{R}^n$. Set $x_0 = 0$ and let $\{x_k\}_{k=0}^\infty$ be the sequence generated by AAMR (3.3) with parameters $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. Set $z_0 = z$ and let $\{z_k\}_{k=0}^\infty$ be the sequence generated by GAP (2.12) with parameters α and $\alpha_1 = \alpha_2 = 2\beta$. Then, for all nonnegative integer k , one has*

$$P_U(x_k + z) = P_U(z_k) \quad \text{and} \quad P_V(x_k) = (2\beta - 1)P_V(z_k - z). \quad (3.39)$$

Proof. Using the linearity of the projection operator onto subspaces (Proposition 1.11), we deduce that the AAMR iteration (3.3) takes the form

$$\begin{aligned} x_{k+1} &= (1 - \alpha)x_k + \alpha(2\beta P_{V-z} - I)(2\beta P_{U-z} - I)(x_k) \\ &= (1 - \alpha)x_k + \alpha(2\beta P_{V-z} - I)(2\beta(P_U(x_k + z) - z) - x_k) \\ &= (1 - \alpha)x_k + \alpha(2\beta P_{V-z}(2\beta(P_U(x_k + z) - z) - x_k) - 2\beta(P_U(x_k + z) - z) + x_k) \\ &= x_k + 2\alpha\beta(P_V(2\beta(P_U(x_k + z) - z) - x_k + z) - P_U(x_k + z)) \\ &= x_k + 2\alpha\beta(2\beta P_V P_U(x_k + z) + (1 - 2\beta)P_V(z) - P_V(x_k) - P_U(x_k + z)). \end{aligned} \quad (3.40)$$

To simplify the notation, let $\eta := 2\beta$. We shall prove (3.39) by induction. Since both equalities clearly hold for $k = 0$, we can assume that they are valid for some $k \geq 0$. By (3.40), the sequence generated by the AAMR scheme satisfies

$$\begin{aligned} P_U(x_{k+1}) &= P_U(x_k) + \alpha\eta(\eta P_U P_V P_U(x_k + z) \\ &\quad + (1 - \eta)P_U P_V(z) - P_U P_V(x_k) - P_U(x_k + z)) \\ &= (\alpha\eta^2 P_U P_V P_U - \alpha\eta P_U P_V + (1 - \alpha\eta)P_U)(x_k) \\ &\quad + \alpha(\eta^2 P_U P_V P_U + \eta(1 - \eta)P_U P_V - \eta P_U)(z), \end{aligned} \quad (3.41)$$

and,

$$\begin{aligned} P_V(x_{k+1}) &= P_V(x_k) + \alpha\eta((\eta - 1)P_V P_U(x_k + z) + P_V((1 - \eta)z - x_k)) \\ &= \alpha\eta(\eta - 1)P_V P_U(x_k + z) + P_V(\alpha\eta(1 - \eta)z + (1 - \alpha\eta)x_k). \end{aligned} \quad (3.42)$$

Thanks to the linearity of the projectors onto subspaces and using $\alpha_1 = \alpha_2 = \eta$, the GAP iteration (2.12) takes the form

$$z_{k+1} = (1 - \alpha\eta(2 - \eta))z_k + \alpha(\eta^2 P_V P_U z_k + \eta(1 - \eta)P_V z_k + \eta(1 - \eta)P_U z_k);$$

and thus this scheme verifies

$$P_U(z_{k+1}) = (\alpha\eta^2 P_U P_V P_U + \alpha\eta(1 - \eta)P_U P_V + (1 - \alpha\eta)P_U)(z_k), \quad (3.43)$$

and

$$P_V(z_{k+1}) = (1 - \alpha\eta)P_V(z_k) + \alpha\eta P_V P_U(z_k). \quad (3.44)$$

Then, by (3.42), the induction hypothesis (3.39) and (3.44), we obtain

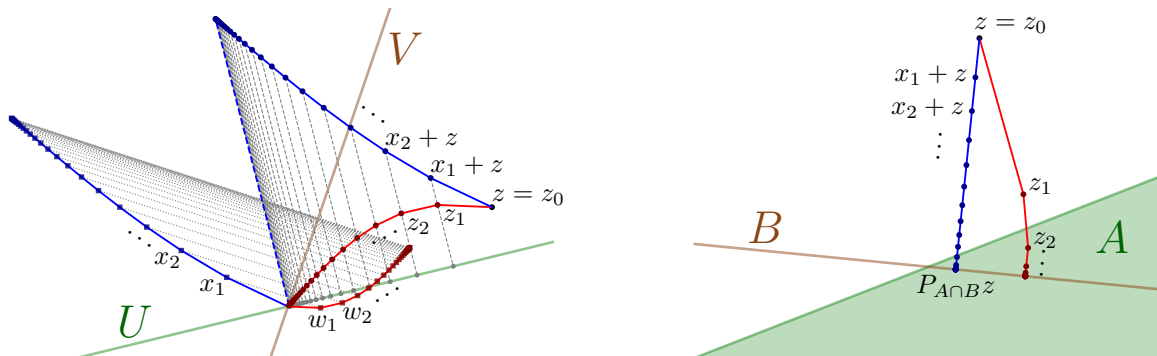
$$\begin{aligned} P_V(x_{k+1}) &= \alpha\eta(\eta - 1)P_V P_U(z_k) + (1 - \alpha\eta)P_V(x_k) + P_V(\alpha\eta(1 - \eta)z) \\ &= \alpha\eta(\eta - 1)P_V P_U(z_k) + (1 - \alpha\eta)(\eta - 1)P_V(z_k - z) + P_V(\alpha\eta(1 - \eta)z) \\ &= (\eta - 1)(\alpha\eta P_V P_U(z_k) + (1 - \alpha\eta)P_V(z_k)) + (1 - \eta)P_V(z) \\ &= (\eta - 1)P_V(z_{k+1} - z), \end{aligned}$$

which proves the second equation in (3.39) for $k + 1$. Finally, by (3.43), (3.39) and (3.41), we have that

$$\begin{aligned} P_U(z_{k+1}) &= \alpha\eta^2 P_U P_V P_U(x_k + z) + \alpha\eta P_U P_V(-x_k + (1 - \eta)z) + (1 - \alpha\eta)P_U(x_k + z) \\ &= (\alpha\eta^2 P_U P_V P_U - \alpha\eta P_U P_V + (1 - \alpha\eta)P_U)(x_k) \\ &\quad + \alpha(\eta^2 P_U P_V P_U + \eta(1 - \eta)P_U P_V - \eta P_U)(z) + P_U(z) = P_U(x_{k+1} + z), \end{aligned}$$

which proves the first equation in (3.39) for $k + 1$ and completes the proof. \square

Theorem 3.34 shows that, for subspaces, the shadow sequences of GAP and AAMR coincide when $\alpha_1 = \alpha_2 = 2\beta$ and the starting point of AAMR is chosen as $x_0 = 0$; see Figure 3.18(a) for a simple example in \mathbb{R}^2 . This is not the case for general convex sets, as shown in Figure 3.18(b).



(a) U and V are two lines in \mathbb{R}^2 . Then $P_U(x_k + z) = P_U(z_k)$ and these converge to $P_{U \cap V}(z)$. We also represent $w_k := (2\beta - 1)(z_k - z)$

(b) B is a line but A is a halfspace. The sequence $\{x_k + z\}_{k=0}^{\infty}$ converges to $P_{A \cap B}(z)$, while $\{z_k\}_{k=0}^{\infty}$ only converges to some point in $A \cap B$

FIGURE 3.18: Graphical representation of Theorem 3.34 for two subspaces (a) and failure of the result for two arbitrary closed and convex sets (b). The sequence $\{x_k + z\}_{k=0}^{\infty}$ is generated by AAMR with $x_0 = 0$, while the sequence $\{z_k\}_{k=0}^{\infty}$ is generated by GAP with $z_0 = z$.

3.2.3 Computational experiments

In this section we demonstrate the theoretical results obtained in the previous sections with two different numerical experiments. In both of them we considered randomly generated subspaces U and V in \mathbb{R}^{50} with $U \cap V \neq \{0\}$.

3.2.3.1 Rate of convergence with fixed parameters

The purpose of our first computational experiment is to exhibit the piecewise expression of the convergence rate $\gamma(T_{\alpha,\beta})$ given in Theorem 3.28. To this aim, we generated 500 pairs of random subspaces. For each pair of subspaces, we chose 10 random starting points with $\|x_0\| = 1$. Then, for each of these instances, we ran the AAMR method with $\alpha = 0.8$ and $\beta \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. The algorithm was stopped when the shadow sequence satisfies

$$\|P_U(x_k + x_0) - P_{U \cap V}(x_0)\| < \epsilon := 10^{-8}, \quad (3.45)$$

where $(x_k)_{k=0}^{+\infty}$ is the sequence iteratively defined by (3.3) with $z = x_0$. According to Corollary 3.31, for any $\mu \in]\gamma(T_{\alpha,\beta}), 1[$, the left-hand side of (3.45) is bounded by μ^k for n big enough. Therefore, an estimate of the maximum number of iterations is given by

$$\frac{\log \epsilon}{\log \gamma(T_{\alpha,\beta})}. \quad (3.46)$$

The results are shown in Figure 3.19, where the points represent the number of iterations required by AAMR to satisfy (3.45), and the lines correspond to the estimated upper bounds given by (3.46). We clearly observe that the algorithm behaves in accordance with the theoretical rates. We emphasize the fact that (3.46) is expected to be a good upper bound on the number of iterations only when this number is *sufficiently* large. We can indeed find a few instances in the plot, especially those which require a small number of iterations, exceeding its estimated upper bound.

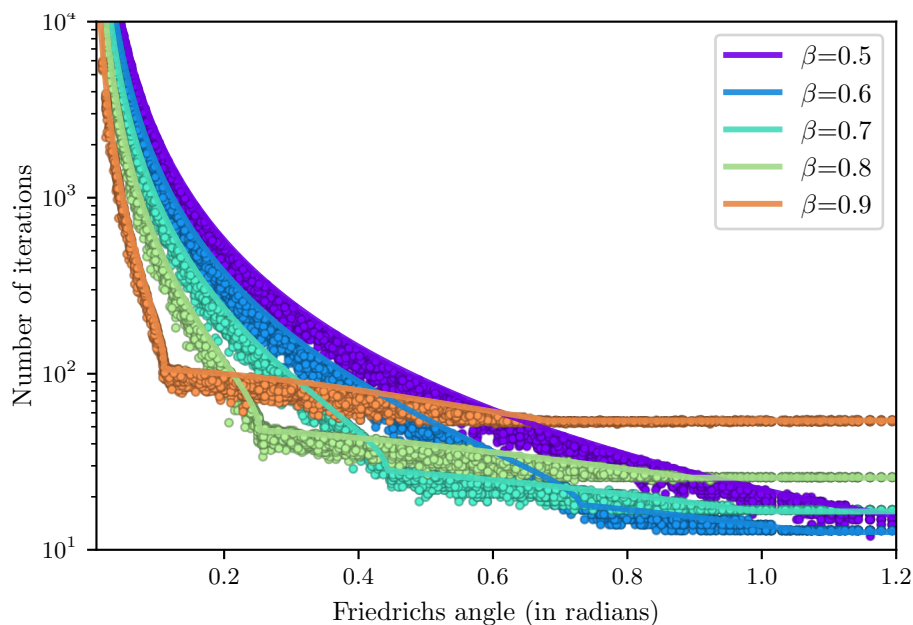


FIGURE 3.19: Number of iterations required to converge for the AAMR algorithm with $\alpha = 0.8$ and five different values of the parameter β , with respect to the Friedrichs angle. The lines correspond to the approximate upper bounds given by (3.46) and the theoretical rates (3.32).

3.2.3.2 Behavior of the algorithms with optimal parameters

In our second experiment we compare the performance of AP, SP, RAP, DR and AAMR, when their parameters are selected to be optimal (see Table 3.1). For 100 pairs of subspaces, we generated 50 random starting points with $\|x_0\| = 1$. For a fair comparison, we monitored the shadow sequence for all the algorithms. We also used the stopping criterion (3.45), with $\epsilon = 10^{-8}$. The results of this experiment are summarized in Figure 3.20, where we show in three different graphics the median, the difference between the maximum and the median, and the coefficient of variation of the number of iterations needed to converge for each pair of subspaces.

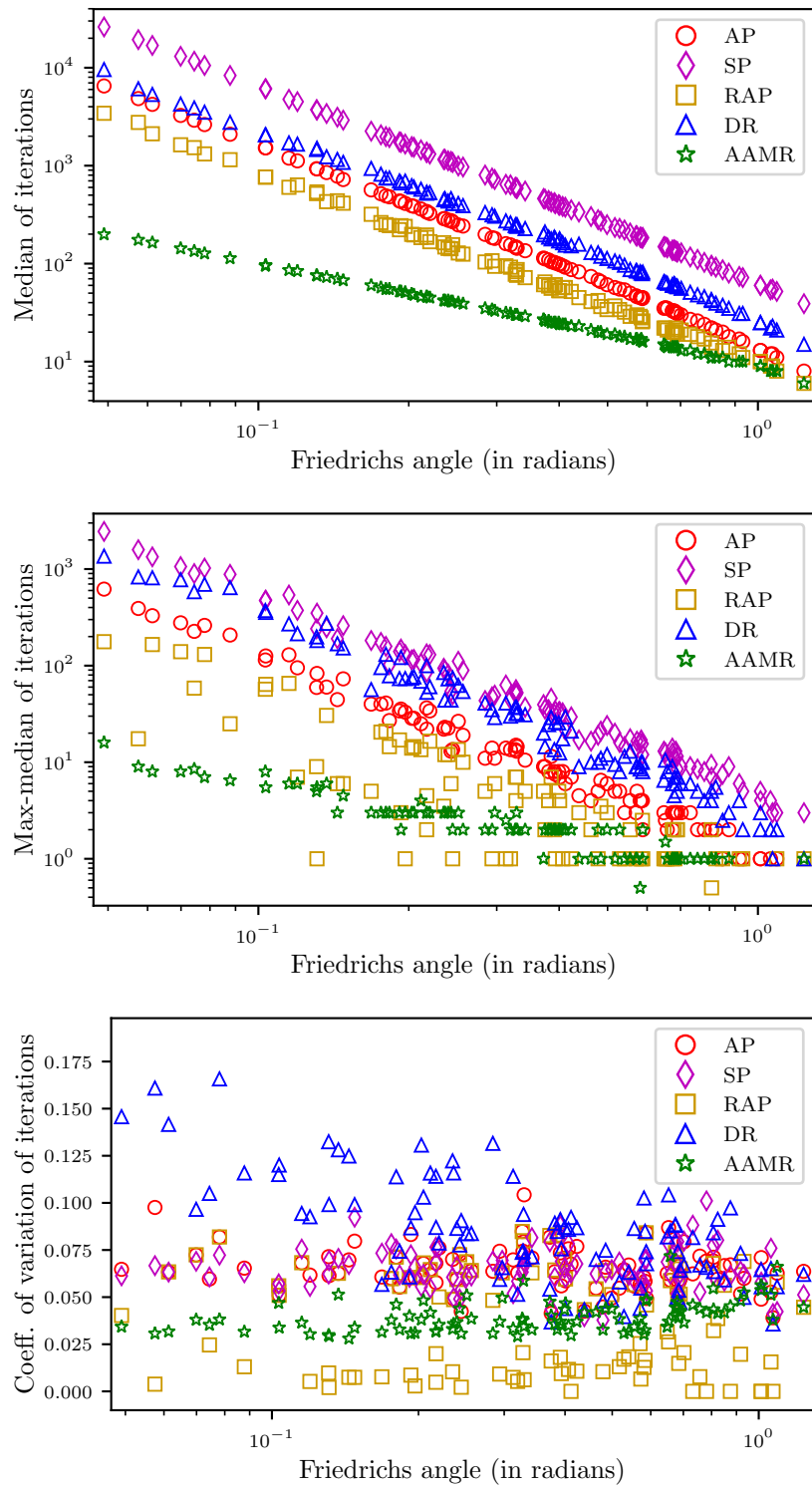


FIGURE 3.20: Median, difference between the maximum and the median, and coefficient of variation of the required number of iterations with respect to the Friedrichs angle of AP, SP, RAP, DR and AAMR for their respective optimal parameters.

As expected, since the rate of convergence of AAMR is the smallest amongst all the compared methods, this algorithm is clearly the fastest, particularly for small angles. Moreover, we can observe that AAMR is one of the most robust methods (together with RAP), which makes the median to be a good representative of the rate of convergence.

REMARK 3.35. In a more practical context, the AAMR algorithm has been recently employed in [32] to solve a continuous-time optimal control problem, under the name *Aragón Artacho–Campoy (AAC) algorithm*. Their numerical results show a very good performance of the algorithm, compared to the other methods considered. Moreover, the optimal parameters in their tests are in accordance with that stated in Theorem 3.33, since the best choice was $\alpha = 1$ and some $\beta \in [0.5, 1[$ depending on the problem.

3.3 Extension to monotone operator theory

The AAMR algorithm has been shown to be a modification of the Douglas–Rachford algorithm, allowing to solve best approximation problems rather than just feasibility ones. As explained in Section 2.2.2.5, the DR algorithm can be generally applied to maximally monotone operators, where it solves the sum problem (2.22). The objective of this section is to extend the AAMR scheme to this more general context. As explained in Remark 2.24, the generalized version of a best approximation problem (3.1), can be stated as

$$\text{Find } w = J_{A+B}(z), \quad (3.47)$$

for some maximally monotone operators $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ and any $z \in \text{ran}(\text{Id} + A + B)$.

The AAMR method can be naturally extended from the convex feasibility framework to the context of maximally monotone operators by considering resolvents instead of projectors in the AAMR operator (2.19). The iterative scheme is thus given by

$$x_{k+1} := (1 - \alpha)x_k + \alpha(2\beta J_B - \text{Id})(2\beta J_A - \text{Id})(x_k), \quad k = 0, 1, 2, \dots, \quad (3.48)$$

with $\alpha \in]0, 1]$ and $\beta \in]0, 1[$. The Douglas–Rachford splitting algorithm (2.23) can be viewed now as a limiting case of (3.48) when $\beta = 1$.

The analysis of AAMR for monotone operators presented in this section is inspired by the work of Combettes [77], where he proposed the algorithm in (2.32) for computing the resolvent of the sum (see Remark 2.24(i)). Our approach consists in reformulating the AAMR iteration so that it can be viewed as the one generated by the DR splitting algorithm for finding a zero of the sum of an appropriate modification of the operators.

3.3.1 AAMR splitting algorithm for maximally monotone operators

We begin this section with the definition of a *modified reflected resolvent*, which is the natural extension of the modified reflector introduced in Definition 3.2.

Definition 3.36 (Modified reflected resolvent). *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be an operator. Given any $\beta \in]0, 1]$, the operator $2\beta J_A - \text{Id}$ is called a modified reflected resolvent of A .*

Our analysis is mainly based on the connection of the modified reflected resolvent with the classical reflected resolvent of a different operator, which we define next.

Definition 3.37 (Strengthening of an operator). *Given an operator $A : \mathcal{H} \rightrightarrows \mathcal{H}$ and given any $\beta \in]0, 1[$, we define the β -strengthening of A as the operator $A^{(\beta)} : \mathcal{H} \rightrightarrows \mathcal{H}$ defined by*

$$A^{(\beta)}(x) := (A + (1 - \beta)\text{Id}) \left(\frac{x}{\beta} \right), \quad \text{for all } x \in \mathcal{H}.$$

Proposition 3.38. *Let $A : \mathcal{H} \rightrightarrows \mathcal{H}$ be an operator and let $\beta \in]0, 1[$. Then,*

$$J_{A^{(\beta)}} = \beta J_A.$$

Further, A is monotone if and only if $A^{(\beta)}$ is $\frac{1-\beta}{\beta}$ -strongly monotone, and A is maximally monotone if and only if $A^{(\beta)}$ is so.

Proof. The β -strengthening of A can be expressed as

$$A^{(\beta)} = A \circ \left(\frac{1}{\beta} \text{Id} \right) + \frac{1 - \beta}{\beta} \text{Id}. \quad (3.49)$$

Now observe that,

$$\text{ran} (\text{Id} + A^{(\beta)}) = \text{ran} \left((\text{Id} + A) \circ \left(\frac{1}{\beta} \text{Id} \right) \right) = \text{ran} (\text{Id} + A). \quad (3.50)$$

For any $x \in \text{dom } J_{A^{(\beta)}} = \text{dom } J_A$ and $w \in \mathcal{H}$, we have

$$w \in J_{A^{(\beta)}}(x) \Leftrightarrow x \in (\text{Id} + A^{(\beta)})(w) \Leftrightarrow x \in (\text{Id} + A) \left(\frac{w}{\beta} \right) \Leftrightarrow \frac{w}{\beta} \in J_A(x) \Leftrightarrow w \in \beta J_A(x),$$

which proves that $J_{A^{(\beta)}} = \beta J_A$, as claimed.

By Lemma 1.19, A is monotone if and only if $A \circ \left(\frac{1}{\beta} \text{Id} \right)$ is monotone, so the assertion about the $\frac{1-\beta}{\beta}$ -strong monotonicity of $A^{(\beta)}$ directly follows from (3.49). Finally, A is maximally monotone if and only if $A^{(\beta)}$ is so, according to Theorem 1.20 and (3.50). \square

Proposition 3.38 establishes strong monotonicity of $A^{(\beta)}$ when A is monotone. The set of zeros of a strongly monotone operator is known to be at most a singleton (see Proposition 1.29). Hence, the sum of the β -strengthenings of two monotone operators will have at most one zero. In the next result, we characterize this set for any pair of general operators.

Proposition 3.39. *Let $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ be two operators and let $\beta \in]0, 1[$. Then, the set of zeros of the sum of their β -strengthenings $A^{(\beta)}$ and $B^{(\beta)}$ is given by*

$$\text{zer} \left(A^{(\beta)} + B^{(\beta)} \right) = \beta J_{\frac{1}{2(1-\beta)}(A+B)}(0).$$

Consequently, $\text{zer} \left(A^{(\beta)} + B^{(\beta)} \right) \neq \emptyset$ if and only if $0 \in \text{ran} \left(\text{Id} + \frac{1}{2(1-\beta)}(A+B) \right)$.

Proof. For any $x \in \mathcal{H}$, one can easily check that $x \in \text{zer} \left(A^{(\beta)} + B^{(\beta)} \right)$ if and only if

$$\begin{aligned} 0 \in A^{(\beta)}(x) + B^{(\beta)}(x) &\Leftrightarrow 0 \in A \left(\frac{x}{\beta} \right) + B \left(\frac{x}{\beta} \right) + 2(1-\beta) \frac{x}{\beta} \\ &\Leftrightarrow 0 \in \frac{x}{\beta} + \frac{1}{2(1-\beta)}(A+B) \left(\frac{x}{\beta} \right) \Leftrightarrow \frac{x}{\beta} \in J_{\frac{1}{2(1-\beta)}(A+B)}(0), \end{aligned}$$

which is equivalent to $x \in \beta J_{\frac{1}{2(1-\beta)}(A+B)}(0)$ and proves the result. \square

We are ready to prove our main result, which shows that the AAMR method can be applied to compute the resolvent of the sum of two maximally monotone operators.

Theorem 3.40 (AAMR splitting algorithm). *Let $A, B : \mathcal{H} \rightrightarrows \mathcal{H}$ be two maximally monotone operators, let $\gamma > 0$ and let $(\lambda_k)_{k=0}^\infty$ be a sequence in $[0, 1]$. Fix any $\beta \in]0, 1[$ and suppose that $z \in \text{ran} \left(\text{Id} + \frac{\gamma}{2(1-\beta)}(A+B) \right)$. Given any $x_0 \in \mathcal{H}$, set*

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k(2\beta J_{\gamma B-z} - \text{Id})(2\beta J_{\gamma A-z} - \text{Id})(x_k), \quad k = 0, 1, 2, \dots \quad (3.51)$$

Then there exists $x^* \in \text{Fix} \left((2\beta J_{\gamma B-z} - \text{Id})(2\beta J_{\gamma A-z} - \text{Id}) \right)$ such that the following hold.

(a) If $\sum_{k=0}^\infty \lambda_k(1 - \lambda_k) = +\infty$, then

(i) $(x_{k+1} - x_k)_{k=0}^\infty$ converges strongly to 0;

(ii) $(x_k)_{k=0}^\infty$ converges weakly to x^* , and $J_{\gamma A}(z + x^*) = J_{\frac{\gamma}{2(1-\beta)}(A+B)}(z)$.

(b) If $\sum_{k=0}^\infty \lambda_k(1 - \lambda_k) = +\infty$ or $\lambda_k = 1$, for all $k = 0, 1, \dots$, then

$(J_{\gamma A}(z + x_k))_{k=0}^\infty$ converges strongly to $J_{\frac{\gamma}{2(1-\beta)}(A+B)}(z)$.

Proof. Since A and B are maximally monotone, by Lemma 1.19, the operators γA_{-z} and γB_{-z} are also maximally monotone. Thus, in view of Proposition 3.38, the iterative scheme in (3.51) becomes

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k R_{(\gamma B_{-z})^{(\beta)}} R_{(\gamma A_{-z})^{(\beta)}}(x_k), \quad k = 0, 1, 2, \dots,$$

with $(\gamma A_{-z})^{(\beta)}$ and $(\gamma B_{-z})^{(\beta)}$ maximally monotone and $\frac{\beta}{1-\beta}$ -strongly monotone. Now observe that $z \in \text{ran} \left(\text{Id} + \frac{\gamma}{2(1-\beta)}(A+B) \right)$ if and only if there exists $w \in \mathcal{H}$ such that

$$\begin{aligned} z = w + \frac{\gamma}{2(1-\beta)}(A+B)(w) &\Leftrightarrow 0 = w - z + \frac{1}{2(1-\beta)}\gamma(A_{-z} + B_{-z})(w - z) \\ &\Leftrightarrow 0 \in \text{ran} \left(\text{Id} + \frac{1}{2(1-\beta)}(\gamma A_{-z} + \gamma B_{-z}) \right). \end{aligned}$$

Hence, Proposition 3.39 implies

$$\text{zer} \left((\gamma A_{-z})^{(\beta)} + (\gamma B_{-z})^{(\beta)} \right) = \left\{ \beta J_{\frac{\gamma}{2(1-\beta)}(A_{-z} + B_{-z})}(0) \right\} \neq \emptyset. \quad (3.52)$$

We are then in position to apply Theorem 2.14(i)–(ii), which yields the existence of

$$x^* \in \text{Fix} \left(R_{(\gamma B_{-z})^{(\beta)}} R_{(\gamma A_{-z})^{(\beta)}} \right) = \text{Fix} \left((2\beta J_{\gamma B_{-z}} - \text{Id})(2\beta J_{\gamma A_{-z}} - \text{Id}) \right)$$

such that $(x_{k+1} - x_k)_{k=0}^\infty \rightarrow 0$, $(x_k)_{k=0}^\infty \rightarrow x^*$ and

$$J_{(\gamma A_{-z})^{(\beta)}}(x^*) \in \text{zer} \left((\gamma A_{-z})^{(\beta)} + (\gamma B_{-z})^{(\beta)} \right). \quad (3.53)$$

According to Proposition 3.38, together with Lemma 1.27, we have that

$$J_{(\gamma A_{-z})^{(\beta)}}(x) = \beta J_{\gamma A_{-z}}(x) = \beta (J_{\gamma A}(x+z) - z), \quad \text{for all } x \in \mathcal{H}; \quad (3.54)$$

and also by Lemma 1.27,

$$J_{\frac{\gamma}{2(1-\beta)}(A_{-z} + B_{-z})}(0) = J_{\left(\frac{\gamma}{2(1-\beta)}(A+B)\right)_{-z}}(0) = J_{\frac{\gamma}{2(1-\beta)}(A+B)}(z) - z. \quad (3.55)$$

Therefore, by combining (3.52), (3.53), (3.54) and (3.55), we obtain that

$$J_{\gamma A}(z + x^*) = J_{\frac{\gamma}{2(1-\beta)}(A+B)}(z),$$

and thus statement (a) has been proved.

Finally, thanks to the strong monotonicity of $(\gamma A_{-z})^{(\beta)}$, we can use Theorem 2.14(iv) if $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$, or Theorem 2.15 if $\lambda_k = 1$, to get that $(J_{(\gamma A_{-z})^{(\beta)}}(x_k))_{k=0}^{\infty}$ converges strongly to the unique zero of $(\gamma A_{-z})^{(\beta)} + (\gamma B_{-z})^{(\beta)}$. Again, taking into account (3.52), (3.54) and (3.55), this is equivalent to

$$(\beta (J_{\gamma A}(z + x_k) - z))_{k=0}^{\infty} \rightarrow \beta \left(J_{\frac{\gamma}{2(1-\beta)}(A+B)}(z) - z \right),$$

which implies (b) and completes the proof. \square

Let us show in the next result how the convergence of the AAMR method in the convex feasibility setting, stated in Theorem 3.17, can be derived as a direct consequence of Theorem 3.40 when we turn from resolvents to projectors.

Corollary 3.41 (AAMR for best approximation problems). *Let $A, B \subseteq \mathcal{H}$ be nonempty, closed and convex sets. Let $(\lambda_k)_{k=0}^{\infty}$ be a sequence in $[0, 1]$ and fix any $\beta \in]0, 1[$. Given $z \in \mathcal{H}$, choose any $x_0 \in \mathcal{H}$ and consider the sequence defined by*

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k(2\beta P_{B-z} - \text{Id})(2\beta P_{A-z} - \text{Id})(x_k), \quad k = 0, 1, 2, \dots$$

If $A \cap B \neq \emptyset$ and $z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z))$, then the following assertions hold.

(a) *If $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$, then*

(i) *$(x_{k+1} - x_k)_{k=0}^{\infty}$ is strongly convergent to 0;*

(ii) *$(x_k)_{k=0}^{\infty}$ is weakly convergent to a point $x^* \in \text{Fix}((2\beta P_{B-z} - \text{Id})(2\beta P_{A-z} - \text{Id}))$ such that*

$$P_A(z + x^*) = P_{A \cap B}(z).$$

(b) *If $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$ or $\lambda_k = 1$, for all $k = 0, 1, \dots$, then*

$$(P_A(z + x_k))_{k=0}^{\infty} \text{ converges strongly to } P_{A \cap B}(z).$$

Proof. We know from Examples 1.18(ii) and 1.25(ii) that the normal cones N_A and N_B are maximally monotone operators with $J_{N_A} = P_A$ and $J_{N_B} = P_B$. Moreover, it can be easily checked that the normal cones to the displaced sets $A - z$ and $B - z$ coincide with the inner $(-z)$ -perturbations of N_A and N_B , i.e.,

$$N_{(A-z)} = (N_A)_{-z} \quad \text{and} \quad N_{(B-z)} = (N_B)_{-z}.$$

Then, according to Example 1.25(ii), it holds that $J_{(N_A)_{-z}} = P_{A-z}$ and $J_{(N_B)_{-z}} = P_{B-z}$. Now observe that

$$z - P_{A \cap B}(z) \in (N_A + N_B)(P_{A \cap B}(z)) = \frac{1}{2(1-\beta)}(N_A + N_B)(P_{A \cap B}(z)),$$

which implies that $z \in \text{ran} \left(\text{Id} + \frac{1}{2(1-\beta)}(N_A + N_B) \right)$ and

$$P_{A \cap B}(z) = J_{\frac{1}{2(1-\beta)}(N_A + N_B)}(z).$$

Hence, the result follows from applying Theorem 3.40 to N_A and N_B , with $\gamma = 1$. \square

Example 3.42 (Proximity operator of the sum of two functions). *Given two proper lower semicontinuous convex functions $f, g : \mathcal{H} \rightarrow]-\infty, +\infty]$, Theorem 3.40 can be applied to their subdifferentials ∂f and ∂g . Hence, given a point $z \in \mathcal{H}$, this gives rise to a sequence $(x_k)_{k=0}^\infty$ such that*

$$\left(\text{prox}_f(z + x_k) \right)_{k=0}^\infty \rightarrow \text{prox}_{\frac{1}{2(1-\beta)}(f+g)}(z),$$

provided that

$$z \in \text{ran} \left(\text{Id} + \frac{1}{2(1-\beta)}(\partial f + \partial g) \right). \quad (3.56)$$

Note that the latter holds for all $z \in \mathcal{H}$ when $\partial f + \partial g = \partial(f+g)$, so a sufficient condition for (3.56) is

$$0 \in \text{sri}(\text{dom } f - \text{dom } g);$$

see, e.g., [35, Corollary 16.48].

REMARK 3.43. According to Theorem 3.40, the AAMR splitting algorithm solves (3.47), which can be seen as a generalization of the best approximation problem described by two closed and convex sets (Corollary 3.41). However this is not the only possible generalization. Given two convex sets $C_1, C_2 \subseteq \mathcal{H}$, we know by (2.24) that $C_1 \cap C_2 = \text{zer}(N_{C_1} + N_{C_2})$. Hence, an alternative generalization of (3.1) is

$$\text{Find } w = P_{\text{zer}(A+B)}(z), \quad (3.57)$$

with A and B maximally monotone operators and $z \in \mathcal{H}$. In the recent work by Alwadani, Bauschke, Moursi and Wang [5], the authors have complemented our analysis by establishing that the *underlying curve* defined by the AAMR splitting algorithm converges to the closest zero of the sum (see [5, Theorem 3.4]), solving thus problem (3.57).

3.3.2 Parallel AAMR splitting for the resolvent of a finite sum

In this section we discuss how to implement the AAMR scheme to compute the resolvent of a finite sum of maximally monotone operators. Given a collection of r operators $A_i : \mathcal{H} \rightrightarrows \mathcal{H}$, $i = 1, 2, \dots, r$, and $z \in \text{ran}(\text{Id} + \sum_{i=1}^r A_i)$, the problem of interest is now

$$\text{Find } w \in J_{\sum_{i=1}^r A_i}(z). \quad (3.58)$$

According to Proposition 2.16, the product space reformulation is a powerful trick for reducing the problem of finding zeros of the sum of finitely many operators to an equivalent problem involving only two, while keeping their monotonicity properties. As we show next, it turns out to be very useful in our context, where we are interested in computing the resolvent of the sum. The following proposition can be seen as an extension of Lemma 3.25 for maximally monotone operators.

Proposition 3.44. *Let $A_i : \mathcal{H} \rightrightarrows \mathcal{H}$ be maximally monotone operators for $i = 1, 2, \dots, r$. Let \mathcal{H} be the product Hilbert space as in (2.3), and consider the diagonal set $\mathbf{D} \subset \mathcal{H}$ defined in (2.4) and the product operator $\mathbf{A} = A_1 \times \dots \times A_r$ as in (2.25). For any $\mathbf{x} = \mathbf{j}(x) = (x, x, \dots, x) \in \mathbf{D}$, we have*

$$J_{\mathbf{A} + N_{\mathbf{D}}}(\mathbf{x}) = \mathbf{j} \left(J_{\frac{1}{r} \sum_{i=1}^r A_i}(x) \right).$$

Consequently,

$$\text{ran}(\text{Id} + \mathbf{A} + N_{\mathbf{D}}) \cap \mathbf{D} = \mathbf{j} \left(\text{ran} \left(\text{Id} + \frac{1}{r} \sum_{i=1}^r A_i \right) \right).$$

Proof. Fix $\mathbf{x} = (x, x, \dots, x) \in \mathbf{D}$. To prove the direct inclusion, pick any $\mathbf{y} \in J_{\mathbf{A} + N_{\mathbf{D}}}(\mathbf{x})$. Then, we have that

$$\mathbf{x} \in \mathbf{y} + \mathbf{A}(\mathbf{y}) + N_{\mathbf{D}}(\mathbf{y}).$$

This ensures the nonemptiness of $N_{\mathbf{D}}(\mathbf{y})$, and then it necessarily holds that $\mathbf{y} = \mathbf{j}(y) \in \mathbf{D}$, for some $y \in \mathcal{H}$. Moreover, there must exist $\mathbf{u} = (u_1, u_2, \dots, u_r) \in \mathcal{H}$ with $\sum_{i=1}^r u_i = 0$ such that

$$x \in y + A_i(y) + u_i, \quad \text{for all } i = 1, 2, \dots, r.$$

By adding up all these equations and dividing by r , we deduce that $x \in y + \frac{1}{r} \sum_{i=1}^r A_i(y)$, or equivalently, that

$$y \in J_{\frac{1}{r} \sum_{i=1}^r A_i}(x).$$

To prove the reverse inclusion, take any $\mathbf{y} = \mathbf{j}(y)$ with $y \in J_{\frac{1}{r} \sum_{i=1}^r A_i}(x)$. Then, for each $i = 1, 2, \dots, r$, there exists $a_i \in A_i(y)$ such that

$$x = y + \frac{1}{r} \sum_{i=1}^r a_i \Leftrightarrow r(x - y) - \sum_{i=1}^r a_i = 0 \Leftrightarrow \sum_{i=1}^r (x - y - a_i) = 0.$$

Set $\mathbf{a} := (a_1, a_2, \dots, a_r)$ and $\mathbf{u} := (u_1, u_2, \dots, u_r)$, where $u_i := x - y - a_i$, for each $i = 1, 2, \dots, r$. By construction, we get that $\mathbf{x} = \mathbf{y} + \mathbf{a} + \mathbf{u}$, with $\mathbf{a} \in \mathbf{A}(\mathbf{y})$ and $\mathbf{u} \in N_{\mathbf{D}}(\mathbf{y})$. This implies $\mathbf{y} \in J_{\mathbf{A} + N_{\mathbf{D}}}(\mathbf{x})$ and completes the proof. \square

Thanks to Proposition 3.44, problem (3.58) can be fitted within the framework of Theorem 3.40, allowing us to derive the following parallel splitting algorithm.

Theorem 3.45 (Parallel AAMR splitting algorithm). *Let $A_i : \mathcal{H} \rightrightarrows \mathcal{H}$ be maximally monotone operators for $i = 1, 2, \dots, r$, let $\gamma > 0$ and let $(\lambda_k)_{k=0}^{\infty}$ be a sequence in $[0, 1]$. Fix any $\beta \in]0, 1[$ and suppose that $z \in \text{ran} \left(\text{Id} + \frac{\gamma}{2r(1-\beta)} \sum_{i=1}^r A_i \right)$. Given r arbitrary points $x_{1,0}, x_{2,0}, \dots, x_{r,0} \in \mathcal{H}$, set*

$$\begin{aligned} & \text{for } k = 0, 1, 2, \dots : \\ & \left[\begin{array}{l} p_k = \frac{1}{r} \sum_{i=1}^r x_{i,k}, \\ \text{for } i = 1, 2, \dots, r : \\ \quad \left[\begin{array}{l} x_{i,k+1} = (1 - \lambda_k)x_{i,k} + \lambda_k (2\beta J_{\gamma(A_i)_{-z}} - \text{Id}) (2\beta p_k - x_{i,k}). \end{array} \right. \end{array} \right. \end{aligned} \quad (3.59)$$

Then the following hold.

(a) If $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$, then

- (i) $(x_{i,k+1} - x_{i,k})_{k=0}^{\infty}$ converges strongly to 0, for all $i = 1, 2, \dots, r$;
- (ii) $(x_{i,k})_{k=0}^{\infty}$ converges weakly to $x_i^* \in \mathcal{H}$, for all $i = 1, 2, \dots, r$, and

$$z + \frac{1}{r} \sum_{i=1}^r x_i^* = J_{\frac{\gamma}{2r(1-\beta)} \sum_{i=1}^r A_i}(z).$$

(b) If $\sum_{k=0}^{\infty} \lambda_k(1 - \lambda_k) = +\infty$ or $\lambda_k = 1$, for all $k = 0, 1, \dots$, then

$$(z + p_k)_{k=0}^{\infty} \text{ converges strongly to } J_{\frac{\gamma}{2r(1-\beta)} \sum_{i=1}^r A_i}(z).$$

Proof. Let \mathcal{H} be the product Hilbert space as in (2.3) and let \mathbf{D} be the diagonal set defined in (2.4). Consider the product operator \mathbf{A} defined as in (2.25), and the normal

cone to the diagonal set $N_{\mathbf{D}}$. By Proposition 2.16(i)–(ii), both operators are maximally monotone. For each $k = 0, 1, 2, \dots$, set

$$\mathbf{x}_k := (x_{1,k}, x_{2,k}, \dots, x_{r,k}) \in \mathcal{H} \quad \text{and} \quad \mathbf{p}_k := \mathbf{j}(p_k) \in \mathbf{D}.$$

Observe that $\mathbf{p}_k = P_{\mathbf{D}}(\mathbf{x}_k) = J_{N_{\mathbf{D}}}(\mathbf{x}_k)$. Further, set $\mathbf{z} := \mathbf{j}(z)$ and note that, since \mathbf{D} is a linear subspace and $\mathbf{z} \in \mathbf{D}$, we have

$$N_{\mathbf{D}} = N_{\mathbf{D}-\mathbf{z}} = (N_{\mathbf{D}})_{-\mathbf{z}} = (\gamma N_{\mathbf{D}})_{-\mathbf{z}}.$$

Therefore, the iterative scheme in (3.59) can be expressed as

$$\mathbf{x}_{k+1} = (1 - \lambda_k)\mathbf{x}_k + \lambda_k (2\beta J_{\gamma \mathbf{A}-\mathbf{z}} - \text{Id}) (2\beta J_{\gamma(N_{\mathbf{D}})_{-\mathbf{z}}} - \text{Id}) (\mathbf{x}_k), \quad \text{for } k = 0, 1, 2, \dots$$

According to Proposition 3.44, we have that

$$z \in \text{ran} \left(\text{Id} + \frac{\gamma}{2r(1-\beta)} \sum_{i=1}^r A_i \right) \Leftrightarrow z \in \text{ran} \left(\text{Id} + \frac{\gamma}{2(1-\beta)} (\mathbf{A} + N_{\mathbf{D}}) \right),$$

and

$$J_{\frac{\gamma}{2(1-\beta)}(\mathbf{A}+N_{\mathbf{D}})}(\mathbf{z}) = \mathbf{j} \left(J_{\frac{\gamma}{2r(1-\beta)} \sum_{i=1}^r A_i}(z) \right).$$

Finally, note that the shadows can be expressed, for any $\mathbf{x} = (x_1, x_2, \dots, x_r) \in \mathcal{H}$, as

$$J_{\gamma N_{\mathbf{D}}}(\mathbf{z} + \mathbf{x}) = P_{\mathbf{D}}(\mathbf{z} + \mathbf{x}) = \mathbf{j} \left(z + \frac{1}{r} \sum_{i=1}^r x_i \right).$$

In particular, $J_{\gamma N_{\mathbf{D}}}(\mathbf{z} + \mathbf{x}_k) = \mathbf{j}(z + p_k)$. Hence, the result follows by applying Theorem 3.40 to \mathbf{A} and $N_{\mathbf{D}}$. \square

REMARK 3.46. As done in Corollary 3.41, if the operators involved in Theorem 3.45 are chosen to be the normal cones to r closed and convex sets $C_1, C_2, \dots, C_r \subseteq \mathcal{H}$, and z is a point in \mathcal{H} satisfying

$$z - P_{\bigcap_{i=1}^r C_i}(z) \in \sum_{i=1}^r N_{C_i} (P_{\bigcap_{i=1}^r C_i}(z)),$$

we can deduce the convergence of the parallel AAMR algorithm established in Theorem 3.26 for finding the projection of the point z onto the intersection of finitely many sets.

3.3.2.1 An alternative parallel splitting

Recall that the AAMR method for two operators (Theorem 3.40) has been shown to be, in essence, a Douglas–Rachford iteration for finding a zero of the sum of the β -strengthenings. The following result is a generalization of Proposition 3.39, and characterizes the set of zeros of the sum of the β -strengthenings of a finite collection of operators. The proof is completely analogous so it is omitted.

Proposition 3.47. *Let $A_i : \mathcal{H} \rightrightarrows \mathcal{H}$ be some operators for $i = 1, 2, \dots, r$ and fix any $\beta \in]0, 1[$. Then, the set of zeros of the sum of their β -strengthenings is given by*

$$\text{zer} \left(\sum_{i=1}^r A_i^{(\beta)} \right) = \beta J_{\frac{1}{r(1-\beta)} \sum_{i=1}^r A_i} (0).$$

Consequently, $\text{zer} \left(\sum_{i=1}^r A_i^{(\beta)} \right) \neq \emptyset$ if and only if $0 \in \text{ran} \left(\text{Id} + \frac{1}{r(1-\beta)} \sum_{i=1}^r A_i \right)$.

In view of the previous proposition, we derive the following alternative splitting algorithm for computing the resolvent of a finite sum of maximally monotone operators.

Theorem 3.48 (Alternative parallelized AAMR-like splitting algorithm). *Let $A_i : \mathcal{H} \rightrightarrows \mathcal{H}$ be maximally monotone operators for $i = 1, 2, \dots, r$, let $\gamma > 0$ and let $(\lambda_k)_{k=0}^\infty$ be a sequence in $[0, 1]$ such that $\sum_{k=0}^\infty \lambda_k(1 - \lambda_k) = +\infty$. Let $\beta \in]0, 1[$ and suppose that $z \in \text{ran} \left(\text{Id} + \frac{\gamma}{r(1-\beta)} \sum_{i=1}^r A_i \right)$. Given $x_{1,0}, x_{2,0}, \dots, x_{r,0} \in \mathcal{H}$, set*

$$\begin{aligned} & \text{for } k = 0, 1, 2, \dots : \\ & \left[\begin{array}{l} p_k = \frac{1}{r} \sum_{i=1}^r x_{i,k}, \\ \text{for } i = 1, 2, \dots, r : \\ \quad \left[\begin{array}{l} x_{i,k+1} = (1 - \lambda_k)x_{i,k} + \lambda_k (2\beta J_{\gamma(A_i)_{-z}} - \text{Id}) (2p_k - x_{i,k}). \end{array} \right. \end{array} \right. \end{aligned} \quad (3.60)$$

Then the following hold:

- (i) $(x_{i,k+1} - x_{i,k})_{k=0}^\infty$ converges strongly to 0, for all $i = 1, 2, \dots, r$;
- (ii) $(x_{i,k})_{k=0}^\infty$ converges weakly to $x_i^* \in \mathcal{H}$, for all $i = 1, 2, \dots, r$, and

$$z + \frac{1}{\beta r} \sum_{i=1}^r x_i^* = J_{\frac{\gamma}{r(1-\beta)} \sum_{i=1}^r A_i} (z);$$

- (iii) $\left(z + \frac{1}{\beta} p_k \right)_{k=0}^\infty$ converges strongly to $J_{\frac{\gamma}{r(1-\beta)} \sum_{i=1}^r A_i} (z)$.

Proof. After rewriting the iterative scheme in (3.60) as

$$\mathbf{x}_{k+1} = (1 - \lambda_k)\mathbf{x}_k + \lambda_k R_{(\gamma\mathbf{A}-z)^{(\beta)}} R_{N_{\mathcal{D}}}(\mathbf{x}_k), \quad \text{for } k = 0, 1, 2, \dots, \quad (3.61)$$

repeat the proof of Theorem 3.40, using Proposition 3.47 together with Proposition 2.16(iii) instead of Proposition 3.39. The convergence is thus derived from Theorem 2.14. \square

REMARK 3.49. Two comments concerning Theorem 3.48 are provided next.

- (i) Observe that the parallel AAMR-like splitting algorithm in (3.60) differs from the one in (3.59). Particularly, we can derive an algorithm for projecting onto the intersection of finitely many sets which is different from the one stated in Theorem 3.26. The new algorithm is illustrated in Figure 3.21 (compare it with Figure 3.7).
- (ii) Unlike in Theorem 3.45(b), the sequence $(\lambda_k)_{k=0}^{\infty}$ is not allowed to be constantly equal to 1 in Theorem 3.48(iii). The reason is that the first resolvent of the DR-like iteration in (3.61) is computed with respect to $N_{\mathcal{D}}$, rather than $(N_{\mathcal{D}})^{(\beta)}$. Then, since $N_{\mathcal{D}}$ is not strongly monotone we cannot apply Theorem 2.15, which is the result that covers the case $\lambda_k = 1$. This could be fixed by reverting the order of the reflected resolvents. However, in that case the shadow sequence $(J_{(\gamma\mathbf{A}-z)}(\mathbf{x}_k))_{k=0}^{\infty} \not\subset \mathcal{D}$, and thus, it cannot be identified with a sequence in the original space \mathcal{H} as in (3.60).

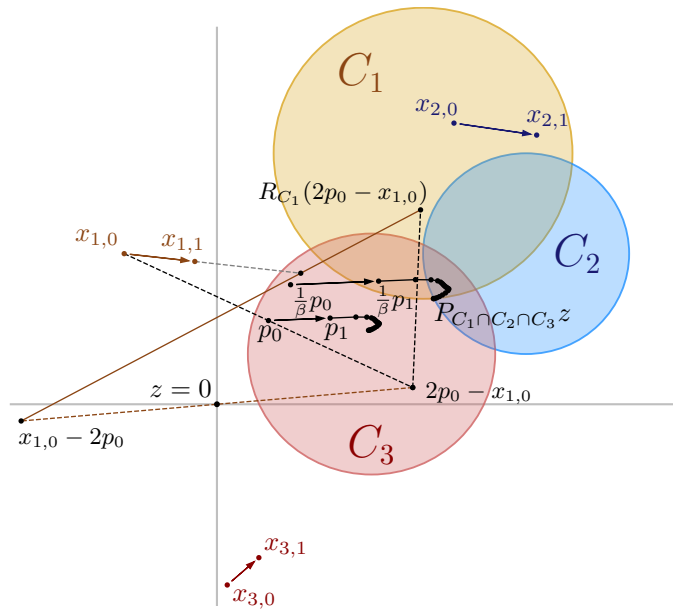


FIGURE 3.21: Illustration of the AAMR iterative scheme for many sets in Theorem 3.48.

Chapter 4

Solving combinatorial problems with the Douglas–Rachford algorithm

In this chapter we present some well-known combinatorial problems which, after appropriately reformulating them as feasibility problems of the form (2.1), can be successfully tackled with the Douglas–Rachford algorithm introduced in Section 2.2.2. Namely, those applications consist of the so-called *graph coloring problem* and the construction of *combinatorial designs of circulant type*.

The combinatorial nature of the underlying problems yields the nonconvexity of some of the sets defining the feasibility models. We have then no convergence guarantees since the general convex theory cannot be applied. Moreover, none of the feasibility problems presented in this chapter can be cast into any of the particular nonconvex settings described in Section 2.2.2.6, for which global convergence of the DR iteration is proved; while local convergence of the algorithm is rather useless when addressing a combinatorial problem, as the algorithm is never run locally in practice. Nonetheless, as pointed out in that section, the method has already been successfully applied to nonconvex (specially combinatorial) problems. This is also the case for the problems in this chapter, as we shall show by an extensive numerical experimentation.

The goal of this chapter is twofold. Firstly, we present the DR algorithm as a successful heuristic for solving the considered problems, something that one might find interesting by itself. Secondly, a better understanding of the behavior of the method onto certain nonconvex scenarios is acquired. This can provide some insights and play a fundamental role in the development of new convergence results of the algorithm in nonconvex settings.

The chapter is structured in two sections. In Section 4.1 we focus on a wide variety of *graph coloring* problems, where we introduce two different feasibility models of the problem, and collect various numerical experiments that show the good performance of

the DR algorithm when applied to them. Then, in Section 4.2, we model a general *combinatorial design of circulant type* as a feasibility problem, with application to constructing *circulant weighing matrices*, *D-optimal designs* and *double core Hadamard matrices*. New constructions and computational experiments are also provided.

4.1 Graph coloring problems

4.1.1 Introduction

A *graph* $G = (V, E)$ is a collection of points V that are connected by links $E \subset V \times V$. The points are usually known as *nodes* or *vertices* while the links are called *edges*, *arcs* or *lines*. An *undirected graph* is a graph in which the edges have no orientation; that is, the edges are not ordered pairs of vertices but sets of two vertices.

A *proper m -coloring* of an undirected graph G is an assignment of one of m possible colors to each vertex of G such that no two adjacent vertices share the same color. More specifically, given the set of colors $K = \{1, \dots, m\}$, an *m -coloring* of G is a mapping $c : V \mapsto K$, assigning a color to each vertex. We say that c is *proper* if

$$c(i) \neq c(j) \text{ for all } \{i, j\} \in E.$$

An example of proper 3-coloring of the so-called *Petersen graph* is depicted in Figure 4.1. The *graph coloring problem* consists in determining whether it is possible to find a proper m -coloring of the graph G . For a basic reference on graph coloring, see e.g. [123].

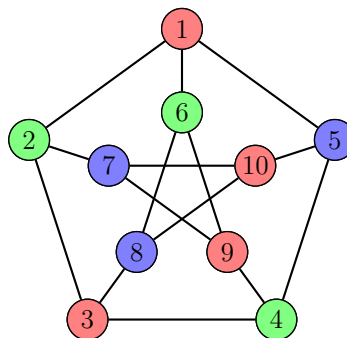


FIGURE 4.1: A 3-coloring of Petersen graph.

Graph coloring has been used in many practical applications such as timetabling and scheduling [135], computer register allocation [70, 71], radio frequency assignment [112], and printed circuit board testing [106]. The graph coloring problem was proved to be

NP-complete [127], so it is reasonable to believe that no polynomial-time exact algorithm solving these problems can be found. For this reason, a wide variety of heuristics and approximation algorithms have been developed. For basic references on graph coloring algorithms and applications, see the surveys [102, 148], or the more recent monograph [139].

In this section we show that the Douglas–Rachford algorithm can be successfully used as a heuristic for solving a wide variety of graph coloring problems when they are conveniently modeled as feasibility problems. This is not the first time that the DR scheme has been used to solve graph coloring problems. It was first employed by Elser et al. in [94], where the authors showed the good performance of the algorithm for edge-colorings, in particular, colorings that avoid monochromatic triangles. As far as we know, this is the only instance of a graph coloring problem whose solution with the Douglas–Rachford algorithm has been studied in the literature. Further, the graph coloring problem considered in [94] is a very specific problem dealing with the coloring of the edges of a complete graph, while we consider any possible graph, and we study node-coloring problems instead of edge-coloring ones.

Before presenting how to reformulate various problems as graph coloring problems, let us shortly summarize some basic concepts of graph theory. A *complete graph* is an undirected graph in which every pair of nodes is connected by an edge. A *clique* is a subset of vertices of an undirected graph such that its induced graph is complete. A *maximal clique* is a clique that cannot be extended by adding one more vertex. A *path* is a sequence of edges that connects a sequence of distinct vertices. A path is said to be a *cycle* if there is an edge from the last vertex in the path to the first one.

Example 4.1 (Formulating 3-SAT as 3-coloring). A Boolean variable *takes logical values: True (T) or False (F)*. A *literal is either a variable or its negation (\neg)*. A *clause is a disjunction (\vee) of literals*. A *formula in conjunctive normal form is a conjunction (\wedge) of clauses*. Given a formula in conjunctive normal form with 3 literals per clause, the 3-SAT (3-satisfiability) problem consists in determining if there exists an assignment of variables that makes the formula true. Specifically, let x_1, \dots, x_n be n Boolean variables and consider m clauses $\theta_1, \dots, \theta_m$, where each clause is the disjunction of 3 literals,

$$\theta_j = t_1^j \vee t_2^j \vee t_3^j, \quad \text{for all } j = 1, 2, \dots, m;$$

with $t_1^j, t_2^j, t_3^j \in \bigcup_{i=1}^n \{x_i, \neg x_i\}$. Let ϕ be the formula comprising the conjunction of all the clauses:

$$\phi = \theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_m.$$

Then, the 3-SAT problem consists in determining if there exists an assignment of the variables that makes the formula ϕ true. Consider for instance the following 3-SAT problem with 3 variables and 2 clauses:

$$\phi = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3). \quad (4.1)$$

There are several solutions to ϕ such as (F, T, F) , (T, T, F) or (F, F, T) , among others.

A 3-SAT problem can be reduced to a 3-coloring problem by using gadgets. A gadget is a small graph whose coloring solves some part of the problem. Using a set of gadgets and connecting them in an appropriate manner, the 3-coloring problem of the full graph can be made equivalent to solving the 3-SAT problem. We start by creating $n + 1$ gadgets, one for each variable and an additional one for setting the interpretation of the colors.

- (a) Create a gadget formed by a complete graph with 3 “color-meaning” nodes named T , F and G , see Figure 4.2(a). As this gadget is a complete graph, a different color must be assigned to each node. The color assigned to node T will be interpreted as True, the color assigned to F as False, and the remaining color assigned to G (ground node) will not have any special interpretation.
- (b) For each variable x_i , construct a gadget with 2 connected nodes, one associated to x_i and the other to $\neg x_i$. Link both of them to the node G to create a gadget of the form in Figure 4.2(b). This gadget forces a logical choice in the value of the variables. Thus, every variable will be assigned to either T or F , and the assignment of every variable and its complement will be consistent.

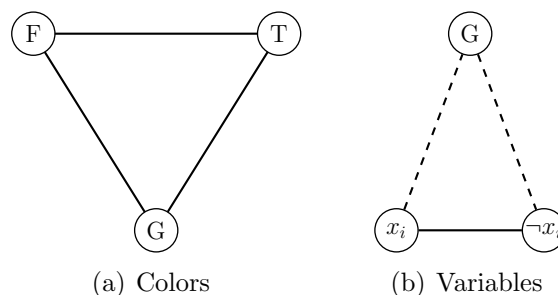


FIGURE 4.2: Gadgets of the variables and colors.

We present next two different formulations of the gadgets corresponding to the clauses.

(c) For the 4-nodes formulation¹, take each clause $\theta = t_1 \vee t_2 \vee t_3$ and create the gadget in Figure 4.3(a) with the nodes associated to t_1, t_2, t_3, F, G , and 4 new nodes. The new unlabeled nodes do not have any special meaning, but, by the construction of the gadgets, every 3-coloring of a clause gadget will assign the same color as T to at least one of the literals t_1, t_2 or t_3 . Thus, a valid 3-coloring of the gadget will make the corresponding clause to be True.

For the 5-nodes formulation², the process is similar but introduces five new nodes instead of four: the gadget is shown in Figure 4.3(b).

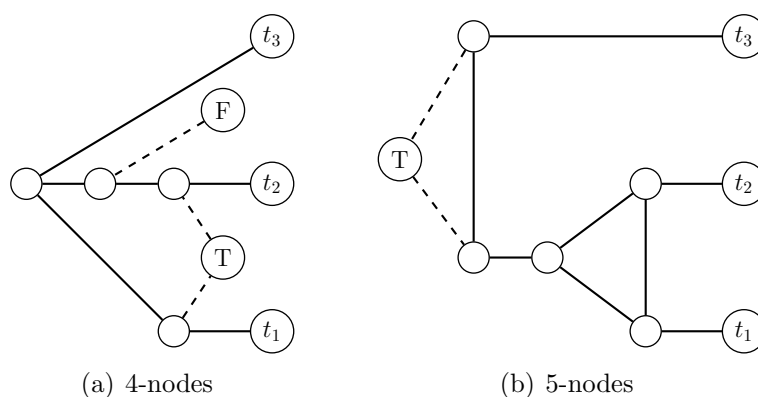


FIGURE 4.3: Gadgets of the clauses.

(d) Finish building the graph by connecting the clause gadgets together using the edges from the common gadgets from Figure 4.2. Full graphs for the four and five node formulations of the 3-SAT problem in (4.1) are shown in Figure 4.4.

The graph resulting from putting all these gadgets together in the 4-nodes formulation has a total of $3 + 2n + 4m$ nodes and $3 + 3n + 9m$ edges. Observe that the graph has $n + 1$ maximal cliques with 3 nodes, one for each gadget of type (a) and (b). In the 5-nodes formulation, the resulting graph has a total of $3 + 2n + 5m$ nodes and $3 + 3n + 10m$ edges. The number of maximal cliques with 3 nodes has increased up to $n + 1 + 2m$, one for each gadget of type (a) and (c) and two for each gadget of type (b). A 3-coloring of the graph built under one of these two formulations corresponds to a solution of the associated 3-SAT problem. A solution to the 3-SAT problem in (4.1) using both formulations is shown in Figure 4.4.

¹We inquired of various experts in the field about the origin of the 4-nodes gadget, but none of them knew about it. We found it in the class notes prepared by Keith Schwarz [160, p. 27]. As he independently came up with it and we have not been able to find it elsewhere, we believe K. Schwarz is its originator.

²The gadget corresponding to the 5-nodes formulation is well-known and first appeared in [107, Fig. 1].

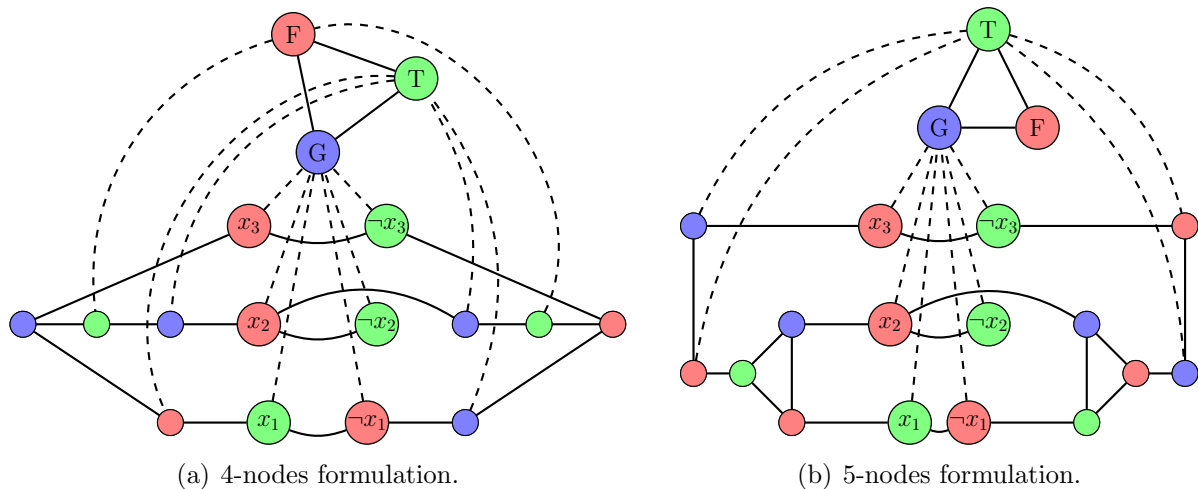


FIGURE 4.4: Two different formulations of the 3-SAT problem in (4.1) as a 3-coloring problem. The same solution of the 3-SAT problem is shown for both formulations.

REMARK 4.2 (Douglas–Rachford for 3-SAT problems). We shall use the DR algorithm to solve 3-SAT instances by reformulating them as a 3-coloring problems, as explained in Example 4.1. With a totally different direct formulation, the Douglas–Rachford method was first shown to be successful for solving 3-SAT problems in [94].

We present next some generalizations or modifications of the graph coloring problem.

4.1.1.1 Precoloring and list coloring problems

In many practical graph coloring problems, the set of eligible colors for each of the nodes can be different. This is the case in the *precoloring problem*, a slight modification of the graph coloring problem in which a subset of the vertices has been preassigned to some colors. The task is to color the remaining vertices to obtain a valid coloring of the entire graph. More generally, in the *list coloring problem*, each vertex can only be colored from a list of admissible colors.

The notion of list coloring was independently introduced by Vizing [173], and Erdős, Rubin and Taylor [97]. Given a graph $G = (V, E)$ and a set of m colors $K = \{1, \dots, m\}$, let $L : V \rightrightarrows K$ be a mapping assigning to each vertex $v \in V$ a list of admissible colors $L(v) \subseteq K$. Thus, the list coloring problem consists in finding a proper coloring of the vertices of the graph G verifying that the color assigned to each vertex belongs to its list of admissible colors; that is,

$$c(i) \neq c(j) \text{ for all } \{i, j\} \in E \quad \text{and} \quad c(i) \in L(i) \text{ for all } i \in V.$$

Note that an ordinary graph coloring problem is a special case of list coloring where $L(i) = K$ for every vertex $i \in V$, and so are the precoloring problems, where the precolored vertices have a list of admissible colors with size one.

List coloring problems can be reduced to standard graph coloring problems. To this aim, one shall add a complete subgraph with m new nodes, each one representing a color in K , and connect each vertex $i \in V$ with the new nodes that represent the colors not belonging to $L(i)$. If we denote by $|A|$ the cardinality of a finite set A , the new graph will have $n + m$ nodes, $l^* = |E| + \frac{m(m-1)}{2} + nm - \sum_{i=1}^n |L(i)|$ edges, and an additional maximal clique of size m . In this way, any valid m -coloring of the extended graph will lead to a solution for the original list coloring problem. An example of such construction is shown in Figure 4.5.

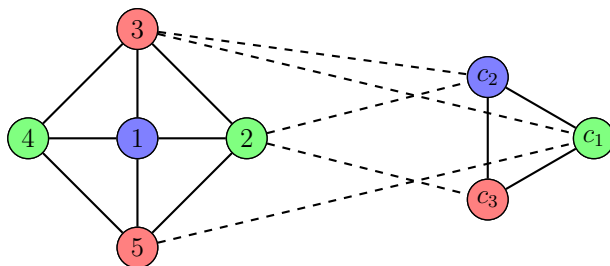


FIGURE 4.5: List coloring reduced to graph coloring of a wheel graph of 5 nodes with admissible colors lists $L(1) = L(4) = \{1, 2, 3\}$, $L(2) = \{1\}$, $L(3) = \{3\}$, and $L(5) = \{2, 3\}$. Nodes c_1 , c_2 and c_3 represent colors 1, 2 and 3, respectively.

Example 4.3 (Formulating Sudokus as 9-precoloring problems). *It is easy to formulate Sudoku puzzles as graph coloring problems. This kind of puzzles consist in a 9×9 grid, divided in nine 3×3 subgrids, with some entries already prefilled. The objective is to fill the remaining cells in such a way that each row, each column and each subgrid contains the digits from 1 to 9 exactly once.*

We shall model Sudokus as 9-precoloring problems, with the aim of applying DR. The construction of the graph is very simple and intuitive. Each cell in the grid shall be represented by a node. Then, we link two nodes if their respective associated cells lay in the same row, same column or same subgrid (see Figure 4.6). The graph obtained contains 81 nodes and 810 edges. Furthermore, a rich maximal clique information is known. Namely, there are 27 maximal cliques of size 9, one per row, one per column and one per subgrid.

We associate a color to each of the 9 digits of the puzzle. Since some cells of the Sudoku are prefilled, this is actually a graph precoloring problem. A valid coloring of the graph will lead to a solution of the Sudoku, as shown in the example in Figure 4.7.

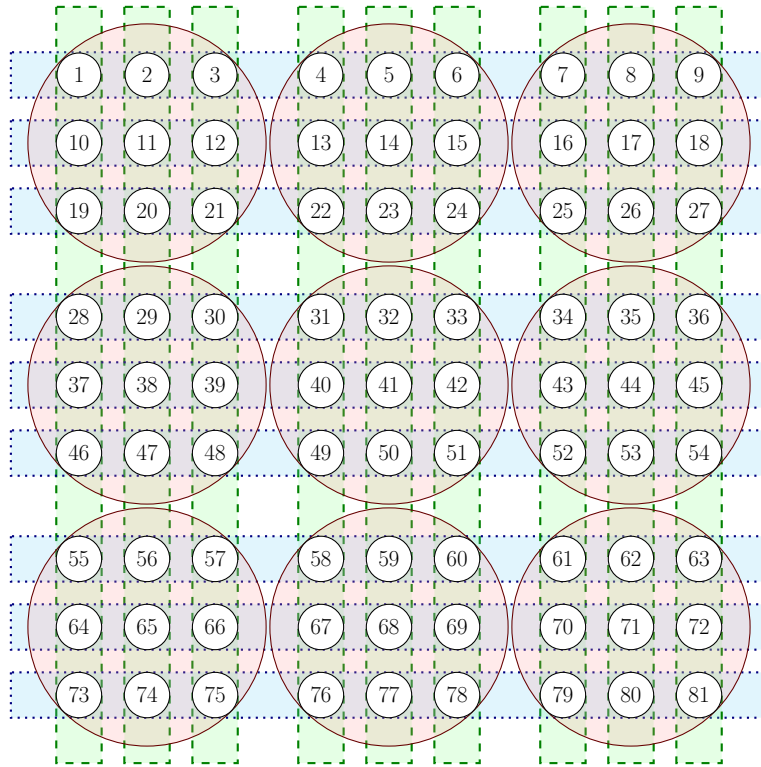


FIGURE 4.6: Graph formulation of a Sudoku, with maximal cliques highlighted.

1				7	9
	4		7	2	
8					
	7		1		6
3					5
	6		4		2
					8
		5	3		7
7	2			4	6

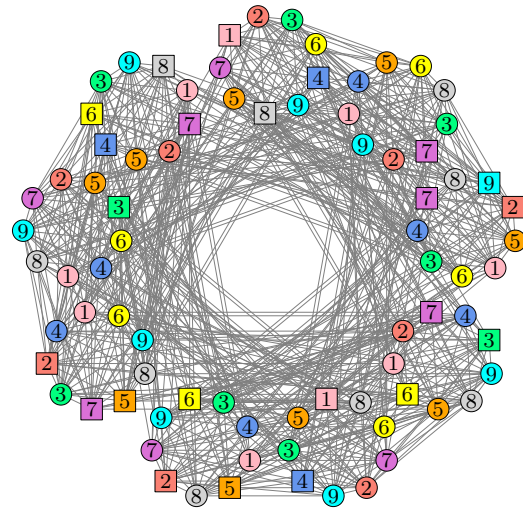


FIGURE 4.7: Unsolved Sudoku puzzle (left) and its graph representation (right): each complete subgraph represents a subgrid and the squared nodes correspond to pre-filled cells. The valid 9-coloring of the graph leads to a solution of the Sudoku.

REMARK 4.4 (A direct feasibility formulation for Sudoku). Sudoku puzzles can be directly modeled as integer feasibility programs as a matrix $A \in \{1, 2, \dots, 9\}^{9 \times 9}$ verifying some constraints. Despite that the Douglas–Rachford algorithm use to fail when tackling these

integer problems, it can be successfully used for solving the puzzles after reformulating them as zero-one programs. Specifically, the matrix A it is reformulated as $B \in \{0, 1\}^{9 \times 9 \times 9}$ where

$$B[i, j, k] := \begin{cases} 1, & \text{if } A[i, j] = k, \\ 0 & \text{otherwise;} \end{cases}$$

see [10, Section 6.2] for a more detailed explanation. We must acknowledge here the fundamental contribution of Elser, Rankenburg and Thibault in [94], who first realized the usefulness of this binary reformulation for the success of the DR algorithm. This formulation will be referred to as the *cubic formulation*.

4.1.1.2 Partial graph coloring

In a *partial graph coloring problem* not all the nodes are necessarily colored. Instead, we require each color in K to be assigned to exactly q nodes. We considered this variant of the problem motivated by the well-known puzzle in the following example.

Example 4.5 (The 8-queens puzzle as a partial graph coloring problem). *The 8-queens puzzle consists in placing eight chess queens on an 8×8 chessboard, so that none of them attack any other. Since a chess queen can be moved any number of squares vertically, horizontally or diagonally, the puzzle's constraints can be formulated as: there is at most one queen at each row, each column and each diagonal. The reformulation of an 8-queens puzzle as a graph coloring problem is similar to the one shown for Sudokus. Each square in the chessboard is represented by a node, and two nodes are linked if their corresponding squares lay on the same column, row or diagonal. The graph has 64 nodes, 728 links and 42 maximal cliques. A partial coloring of this graph, with only one color used $q = 8$ times, leads to a solution of the puzzle (see Figure 4.8).*

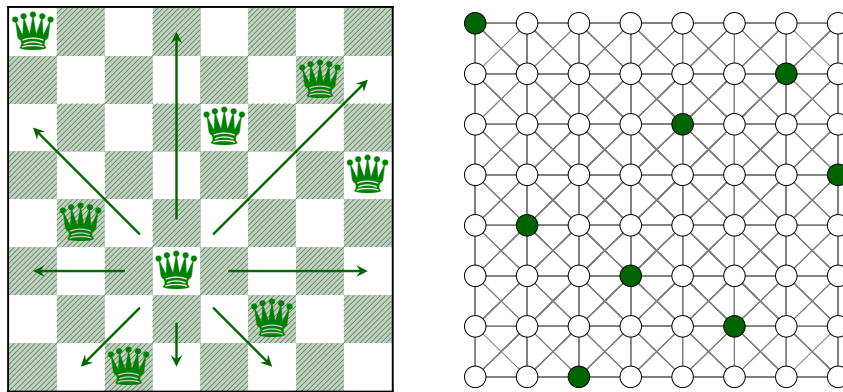


FIGURE 4.8: A solution to the 8-queens puzzle (left) and its graph representation (right).

REMARK 4.6 (Douglas–Rachford for 8-queens puzzles). The use of the Douglas–Rachford algorithm for solving the 8-queens puzzle was already proposed and studied through a direct formulation in [159]. One of the main advantages of formulating these puzzles as graph coloring problems is that it is straightforward to model many variations of the problem. For instance, to model the knights puzzle, a similar puzzle played with knights instead of queens, one only needs to change the links of the chessboard graph, see Figure 4.9(a). Different shapes can also be considered: we show in Figure 4.9(b) a chessboard with a hole, and in Figure 4.9(c) a puzzle dedicated to Jonathan Borwein.

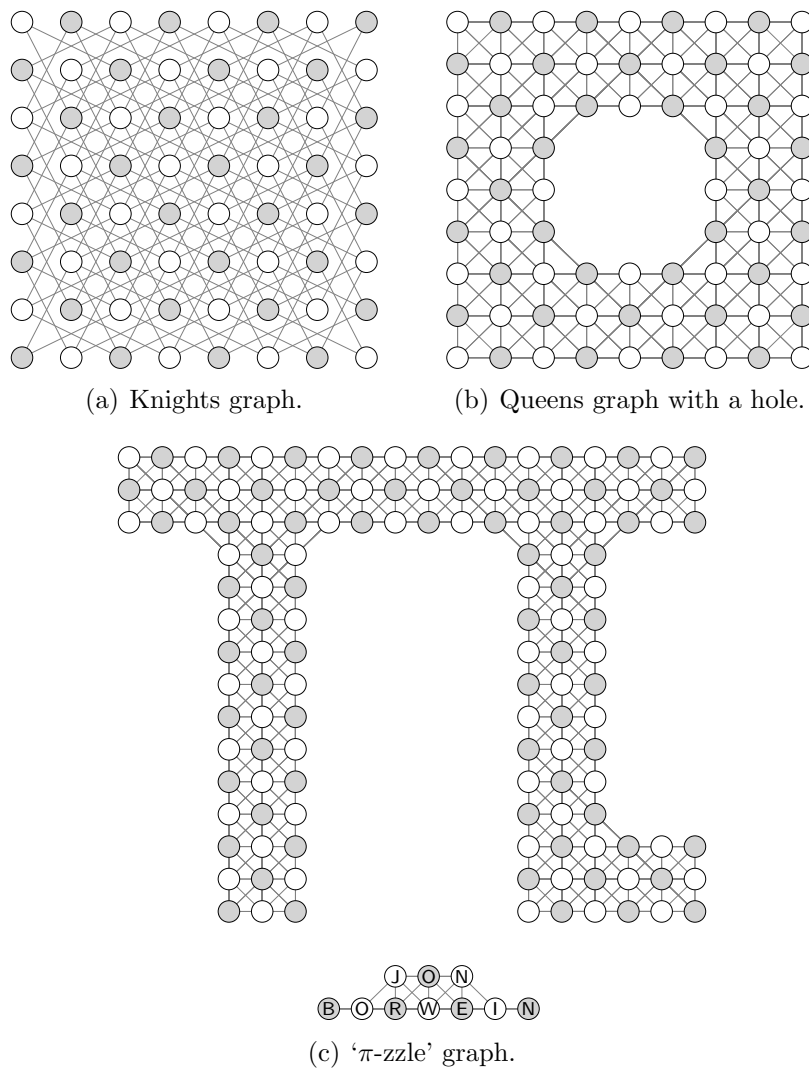


FIGURE 4.9: (a) A 16-knights puzzle with 4 colors: a solution will fill the chessboard. (b) A 10-queens puzzle with 3 colors played in a 9×9 chessboard with a hole. (c) Empty 'π-zzle'. The goal of this puzzle is to place on the board 8 times each of the 18 letters A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, R and W. Ten cells have been prefilled.

REMARK 4.7 (Generalizations of the 8-queens problem). Note that the 8-queens puzzle can be easily posed for any size of the chessboard. The problem has been generalized in many different directions, see [53] for a recent survey. One of these generalizations is the Queens- n^2 puzzle, where one must cover an entire chessboard $n \times n$ with n^2 queens, so that two queens of the same color do not attack each other. This problem is actually a classical graph coloring problem of the queens' chessboard graph.

4.1.1.3 The Hamiltonian path problem

A *Hamiltonian path* is a path in a graph that visits every vertex exactly once. The Hamiltonian path problem consists in determining whether or not such a path exists. A Hamiltonian path can be constructed from a proper coloring of the graph satisfying certain constraints, as we explain next.

Given a graph G with n nodes, our objective will be to find an n -coloring of the graph, where each color $1, 2, \dots, n$ will represent a position in the path. In order to ensure that the coloring represents a valid path, we will impose that two nodes assigned with two consecutive colors must be linked; i.e., for all $i, j \in V$, and for all $k \in K$, it holds that

$$c(i) = k \text{ and } c(j) = k \pm 1 \quad \Rightarrow \quad \{i, j\} \in E.$$

The construction of a Hamiltonian path from such coloring is illustrated in Figure 4.10.

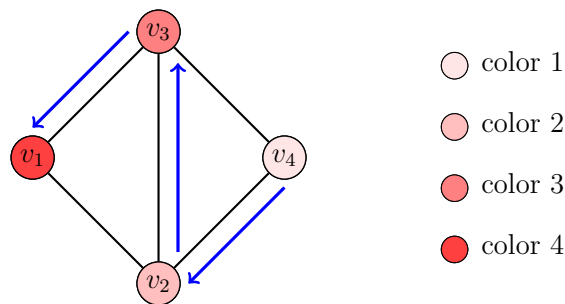


FIGURE 4.10: A Hamiltonian path constructed from a coloring of the graph.

A *Hamiltonian cycle* is a Hamiltonian path that is also a cycle, that is, there is a link connecting the last node in the path and the first one. The problem of finding such a cycle can be cast as a Hamiltonian path problem as we show next.

Given a graph $G = (V, E)$, select any node $v \in V$ and make a copy of it, i.e., create a new node v' that is connected with all nodes linked to v . Then, create another two new nodes t and s , and link t with v and s with v' (see Figure 4.11).

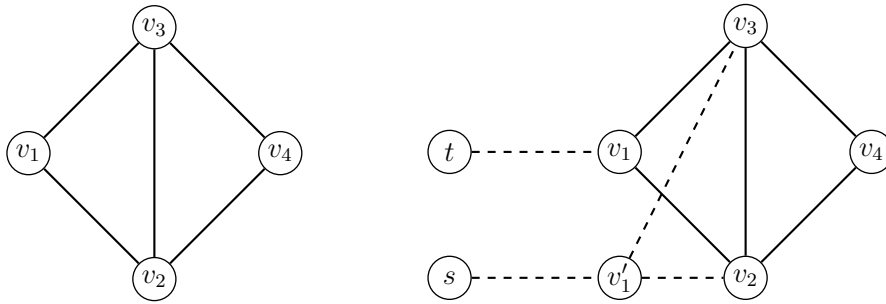


FIGURE 4.11: Hamiltonian cycle reduced to Hamiltonian path.

Since t and s have *degree one* (i.e., they are only linked with another node), every admissible Hamiltonian path in the new graph needs to start in one of these nodes and finish in the other. Thus, after removing t and s , we end up with a path going from v to v' . As these nodes were originally the same, we have actually found a Hamiltonian cycle.

Example 4.8 (The knight’s tour problem). *An example of Hamiltonian path/cycle arises in the knight’s tour problem. The knight’s path problem consists in finding a sequence of moves of a knight on a chessboard such that it visits exactly once every square. If the final position of such a path is one knight’s move away from the starting position of the knight, the path is called a knight’s cycle. Thus, to find a knight’s cycle, one only needs to build the graph corresponding to the knight’s movements on a chessboard shown in Figure 4.9(a), and find a Hamiltonian cycle in the graph. A solution for a 12×12 chessboard is shown in Figure 4.12.*


143	14	127	110	141	108	3	132	139	106	91	134
126	111	142	15	128	77	140	107	4	133	138	105
13		11	76	109	2	67	78	131	92	135	90
10	125	112	1	16	129	22	5	68	79	104	137
113	12	75	8	23	66	69	130	93	136	89	80
124	9	114	17	70	19	6	21	88	63	94	103
115	74	123	24	7	86	65	84	45	102	81	62
122	41	116	73	18	71	20	87	64	83	100	95
117	28	121	42	25	32	85	44	101	46	61	82
40	37	118	27	72	43	56	49	58	53	96	99
29	120	35	38	31	26	33	54	51	98	47	60
36	39	30	119	34	55	50	57	48	59	52	97

FIGURE 4.12: A knight’s cycle on a 12×12 chessboard found with DR.

4.1.2 A feasibility model based on a binary linear program

We present now our first formulation of the graph coloring problem as a feasibility one. This is expressed in terms of binary indicator variables, the same variables that would be used in an integer programming formulation. The model can be easily adapted to cover all the variants and generalizations of the graph coloring problem discussed in the previous section. The good performance of the Douglas–Rachford when applied to the proposed formulations is shown in various computational experiments.

4.1.2.1 Modelling the classical graph coloring problem with binary variables

The m -coloring of a graph $G = (V, E)$ with n nodes shall be modeled as a binary matrix $X = (x_{ik}) \in \{0, 1\}^{n \times m}$, where $x_{ik} = 1$ indicates that vertex i receives color k . Then, we have the following constraints:

$$\sum_{k=1}^m x_{ik} = 1, \quad \text{for all } i = 1, \dots, n; \quad (4.2)$$

$$x_{ik} + x_{jk} \leq 1, \quad \text{for all } \{i, j\} \in E, k = 1, \dots, m; \quad (4.3)$$

$$x_{ik} \in \{0, 1\}, \quad \text{for all } i = 1, \dots, n, k = 1, \dots, m. \quad (4.4)$$

Constraint (4.2) together with (4.4) determine that each node is assigned with exactly one color. Constraint (4.3) combined with (4.4) impose the requirement that any two adjacent nodes cannot be assigned with the same color.

The formulation of the constraints has a big effect on the behavior of the Douglas–Rachford scheme when applied to nonconvex problems. On the one hand, ones needs a formulation where the projectors onto the sets are easy to compute. On the other hand, the formulation chosen often determines whether or not the Douglas–Rachford scheme can successfully solve the problem at hand always, frequently or never [10]. For these two reasons, we have realized that it is convenient to reformulate constraint (4.3) as

$$x_{ik} + x_{jk} - y_{ek} = 0, \quad \text{for all } e = \{i, j\} \in E, k = 1, \dots, m; \quad (4.5)$$

where $y_{ek} \in \{0, 1\}$, for all $i, j \in \{1, \dots, n\}$ and $k \in \{1, \dots, m\}$. Although we have considerably increased the number of variables of the feasibility problem by adding lm new variables, where l is the number of edges in the graph, we have empirically observed that the Douglas–Rachford scheme becomes much more successful with this formulation.

Note that every permutation of a proper coloring is also a proper coloring. In our numerical tests we observed that this abundance of equivalent solutions significantly decreases the rate of success of the Douglas–Rachford algorithm. To avoid this problem, we may restrict the set of possible colorings to those that assign the first color to the first vertex. Without loss of generality, we can assume that the first vertex is connected to at least another node, which thus cannot be colored with the first color. Hence, we can reduce the number of solutions by forcing one of these nodes to be assigned with the second color. For this reason, we add to the formulation the constraint

$$x_{1,1} = 1 \text{ and } x_{i_0,2} = 1, \quad \text{for some fixed } i_0 \in \{2, \dots, n\} \text{ such that } \{1, i_0\} \in E. \quad (4.6)$$

In our experiments, vertex i_0 was chosen as $i_0 := \min \{i \in V : \{1, i\} \in E\}$. It would be possible to further reduce the number of solutions by following the same strategy with the largest complete subgraph contained in the graph G . As finding such a subgraph is not a trivial task and we did not observe in our numerical tests a clear improvement in the rate of success of the algorithm, we decided to only add constraint (4.6) into our formulation.

We shall also add the additional constraint that all m colors have to be used, i.e.,

$$\sum_{i=1}^n x_{ik} \geq 1, \quad \text{for all } k = 1, \dots, m. \quad (4.7)$$

Let $E = \{e_1, \dots, e_l\}$ be the set of edges, where $e_p \in \{1, \dots, n\}^2$ for every $p = 1, \dots, l$. Let $I := \{1, \dots, n\}$ and $P := \{n+1, \dots, n+l\}$, and let $K := \{1, \dots, m\}$ be the set of colors. Then, the m -coloring problem determined by constraints (4.2), (4.4), (4.5), (4.6) and (4.7) can be formulated as a feasibility problem with four constraints:

$$\text{Find } Z \in C_1 \cap C_2 \cap C_3 \cap C_4, \quad (4.8)$$

where $Z = (z_{ik}) \in \mathbb{R}^{(n+l) \times m}$ and

$$\begin{aligned} C_1 &:= \left\{ Z \in \mathbb{R}^{(n+l) \times m} : z_{ik} \in \{0, 1\}, \forall (i, k) \in I \times K \text{ and } \sum_{k=1}^m z_{ik} = 1, \forall i \in I \right\}, \\ C_2 &:= \left\{ Z \in \mathbb{R}^{(n+l) \times m} : z_{ik} + z_{jk} - z_{pk} = 0, \text{ with } e_{p-n} = \{i, j\} \in E, \forall (p, k) \in P \times K \right\}, \\ C_3 &:= \left\{ Z \in \{0, 1\}^{(n+l) \times m} : \sum_{i=1}^n z_{ik} \geq 1, \forall k \in K \right\}, \\ C_4 &:= \left\{ Z \in \mathbb{R}^{(n+l) \times m} : z_{1,1} = 1 \text{ and } z_{i_0,2} = 1 \right\}. \end{aligned}$$

Observe that constraint C_2 can be expressed in matrix form as

$$C_2 = \{Z \in \mathbb{R}^{(n+l) \times m} : AZ = 0_{l \times m}\}, \quad (4.9)$$

where $A = (a_{pq}) \in \mathbb{R}^{l \times (n+l)}$ is defined by

$$a_{pq} := \begin{cases} 1, & \text{if } e_p = \{i, j\} \text{ and } q \in \{i, j\}, \\ -1, & \text{if } q = n + p, \\ 0, & \text{elsewhere;} \end{cases}$$

for each $p = 1, \dots, l$ and $q \in I \cup P$.

The projectors onto C_1 and C_3 can be derived from Propositions 1.45 and 1.46, respectively; while the projector onto C_4 is trivially obtained. For any $Z \in \mathbb{R}^{(n+l) \times m}$, these projectors are given, pointwise, by

$$\begin{aligned} P_{C_1}(Z)[i, :] &= \begin{cases} \{e_k^T : z_{ik} = \max\{z_{i1}, z_{i2}, \dots, z_{im}\}\}, & \text{if } i \in I, \\ (z_{i1}, z_{i2}, \dots, z_{im}), & \text{if } i \in P; \end{cases} \\ P_{C_3}(Z)[:, k] &= \begin{cases} P_{\{0,1\}^{n+l}}(Z[:, k]) \setminus \{0_{n+l}\}, & \text{if } P_{\{0,1\}^{n+l}}(Z[:, k]) \neq \{0_{n+l}\}, \\ \{e_i : z_{ik} = \max\{z_{1k}, \dots, z_{nk}\}\}, & \text{otherwise;} \end{cases} \\ (P_{C_4}(Z))[i, k] &= \begin{cases} 1, & \text{if } (i, k) \in \{(1, 1), (i_0, 2)\}, \\ z_{ik}, & \text{otherwise;} \end{cases} \end{aligned}$$

for each $i \in I \cup P$ and $k \in K$. Since A is full row rank, according to Proposition 1.47, the projector onto C_2 is given by

$$P_{C_2}(Z) = \left(\text{Id}_{n+l} - A^T (AA^T)^{-1} A \right) Z.$$

Finally, observe that the projectors onto C_1 and C_3 may be multivalued. A projection onto these sets $\pi_{C_1}(Z) \in P_{C_1}(Z)$ and $\pi_{C_3}(Z) \in P_{C_3}(Z)$ is given, pointwise, by

$$\begin{aligned} (\pi_{C_1}(Z))[i, k] &= \begin{cases} 1, & \text{if } i \in I, k = \text{argmax}\{z_{i1}, z_{i2}, \dots, z_{im}\}, \\ z_{ik}, & \text{if } i \in P, \\ 0, & \text{otherwise;} \end{cases} \\ (\pi_{C_3}(Z))[i, k] &= \begin{cases} 1, & \text{if } i = \text{argmax}\{z_{1k}, z_{2k}, \dots, z_{nk}\}, \\ \min\{1, \max\{0, \text{round}(z_{ik})\}\}, & \text{otherwise;} \end{cases} \end{aligned}$$

where the lowest index is chosen in argmax and $\text{round}(0.5) = 0$.

Adding maximal clique information

Let us illustrate with an example the need of adding maximal clique information into our formulation, whenever this information is available.

Example 4.9. *Let us consider now the so-called windmill graph $\text{Wd}(a, b)$, which is the graph constructed for $a \geq 2$ and $b \geq 2$ by joining b copies of a complete graph with a vertices at a shared vertex. A plot of $\text{Wd}(6, 5)$ is shown in Figure 4.13.*

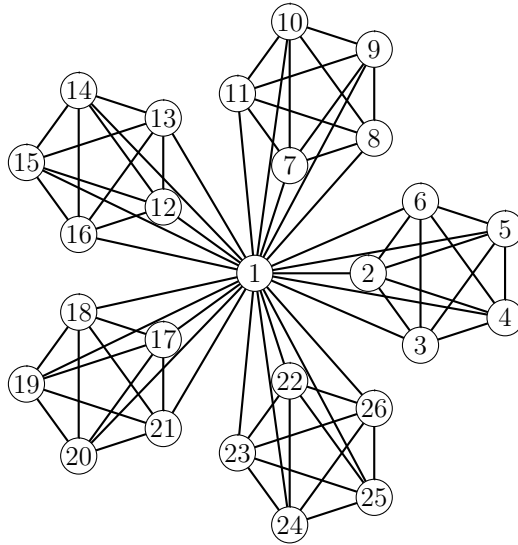


FIGURE 4.13: Plot of the windmill graph $\text{Wd}(6, 5)$.

Every windmill graph $\text{Wd}(a, b)$ can be easily a -colored (there are $a((a-1)!)^b$ different ways). Despite this abundance of valid colorings, the Douglas–Rachford scheme fails to find a solution rather often, see the results in Figure 4.17. This graph has an additional available information that can be used: it has b maximal cliques of size a , and each color can be used at most once within each maximal clique.

Let $Q \subset 2^V$ be a nonempty subset of maximal cliques of the graph $G = (V, E)$ and let $\widehat{E} := E \cup Q$. Let $Q = \{e_{l+1}, \dots, e_r\}$, with $r \geq l+1$. Thus, $\widehat{E} = \{e_1, \dots, e_l, e_{l+1}, \dots, e_r\}$. The maximal clique information can be easily added into constraint C_2 in (4.9). Indeed, let

$$\widehat{C}_2 := \left\{ Z \in \mathbb{R}^{(n+r) \times m} : \widehat{A}Z = 0_{r \times m} \right\},$$

where $\widehat{A} = (\widehat{a}_{pq}) \in \mathbb{R}^{r \times (n+r)}$ is defined, for each $p = 1, \dots, r$ and $q \in \{1, \dots, n+r\}$, by

$$\widehat{a}_{pq} := \begin{cases} 1, & \text{if } q \in e_p; \\ -1, & \text{if } q = n+p; \\ 0, & \text{elsewhere.} \end{cases}$$

This is clearly an equivalent formulation of the m -coloring problem, where we have added $(r - l)m$ new variables (now $Z \in \mathbb{R}^{(n+r) \times m}$), which correspond to the (redundant) information that each color can only be used once within each maximal clique. Despite that, this formulation can be advantageous, as shown in Figure 4.17. For some particular graphs, adding this information can be crucial, see Table 4.5, where we compare two reformulations of 3-SAT problems with and without maximal clique information.

4.1.2.2 Modelling other variants of the graph coloring problem

The feasibility problem in (4.8) can be adequately adapted so that it permits to solve any of the previously presented variants of the graph coloring problem. To do so, some of the constraint sets defining the model must be removed or replaced. Such modifications of the formulation are explained in detail hereafter.

Precoloring and list coloring

Consider first the case where a list coloring problem has been reduced to a classical one through the reformulation shown in Figure 4.5. Note that the new feasibility problem is defined in $\mathbb{R}^{(n+m+l^*) \times m}$. Constraint C_4 has to be changed, as it no longer makes sense. We have m new nodes, labeled $n + 1, \dots, n + m$, and each of them represents a color. To include this information, we shall replace C_4 by

$$C_4^* := \{Z \in \mathbb{R}^{(n+m+l^*) \times m} : z_{n+k,k} = 1, \forall k \in K\}.$$

Thereby, the solution set is $C_1 \cap C_2 \cap C_3 \cap C_4^*$. The projector onto C_4^* at any point $Z \in \mathbb{R}^{(n+m+l^*) \times m}$ is given componentwise by

$$(P_{C_4^*}(Z)) [i, k] = \begin{cases} 1, & \text{if } i = k + n; \\ z_{ik}, & \text{otherwise.} \end{cases}$$

However, note that the increase in the number of nodes and edges may cause the DR algorithm to become slower.

Another option here would be to directly modify the constraint C_1 to only allow admissible colors, that is, to replace it by the set

$$\bar{C}_1 := \left\{ Z \in \mathbb{R}^{(n+l) \times m} : z_{ik} \in \{0, 1\}, \forall (i, k) \in I \times K \text{ and } \sum_{k \in L(i)} z_{ik} = 1, \sum_{k \notin L(i)} z_{ik} = 0, \forall i \in I \right\}.$$

A projection onto \overline{C}_1 is given, pointwise, by

$$(\pi_{\overline{C}_1}(Z)) [i, k] = \begin{cases} 1, & \text{if } i \in I, k = \operatorname{argmax}\{z_{ij}, j \in L(i)\}; \\ z_{ik}, & \text{if } i \in P; \\ 0, & \text{otherwise.} \end{cases}$$

In this formulation, constraint C_4 cannot be adapted and it has to be removed from the feasibility problem. Then, the solution set becomes $\overline{C}_1 \cap C_2 \cap C_3$.

Partial coloring

Recall that in a partial coloring, not all the nodes need to be colored, but only q nodes per color. We must then remove the set C_4 in (4.8) and replace the sets C_1 and C_3 by

$$\check{C}_1 := \left\{ Z \in \mathbb{R}^{(n+l) \times m} : z_{ik} \in \{0, 1\}, \forall (i, k) \in I \times K \text{ and } \sum_{k=1}^m z_{ik} \leq 1, \forall i \in I \right\},$$

$$\check{C}_3 := \left\{ Z \in \{0, 1\}^{(n+l) \times m} : \sum_{i=1}^n z_{ik} = q, \forall k \in K \right\}.$$

For instance, for the 8-queens puzzle (Example 4.5) we need $q = 8$ and $m = 1$. Hence, the solution set of these problems is $\check{C}_1 \cap C_2 \cap \check{C}_3$. A projection onto \check{C}_1 and \check{C}_3 is given by

$$(\pi_{\check{C}_1}(Z)) [i, k] = \begin{cases} \min \{1, \max\{0, \operatorname{round}(z_{ik})\}\}, & \text{if } i \in I, k = \operatorname{argmax}\{z_{i1}, z_{i2}, \dots, z_{im}\}, \\ z_{ik}, & \text{if } i \in P, \\ 0, & \text{otherwise;} \end{cases}$$

$$(\pi_{\check{C}_3}(Z)) [i, k] = \begin{cases} 1, & \text{if } i \in Q_{k,q}, \\ \min \{1, \max\{0, \operatorname{round}(z_{ik})\}\}, & \text{if } i \in P, \\ 0, & \text{otherwise;} \end{cases}$$

where, for a given color $k \in K$, we denote by $Q_{k,q} \subset I$ the set of indices corresponding to the q largest values in $\{z_{1k}, z_{2k}, \dots, z_{nk}\}$ (lowest index is chosen in case of tie).

Hamiltonian paths

In a Hamiltonian path problem every node must be assigned with a different color. It is thus no longer necessary to work in $\mathbb{R}^{(n+l) \times n}$, but in $\mathbb{R}^{n \times n}$, so constraint C_1 becomes

$$\tilde{C}_1 := \left\{ X \in \mathbb{R}^{n \times n} : x_{ik} \in \{0, 1\}, \forall (i, k) \in I \times K \text{ and } \sum_{k=1}^m x_{ik} = 1, \forall i \in I \right\}.$$

To ensure that the coloring leads to a path, the set C_3 must be modified and replaced by

$$\tilde{C}_3 := \{X \in \{0, 1\}^{n \times n} : \forall k = 1, \dots, n-1, \exists \{i, j\} \in E \text{ s.t. } x_{i,k}x_{j,k+1} = 1\}.$$

Constraint C_2 is no longer needed. We have observed that the rate of success of DR is decreased if C_2 is removed, and that it is better to replace it by the redundant constraint $\tilde{C}_2 := \mathbb{R}^{n \times n}$, see the experiment shown in Figure 4.21. Note that constraint C_4 forces the path to start on node 1 (a path which may not even exist), so it is eliminated.

The projector onto \tilde{C}_3 is hard to compute because of the recurrent dependence between all the columns in the matrix X . To overcome this problem, we propose to split the set \tilde{C}_3 into two constraints, one relating each odd column with its following one, and another similar constraint for the even columns. That is, we define the constraints

$$\begin{aligned} \tilde{C}_{3,\text{odd}} &:= \left\{ X \in \{0, 1\}^{(n+1) \times n} : \forall k = 1, \dots, \left\lfloor \frac{n}{2} \right\rfloor, \exists \{i, j\} \in E \text{ s.t. } x_{i,2k-1}x_{j,2k} = 1 \right\}, \\ \tilde{C}_{3,\text{even}} &:= \left\{ X \in \{0, 1\}^{(n+1) \times n} : \forall k = 1, \dots, \left\lfloor \frac{n-1}{2} \right\rfloor, \exists \{i, j\} \in E \text{ s.t. } x_{i,2k}x_{j,2k+1} = 1 \right\}, \end{aligned}$$

which satisfy $\tilde{C}_3 = \tilde{C}_{3,\text{odd}} \cap \tilde{C}_{3,\text{even}}$, where $\lfloor \cdot \rfloor$ denotes the integer part of a number. Therefore, the solution set of the Hamiltonian path problem is $\tilde{C}_1 \cap \tilde{C}_2 \cap \tilde{C}_{3,\text{odd}} \cap \tilde{C}_{3,\text{even}}$.

To compute a projection onto $\tilde{C}_{3,\text{odd}}$ and $\tilde{C}_{3,\text{even}}$, consider the function $h : \mathbb{R} \mapsto \mathbb{R}$ defined by

$$h(x) := \begin{cases} x, & \text{if } x \leq 0.5, \\ 1, & \text{if } x > 0.5; \end{cases}$$

and let us denote by

$$(s_{k_1, k_2}^0, s_{k_1, k_2}^1) = \operatorname{argmin} \{(1 - h(x_{i, k_1}))^2 + (1 - h(x_{j, k_2}))^2, \{i, j\} \in E\},$$

where the lowest index is taken in argmin to avoid multivaluedness. Then, a projection onto $\tilde{C}_{3,\text{odd}}$ and $\tilde{C}_{3,\text{even}}$ can be obtained as follows

$$\begin{aligned} \left(\pi_{\tilde{C}_{3,\text{odd}}} (Z) \right) [i, k] &= \begin{cases} 1, & \text{if } i = s_{k, k+1}^0, k < n \text{ and } k \text{ is odd,} \\ 1, & \text{if } i = s_{k-1, k}^1 \text{ and } k \text{ is even,} \\ \min \{1, \max\{0, \operatorname{round}(x_{ik})\}\}, & \text{otherwise;} \end{cases} \\ \left(\pi_{\tilde{C}_{3,\text{even}}} (Z) \right) [i, k] &= \begin{cases} 1, & \text{if } i = s_{k, k+1}^0, k < n \text{ and } k \text{ is even,} \\ 1, & \text{if } i = s_{k-1, k}^1, 1 < k \text{ and } k \text{ is odd,} \\ \min \{1, \max\{0, \operatorname{round}(x_{ik})\}\}, & \text{otherwise.} \end{cases} \end{aligned}$$

4.1.2.3 Numerical experiments

In this section we test the performance of the Douglas–Rachford algorithm for solving a representative sample of the graph coloring problems previously presented. Note that the feasibility problem in (4.8), as well as all its adjustments presented in Section 4.1.2.2, are described by more than two constraints. We have thus to turn to a many-set version of the DR algorithm (see Section 2.2.2.2). One option would be to use the cyclic DR method defined by (2.18). However this method suffers the same unsuitable behavior than alternating projections when tackling combinatorial problems (see Figure 2.11). Hence, we use the Douglas–Rachford scheme in the product space, whose iteration is generated by (2.17). For each formulation defined by r sets, we use the stopping criterion

$$\sum_{i=1}^r \|p_k - P_{C_i}(p_k)\|^2 < \varepsilon := 10^{-10}. \quad (4.10)$$

All codes were written in Python 2.7 and the tests were run on an Intel Core i7-4770 CPU 3.40 GHz with 12 GB RAM, under Windows 10 (64-bit).

Testing some simple graphs

We begin our tests with one of the most well-known graphs: *Petersen graph* (see Figure 4.1). This graph has 10 vertices, 15 edges and can be 3-colored in 120 different ways. The results of our first experiment are shown in Figure 4.14. For 100,000 random starting points and using formulation (4.8), we report the number of iterations needed by the Douglas–Rachford algorithm until it obtained a solution or it reached 500 iterations. The success rate was nearly 100% in this experiment: the algorithm was able to find a solution for almost every starting point (with the exception of 2 instances out of 100,000).

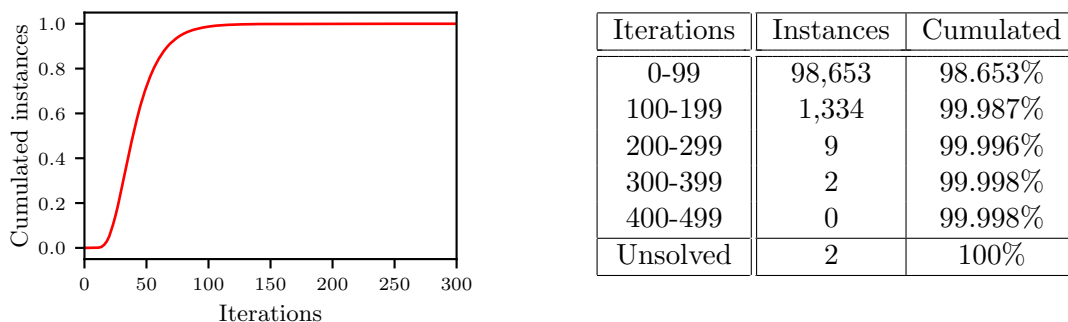


FIGURE 4.14: Number of iterations spent by DR to find a solution of a 3-coloring of Petersen graph for 100,000 random starting points. On average, each solution was found in 0.00533 seconds. Instances were labeled as “Unsolved” after 500 iterations.

We also tested the performance of the Douglas–Rachford algorithm with formulation (4.8) for finding a valid coloring of some other different simple graphs. Namely, we used *complete graphs* of 4, 5 and 6 nodes; *wheel graphs* of 5 and 6 nodes; and *cycle graphs* of 10, 15 and 20 nodes. These graphs are explained next and are illustrated in Figure 4.15.

- In a *complete graph* every pair of nodes is connected. A complete graph with n vertices has $n(n-1)/2$ edges and can be n -colored in $n!$ different ways.
- A *cycle graph* consists in a collection of vertices connected in a closed chain. A cycle graph with n vertices has n edges. If n is even, it can be 2-colored in 2 different ways; if n is odd, it can be 3-colored in $2^n - 2$ different ways.
- A *wheel graph* is a cycle with an extra node which is connected to all other nodes. A wheel graph with n vertices has $2(n-1)$ edges. If n is even, it can be 4-colored in $4(2^{n-1} - 2)$ different ways; if n is odd, it can be 3-colored in 6 different ways.

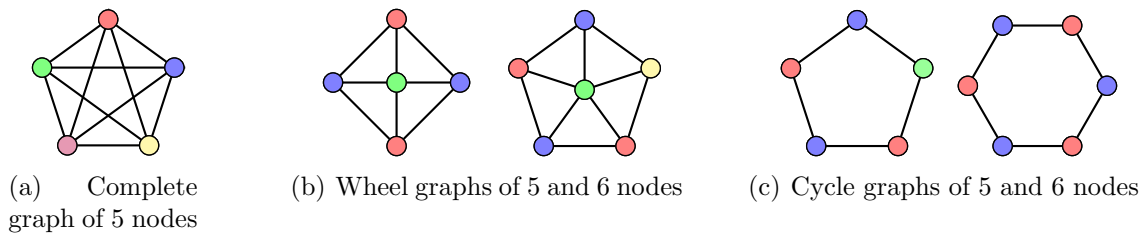


FIGURE 4.15: Proper colorings of some simple graphs.

The DR algorithm was run on each of the considered problems from 10,000 random starting points, and it was stopped after a maximum of 500 iterations. For each type of graph, we plot in Figure 4.16 the cumulated number of solved instances with respect to the first 300 iterations, while some more detailed results are shown in Tables 4.1 to 4.3.

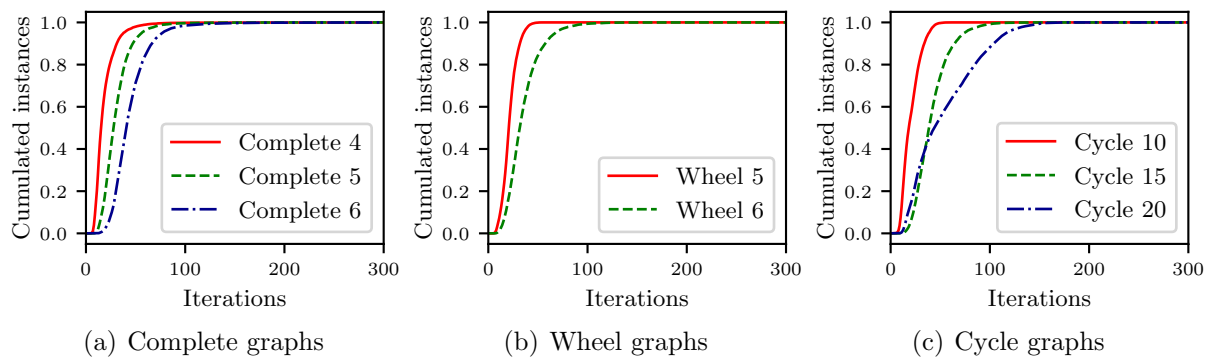


FIGURE 4.16: Cumulated number of instances solved by DR with respect to the 300 first iterations for some complete, cycle and wheel graphs of different sizes.

Iterations	Complete 4		Complete 5		Complete 6	
	Instances	Cumul.	Instances	Cumul.	Instances	Cumul.
0-99	9,981	99.81%	9,961	99.61%	9,847	98.47%
100-199	14	99.95%	37	99.98%	143	99.9%
200-299	0	99.95%	2	100.0%	6	99.96%
300-399	0	99.95%	0	100.0%	4	100.0%
400-499	0	99.95%	0	100.0%	0	100.0%
Unsolved	5	100%	0	100%	0	100%

TABLE 4.1: Number of iterations spent by DR to find a solution of an n -coloring of a complete graph with n vertices for 10,000 random starting points, with $n = 4, 5, 6$. Each solution was found, on average, in 0.00215 seconds for $n = 4$, 0.00371 seconds for $n = 5$, and 0.00569 seconds for $n = 6$. Instances were labeled as “Unsolved” after 500 iterations.

Iterations	Wheel 5		Wheel 6	
	Instan.	Cumul.	Instan.	Cumul.
0-49	9,988	99.88%	8,332	83.32%
50-99	12	100%	1,600	99.32%
100-149	0	100%	65	99.97%
150-249	0	100%	2	99.99%
250-499	0	100%	0	99.99%
Unsolved	0	100%	1	100%

TABLE 4.2: Number of iterations spent by DR to find a solution of two wheel graphs for 10,000 random starting points. Each solution was found, on average, in 0.00266 seconds for wheel 5, and 0.00455 seconds for wheel 6. Instances were labeled as “Unsolved” after 500 iterations.

Iterations	Cycle 10		Cycle 15		Cycle 20	
	Instances	Cumul.	Instances	Cumul.	Instances	Cumul.
0-49	9,967	99.67%	7,276	72.76%	5,438	54.38%
50-99	33	100%	2,640	99.16%	3,349	87.87%
100-149	0	100%	81	99.97%	1,156	99.43%
150-199	0	100%	2	99.99%	57	100%
200-499	0	100%	0	99.99%	0	100%
Unsolved	0	100%	1	100%	0	100%

TABLE 4.3: Number of iterations spent by DR to find a solution of three cycle graphs for 10,000 random starting points. Each solution was found, on average, in 0.0025 seconds for cycle 10, 0.00561 seconds for cycle 15, and 0.00731 seconds for cycle 20. Instances were labeled as “Unsolved” after 500 iterations.

We note the good behavior of the DR algorithm: it was able to find a solution for every random starting point for the complete graphs of 5 and 6 nodes, the wheel graph of 5 nodes, and the cycle graphs of 10 and 20 nodes; while it failed in at most the 0.05% of the instances in the remaining cases.

In our following experiment, whose results are shown in Figure 4.17 and Table 4.4, we compare the performance of the Douglas–Rachford algorithm with and without maximal clique information when it is applied for finding a solution of the windmill graph $Wd(6, 5)$. Observe that, even having increased the number of variables in the feasibility problem, both the rate of success and the rate of convergence (in terms of iterations as well as computing time) are much improved.

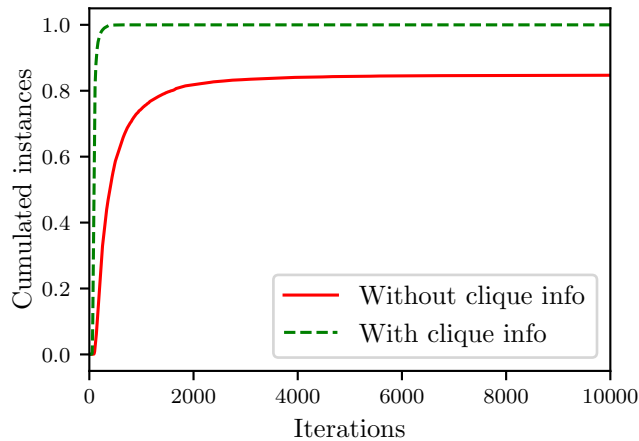
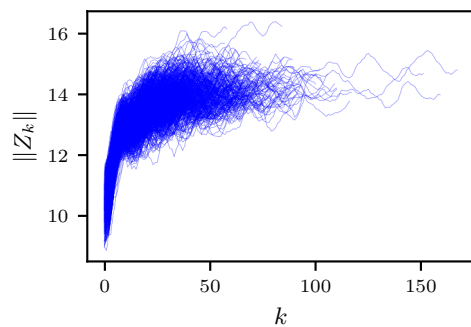


FIGURE 4.17: Cumulated number of instances solved by DR (out of 10,000 random starting points) to find a solution of the windmill graph $Wd(6, 5)$, with and without maximal clique information, with respect to the iterations.

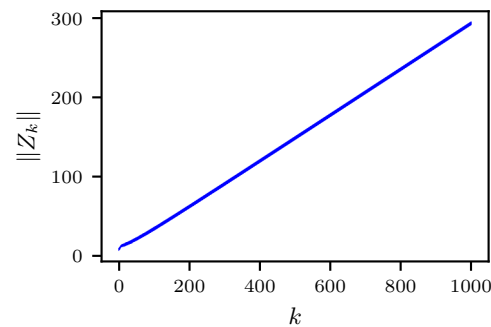
Iterations	Without maximal clique information		With maximal clique information	
	Instances	Cumul.	Instances	Cumul.
0-999	7,448	74.48%	9,999	99.99%
1,000-1,999	728	81.76%	0	99.99%
2,000-2,999	165	83.41%	0	99.99%
3,000-3,999	64	84.05%	0	99.99%
4,000-4,999	29	84.34%	0	99.99%
5,000-5,999	17	84.51%	0	99.99%
6,000-6,999	7	84.58%	0	99.99%
7,000-7,999	7	84.65%	0	99.99%
8,000-8,999	4	84.69%	0	99.99%
9,000-9,999	1	84.7%	0	99.99%
Unsolved	1,530	100%	1	100%

TABLE 4.4: Comparison of the number of iterations spent by DR to find a solution of the windmill graph $Wd(6, 5)$ for 10,000 random starting points. Complete maximal clique information was used in the right columns. Each solution was found, on average, in 0.13347 seconds without clique information, and 0.02424 seconds with maximal clique information. Instances were labeled as “Unsolved” after 10,000 iterations.

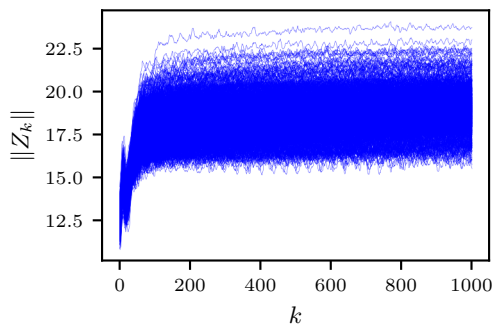
If the norm of the iterates, $\|Z_k\|$, increases as k increases, the Douglas–Rachford algorithm may serve to detect infeasibility of the corresponding coloring problem, see Figures 4.18(a) and 4.18(b). In the convex case, according to Theorem 2.9(ii), this behavior is ensured for infeasible problems. However, due to the lack of convexity, this is not always the case in our context, as shown in Figures 4.18(c) and 4.18(d). Interestingly, when we removed the extra constraints (4.6) and (4.7), which is something that does not change the feasibility of the problems, the algorithm was not able to detect any infeasible problem.



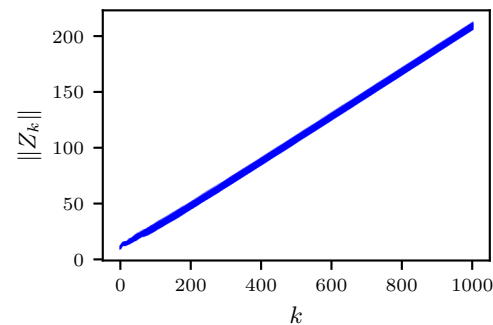
(a) 3-coloring of Petersen graph (feasible)



(b) 2-coloring of Petersen graph (infeasible)



(c) 4-coloring of a 7-complete graph (infeasible)



(d) 3-coloring of a 7-complete graph (infeasible)

FIGURE 4.18: For 1,000 random starting points, we represent the iteration k in the horizontal axis and $\|Z_k\|$ in the vertical axis for 1,000 iterations of the Douglas–Rachford algorithm.

3-SAT problems

Next, we tested the performance of DR for the 4-nodes and the 5-nodes formulations for the first 50 3-SAT problems with 20 variables and 91 clauses in SATLIB³. For each of the formulations, we run the Douglas–Rachford algorithm with and without maximal clique information for 10 random starting points. The results are shown in Table 4.5. Clearly, the addition of the maximal clique information turns out to be crucial for the success of the Douglas–Rachford algorithm, specially for the 5-nodes formulation.

³SATLIB: www.cs.ubc.ca/~hoos/SATLIB/Benchmarks/SAT/RND3SAT/uf20-91.tar.gz

Time	4-nodes without clique info.		4-nodes with clique info.		5-nodes without clique info.		5-nodes with clique info.	
	Inst.	Cumul.	Inst.	Cumul.	Inst.	Cumul.	Inst.	Cumul.
0-49	278	55.6%	323	64.6%	0	0.0%	249	49.8%
50-99	60	67.6%	65	77.6%	0	0.0%	69	63.6%
100-149	31	73.8%	24	82.4%	0	0.0%	42	72.0%
150-199	20	77.8%	14	85.2%	0	0.0%	31	78.2%
200-249	16	81.0%	11	87.4%	0	0.0%	21	82.4%
250-299	11	83.2%	10	89.4%	0	0.0%	8	84.0%
Unsolved	84	100%	53	100%	500	100%	80	100%

TABLE 4.5: Time spent (in seconds) by DR to find a solution of 50 different 3-SAT problems with 20 variables and 91 clauses. For each problem, 10 random starting points were chosen. After 300 seconds without finding a solution, instances were labeled as “Unsolved”. Two formulations of the gadgets were considered, with 4 and 5 nodes.

REMARK 4.10 (Performance profiles). For an appropriate visualization of the results and comparison of the formulations, we turn to *performance profiles* (see [88]). We use the modification proposed in [122], since it suits better our experiment, where we have multiple runs for every formulation and problem. Let Φ denote the (finite) set of all formulations. A performance profile is constructed as follows.

1. For each formulation $f \in \Phi$, let $t_{f,p}$ be the average time required by DR to solve problem p among all the successful runs, and let us denote by $s_{f,p}$ the portion of successful runs for problem p .
2. Compute $t_p^* := \min_{f \in \Phi} t_{f,p}$, for all $p \in \{1, \dots, n_p\}$, where n_p is the number of problems in the experiment.
3. Then, for any $\tau \geq 1$, define $R_f(\tau) := \{p \in \{1, \dots, n_p\}, t_{f,p} \leq \tau t_p^*\}$; that is, $R_f(\tau)$ is the set of problems for which formulation f is at most τ times slower than the best one.
4. The *performance profile* function of formulation f is given by

$$\begin{aligned} \rho_f : [1, +\infty) &\longmapsto [0, 1] \\ \tau &\longmapsto \rho_f(\tau) := \frac{1}{n_p} \sum_{p \in R_f(\tau)} s_{f,p}. \end{aligned}$$

The value $\rho_f(1)$ indicates the portion of runs for which f was the fastest formulation. When $\tau \rightarrow +\infty$, then $\rho_f(\tau)$ shows the portion of successful runs for formulation f .

The performance profiles for the results of the 3-SAT experiment in Table 4.5 are displayed in Figure 4.19. It clearly shows that the 4-nodes formulation with maximal clique information is the best one. Despite that the addition of maximal clique information increases the dimension of the problem, it improves both the rate of success and the speed of convergence of the algorithm for both formulations (specially for the 5-nodes one).

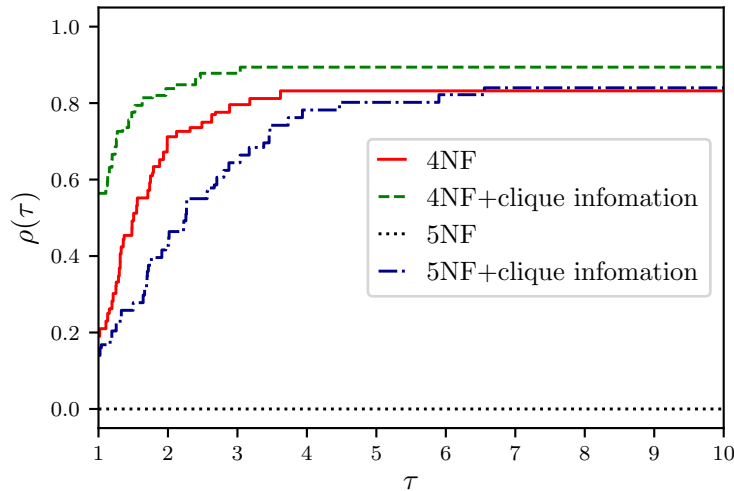


FIGURE 4.19: Performance profiles for the results of the 3-SAT experiment in Table 4.5.

Sudokus

In our next numerical experiment, for solving Sudoku puzzles, we compared the performance of the DR algorithm applied to the cubic formulation for Sudokus (see Remark 4.4), with the reformulations as a graph coloring ($C_1 \cap C_2 \cap C_3 \cap C_4^*$) and as a graph precoloring ($\overline{C}_1 \cap C_2 \cap C_3$), explained in Section 4.1.2.2. We considered the 95 hard puzzles from the library `top95`⁴, which was the one among the libraries tested in [10, Table 2] where DR was most unsuccessful. For each formulation and each puzzle, Douglas–Rachford was run for 20 random starting points. Results are shown in Table 4.6 and performance profiles are displayed in Figure 4.20.

As it was expected, the cubic formulation was much faster, since this formulation is specifically designed for solving these puzzles. On average, the cubic formulation solved a Sudoku in 5.76 seconds, while the graph precoloring formulation needed 33.78 seconds. The worst method was the reformulation as a graph coloring problem, which needed 112.25 seconds on average to solve a Sudoku. Even so, it was surprising to see that the rate of success for these three formulations was very similar, around 90%.

⁴`top95`: <http://magictour.free.fr/top95>

Time	Cubic formulation		Graph precoloring		Reformulation as graph coloring	
	Inst.	Cumul.	Inst.	Cumul.	Inst.	Cumul.
0-49	1,688	88.8%	1,451	76.4%	261	13.7%
50-99	19	89.8%	173	85.5%	534	41.8%
100-149	15	90.6%	40	87.6%	451	65.6%
150-199	6	90.9%	22	88.7%	267	79.6%
200-249	4	91.2%	12	89.4%	118	85.8%
250-299	2	91.3%	5	89.6%	45	88.2%
Unsolved	166	100%	197	100%	224	100%

TABLE 4.6: Time spent (in seconds) to find the solution of 95 different Sudoku problems by DR with the graph precoloring, the cubic, and the graph coloring formulations. For each problem, 20 starting points were randomly chosen. We stopped the algorithm after a maximum time of 300 seconds, in which case the problem was labeled as “Unsolved”.

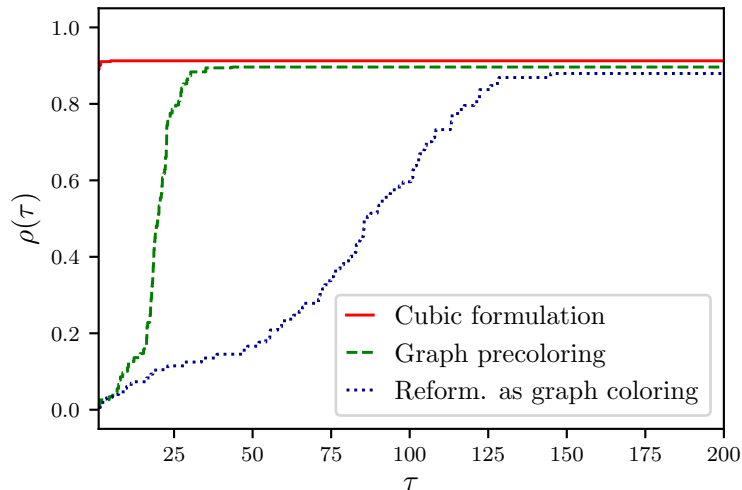


FIGURE 4.20: Performance profiles for the results of the Sudoku experiment in Table 4.6.

In Table 4.7 we list the Sudokus for which either the cubic or the graph precoloring formulation failed to find a solution for some starting points. It is apparent that the three methods tend to fail on the same Sudokus. The reformulation as graph coloring was clearly the most successful method for Sudoku 19. The graph precoloring formulation had a very bad performance on Sudoku 22, compared to the other two methods. On the other hand, it is remarkable that the cubic formulation was significantly less successful than the graph precoloring for Sudoku 90, and that it failed to find any solution at all for Sudoku 25. Both the graph precoloring formulation and the reformulation as graph coloring also had troubles with this Sudoku, and were only able to find a solution for 3 and 2 out of the 20 starting points, respectively. This Sudoku is the one shown in Figure 4.7.

Sudoku Number	5	12	13	17	19	22	25	29	38
Cubic formulation	0	0	16	19	5	1	20	1	17
Graph precoloring	6	1	18	18	13	19	17	7	15
Reformulation as graph coloring	13	1	16	18	1	9	18	15	12
Sudoku Number	53	59	66	82	83	85	86	90	94
Cubic formulation	0	0	14	0	5	18	17	14	19
Graph precoloring	5	3	13	3	5	15	15	8	16
Reformulation as graph coloring	6	1	11	7	4	14	16	14	15

TABLE 4.7: Number of failed runs in either the cubic or the graph precoloring formulation. Sudokus not listed here were solved by these two formulations for every starting point.

Knight’s tours

In our next experiment, we explored the behavior of DR for solving the knight’s tour problem when the size of the chessboard is increased. Results are displayed in Figure 4.21, where we analyze both paths and cycles with the two formulations $\tilde{C}_1 \cap \tilde{C}_{3,\text{odd}} \cap \tilde{C}_{3,\text{even}}$ (red crosses) and $\tilde{C}_1 \cap \tilde{C}_2 \cap \tilde{C}_{3,\text{odd}} \cap \tilde{C}_{3,\text{even}}$ (blue dots). Clearly, the formulation including the redundant constraint $\tilde{C}_2 = \mathbb{R}^{n \times n}$ is much faster. For this reason, no knight’s paths of size 10 or 11 are shown for the formulation without \tilde{C}_2 , as the algorithm was stopped before it had enough time to converge. The rate of success of both formulations for paths and cycles was very similar, around 95%. It can be observed an exponential dependence between time and size, which makes DR to be inappropriate for big puzzles.

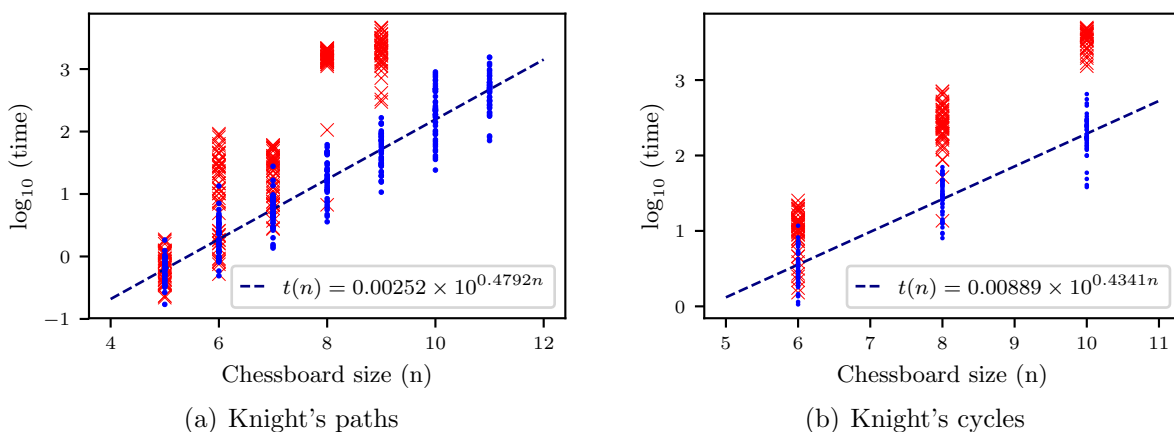


FIGURE 4.21: Time (in \log_{10}) required by DR for finding knight’s paths and cycles on chessboards of different size. For each size, 50 random starting points were chosen. Blue dots represent instances of the DR method applied with the addition of the redundant constraint $\tilde{C}_2 = \mathbb{R}^{n \times n}$, while red crosses represent instances where the method was run without \tilde{C}_2 . The dotted lines were obtained by linear regression. The algorithm was stopped after a maximum time of 5,000 seconds, in which case the instance is not displayed.

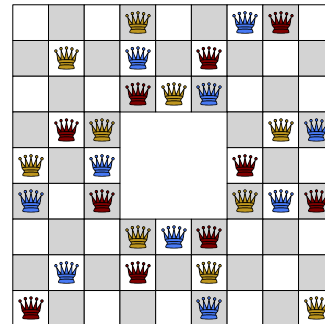
It is remarkable that the line $t(n)$ obtained by linear regression in Figure 4.21(b) predicts an average time of $t(12) = 1,439$ seconds for finding a knight's cycle in a 12×12 chessboard, and this totally fits with the average time of 1,397 seconds spent by DR to obtain the tour previously shown in Figure 4.12.

Generalizations of the 8-queens puzzle

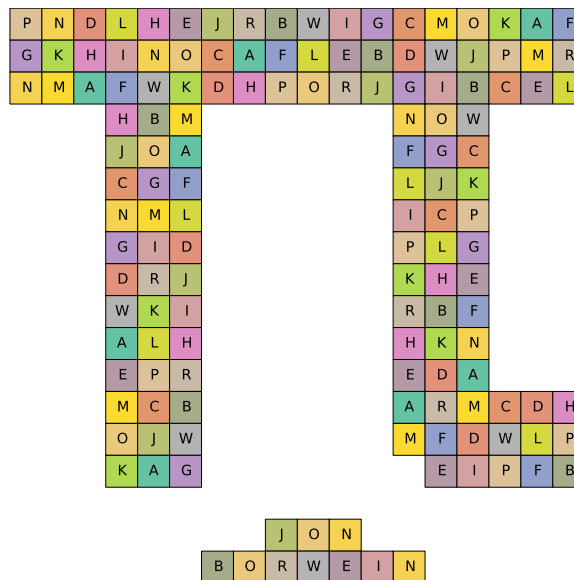
Finally, we apply the DR algorithm for solving the generalizations of the 8-queen puzzle proposed in Figure 4.9. For each of them, we run the algorithm from 10 random starting points. Solutions and results are shown in Figure 4.22.



(a) Knights on a classic chessboard



(b) Queens in a chessboard with a hole



(c) 'π-zzle'

FIGURE 4.22: Solution to the puzzles in Figure 4.9 computed with DR. For 10 random starting points, the average (maximum) time spent for puzzles (a), (b) and (c) was 0.23, 3.32 and 252.82 seconds (0.35, 11.49 and 424.67 seconds), respectively.

4.1.3 A feasibility model based on a low-rank constrained matrix

In this section we consider a different formulation of graph coloring, which is based on semi-definite programming. It is due to Karger, Motwani and Sudan [126], who proposed using the geometry of the regular simplex to color the vertices of a graph. The associated feasibility problem relies on the reconstruction of a non-negative rank-constrained matrix. This geometrical encoding of the problem, which respects all its symmetries, is well suited to projection based algorithms. Although this model cannot be so easily adapted to cover other type of graph coloring problems apart from precoloring, our numerical experiments indicate that the KMS formulation appears to be superior to the indicator variable formulation proposed in Section 4.1.2, when using the DR heuristic. While we do not have an interpretation of this result, it is empirically supported on a wide spectrum of problem instances.

4.1.3.1 Modelling graph coloring with vertices of the regular simplex

Suppose that we have a proper m -coloring of the graph $G = (V, E)$ given by $c : V \mapsto K$. The m -coloring c can be represented by a matrix as follows. Let $u_1, u_2, \dots, u_m \in \mathbb{R}^{m-1}$ be the vertices of a *standard centered regular $(m-1)$ -simplex*, i.e.,

$$u_1 + \dots + u_m = 0 \quad \text{and} \quad \langle u_i, u_j \rangle = \begin{cases} 1, & \text{if } i = j, \\ \mu, & \text{if } i \neq j; \end{cases} \quad (4.11)$$

for some constant $\mu \in \mathbb{R}$. It directly follows from (4.11) that

$$0 = \left\langle \sum_{i=1}^m u_i, \sum_{i=1}^m u_i \right\rangle = m + 2m(m-1)\mu \quad \Leftrightarrow \quad \mu = \frac{-1}{m-1}.$$

Each of the vertices of the $(m-1)$ -simplex shall represent one of the m colors. Hence, the m -coloring of the graph G can be recovered from the matrix

$$U_c := [u_{c(1)}, u_{c(2)}, \dots, u_{c(n)}] \in \mathbb{R}^{(m-1) \times n}, \quad (4.12)$$

whose rows are vertices of the $(m-1)$ -simplex (possibly repeated). Finally, let us construct the Gram matrix associated with U_c , namely,

$$W_c := U_c^T U_c \in \mathbb{R}^{n \times n}. \quad (4.13)$$

Since c is a proper coloring, and by (1.4), the matrix W_c has the following properties:

(P1) $W_c \in \mathcal{S}_+^n$,

(P2) $\text{rank}(W_c) \leq m - 1$,

(P3) $W_c \in \{1, \frac{-1}{m-1}\}^{n \times n}$ and some of the entries of $W_c = [w_{ij}]$ are determined as follows:

$$w_{ii} = 1, \forall i \in V, \quad \text{and} \quad w_{ij} = \frac{-1}{m-1}, \forall \{i, j\} \in E.$$

Hence, every valid m -coloring of the graph G leads to a matrix verifying (P1)–(P3). In fact, this is an equivalence, as we shall show after the next illustrative example.

Example 4.11. Consider a graph $G = (V, E)$ where the set of vertices is $V = \{1, 2, 3, 4, 5\}$, and the set of edges is $E = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 5\}\}$. A proper 3-coloring of G is shown in Figure 4.23(a). We identify each of the colors with one of the vertices $u_1, u_2, u_3 \in \mathbb{R}^2$ of a standard centered regular 2-simplex (see Figure 4.23(b)), where

$$u_1 = (1, 0)^T, \quad u_2 = \frac{1}{2}(-1, \sqrt{3})^T \quad \text{and} \quad u_3 = \frac{1}{2}(-1, -\sqrt{3})^T.$$

Then the matrix representation of c given in (4.13) becomes

$$W_c = U_c^T U_c = \begin{pmatrix} \boxed{1} & \boxed{-0.5} & \boxed{-0.5} & 1 & -0.5 \\ \boxed{-0.5} & \boxed{1} & \boxed{-0.5} & -0.5 & 1 \\ \boxed{-0.5} & \boxed{-0.5} & \boxed{1} & -0.5 & \boxed{-0.5} \\ 1 & -0.5 & -0.5 & \boxed{1} & -0.5 \\ -0.5 & 1 & \boxed{-0.5} & -0.5 & \boxed{1} \end{pmatrix}, \quad \text{with } U_c = [u_1, u_2, u_3, u_1, u_2].$$

The boxed entries in W_c correspond to those determined by (P3).

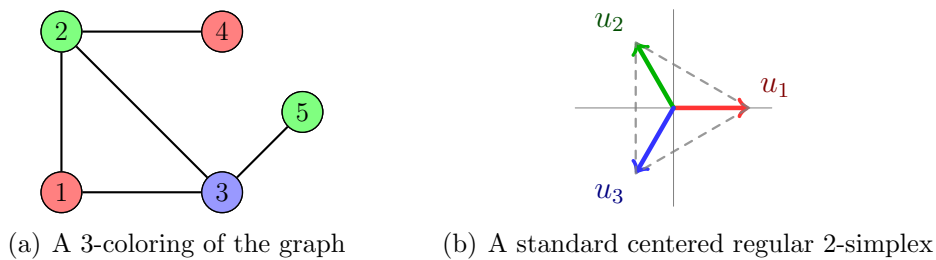


FIGURE 4.23: Graphical representation of Example 4.11.

Proposition 4.12. *Let $G = (V, E)$ be a graph with n nodes and let K be a set of m colors. Consider a matrix $X \in \mathbb{R}^{n \times n}$ that verifies properties (P1)–(P3). Then, there exists a proper m -coloring $c : V \mapsto K$ such that*

$$X = U_c^T U_c,$$

where U_c is given by (4.12).

Proof. Consider the spectral decomposition $X = Q\Lambda Q^T$, where $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$ is the diagonal matrix of eigenvalues. Since X is positive definite and has rank not greater than $m - 1$, we can assume without loss of generality that

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_{m-1} \geq 0 = \lambda_m = \dots = \lambda_n.$$

Then, we can express

$$X = Q\Lambda Q^T = \begin{pmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{pmatrix} \begin{pmatrix} \hat{\Lambda} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} Q_{11}^T & Q_{21}^T \\ Q_{12}^T & Q_{22}^T \end{pmatrix} = \begin{pmatrix} Q_{11} \\ Q_{21} \end{pmatrix} \hat{\Lambda} \begin{pmatrix} Q_{11}^T & Q_{21}^T \end{pmatrix},$$

with $\hat{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_{m-1})$. Hence, we can factorize $X = Y^T Y$, with

$$Y = \hat{\Lambda}^{\frac{1}{2}} \begin{pmatrix} Q_{11}^T & Q_{21}^T \end{pmatrix}.$$

Let $y_1, \dots, y_n \in \mathbb{R}^{m-1}$ be the columns of Y , i.e., $Y = [y_1 | y_2 | \dots | y_n]$. Observe that y_1, \dots, y_n are unit vectors because X has ones on the diagonal, and thus

$$\langle y_i, y_j \rangle = \begin{cases} 1, & \text{if } y_i = y_j, \\ \frac{-1}{m-1}, & \text{if } y_i \neq y_j; \end{cases} \quad (4.14)$$

for all $i, j = 1, \dots, n$. Let us show now that there are at most m distinct vectors among them. To this aim, suppose that $y_{i_1}, y_{i_2}, \dots, y_{i_{m+1}}$ are $m + 1$ different vectors. Consider

$$\tilde{X} := \begin{bmatrix} y_{i_1}^T \\ y_{i_2}^T \\ \vdots \\ y_{i_{m+1}}^T \end{bmatrix} [y_{i_1} | y_{i_2} | \dots | y_{i_{m+1}}] = \begin{pmatrix} 1 & \frac{-1}{m-1} & \dots & \frac{-1}{m-1} \\ \frac{-1}{m-1} & 1 & \dots & \frac{-1}{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{-1}{m-1} & \frac{-1}{m-1} & \dots & 1 \end{pmatrix} \in \mathbb{R}^{(m+1) \times (m+1)}.$$

It holds that $\text{rank}(\tilde{X}) \leq \text{rank}(X) \leq m - 1$, since \tilde{X} is a submatrix of X , but this is a contradiction with the fact that

$$\det(\tilde{X}) = \left(1 - \frac{m}{m-1}\right) \left(1 + \frac{1}{m-1}\right)^m \neq 0.$$

Therefore, it must hold that $\cup_{j=1}^n \{y_j\} = \{u_1, \dots, u_r\}$, where u_1, \dots, u_r are $r \leq m$ distinct vertices of a regular $(m-1)$ -simplex (a rotation of the standard simplex). Finally, define $c : V \mapsto K$ by $c(i) = \{k \in \{1, \dots, r\} : y_i = u_k\}$, so that we trivially get $Y = U_c$, where U_c is as in (4.12). According to (P3), together with (4.14), we have that c is a proper m -coloring of G , as claimed. \square

In view of Proposition 4.12, finding a proper m -coloring of a graph with n vertices is equivalent to finding an $n \times n$ matrix verifying properties (P1), (P2) and (P3). Therefore, the latter will be tackled by solving the following feasibility problem:

$$\text{Find } X \in C_1 \cap C_2 \subseteq \mathbb{R}^{n \times n}, \quad (4.15)$$

where the constraint sets are defined by

$$C_1 := \left\{ X \in \left\{ 1, \frac{-1}{m-1} \right\}^{n \times n} : x_{ii} = 1, \forall i \in V \text{ and } x_{ij} = \frac{-1}{m-1}, \forall \{i, j\} \in E \right\}, \quad (4.16a)$$

$$C_2 := \{X \in \mathcal{S}_+^n : \text{rank}(X) \leq m - 1\}. \quad (4.16b)$$

REMARK 4.13. One advantage of the feasibility problem (4.15) is the avoidance of equivalent colorings in the following sense. Suppose that $c : V \mapsto K$ is a proper m -coloring of a graph G , and let W_c be its associated matrix given by (4.13). For any permutation of the colors, $\sigma : K \mapsto K$, we have that $\sigma \circ c$ is also a proper m -coloring of G , so there exist many equivalent valid colorings. However, observe that $W_{\sigma \circ c} = W_c$, and thus all of them lead to a unique solution of (4.15).

Modeling precoloring problems

Precoloring problems can be modeled by a slight modification of the feasibility problem in (4.15). Let $\tilde{V} \subseteq V$ be the subset of precolored nodes and denote by $p_i \in K$ the preassigned color to node $i \in \tilde{V}$. The task then is to find a coloring $c : V \mapsto K$ such that

$$c(i) \neq c(j), \text{ for all } \{i, j\} \in E \quad \text{and} \quad c(i) = p_i, \text{ for all } i \in \tilde{V}. \quad (4.17)$$

Notice that any coloring satisfying (4.17) also verifies

$$c(i) \neq c(j), \text{ for all } \{i, j\} \in E \quad \text{and} \quad c(i) = c(j) \Leftrightarrow p_i = p_j, \text{ for all } i, j \in \tilde{V}. \quad (4.18)$$

In fact, both conditions can be shown to be equivalent in the following sense. Suppose that $c : V \mapsto K$ is a coloring verifying (4.18). Then, for any permutation of the colors $\sigma : K \mapsto K$ such that $\sigma(c(i)) = p_i$ for all $i \in \tilde{V}$, one can easily check that $\sigma \circ c$ is a proper coloring for which (4.17) holds.

Therefore, we shall focus on finding colorings fulfilling condition (4.18). The matrix W_c constructed from c as in (4.13), shall verify now (P1), (P2) and

(P3') $W_c \in \left\{1, \frac{-1}{m-1}\right\}^{n \times n}$ and some of the entries of $W_c = [w_{ij}]$ are determined as follows:

$$w_{ij} = 1, \forall \{i, j\} \in \hat{I} \quad \text{and} \quad w_{ij} = \frac{-1}{m-1}, \forall \{i, j\} \in \hat{E};$$

where $\hat{I} := \{\{i, i\} : i \in V\} \cup \{\{i, j\} \subseteq \tilde{V} : p_i = p_j\}$ and $\hat{E} := E \cup \{\{i, j\} \subseteq \tilde{V} : p_i \neq p_j\}$.

The new modified property (P3') can be incorporated into the formulation of the feasibility problem (4.15) by replacing the constraint C_1 by

$$\hat{C}_1 := \left\{ X \in \left\{1, \frac{-1}{m-1}\right\}^{n \times n} : x_{ij} = 1, \forall \{i, j\} \in \hat{I} \text{ and } x_{ij} = \frac{-1}{m-1}, \forall \{i, j\} \in \hat{E} \right\}. \quad (4.19)$$

Example 4.14 (Example 4.11 revisited). Consider the graph in Example 4.11 and suppose that node 2 is precolored red (R), and nodes 4 and 5 are precolored blue (B). The precoloring problem is shown in Figure 4.24(a). Following the notation established above, we have

$$\hat{I} = \{\{i, i\} : i \in V\} \cup \{\{4, 5\}\} \quad \text{and} \quad \hat{E} = E \cup \{\{2, 4\}, \{2, 5\}\} = E \cup \{\{2, 5\}\}.$$

The unique solution to the feasibility problem $\hat{C}_1 \cap C_2$ is the matrix

$$W_c = \begin{pmatrix} \boxed{1} & \boxed{-0.5} & \boxed{-0.5} & 1 & 1 \\ \boxed{-0.5} & \boxed{1} & \boxed{-0.5} & \boxed{-0.5} & \boxed{-0.5} \\ \boxed{-0.5} & \boxed{-0.5} & \boxed{1} & -0.5 & \boxed{-0.5} \\ 1 & \boxed{-0.5} & -0.5 & \boxed{1} & \boxed{1} \\ 1 & \boxed{-0.5} & \boxed{-0.5} & \boxed{1} & \boxed{1} \end{pmatrix},$$

where the boxed entries in W_c correspond to those determined by (P3'). The entries whose values are fixed by (P3') but not by (P3) are marked with a double-box.

Suppose that we obtain the factorization $W_c = U_c^T U_c$, with $U_c = [u_1, u_2, u_3, u_1, u_1]$. The 3-coloring determined by U_c is represented in Figure 4.24(b). Then, in order to make this coloring consistent with the precoloring of the vertices, we need to suitably permute the set of colors. Precisely, we require $\sigma(G) = R$ and $\sigma(R) = B$. It must therefore be $\sigma(B) = G$. The permuted 3-coloring consistent with the precoloring, given by $U_{\sigma \circ c} = [u_3, u_1, u_2, u_3, u_3]$, is shown in Figure 4.24(c).

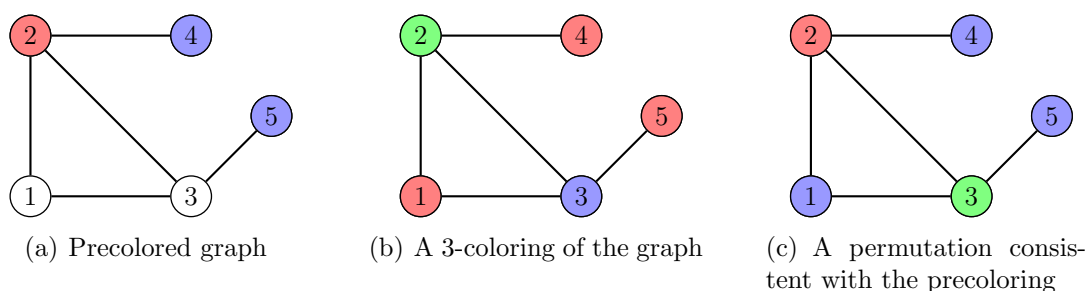


FIGURE 4.24: Graphical representation of Example 4.14

4.1.3.2 Implementation of the Douglas–Rachford algorithm

Projecting onto the constraints

In order to apply any projection algorithm to feasibility problems, and (4.15) in particular, it must be possible to efficiently compute the projections onto the two constraint sets, in our case (4.16). This is indeed the case, as shown in the following results.

Proposition 4.15 (Projection onto C_1). Consider any $X = (x_{ij}) \in \mathbb{R}^{n \times n}$. A projection of X onto the set C_1 defined in (4.16a) is given componentwise by

$$\left(\pi_{C_1}(X)\right)[i, j] = \begin{cases} 1, & \text{if } x_{ij} > \frac{m-2}{2(m-1)} \text{ and } \{i, j\} \notin E, \text{ or } i = j; \\ \frac{-1}{m-1}, & \text{if } x_{ij} \leq \frac{m-2}{2(m-1)} \text{ and } i \neq j, \text{ or } \{i, j\} \in E. \end{cases} \quad (4.20)$$

A projection of X onto the set \widehat{C}_1 in (4.19) is given componentwise by

$$\left(\pi_{\widehat{C}_1}(X)\right)[i, j] = \begin{cases} 1, & \text{if } x_{ij} > \frac{m-2}{2(m-1)} \text{ and } \{i, j\} \notin \widehat{E}, \text{ or } \{i, j\} \in \widehat{I}; \\ \frac{-1}{m-1}, & \text{if } x_{ij} \leq \frac{m-2}{2(m-1)} \text{ and } \{i, j\} \notin \widehat{I}, \text{ or } \{i, j\} \in \widehat{E}. \end{cases} \quad (4.21)$$

Proof. Clearly, the projector of X onto C_1 can be computed componentwise. Taking into account the constraints in (4.16a), the projection of an entry x_{ij} is 1 if $i = j$, and is $\frac{-1}{m-1}$

if $\{i, j\} \in E$. Otherwise, it is equal to $P_{\{1, \frac{m-1}{2}\}}(x_{ij})$. As the middle point between these two values is $\frac{m-2}{2(m-1)}$, then (4.20) follows. The proof of (4.21) is analogous. \square

Proposition 4.16 (Projection onto C_2). *Let $X \in \mathcal{S}^n$ and consider its spectral decomposition $X = Q\Lambda Q^T$, with $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$ and $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$. A projection of X onto the set C_2 defined in (4.16b) is given componentwise by*

$$\pi_{C_2}(X) = Q\Lambda_{m-1}^+ Q^T, \quad (4.22)$$

where $\Lambda_{m-1}^+ = \text{diag}(\max\{0, \lambda_1\}, \dots, \max\{0, \lambda_{m-1}\}, 0, \dots, 0)$.

Proof. See Proposition 1.49. \square

REMARK 4.17. According to Propositions 4.15 and 4.16, computing a projection onto C_1 is a simple rounding operation, while a projection onto C_2 requires the computation of the spectral decomposition of an $n \times n$ matrix. From a computational point of view, the former is not a problem but the later may be time-consuming, especially for big problems. However, observe that we do not need to compute the whole spectrum in (4.22), but only the $m - 1$ largest eigenvalues and their associated eigenvectors. In large-scale problems, m is usually much smaller than n and hence π_{C_2} can be computed reasonably fast.

The nonconvexity of the constraints manifests itself in the equality case of the conditionals in Proposition 4.15, and the case of degenerate eigenvalues in Proposition 4.16. None of them can be acted upon in practice, given the finite precision of the computations.

REMARK 4.18. In order to find $\pi_{C_2}(X)$, Proposition 4.16 requires the matrix X to be symmetric. Observe that, according to (4.20) and by definition of C_2 , we get that

$$\pi_{C_1}(X), \pi_{C_2}(X) \in \mathcal{S}^n, \quad \text{for all } X \in \mathcal{S}^n.$$

Hence, since \mathcal{S}^n is a subspace, the iterates generated by DR (2.26) will remain symmetric (with due attention to numerical precision), as long as the initial point is chosen in \mathcal{S}^n .

Variety of implementations

For a more complete experimentation, instead of the classical Douglas–Rachford algorithm we consider its generalized version GDR (2.19), so that the parameter α can play a role in the rate of convergence and success of the method. There are several options for implementing the GDR algorithm. The simplest choice would be to directly apply GDR in the original space $\mathbb{R}^{n \times n}$, since the feasibility problem to be solved (4.15) only involves

two constraint sets. Then, we can iterate by using either $T_{C_1, C_2, \alpha}$ or $T_{C_2, C_1, \alpha}$. On the other hand, although the product space reformulation (Section 2.1.1) is typically employed for feasibility problems involving more than two sets, it can be still applied to two sets. In this way, we obtain two additional implementations by either using the operator $T_{D, C, \alpha}$ or $T_{C, D, \alpha}$, where C and D are the sets defined in (2.4). The purpose of the next section is to numerically compare these different implementations. In our tests we observed that the numerical behavior of $T_{D, C, \alpha}$ and $T_{C, D, \alpha}$ is similar; thus, to simplify, we only show the results with the operator $T_{D, C, \alpha}$, whose shadow sequence is easier to track, as it can be identified with a sequence in the original space $\mathbb{R}^{n \times n}$.

4.1.3.3 Numerical experiments

In this section we run various numerical experiments to test the performance of the GDR algorithm for solving different graph coloring problems. We compare the formulation discussed in Section 4.1.3.1 with the one proposed in Section 4.1.2. To distinguish them, we shall refer to the model proposed in Section 4.1.2 as the *binary formulation*, and to the new one developed in Section 4.1.3.1 as the *rank formulation*.

Different families of graphs are taken into consideration: the Queens $_n$ puzzles, random colorable graphs, the windmill graphs, Sudokus, and the DIMACS benchmark instances. Each of these families is employed for a different purpose. We start with a difficult coloring problem, the Queens $_n$ puzzle, where we show the effect that the parameter α has in the different implementations. We also use these puzzles to draw attention to something that is usually overlooked: finite machine precision. Next, to test how the method scales, we run an experiment on random colorable graphs with controlled asymptotic complexity. The windmill graphs and the Sudoku puzzles are used to show that the rank formulation is superior to the binary formulation, even when we allow maximal clique information. We finish this experimental section by testing the algorithm on the DIMACS benchmark instances, a widely used collection of diverse graph types.

Unless otherwise is stated, the stopping criterion used for each implementation of the form $T_{A, B, \alpha}$ was

$$\text{Error}_{A, B}(x_k) := \|P_B(P_A(x_k)) - P_A(x_k)\| \leq 10^{-10}, \quad (4.23)$$

where x_k is the current iterate, in which case the instance was labeled as successful. All codes were written in Python 2.7 and the tests were run on an Intel Core i7-4770 CPU 3.40 GHz with 32 GB RAM, under Windows 10 (64-bit).

Queens $_n^2$ puzzles

A well-known and challenging graph coloring problem is the Queens $_n^2$ puzzle. This is a puzzle that consists in covering an entire $n \times n$ chessboard with queens of different colors, so that two queens of the same color do not attack each other (see Remark 4.7). The puzzle is equivalent to finding a proper coloring of the queens graph. In Table 4.8 we show the chromatic number of the graph for the first nine puzzles. The smallest open case for which the chromatic number is currently unknown is $n = 27$, see [121].

n	2	3	4	5	6	7	8	9	10
$\chi(n)$	4	5	5	5	7	7	9	10	11

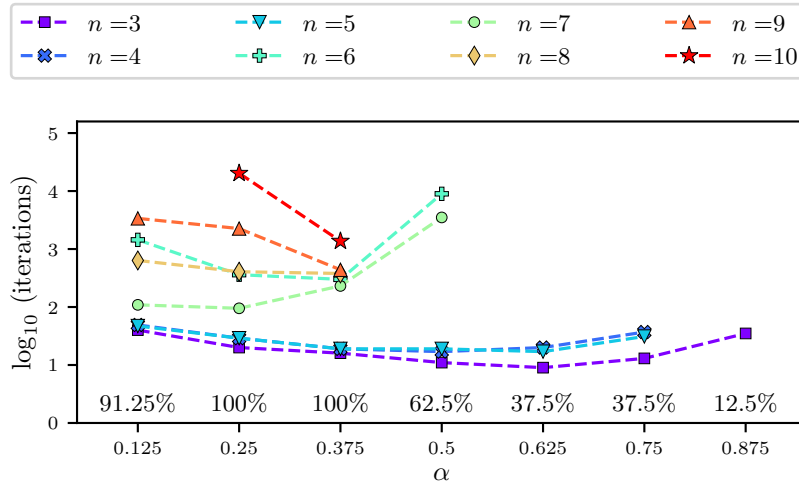
TABLE 4.8: Chromatic number $\chi(n)$ of the Queens $_n^2$ graph.

In our first experiment, we analyze how both the implementation and the choice in the relaxation parameter α affects the behavior of GDR for solving this type of puzzles. For each $n \in \{3, 4, \dots, 10\}$ and each $\alpha \in \{0.125, 0.25, \dots, 0.875\}$, we ran three different implementations of GDR (namely, $T_{C_1, C_2, \alpha}$, $T_{C_2, C_1, \alpha}$ and $T_{\mathcal{D}, \mathcal{C}, \alpha}$) from 10 random starting points. The results are shown in Figure 4.25, where the markers correspond to the median among the solved instances. We also show the percentage of instances solved for each value of α , among all the problems and repetitions. According to these results, it seems that the value of the parameter α that suits best each of the formulations $T_{C_1, C_2, \alpha}$, $T_{C_2, C_1, \alpha}$ and $T_{\mathcal{D}, \mathcal{C}, \alpha}$, is $\alpha = 0.375$, $\alpha = 0.25$ and $\alpha = 0.5$, respectively.

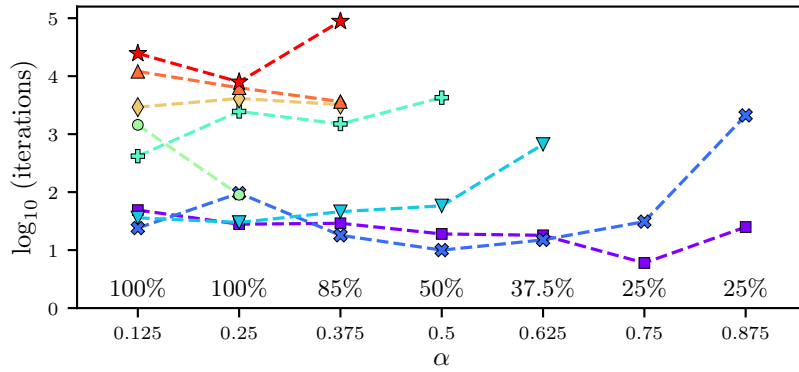
To corroborate the previous conclusion, we visualize the results using performance profiles (see Remark 4.10). We show in Figure 4.26 the performance profiles for each value of α and each of the three implementations $T_{C_1, C_2, \alpha}$, $T_{C_2, C_1, \alpha}$ and $T_{\mathcal{D}, \mathcal{C}, \alpha}$. This corroborates our previous choice of best parameters α for each implementation. Finally, we compare $T_{C_1, C_2, 0.375}$, $T_{C_2, C_1, 0.25}$ and $T_{\mathcal{D}, \mathcal{C}, 0.5}$ in Figure 4.27, where we can clearly observe that the first implementation dominates the others.

REMARK 4.19 (On the machine precision). Our numerical tests show no systematic effect of the machine precision on the average number of iterations per solution, provided the precision is above a modest threshold of about 6 decimal digits. This is consistent with the chaotic dynamics displayed by GDR when solving hard problems.

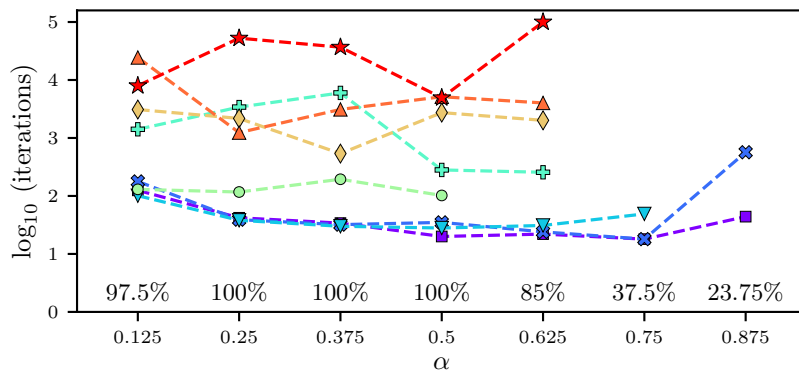
The behavior explained in Remark 4.19 is demonstrated in the next experiment, where $T_{\mathcal{D}, \mathcal{C}, 0.5}$ was implemented for solving the Queens $_6^2$ and the Queens $_7^2$ puzzles. For each problem, the algorithm was run from the same starting point using different values of the machine precision. The stopping criterion (4.23) was decreased to 10^{-5} to accommodate the reduced precision. We believe this is still adequate to recover a unique, discrete



(a) GDR implemented with $T_{C_1, C_2, \alpha}$



(b) GDR implemented with $T_{C_2, C_1, \alpha}$



(c) GDR implemented with $T_{D, C, \alpha}$

FIGURE 4.25: Results of the $Queens_{n^2}$ experiment for three implementations of GDR. Each marker corresponds to the median of the solved instances among 10 random starting points. At the bottom of each graph, we show the percentage of solved instances for each value of α . Instances were considered as unsolved after 100,000 iterations.

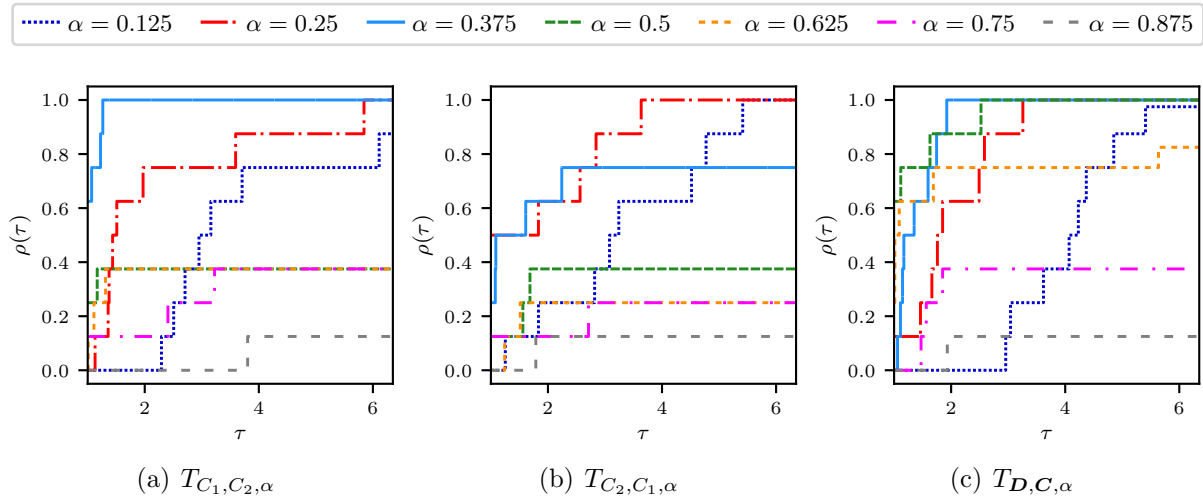


FIGURE 4.26: Performance profiles of the Queens_{n^2} experiment for three implementations of GDR.

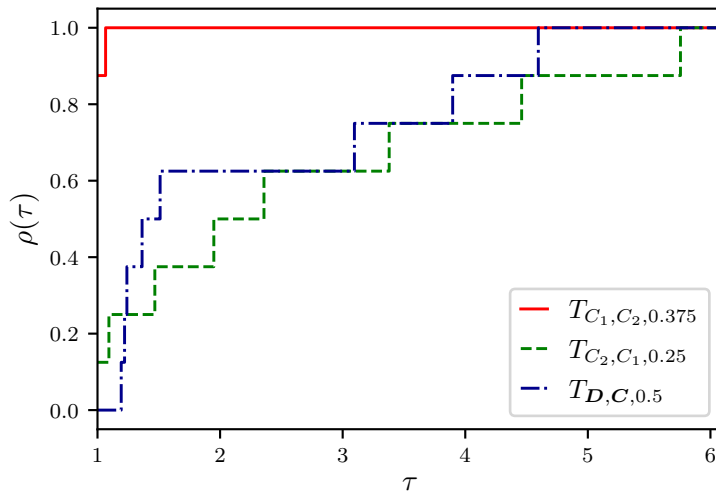


FIGURE 4.27: Performance profiles of the Queens_{n^2} experiment comparing the implementations $T_{C_1, C_2, 0.375}$, $T_{C_2, C_1, 0.25}$ and $T_{D, C, 0.5}$.

coloring from the Gram matrix. The results of repeating this experiment for 10 different random starting points are shown in Figure 4.28. In Figure 4.29 we plot the value of $\text{Error}_{D, C}$ in (4.23) with respect to the number of iterations for up to 15 digits of precision for one particular random starting point. While these results indicate a high sensitivity to the numerical precision, there is no evidence of a systematic effect. For these experiments, we employed the `mpmath` library [124], which drastically increases the time needed to compute the iterations of the GDR algorithm.

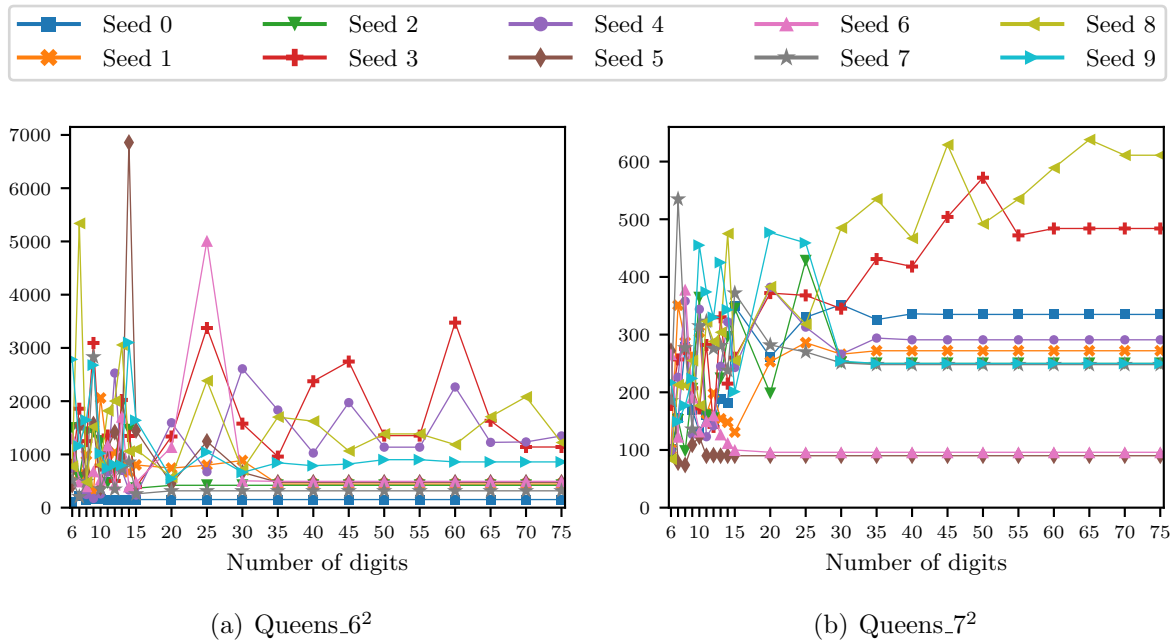


FIGURE 4.28: Comparison of the number of number of iterations and the number of digits used in the machine precision for 10 random starting points, when $T_{D,C,0.5}$ was employed to solve the Queens. $_6^2$ and the Queens. $_7^2$ puzzles. For every starting point and every value of the machine precision, the algorithm found a solution to the puzzle.

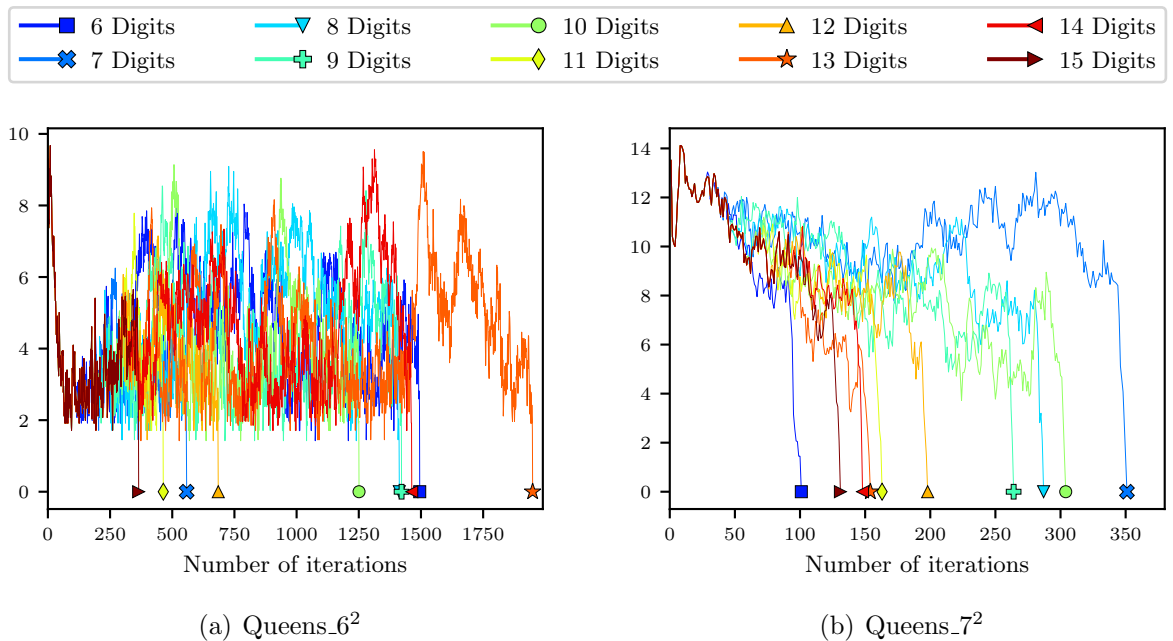


FIGURE 4.29: Comparison of the value of Error in (4.23) and the number of iterations for different number of decimal digits used in the machine precision, when $T_{D,C,0.5}$ was employed to solve the Queens. $_6^2$ and the Queens. $_7^2$ puzzles.

Random colorable graphs

The hardness of finding a proper coloring of a graph depends on many factors, the single most significant of which is the number of valid colorings. Random colorable graphs are easily constructed, but to be able to draw some consequences from the experiments we run on them, we must generate them in such a way that their complexity is controlled.

We consider the Erdős–Renyi model [96], $\mathbb{G}(\delta, n)$, which is the ensemble of all graphs with n vertices and $l = \lfloor \delta n \rfloor$ edges, where $\lfloor \cdot \rfloor$ denotes the integer part, endowed with the uniform measure. Hence, δ represents the averaged number of edges per node. The probability that a random graph with this distribution is m -colorable depends on the magnitude of the parameter δ . Precisely, the expected number of proper colorings decreases as δ increases. There is an *asymptotic threshold* in the colorable-uncolorable transition denoted $\delta_s(m)$ (see [1, Theorem 1.1]). This means that the probability an m -coloring exists tends to one as n increases, provided that $\delta < \delta_s(m)$, and conversely, it converges to zero for $\delta > \delta_s(m)$. The asymptotic threshold is known to be upper-bounded by $\delta_s(m) \leq \bar{\delta}_s(m) := \frac{\log m}{\log \frac{m}{m-1}}$ (see, e.g., [2, Section 2]).

With the number of vertices n and the number of colors m fixed, random graphs sampled from $\mathbb{G}(\bar{\delta}_s(m), n)$ are at the m -colorability transition and expected to be hard instances, when solvable. In order to avoid non-colorable graphs, the sampling can be modified to ensure the existence of a coloring as follows. First, a partition of V into m groups with approximately equal size is chosen, e.g. consider the equivalence classes defined by the congruence modulo m of the integer vertex labels. Then, $\lfloor \bar{\delta}_s(m)n \rfloor$ edges are randomly generated from the uniform distribution over the set of all edges connecting two nodes in different groups. Algorithm 1 contains the discussed routine that generates such graphs.

Algorithm 1: Generate an m -colorable random graph with low expected number of m -colorings.

Input: $V = \{1, \dots, n\}$, $m \geq 2$

Set $\bar{\delta}_s(m) := \frac{\log m}{\log \frac{m}{m-1}}$, $E := \emptyset$ and $l = 0$;

while $l < \bar{\delta}_s(m)n$ **do**

Generate randomly $e := \{i, j\} \in V \times V$;

if $e \notin E$ and $(i - j) \not\equiv 0 \pmod{m}$ **then**

$E = E \cup \{e\}$;

$l = l + 1$;

Output: $G = (V, E)$

The goal of our next experiment is to show how the GDR algorithm complexity grows with respect to the number of vertices in the graph. We make use of colorable random graphs with low expected number of colors so that we have control of the complexity of our instances. For each $m \in \{8, 9, 10\}$ and for each $n \in \{50, 75, \dots, 200\}$, we generated 5 random graphs using Algorithm 1. Then, for each graph, the GDR algorithm was run from 5 different starting points (this makes a total of 25 runs per each pair (m, n)). Based on the results in the Queens- n^2 experiment, we implemented GDR with $T_{C_1, C_2, 0.375}$. In Figure 4.30 we plot the number of iterations needed by the algorithm with respect to the size of the graph, for each m . We can observe that the number of colors does not have a noticeable effect on the performance of the algorithm. As expected, we come upon an exponential dependence between size and number of iterations to find a coloring, which is consistent with the NP-hardness of the problem.

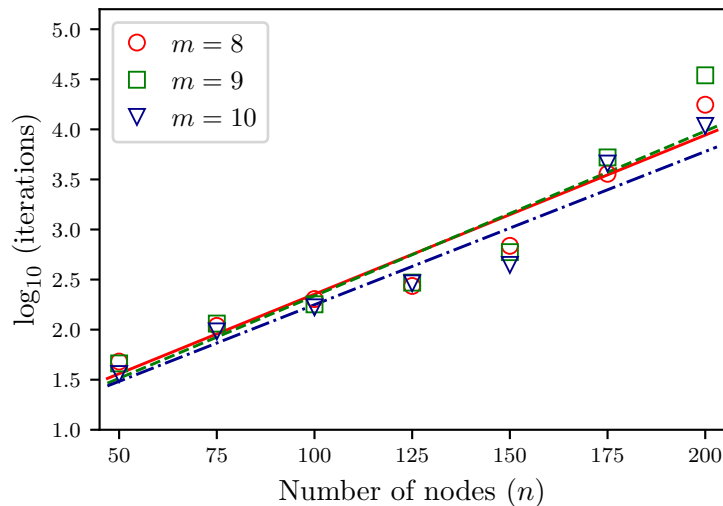
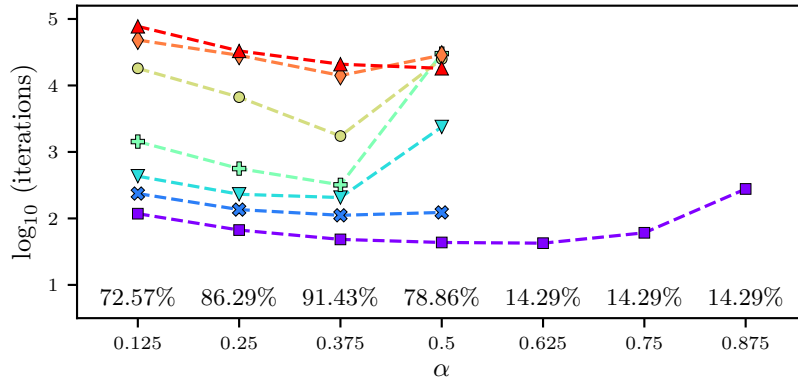
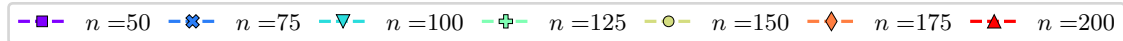
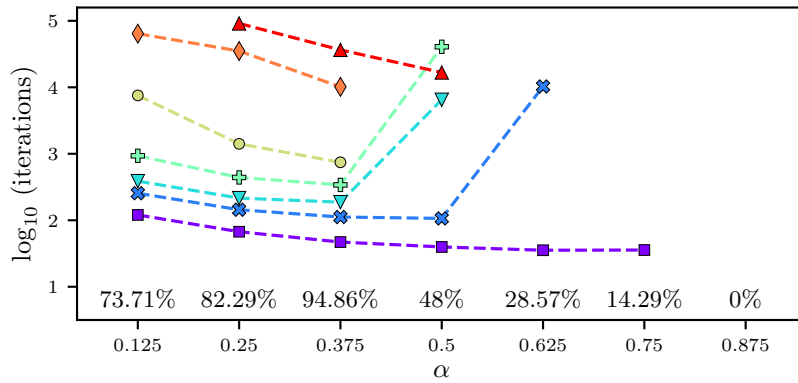


FIGURE 4.30: Results of the experiment on m -colorable random graphs for $m = 8, 9, 10$, for GDR implemented with $T_{C_1, C_2, 0.375}$. Each marker corresponds to the median of the solved instances among 10 random starting points, and the lines were obtained by linear regression among all the solved instances.

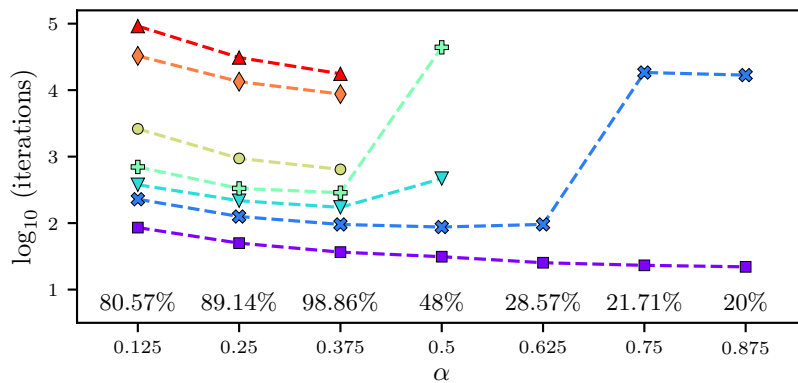
To confirm our previous choice of the best parameter $\alpha = 0.375$, we repeated the experiment for each $\alpha \in \{0.125, 0.375, \dots, 0.875\}$ with the implementation $T_{C_1, C_2, \alpha}$. The results are shown in Figure 4.31, where again the markers correspond to the median among the solved instances, and the percentage of instances solved for each value of α is computed among all the problems and repetitions. This results confirm that the best choice for general purposes is $\alpha = 0.375$.



(a) 8 colors



(b) 9 colors



(c) 10 colors

FIGURE 4.31: Results of the experiment on m -colorable random graphs for $m = 8, 9, 10$, for the implementation $T_{C_1, C_2, \alpha}$ of GDR. Each marker corresponds to the median of the solved instances among 10 random starting points. At the bottom of each graph, we show the percentage of solved instances for each value of α . Instances were considered as unsolved after 100,000 iterations.

Windmill graphs

We turn our attention to windmill graphs (see Figure 4.13). Recall that a windmill graph $Wd(a, b)$ can be easily a -colored in $a((a-1)!)^b$ different ways. Despite this abundance of valid colorings, all of them are equivalent under a permutation of the colors. Thus, by Remark 4.13, there exists a unique solution to the rank feasibility problem. This is not the case, however, for the binary formulation.

In our tests with the binary formulation (see Figure 4.17 and Table 4.4), the DR algorithm fails to find colorings of windmill graphs rather often. We tentatively attribute this to the high multiplicity of solutions in the binary formulation. In Section 4.1.2.3 we have addressed this problem by augmenting the model with information about the maximal cliques of the graph (see Table 4.4). The set of maximal cliques would normally be unknown (and difficult to find) for general graphs, so a formulation that does not require this information would generally be preferred.

In our next experiment, whose results are shown in Table 4.9, we compare the binary formulation with and without maximal clique information, and the rank formulation for coloring sixteen windmill graphs of different parameters.

Wd(a, b)		Binary formulation			Binary formulation with clique info.			Rank formulation		
a	b	Success	Time	Iter.	Success	Time	Iter.	Success	Time	Iter.
5	5	10/10	0.05	226	10/10	0.02	63	10/10	0.01	20
	10	10/10	0.13	375	10/10	0.04	93	10/10	0.02	33
	15	9/10	0.23	503	10/10	0.06	135	10/10	0.03	43
	20	9/10	0.3	521	10/10	0.1	170	10/10	0.05	51
10	5	1/10	1.12	1886	10/10	0.12	200	10/10	0.03	33
	10	0/10	-	-	10/10	0.27	242	10/10	0.08	54
	15	0/10	-	-	10/10	3.47	1729	10/10	0.24	86
	20	0/10	-	-	10/10	5.2	1531	10/10	0.45	109
15	5	0/10	-	-	10/10	0.54	330	10/10	0.06	44
	10	0/10	-	-	10/10	1.69	369	10/10	0.29	92
	15	0/10	-	-	10/10	5.21	588	10/10	0.85	144
	20	0/10	-	-	10/10	13.35	949	10/10	1.69	180
20	5	0/10	-	-	10/10	2.62	642	10/10	0.15	68
	10	0/10	-	-	10/10	12.52	1059	10/10	0.63	119
	15	0/10	-	-	10/10	16.95	729	10/10	1.83	170
	20	0/10	-	-	8/10	31.76	828	10/10	7.3	297

TABLE 4.9: Summary of the results of GDR for finding proper colorings of windmill graphs. For each formulation, we show the number of solved instances, the averaged time (in seconds) and the averaged number of iterations. Instances were considered as unsolved after 60 seconds.

We observe that the addition of maximal clique information is crucial for the success of the binary formulation. Without adding it, the DR algorithm was not able to find any solutions for even modestly large values of a . On the other hand, the superior performance of the rank formulation for this graph is apparent, both in terms of number of iterations and time. We emphasize again that the rank formulation does not use maximal clique information, and despite this, it achieved a success rate of 100%.

Sudokus

In our next experiment, we compare the precoloring rank matrix model (with $T_{C_1, C_2, 0.375}$) and the binary formulation for precoloring stated in Section 4.1.2.2, with maximal clique information included in the latter. We also incorporate to the experiment the cubic formulation for Sudokus (see Remark 4.4). For each of these three formulations and each of the 95 puzzles in `top95` (the same Sudoku data set used in the experiments in Section 4.1.2.3), the GDR algorithm was run from 10 random starting points. The performance profiles of the results are displayed in Figure 4.32.

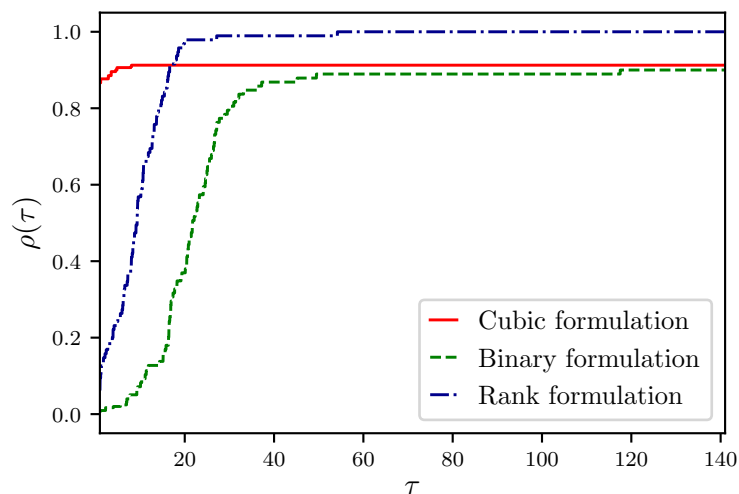


FIGURE 4.32: Performance profiles comparing the binary and the rank matrix formulations for solving 95 Sudoku problems. For each problem, 10 starting points were randomly generated. Instances were considered as unsolved after 300 seconds.

The cubic formulation was the fastest in 86.36% of the instances. On average it solved a Sudoku in 4.65 seconds, while the binary and rank formulations needed 35.8 and 13.79 seconds, respectively. Regarding the success of the algorithm, the cubic and binary formulations solved about 90% of the instances. The rank formulation was the clear winner in terms of success, as it solved every single instance, even for those puzzles listed in Table 4.7, on which DR was observed to be highly unsuccessful.

To further challenge the rank formulation, we performed experiments on the so-called ‘nasty’ Sudoku (shown in Figure 4.33). The ‘nasty’ Sudoku has very low success rate in the cubic formulation (see [10, Section 6.5]), as the algorithm almost always enters a limit cycle (see [10, Table 4]). This is not the case, however, for the rank formulation. In our next experiment we compare the cubic, binary and rank formulations for solving the ‘nasty’ Sudoku from 100 random starting points. The results are summarized in Figure 4.33. The rank formulation obtained again a success rate of 100%. The second most successful formulation was the binary one, which was only able to find a solution for 19% of the starting points. So far, we have not been able to find any Sudoku on which the rank formulation failed to find a solution for some starting point.

7				9	5	
	1				3	
		2	3		7	
		4	5		7	
8					2	
				6	4	
	9			1		
	8			6		
		5	4			7

Time	Cubic		Binary		Rank	
	Inst.	Cumul.	Inst.	Cumul.	Inst.	Cumul.
0-24	12	12%	15	15%	61	61%
25-49	0	12%	2	17%	36	97%
50-99	0	12%	1	18%	3	100%
100-299	0	12%	1	19%	0	100%
Unsolved	88	100%	81	100%	0	100%

FIGURE 4.33: Number of solved instances (right), among 100 random starting points, to find the solution of the ‘nasty’ Sudoku (left) by GDR with the cubic, the binary, and the rank formulations. For each interval of time (in seconds), we show the number of solved instances and the cumulative proportion of solved instances for each formulation. The algorithm was stopped after a maximum of 300 seconds, in which case the problem was labeled as “Unsolved”.

DIMACS benchmark instances

In our final experiment, we test the rank formulation on the widely used graph coloring library from DIMACS benchmark instances⁵. This collection contains various classes of graphs, such as random or quasi-random graphs, problems based on register allocation for variables in real codes, or class scheduling graphs, among others.

The GDR algorithm was applied to a wide sample of the aforementioned benchmark instances. Guided by the results in the previous experiments, we used the implementation $T_{C_1, C_2, 0.375}$. For each graph, the algorithm was run from 10 random starting points and was stopped after a maximum time of one hour. In Table 4.10 we present the results of the experiment, as well as the main features of the selected instances. The unsuccessful instances mainly occurred on the very large graphs, on which the algorithm may have succeeded given more time.

⁵DIMACS benchmark instances: <http://cse.unl.edu/~tnguyen/npbenchmarks/graphcoloring.html>

Instances	Nodes	Edges	Colors	Success	Iter	Time (s)
fpsol2.i.1	496	11,654	65	10/10	8,984	463.94
fpsol2.i.2	451	8,691	30	10/10	13,316	495.94
fpsol2.i.3	425	8,688	30	10/10	14,454	480.27
inithx.i.1	864	18,707	54	10/10	16,174	2,443.43
inithx.i.2	645	13,979	31	10/10	20,049	1,500.45
inithx.i.3	621	13,969	31	10/10	20,604	1,432.43
le450_15a	450	8,168	15	4/10	61,365	1,944.35
le450_15b	450	8,169	15	8/10	65,537	2,076.54
le450_15c	450	16,680	15	10/10	5,464	173.1
le450_15d	450	16,750	15	10/10	19,718	619.74
le450_25a	450	8,260	25	10/10	1,938	68.93
le450_25b	450	8,263	25	10/10	1,849	65.82
le450_25c	450	17,343	25	0/10	-	-
le450_25d	450	17,425	25	0/10	-	-
le450_5a	450	5,714	5	10/10	3,071	82.47
le450_5b	450	5,734	5	10/10	8,885	238.33
le450_5c	450	9,803	5	10/10	3,212	86.68
le450_5d	450	9,757	5	10/10	1,644	44.49
mulsol.i.1	197	3,925	49	10/10	2,331	18.79
mulsol.i.2	188	3,885	31	10/10	8,696	63.18
mulsol.i.3	184	3,916	31	10/10	7,814	55.88
mulsol.i.4	185	3,946	31	10/10	8,584	60.71
mulsol.i.5	186	3,973	31	10/10	8,685	62.72
zeroin.i.1	211	4,100	49	10/10	3,014	27.1
zeroin.i.2	211	3,541	30	10/10	4,775	39.08
zeroin.i.3	206	3,540	30	10/10	4,286	34.51
anna	138	493	11	10/10	354	1.04
david	87	406	11	10/10	167	0.26
homer	561	1,628	13	10/10	1,222	59.01
huck	74	301	11	10/10	81	0.11
jean	80	254	10	10/10	98	0.13
miles1000	128	3,216	42	10/10	570	2.43
miles1500	128	5,198	73	10/10	4,736	24.65
miles250	128	387	8	10/10	173	0.4
miles500	128	1,170	20	10/10	307	1.07
miles750	128	2,113	31	10/10	671	2.54
myciel3	11	20	4	10/10	7	0.0
myciel4	23	71	5	10/10	15	0.0
myciel5	47	236	6	10/10	41	0.03
myciel6	95	755	7	10/10	179	0.26
myciel7	191	2,360	8	9/10	377	1.52
mug88_1	88	146	4	10/10	43	0.05
mug88_25	88	146	4	10/10	46	0.05
mug100_1	100	166	4	10/10	54	0.07
mug100_25	100	166	4	10/10	47	0.06

TABLE 4.10: Summary of the results of the GDR algorithm implemented with $T_{C_1, C_2, 0.375}$ for finding proper colorings of a representative sample of DIMACS benchmark instances. For each problem, we show the number of solved runs, the average time (in seconds) and the average number of iterations. We also include the number of nodes and edges, and the chromatic number of each graph. Runs were considered as unsolved after 3600 seconds.

4.2 Combinatorial designs of circulant type

4.2.1 Introduction

The notion of autocorrelation associated with a finite sequence is a unifying concept that allows for several classes of *combinatorial designs of circulant type* to be concisely described. Designs of this type can be represented in terms of circulant matrices formed from finite complementary sequences (Definition 1.40). Examples of such designs include certain *D-optimal matrices*, *Hadamard matrices* and *circulant weighing matrices* amongst many other possibilities. A precise summary describing several of these designs, the associated sequences and their autocorrelation properties, can be found in [130, Table 1]. For an encyclopedic reference on autocorrelation properties and complementary sequences more generally, see [161, 162], and for an authoritative reference on combinatorial designs, see [76].

Many combinatorial designs can be defined as matrices of a given class which attain certain determinantal bounds. For instance, D-optimal and Hadamard matrices of a given order are precisely the $\{\pm 1\}$ -matrices whose determinant is maximal among all other such matrices of the same order [63, 118, 164]. For this reason, combinatorial designs arise in various fields where the determinantal bounds give rise to “best possible” or “optimal” objects. Specific applications include coding theory [19, 158], quantum computing [101, 172], wireless communication, cryptography and radar [109]. In many such applications, precise knowledge of the relevant combinatorial design is required.

In order to explicitly construct combinatorial designs of non-trivial orders, it is necessary to exploit the underlying structure. Some possibilities include an appropriate group theoretic structure through which the mathematical analysis can proceed, or an efficient representation which is amenable to search algorithms such as metaheuristics. We purpose the Douglas–Rachford algorithm as a search heuristic. The critical feature of the problem, which allows for an efficient implementation of DR to solve it, is that the autocorrelation function gives rise to a projection operator which can be efficiently computed.

4.2.2 Modelling Framework

In this section we explain how to model a general combinatorial design of circulant type as a feasibility problem described by three sets. More precisely, we consider designs belonging to the following class.

Definition 4.20 (Design of circulant type). Consider natural numbers $n, m \in \mathbb{N}$, vectors $\alpha \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$, and let $\mathcal{A} \subset \mathbb{R}$ be finite and nonempty. A design of circulant type of order n with parameters $(m, \alpha, v, \mathcal{A})$ is an m -tuple of vectors,

$$(a^0, a^1, \dots, a^{m-1}) \in (\mathcal{A}^n)^m := \mathcal{A}^n \times \cdots \times \mathcal{A}^n,$$

which satisfy the following two conditions:

$$\sum_{s=0}^{n-1} a_s^j = \alpha_j \quad \forall j \in \{0, 1, \dots, m-1\}, \quad \text{and} \quad \sum_{j=0}^{m-1} a^j \star a^j = v.$$

We remark that the notation “ \mathcal{A} ” will be reserved for a finite subset of \mathbb{R} which we refer to as the *alphabet*. We will be concerned with the alphabets $\{\pm 1\}$ and $\{0, \pm 1\}$.

Let $\mathcal{A} \subset \mathbb{R}$ be finite and nonempty, and let $\alpha \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. A design of circulant type of order n with parameters $(m, \alpha, v, \mathcal{A})$ can be constructed as a solution of the following feasibility problem:

$$\text{Find } (a^0, a^1, \dots, a^{m-1}) \in C_1 \cap C_2 \cap C_3 \subseteq (\mathbb{R}^n)^m, \quad (4.24)$$

where the constraint sets are defined by

$$C_1 := \{(a^0, a^1, \dots, a^{m-1}) \in (\mathbb{R}^n)^m : a^j \in \mathcal{A}^n, \forall j = 0, 1, \dots, m-1\}, \quad (4.25a)$$

$$C_2 := \left\{ (a^0, a^1, \dots, a^{m-1}) \in (\mathbb{R}^n)^m : \sum_{s=0}^{n-1} a_s^j = \alpha_j, \forall j = 0, 1, \dots, m-1 \right\}, \quad (4.25b)$$

$$C_3 := \left\{ (a^0, a^1, \dots, a^{m-1}) \in (\mathbb{R}^n)^m : \sum_{j=0}^{m-1} a^j \star a^j = v \right\}. \quad (4.25c)$$

REMARK 4.21 (Autocorrelation constraints in bit retrieval). In the special case that $m = 1$, the constraint C_3 appears in the formulation of the *bit retrieval* problem used in [94].

REMARK 4.22 (Variants of C_1). The constraint set C_1 in (4.25a) can be easily modified so that the alphabet \mathcal{A} set is different for each vector a^j or even for each individual entry of the vectors a^j . In this way, desired entries of a design can be fixed or avoided by choosing the corresponding alphabet sets to be singleton or to exclude certain values, respectively.

For each set of parameters $(m, \alpha, v, \mathcal{A})$, it transpires that an m -tuple of vectors $(a^j)_{j=0}^{m-1}$ satisfies Definition 4.20 precisely when it is a feasible point for (4.24). This equivalence is justified by the following proposition.

Proposition 4.23. *Let $\mathcal{A} \subset \mathbb{R}$ be nonempty and finite and consider a collection of real sequences $\{a^j\}_{j=0}^{m-1} \subseteq \mathcal{A}^n$. Then the following assertions are equivalent:*

(i) $\{a^j\}_{j=0}^{m-1} \subseteq \mathcal{A}^n$ is complementary with constants ν_0 and ν_1 , i.e.,

$$\sum_{j=0}^{m-1} a^j \star a^j = (\nu_0, \nu_1, \dots, \nu_1);$$

(ii) $(a^j)_{j=0}^{m-1} \in (\mathbb{R}^n)^m$ solves (4.24) with $v = (\nu_0, \nu_1, \dots, \nu_1)$ and some $\alpha \in \mathbb{R}^m$ which satisfies

$$\sum_{j=0}^{m-1} \alpha_j^2 = \nu_1(n-1) + \nu_0.$$

Proof. This is an immediate consequence of Proposition 1.41. \square

In order for the feasibility problem (4.24) to be computationally useful, it is necessary that the projectors onto the constraint sets in (4.25) can be efficiently computed. In what follows, we prove that this is indeed the case.

Proposition 4.24 (Projector onto C_1). *Let $(a^0, a^1, \dots, a^{m-1}) \in (\mathbb{R}^n)^m$ and consider the set C_1 described in (4.25a). Then $P_{C_1}((a^j)_{j=0}^{m-1})$ is the set of points $(\bar{a}^j)_{j=0}^{m-1} \in (\mathbb{R}^n)^m$ which satisfy*

$$\bar{a}_s^j \in \left\{ l \in \mathcal{A} : |l - a_s^j| = \min_{\bar{l} \in \mathcal{A}} |\bar{l} - a_s^j| \right\},$$

for all $j = 0, 1, \dots, m-1$ and $s = 0, 1, \dots, n-1$.

Proof. Let $a \in \mathbb{R}$. We observe that projector onto the set \mathcal{A} is given by

$$P_{\mathcal{A}}(a) = \left\{ l \in \mathcal{A} : |l - a| = \min_{\bar{l} \in \mathcal{A}} |\bar{l} - a| \right\}.$$

Applying this result pointwise and using the definition of the inner product on $(\mathbb{R}^n)^m$, the result follows. \square

Proposition 4.25 (Projector onto C_2). *Let $e = (1, 1, \dots, 1) \in \mathbb{R}^n$. The projector onto the set C_2 in (4.25b) at $(a^0, a^1, \dots, a^{m-1}) \in (\mathbb{R}^n)^m$ is given by*

$$P_{C_2}((a^j)_{j=0}^{m-1}) = \left(a^j + \frac{1}{n} \left(\alpha_j - \sum_{s=0}^{n-1} a_s^j \right) e \right)_{j=0}^{m-1}.$$

Proof. According to Proposition 1.43, the projection of any point $a \in \mathbb{R}^n$ onto the hyperplane $H_j := \{a \in \mathbb{R}^n : e^T a = \alpha_j\}$ is given by

$$P_{H_j}(a) = a + \frac{1}{\|e\|^2} (\alpha_j - e^T a) e = a + \frac{1}{n} \left(\alpha_j - \sum_{s=0}^{n-1} a_s \right) e.$$

The definition of the inner product on $(\mathbb{R}^n)^m$ yields $P_{C_2}((a^j)_{j=0}^{m-1}) = (P_{H_j}(a^j))_{j=0}^{m-1}$, from which the result follows. \square

REMARK 4.26. The implementation of the projectors given in Propositions 4.24 and 4.25, requires only vector arithmetic and finding the minimum of a finite set, respectively. From a computation perspective, the latter poses no problem when the alphabet, \mathcal{A} , is small.

We now turn our attention to describing the projector onto C_3 . In the following proposition, we denote the unit sphere in \mathbb{C}^m by

$$\mathbb{S} := \left\{ (z_j)_{j=0}^{m-1} \in \mathbb{C}^m : \sum_{j=0}^{m-1} |z_j|^2 = 1 \right\},$$

and we define the set

$$Y := \mathcal{F}(\mathbb{R}^n)^m = \mathcal{F}(\mathbb{R}^n) \times \overset{(m)}{\cdots} \times \mathcal{F}(\mathbb{R}^n),$$

where \mathcal{F} denotes the discrete Fourier transform (1.10). In view of Proposition 1.42, the set $\mathcal{F}(\mathbb{R}^n)$ is precisely the set of all conjugate symmetric vectors in \mathbb{C}^n , i.e.,

$$\mathcal{F}(\mathbb{R}^n) = \{(z_s)_{s=0}^{n-1} \in \mathbb{C}^n : z_0 \in \mathbb{R}, z_s = z_{n-s}^*, \forall s = 1, 2, \dots, n-1\}.$$

Proposition 4.27 (Projector onto C_3). *For any $v \in \mathbb{R}^n$, the projector onto the set C_3 defined in (4.25c) can be computed as*

$$P_{C_3} = (\mathcal{F}^{-1}, \dots, \mathcal{F}^{-1}) \circ P_{\widehat{C}_3} \circ (\mathcal{F}, \dots, \mathcal{F}), \quad (4.26)$$

where the set \widehat{C}_3 is described by

$$\widehat{C}_3 := \left\{ (\hat{a}^j)_{j=0}^{m-1} \in Y : \sum_{j=0}^{m-1} |\hat{a}^j|^2 = \hat{v} \right\},$$

with $\hat{v} := \mathcal{F}(v)$.

Furthermore, the projector onto \widehat{C}_3 at $(\hat{a}^1, \hat{a}^2, \dots, \hat{a}^{m-1}) \in Y$, $P_{\widehat{C}_3}((\hat{a}^j)_{j=0}^{m-1})$, is given by the set of all points $(\bar{a}^j)_{j=0}^{m-1} \in Y$ which satisfy, for all $s = 0, 1, \dots, n-1$:

$$\begin{cases} (\bar{a}_s^j)_{j=0}^{m-1} = \frac{\sqrt{\hat{v}_s}}{\sqrt{\sum_{j=0}^{m-1} |\hat{a}_s^j|^2}} (\hat{a}_s^j)_{j=0}^{m-1}, & \text{if } (\hat{a}_s^j)_{j=0}^{m-1} \neq 0_m, \\ (\bar{a}_s^j)_{j=0}^{m-1} \in \sqrt{\hat{v}_s} \mathbb{S}, & \text{if } (\hat{a}_s^j)_{j=0}^{m-1} = 0_m. \end{cases} \quad (4.27)$$

Proof. We first prove the claimed formula for $P_{\widehat{C}_3}$. To this end, note that

$$\widehat{C}_3 = E \cap Y \text{ where } E := \left\{ (\hat{a}^j)_{j=0}^{m-1} \in (\mathbb{C}^n)^m : \sum_{j=0}^{m-1} |\hat{a}^j|^2 = \hat{v} \right\}. \quad (4.28)$$

By Proposition 1.44, the projector onto \mathbb{S} for a point $z \in \mathbb{C}^m$ is given by

$$P_{\mathbb{S}}(z) = \begin{cases} z/\|z\|, & \text{if } z \neq 0_m; \\ \mathbb{S}, & \text{if } z = 0_m. \end{cases} \quad (4.29)$$

Applying (4.29) to each m -tuple $(\hat{a}_s^j)_{j=0}^{m-1}$, we deduce that $(\bar{a}_s^j)_{j=0}^{m-1} \in P_E((\hat{a}_s^j)_{j=0}^{m-1}) \subset (\mathbb{C}^n)^m$ precisely when the vector $(\bar{a}_s^j)_{j=0}^{m-1} \in \mathbb{C}^m$ satisfies (4.27) for all $s = 0, \dots, n-1$. Due to (4.28), any vector $(\bar{a}_s^j)_{j=0}^{m-1}$ which satisfies (4.27) and is contained in Y is an element of $P_{\widehat{C}_3}((\hat{a}_s^j)_{j=0}^{m-1})$. Thus the claimed formula for $P_{\widehat{C}_3}$ follows.

Next we prove (4.26). We first note that since the Fourier transform, \mathcal{F} , is a linear isometry on \mathbb{C}^n (Proposition 1.42(iii)), the operator $(\mathcal{F}, \dots, \mathcal{F})$ is a linear isometry on $(\mathbb{C}^n)^m$ with inverse given by $(\mathcal{F}, \dots, \mathcal{F})^{-1} = (\mathcal{F}^{-1}, \dots, \mathcal{F}^{-1})$. Since distances are invariant under isometries, we therefor have that

$$P_{C_3} = (\mathcal{F}^{-1}, \dots, \mathcal{F}^{-1}) \circ P_{\mathcal{F}(C_3)} \circ (\mathcal{F}, \dots, \mathcal{F}),$$

where $\mathcal{F}(C_3) := \left\{ (\mathcal{F}(a^j))_{j=0}^{m-1} : (a^j)_{j=0}^{m-1} \in C_3 \right\}$. To complete the proof, it therefore suffices to show $\mathcal{F}(C_3) = \widehat{C}_3$. To this end, in view of Proposition 1.42(ii)–(iii), for any tuple $(a^j)_{j=0}^{m-1} \in (\mathbb{C}^n)^m$ we have that

$$\sum_{j=0}^{m-1} a^j \star a^j = v \quad \Leftrightarrow \quad \sum_{j=0}^{m-1} \mathcal{F}(a^j \star a^j) = \hat{v} \quad \Leftrightarrow \quad \sum_{j=0}^{m-1} |\mathcal{F}(a^j)|^2 = \hat{v},$$

which shows that $\mathcal{F}(C_3) \subseteq \widehat{C}_3$. To deduce the reverse inclusion, note that \mathcal{F} is invertible (Proposition 1.42(iii)) and use the same argument with $(a^j)_{j=0}^{m-1} := (\mathcal{F}^{-1}(\hat{a}^j))_{j=0}^{m-1}$. \square

REMARK 4.28. We emphasize that it is important to note that the projector onto \widehat{C}_3 is given by (4.27) for tuples $(\bar{a}^j)_{j=0}^{m-1}$ contained in Y but not $(\mathbb{C}^n)^m$.

We now provide three concrete examples of types of combinatorial designs which can be described in terms of the structure proposed in formulation (4.24).

4.2.2.1 Circulant weighing matrices

Definition 4.29 (Circulant weighing matrix). Let $n, k \in \mathbb{N}$. A circulant weighing matrix of order n and weight k^2 , denoted $CW(n, k^2)$, is a circulant matrix $W \in \{0, \pm 1\}^{n \times n}$ such that

$$WW^T = k^2 I. \quad (4.30)$$

As we show in the next proposition, circulant weighing matrices can be formulated as designs of circulant type in the sense of Definition 4.20.

Proposition 4.30. Let $n, k \in \mathbb{N}$. A matrix $W \in \mathbb{R}^{n \times n}$ is $CW(n, k^2)$ if and only if there exists a vector $a \in \{0, \pm 1\}^n$ with $W = c(a)$ such that

$$(i) \sum_{s=0}^{n-1} a_s = \pm k, \text{ and}$$

$$(ii) a \star a = (k^2, 0, 0, \dots, 0).$$

Proof. Since the matrix W is circulant, there exists a vector $a \in \{0, \pm 1\}^n$ such that $W = c(a)$ where the mapping $c : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$ maps a vector to an associated circulant matrix. For such a vector, the equality (4.30) is equivalent to

$$a \star a = (k^2, 0, 0, \dots, 0).$$

The result follows by applying Proposition 4.23 (with $m = 1$). □

Example 4.31 (A CW matrix of small order). The vector

$$a = (-1, 1, 1, -1, 1, 0, 1, 0, 1, 1, 0, 0, -1)$$

defines a circulant weighing matrix $CW(13, 3^2)$. Indeed, it verifies $\sum_{s=0}^{12} a_s = 3$ and $a \star a = (9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$.

The class of circulant weighing matrices are of interest, in part, because they include all *circulant Hadamard matrices* (specially, a $CW(n, k^2)$ is a circulant Hadamard matrix

whenever $n = k^2$ and $n \equiv 0 \pmod{4}$). The existence of a CW matrices for a given order and weight is, in general, not resolved. *Strassler's table*, which originally appeared 20 years ago in [165], gives the existence status of $CW(n, k^2)$ for $n \leq 200$ and $k \leq 10$. The table has been updated several times, but still contains open cases. The most up-to-date version known to the author at the time of writing is contained in [169, Appendix A]. For other recent progress regarding CW matrices, see [170]. In Section 4.2.3.1 we present two circulant weighing matrices found with the DR algorithm, namely, a $CW(126, 8^2)$ and a $CW(198, 10^2)$, whose existence had been previously marked as an open question.

4.2.2.2 D-optimal designs of circulant type

Let n be an odd positive integer. In [92], the author showed that the determinant of a square matrix of order $2n$ having $\{\pm 1\}$ entries satisfies the bound

$$|\det(D)| \leq 2^n(2n-1)(n-1)^{n-1}.$$

Such a matrix is said to be *D-optimal* if it has maximal determinant, that is, the aforementioned determinate bound is attained.

To construct a D-optimal matrix, it suffices to find two commuting square $\{\pm 1\}$ -matrices, A and B , of order n such that

$$AA^T + BB^T = (2n-2)I + 2J, \quad (4.31)$$

where $J \in \mathbb{R}^{n \times n}$ denotes the matrix of all ones. A D-optimal matrix D of order $2n$ can then be constructed from the matrices A and B as

$$D = \begin{pmatrix} A & B \\ -B^T & A^T \end{pmatrix}. \quad (4.32)$$

This construction, originally proposed in [92] for the case in which matrices A and B are circulant, was later extended in [75] to commuting matrices. The former case constitutes a special type of D-optimal designs known as *D-optimal designs of circulant type*.

Definition 4.32 (D-optimal design of circulant type). *A D-optimal design of circulant type is a matrix D of order $2n$ given by (4.32) for a pair of circulant $\{\pm 1\}$ -matrices A and B of order n satisfying (4.31). In the case when we wish to refer to the underlying matrices A and B explicitly (rather than D), we shall say that (A, B) is a D-optimal design of circulant type.*

As before, we deduce the following characterization of D-optimal designs as particular instances of general combinatorial designs, both of circulant type.

Proposition 4.33. *Let n be an odd integer. A matrix D is a D-optimal design of circulant type of order $2n$ if and only if there exist constants $\alpha, \beta \in \mathbb{Z}$ with $\alpha^2 + \beta^2 = 4n - 2$ and a pair of vectors $(a, b) \in \{\pm 1\}^n \times \{\pm 1\}^n$ such that D satisfies (4.32) for $A = c(a)$ and $B = c(b)$, and the following assertions hold:*

$$(i) \sum_{s=0}^{n-1} a_s = \alpha,$$

$$(ii) \sum_{s=0}^{n-1} b_s = \beta, \text{ and}$$

$$(iii) a \star a + b \star b = (2n, 2, 2, \dots, 2).$$

Proof. Let (A, B) be a D-optimal design of circulant type of order $2n$. Since both matrices A and B are circulant, there exist vectors $a, b \in \{\pm 1\}^n$ such that $A = c(a)$ and $B = c(b)$. For such vectors, (4.31) is equivalent to

$$a \star a + b \star b = (2n, 2, 2, \dots, 2).$$

The result follows by applying Proposition 4.23 (with $m = 2$). □

Example 4.34 (D-optimal design of circulant type of small order). *The vectors*

$$a = (-1, 1, -1, 1, 1, 1, 1, 1, -1) \text{ and } b = (-1, 1, 1, 1, 1, -1, 1, 1, 1)$$

define a D-optimal design of order 9. Let $\alpha = 3$ and $\beta = 5$. Then we have $\alpha^2 + \beta^2 = 4n - 2$ with $\sum_{s=0}^8 a_s = \alpha$ and $\sum_{s=0}^8 b_s = \beta$, and that $a \star a + b \star b = (18, 2, 2, 2, 2, 2, 2, 2, 2)$.

The existence of a D-optimal matrix for values $n < 100$ for which the Diophantine equation $x^2 + y^2 = 4n - 2$ has solutions has been resolved in the affirmative with the exception of $n = 99$; see [86] and [87, Table 1]. In other words, the first unresolved case of existence arises when $n = 99$.

4.2.2.3 Double circulant core Hadamard matrices

Let n be an odd positive integer. Recall that a *Hadamard matrix* of order n is a matrix $H \in \{\pm 1\}^{n \times n}$ such that

$$HH^T = H^T H = nI.$$

There are many equivalent characterization of Hadamard matrices. For instance, they are precisely the $\{\pm 1\}$ -matrices of maximal determinant [118, Chapter 2].

Definition 4.35 (Double circulant core Hadamard matrix). *Let $n \in \mathbb{N}$. A Hadamard matrix, H , of order $2n + 2$ is said to be a Hadamard matrix with two circulant cores (DCHM) if it is of either one of the following two forms*

$$\left(\begin{array}{cc|cccc} - & - & + & \dots & + & + & \dots & + \\ - & + & + & \dots & + & - & \dots & - \\ \hline + & + & & & & & & \\ \vdots & \vdots & & & A & & & B \\ + & + & & & & & & \\ \hline + & - & & & & & & \\ \vdots & \vdots & & & B^T & & & -A^T \\ + & - & & & & & & \end{array} \right), \left(\begin{array}{cc|cccc} + & + & & & & & & \\ \vdots & \vdots & & & A & & & B \\ + & + & & & & & & \\ \hline + & - & & & & & & \\ \vdots & \vdots & & & B^T & & & -A^T \\ + & - & & & & & & \\ \hline - & - & + & \dots & + & + & \dots & + \\ - & + & + & \dots & + & - & \dots & - \end{array} \right), \quad (4.33)$$

where A and B are circulant matrices of order n , and $+$ and $-$ are shorthand for $+1$ and -1 , respectively.

We note that the two Hadamard matrices in (4.33) are *Hadamard equivalent* in the sense that one can be obtained from the other via sequence of row/column negations and row/column permutation [131, §2.1].

Two circulant matrices A and B satisfy Definition 4.35 precisely when [131, p. 3]

$$AA^T + BB^T = (2n + 2)I_n - 2J_n, \quad (4.34)$$

and hence we can deduce the following characterization.

Proposition 4.36. *A pair of matrices A and B satisfy (4.34) and, consequently define a Hadamard matrix with two circulant cores, if and only if there exists vectors $a, b \in \{\pm 1\}^n$ such that $A = c(a), B = c(b)$ with*

- (i) $\alpha := \sum_{s=0}^{n-1} a_s \in \{\pm 1\}$,
- (ii) $\beta := \sum_{s=0}^{n-1} b_s \in \{\pm 1\}$, and
- (iii) $a \star a + b \star b = (2n, -2, -2, \dots, -2)$.

Proof. Denote $A = c(a)$ and $B = c(b)$ for vectors $a, b \in \{\pm 1\}^n$. It follows that (4.34) is equivalent to

$$a \star a + b \star b = (2n, -2, -2, \dots, -2),$$

and thus (iii) holds. Furthermore, as a direct consequence of Proposition 4.23, one has

$$\alpha^2 + \beta^2 = \left(\sum_{s=0}^{n-1} a_s \right)^2 + \left(\sum_{s=0}^{n-1} b_s \right)^2 = 2,$$

from which (i)-(ii) follows, and thus the result is proved. \square

Example 4.37 (A Hadamard matrix with two circulant cores of small order).

The vectors

$$a = (1, -1, -1, 1, -1, 1, 1, 1, -1) \quad \text{and} \quad b = (-1, -1, 1, 1, -1, 1, 1, 1, -1)$$

define a double core circulant Hadamard matrix design. Indeed, note that $\sum_{s=0}^8 a_s = 1$, $\sum_{s=0}^8 b_s = 1$ and $a \star a + b \star b = (18, -2, -2, -2, -2, -2, -2, -2, -2)$.

4.2.3 Computational Results

In this section, we report the results of some numerical experiments which demonstrate the performance of DR for solving (4.24). As that feasibility problem involves three constraint sets, we implement the Douglas–Rachford algorithm in the product space as described in (2.17). We used the same stopping criteria stated in (4.10). All codes were written in Python 2.7 and runs were performed on Intel® Xeon® X5650 @ 2.67 GHz with 99 GB RAM, under Debian 4.9. In order to run DR, the full amount of RAM was not used nor required; our experiments could have easily been performed on a standard desktop computer.

Computational results for CW matrices are summarized in Figure 4.34, while more detailed results are included at the end of the section in Table 4.13. Results for D-optimal designs of circulant type and Hadamard matrices with two circulant cores, respectively, can be found in Tables 4.11 and 4.12.

4.2.3.1 New circulant weighing matrices

We now state and prove our main result regarding the existence of two circulant weighing matrices. Our approach makes use of the following construction which is a consequence of [22, Theorem 2.3]. Since this result appears without a proof in [21, Section 2], we show next how to derive it and give an explicit expression of the components of the constructed matrix in terms of the components of the original matrices.

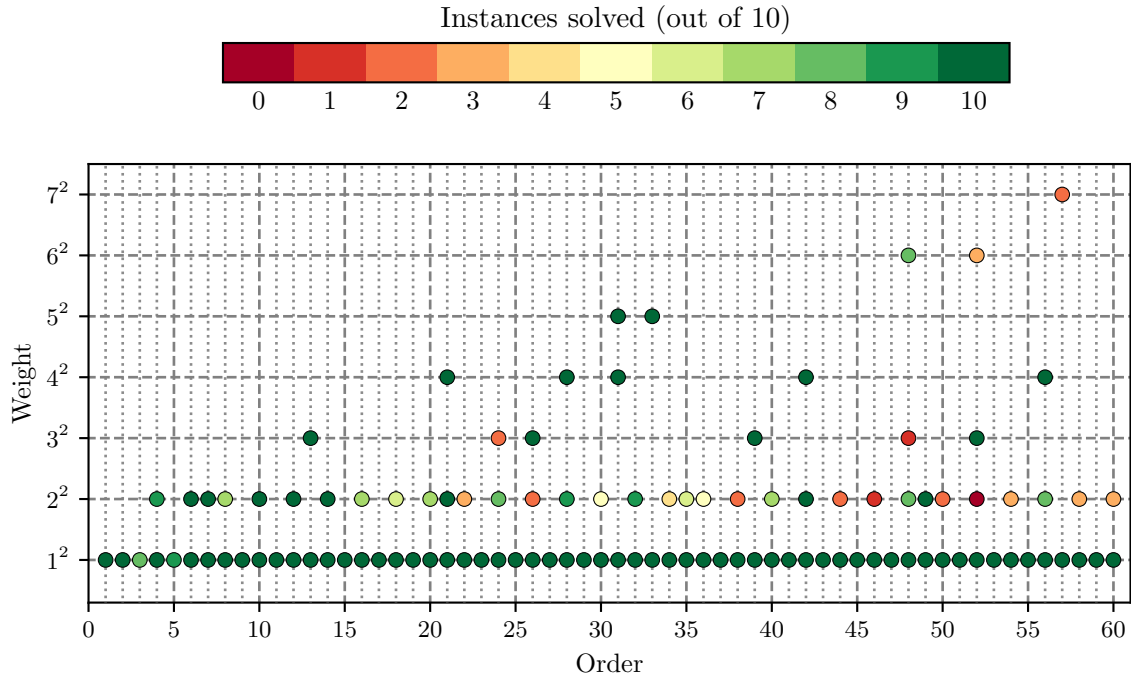


FIGURE 4.34: Results for CW matrices (10 random initialization, 3600s time limit).

n	(α, β)	Solved instances	Average time (s)	Average iterations
3	(1,3)	10	0.00	3.4
5	(3,3)	10	0.00	6.6
7	(1,5)	9	0.01	12.7
9	(3,5)	10	0.19	398.3
13	(1,7)	7	0.13	349.7
13	(5,5)	7	0.16	403.6
15	(3,7)	10	0.24	591.8
19	(5,7)	10	0.81	1,999.1
21	(1,9)	8	1.36	3,424.9
23	(3,9)	8	2.02	5,097.1
25	(7,7)	10	4.64	11,668.6
27	(5,9)	9	116.50	297,617.0
31	(1,11)	10	187.63	460,501.0
33	(3,11)	8	553.44	1,380,160.0
33	(7,9)	8	810.97	2,025,880.0
37	(5,11)	3	1,885.47	4,399,507.0
41	(9,9)	1	586.87	1,352,777.0
43	(1,13)	0	—	—
43	(7,11)	1	1,207.20	2,737,865.0

TABLE 4.11: Experimental results for D-optimal designs with parameters (n, α, β) given by Proposition 4.33 (10 random initialization, 3600s time limit).

n	Solved instances	Average time (s)	Average iterations
1	10	0.00	1.7
3	10	0.01	33.6
5	10	0.00	5.9
7	8	0.01	35.8
9	10	0.01	35.2
11	10	0.04	89.2
13	9	0.10	222.2
15	10	0.10	241.8
17	10	0.22	549.3
19	10	1.68	4,162.5
21	10	1.97	4,764.0
23	10	2.26	5,533.2
25	9	16.08	40,468.1
27	10	76.10	192,706.0
29	10	91.82	223,875.0
31	10	428.61	1,028,850.0
33	10	849.84	2,070,120.0
35	4	2,354.52	5,864,880.0
37	2	1,883.67	4,603,068.0
39	1	2,536.40	5,916,197.0

TABLE 4.12: Experimental results for DCHM designs with parameters $(n, \alpha = 1, \beta = 1)$ given by Proposition 4.36 (10 random initialization, 3600s time limit).

Theorem 4.38. *Let $n, k \in \mathbb{N}$ with n odd. Let A and B be two $CW(n, k^2)$ whose respective first rows, a and b , have disjoint support⁶. Then the circulant matrix $c(w)$ is a $CW(2n, 4k^2)$ where $w = (w_0, w_1, \dots, w_{2n-1}) \in \mathbb{R}^{2n}$ is given component-wise by*

$$w_s := \begin{cases} a_{\frac{s}{2}} + b_{\frac{s}{2}}, & \text{if } s \text{ is even,} \\ a_{\frac{s+n}{2}} - b_{\frac{s+n}{2}}, & \text{if } s \text{ is odd and } s \leq n-2, \\ a_{\frac{s-n}{2}} - b_{\frac{s-n}{2}}, & \text{if } s \text{ is odd and } s > n-2. \end{cases} \quad (4.35)$$

Proof. Let $G = \langle x \rangle = \{1, x, \dots, x^{2n-1}\}$ be a cyclic group of order $2n$ generated by x , where $x^{2n} = 1$. Clearly, the element x^n of the group G has order 2.

Let $a, b \in \mathbb{R}^n$ denote the first rows of A and B , respectively, and consider the generating functions given by the expressions

$$\mathbf{A}(x) := \sum_{s=0}^{n-1} a_s x^s \quad \text{and} \quad \mathbf{B}(x) := \sum_{s=0}^{n-1} b_s x^s.$$

⁶The *support* of $c = (c_0, c_1, \dots, c_{n-1}) \in \mathbb{R}^n$ is the set $\{i \in \{0, \dots, n-1\} : c_i \neq 0\}$.

1, 0, 1, -1, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1].

It has disjoint support with its cyclic permutation, b , given by

$b = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, -1, 0,$
 $0, 0, 0, 0, 1, 1, 0, 1, 0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1,$
 $-1, 0, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, -1, 0].$

The construction in Theorem 4.38 applied to a and b yields

$w = [1, 0, -1, 1, 0, -1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, -1, 0, 0, 1, 0, 1, 0,$
 $-1, 0, 0, -1, 0, 0, -1, -1, 1, 1, 0, 0, 0, 0, 0, -1, 1, 0, 1, 1, -1, 0, 0, 1,$
 $1, 0, 0, 0, 1, 1, -1, 0, 1, 0, 1, 1, 1, 1, 1, 0, -1, -1, 0, -1, 0, 0, 0,$
 $-1, 0, 0, 0, 0, 1, 0, 1, -1, 0, 0, 1, 0, -1, 0, -1, 0, 0, 1, 0, 0, -1, 1, -1,$
 $-1, 0, 0, 0, 0, 0, 1, -1, 0, -1, 1, 1, 0, 0, -1, 1, 0, 0, 0, 1, 1, -1, 0, -1,$
 $0, -1, -1, 1, 1, -1].$

and, consequently, the vector w defines a $CW(126, 8^2)$.

Similarly, using DR, the following $CW(99, 5^2)$ was found

$a = [-1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 1, 0, 0, 0, 0, 0,$
 $-1, 0, 0, 1, 0, 0, 1, 0, 0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0,$
 $1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, -1, 0, 0, 1, 0, 0, -1, 0, 0,$
 $1, 0, 0, 0, 0, 0, 1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0,$
 $-1, 0, 0]$

It has disjoint support with its cyclic permutation, b , given by

$b = [0, -1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, -1, 0, 0, 1, 0, 0, 0, 0,$
 $0, -1, 0, 0, 1, 0, 0, 1, 0, 0, -1, 0, 0, -1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,$
 $0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, -1, 0, 0, 1, 0, 0, -1, 0,$
 $0, 1, 0, 0, 0, 0, 0, 1, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0,$
 $0, -1, 0].$

The construction in Theorem 4.38 applied to a and b yields

$w = [-1, 0, -1, 1, 0, -1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, -1, 1, 0, 1, 1, 0, -1,$
 $0, 0, 0, -1, 0, 1, -1, 0, -1, 1, 0, -1, 1, 0, 1, -1, 0, 1, 0, 0, 0, 1, 0, -1,$
 $-1, 0, -1, 0, 0, 0, 1, 0, 1, 1, 0, -1, 1, 0, 1, -1, 0, 1, -1, 0, -1, 0, 0, 0,$
 $-1, 0, -1, 0, 0, 0, 0, 0, -1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 1, -1, 0, 1,$
 $1, 0, 1, -1, 0, 1, 1, 0, 1, 1, 0, -1, 0, 0, 0, 1, 0, -1, 1, 0, 1, 1, 0, -1,$
 $1, 0, 1, 0, 0, 0, -1, 0, -1, -1, 0, 1, 1, 0, 1, 1, 0, -1, -1, 0, -1, 0, 0, 0,$

$$1, 0, 1, -1, 0, 1, 0, 0, 0, 1, 0, -1, 1, 0, 1, 1, 0, -1, -1, 0, -1, -1, 0, 1, \\ 0, 0, 0, -1, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, -1, 1, 0, -1, 0, 0, 0, 1, 0, -1, \\ -1, 0, -1, 1, 0, -1],$$

and, consequently, the vector w defines a $CW(198, 10^2)$. \square

REMARK 4.40. Both constructions in Theorem 4.39 were marked as open cases in the update of Strassler's Table appearing in the 2016 preprint version of [169]⁷. We also note that a previous version of Strassler's Table published in [20, Table 3] also listed these two cases as open. Despite the fact that these two cases have remained unresolved in multiple updates of Strassler's table, we discovered (after independently proving Theorem 4.39) that existence can actually be deduced by combining either of the aforementioned versions of Strassler's table with a much older result [18, Theorem 2.2] which appeared in 1999. Specifically, the existence of $CW(126, 8^2)$ and $CW(198, 10^2)$ follows by respectively applying this result to $CW(21, 4^2)$ and $CW(33, 5^2)$, with $m = 3$. In fact, the existence of $CW(198, 10^2)$ was already claimed in [18]. After pointing out the errors to the author, Strassler's Table has been correctly updated in the final published version of [169].

REMARK 4.41. Although Strassler's original table [165] correctly states that $CW(196, 4^2)$ exist, in both of updates, [20, Table 3] and the one in the preliminary version of [169], its status is incorrectly shown as not existing. The same error appears in [111, §5] and [170, p. 144]. Indeed, we obtained the following $CW(28, 4^2)$ with DR

$$\mathbf{a} = [1, 0, 1, -1, -1, 1, 0, 1, -1, 0, 0, 1, 0, 0, -1, 0, -1, -1, 1, 1, 0, 1, \\ 1, 0, 0, 1, 0, 0],$$

from which a $CW(196, 4^2)$ can be deduced by appending 0_6 after each component

$$\mathbf{w} = [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0, \\ 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \\ 0, 1, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \\ 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \\ 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, -1, \\ 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \\ 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \\ 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \\ 0, 0, 0, 0].$$

The status of $CW(196, 4^2)$ has also been corrected in the published work [169, Table 3] after notifying the author.

⁷Tan M.M.: Group Invariant Weighing Matrices. *Preprint* (2016). ArXiv: [1610.01914](https://arxiv.org/abs/1610.01914)

(n, k)	Solved	Av. time (s)	Av. iterations	(n, k)	Solved	Av. time (s)	Av. iterations
(1,1)	10	0.00	1.5	(35,1)	10	0.00	8.5
(2,1)	10	0.00	1.4	(36,1)	10	0.00	8.3
(3,1)	8	0.00	3.1	(37,1)	10	0.00	11.7
(4,1)	10	0.00	5.6	(38,1)	10	0.00	6.2
(5,1)	9	0.00	4.0	(39,1)	10	0.00	9.9
(6,1)	10	0.00	4.1	(40,1)	10	0.00	10.5
(7,1)	10	0.00	3.3	(41,1)	10	0.00	11.8
(8,1)	10	0.00	3.5	(42,1)	10	0.00	11.8
(9,1)	10	0.00	4.0	(43,1)	10	0.00	9.1
(10,1)	10	0.00	4.5	(44,1)	10	0.00	8.7
(11,1)	10	0.00	4.0	(45,1)	10	0.00	9.7
(12,1)	10	0.00	3.8	(46,1)	10	0.00	14.5
(13,1)	10	0.00	4.7	(47,1)	10	0.00	9.3
(14,1)	10	0.00	3.8	(48,1)	10	0.00	10.9
(15,1)	10	0.00	5.7	(49,1)	10	0.00	11.9
(16,1)	10	0.00	6.0	(50,1)	10	0.00	13.4
(17,1)	10	0.00	5.7	(51,1)	10	0.00	11.7
(18,1)	10	0.00	4.6	(52,1)	10	0.00	16.3
(19,1)	10	0.00	7.0	(53,1)	10	0.01	17.8
(20,1)	10	0.00	6.2	(54,1)	10	0.00	16.2
(21,1)	10	0.00	6.3	(55,1)	10	0.00	14.7
(22,1)	10	0.00	8.2	(56,1)	10	0.00	10.4
(23,1)	10	0.00	6.9	(57,1)	10	0.00	15.9
(24,1)	10	0.00	6.2	(58,1)	10	0.00	11.6
(25,1)	10	0.00	4.8	(59,1)	10	0.00	12.4
(26,1)	10	0.00	5.2	(60,1)	10	0.00	16.1
(27,1)	10	0.00	6.1	(32,2)	9	0.25	984.3
(28,1)	10	0.00	6.7	(34,2)	4	0.06	211.8
(29,1)	10	0.00	8.7	(35,2)	6	0.14	516.3
(30,1)	10	0.00	7.9	(36,2)	5	0.09	359.4
(4,2)	9	0.00	5.1	(38,2)	2	0.11	398.0
(6,2)	10	0.00	8.1	(40,2)	7	0.34	1,287.3
(7,2)	10	0.09	328.9	(42,2)	10	0.60	2,265.0
(8,2)	7	0.02	81.4	(44,2)	2	0.06	241.0
(10,2)	10	0.04	180.5	(46,2)	1	0.02	65.0
(12,2)	10	0.05	211.7	(48,2)	8	0.21	798.0
(14,2)	10	0.16	649.2	(49,2)	10	1.36	5,031.0
(16,2)	7	0.09	373.0	(50,2)	2	0.05	201.5
(18,2)	6	0.03	110.5	(52,2)	0	-	-
(20,2)	7	0.30	1,213.9	(54,2)	3	0.14	491.7
(21,2)	10	0.32	1,165.9	(56,2)	8	0.29	1,098.4
(22,2)	3	0.02	67.7	(58,2)	3	0.01	44.3
(24,2)	8	0.13	506.5	(60,2)	3	0.28	1,082.0
(26,2)	2	0.03	101.0	(39,3)	10	5.72	22,158.7
(28,2)	9	0.19	703.3	(48,3)	1	13.29	52,189.0
(30,2)	5	0.07	232.6	(52,3)	10	3.92	14,888.2
(13,3)	10	0.05	172.0	(31,4)	10	422.45	1,652,410.0
(24,3)	2	10.93	42,967.5	(42,4)	10	132.12	504,622.0
(26,3)	10	1.89	7,162.3	(56,4)	10	59.63	225,106.0
(21,4)	10	11.47	45,012.3	(31,5)	10	23.10	90,731.5
(28,4)	10	15.89	60,377.3	(33,5)	10	334.83	1,306,620.0
(31,1)	10	0.00	9.0	(48,6)	8	607.04	2,365,024.0
(32,1)	10	0.00	9.8	(52,6)	3	2,314.49	8,309,650.0
(33,1)	10	0.00	9.4	(57,7)	2	482.54	1,812,060.0
(34,1)	10	0.00	9.7				

TABLE 4.13: Detailed results for CW matrices (10 random initialization, 3600s time limit).

Conclusions and future research

This dissertation contributes to the family of projection algorithms, both from the theoretical perspective and also from the point of view of the applications. All the results in Chapters 3 and 4 are new and the content therein is based on the author's joint works [11, 14, 16], and the submitted manuscripts [12, 13, 15]. Detailed contributions, as well as some possible directions of future research, are summarized hereunder.

Theoretical contributions

A new projection algorithm, the averaged alternating modified reflections (AAMR) method, has been introduced and studied in Chapter 3. Even though each iteration of AAMR is very similar to the one of the classical Douglas–Rachford (DR) method, the new scheme yields a solution to the best approximation problem, unlike DR, which only gives a point in the intersection of the sets. Under a constraint qualification, the method was proved to be strongly convergent to the solution of the best approximation problem.

Motivated by the promising results of some numerical experiments, we have computed the rate of linear convergence of the AAMR method for the case of two subspaces in a Euclidean space. We have additionally found the optimal selection of the parameters defining the scheme that minimizes this rate, in terms of the Friedrichs angle. The rate with optimal parameters coincides with the one of the generalized alternating projections (GAP) method, which is the best among all known rates of projection methods. The sharpness of these theoretical results was additionally demonstrated with two computational experiments.

The AAMR algorithm has been extended to deal with monotone operators. Such an extension has been naturally derived by replacing projectors onto convex sets with resolvents of maximally monotone operators. The new splitting method can be applied to compute the resolvent of the sum of the operators, a generalized version of the best approximation problem within this context. This generalization of the method has allowed us to derive two different parallelized variants for dealing with finitely many operators.

- The fixed points of the AAMR operator are responsible for the AAMR algorithm to be able to solve best approximation problems. The existence of such fixed points has been proved equivalent to a constraint qualification to be held. However, the set itself has only been characterized for the case of two subspaces (Proposition 3.15). A precise description of the set of fixed points for arbitrary convex sets appears mandatory, since it implies a better understanding of the algorithm that may lead to new results.
- The AAMR algorithm is defined by two parameters α and β , which have a big effect in the convergence rate of the method (see Figure 3.11). Although α is allowed to vary along the iterations, nothing has been stated for β . It would be useful to derive a version of the algorithm where β can be updated at each step, with the purpose of developing some acceleration technique for the AAMR scheme.
- All the results have been done assuming the convexity of the sets. Because of the similarity of the AAMR scheme and the Douglas–Rachford method, and the effectiveness of the latter in some highly nonconvex settings, it would be interesting to explore whether it would be possible to use the AAMR method as heuristic in nonconvex feasibility problems, either alone, or combined with DR to avoid possible cycles.
- The analysis of the rate of convergence of the AAMR algorithm was done for the case of linear subspaces in a finite-dimensional space. It would be interesting to investigate whether the results can be extended to infinite-dimensional spaces; or even more, to study the rate of convergence of the method when it is applied to two arbitrary convex sets.
- The extension of AAMR for monotone operators has been analyzed by rewriting the iteration as the one generated by DR for a modification of the operators. This may give rise to the possibility of using some known results on Douglas–Rachford to derive new results for the AAMR algorithm.

Contributed applications

In Chapter 4, we have presented feasibility formulations for various known combinatorial problems, and we have showed that the Douglas–Rachford method can be used as a successful heuristic for solving them. This extends the list of nonconvex problems for which DR was already known to be a satisfactory solver.

We have first considered the graph coloring problem, for which two different formulations in the form of feasibility problem has been provided: one relying on binary indicator variables and another based on a semidefinite programming representation. On the one hand, the binary formulation has been shown to be easily adaptive and could successfully address other variants of the problem such as precoloring and list coloring problems (including Sudoku puzzles), 8-queens puzzles and generalizations, and Hamiltonian path problems (as the knight's tour problem). On the other hand, we have only been able to adapt the rank formulation for dealing with precoloring problems. Nonetheless, an extensive numerical experimentation over a wide spectrum of instances, demonstrates the superiority of the rank formulation over the binary one for those problems which can be tackled by both formulations.

The main reason of failure of projection-based heuristic algorithms is that the iteration stays trapped on limit cycles. Our experiments indicate that the rank-constrained matrix formulation appears to be immune to this problem, achieving a 100% success rate in most of the problems tested. The experiments performed with the binary formulation also demonstrate a good behavior of DR, but the success rates and the computing time results are far from those achieved by the rank formulation. Most notable is the experiment on the so-called 'nasty' Sudoku (treated as a graph pre-coloring instance, see Figure 4.33), on which the binary formulation only had a 20% success rate, while the rank formulation solved every single instance. Even more, the rank-constrained approach does not come at a great cost in implementation, and is able to solve graph coloring instances from the DIMACS benchmark collection with hundreds of vertices and thousands of edges, often in much less than an hour (see Table 4.10).

- In the emerging field of projection-based heuristic algorithms for solving combinatorially hard problems, the competition is usually framed to be about the choice of the operator (DR, ADMM, etc.). In our study we have shown that the choice of constraint formulation has a very significant effect, and in the end, it may prove to be even more important than the choice of operator.
- In the convex setting, for infeasible problems, the sequence generated by Douglas–Rachford provably tends to infinity (in norm). In our experiments with the binary formulation, we obtained some similar results for some particular graphs (see Figure 4.18), a behavior that seems to be strongly influenced by the formulation of the feasibility problem. This motivates us to further analyze the detection of infeasibility in nonconvex settings with this algorithm.

- We speculate that the good performance of the rank-constrained matrix formulation may be linked to the elimination of the high symmetry-based solution multiplicity of competing formulations (see Remark 4.13). This is strictly an empirical observation, and we have no proposal on how solution multiplicity might be linked to limit cycle behavior. Our results are offered as motivation for pursuing this direction in future research on projection based algorithms.

Finally, we have developed a feasibility problem which allows to construct any class of combinatorial designs of circulant type with the Douglas–Rachford algorithm. The approach is illustrated on three different classes of such designs: circulant weighing matrices, D-optimal matrices, and Hadamard matrices with two circulant cores. Furthermore, we have explicitly constructed two circulant weighing matrices, a $CW(126, 64)$ and a $CW(198, 100)$, whose existence was previously marked as an open question (see Remark 4.40).

- The latest update of Strassler’s Table [169, Table 3] still contains various cases marked as open. It would be worthwhile to study if some of these cases can be solved with the proposed approach, if we allow more time to the DR algorithm.
- There exists some approaches in the literature, where the construction of combinatorial designs is addressed by first reducing the problem to a simpler one (see [85]). Once the latter is solved, the challenge is then to reconstruct a solution to the original problem. It would be interesting to study whether the DR algorithm can be used for this purpose.

Bibliography

- [1] D. Achlioptas and E. Friedgut: A sharp threshold for k -colorability. *Random Struct. Alg.*, 14:67–70, 1990.
- [2] D. Achlioptas and M. Molloy: Almost all graphs with $2.522n$ edges are not 3-colorable. *Elec. Jour. Of Comb.*, 6(1):R29, 1999.
- [3] S. Adly, L. Bourdin, and F. Caubet: On the proximity operator of the sum of two convex functions, 2017. arXiv: [1707.08509](#).
- [4] S. Agmon: The relaxation method for linear inequalities. *Canad. J. Math.*, 6(3):382–392, 1954.
- [5] S. Alwadani, H.H. Bauschke, W.M. Moursi, and X. Wang: On the asymptotic behaviour of the Aragón Artacho–Campoy algorithm, 2018. arXiv: [1805.11165](#).
- [6] F.J. Aragón Artacho and J.M. Borwein: Global convergence of a non-convex Douglas–Rachford iteration. *J. Glob. Optim.*, 57(3):753–769, 2013.
- [7] F. J. Aragón Artacho, J. M. Borwein, V. Martín-Márquez, and L. Yao: Applications of convex analysis within mathematics. *Math. Program. Ser. B*, 148(1–2):49–88, 2014.
- [8] F. J. Aragón Artacho, J. M. Borwein, and M. K. Tam: Douglas–Rachford feasibility methods for matrix completion problems. *ANZIAM J.*, 55(4):299–326, 2014.
- [9] F. J. Aragón Artacho, J.M. Borwein, and M.K. Tam: Global behavior of the Douglas–Rachford method for a nonconvex feasibility problem. *J. Glob. Optim.*, 65(2):309–327, 2016.
- [10] F. J. Aragón Artacho, J.M. Borwein, and M.K. Tam: Recent results on Douglas–Rachford methods for combinatorial optimization problem. *J. Optim. Theory. Appl.*, 163(1):1–30, 2014.
- [11] F.J. Aragón Artacho and R. Campoy: A new projection method for finding the closest point in the intersection of convex sets. *Comput. Optim. Appl.*, 69(1):99–132, 2018.

-
- [12] F. J. Aragón Artacho and R. Campoy: Computing the resolvent of the sum of maximally monotone operators with the averaged alternating modified reflections algorithm, 2018. arXiv: [1805.09720](#).
- [13] F. J. Aragón Artacho and R. Campoy: Optimal rates of linear convergence of the averaged alternating modified reflections method for two subspaces, 2017. arXiv: [1711.06521](#).
- [14] F. J. Aragón Artacho and R. Campoy: Solving graph coloring problems with the Douglas–Rachford algorithm. *Set-Valued Var. Anal.*, 26(2):277–304, 2018.
- [15] F. J. Aragón Artacho, R. Campoy, and V. Elser: An enhanced formulation for successfully solving graph coloring problems with the Douglas–Rachford algorithm, 2018. arXiv: [1808.01022](#).
- [16] F. J. Aragón Artacho, R. Campoy, I. S. Kotsireas, and M. K. Tam: A feasibility approach for constructing combinatorial designs of circulant type. *J. Comb. Optim.*, 35(4):1061–1085, 2018.
- [17] F. J. Aragón Artacho, Y. Censor, and A. Gibali: The cyclic Douglas–Rachford algorithm with r-sets-Douglas–Rachford operators. *Optim. Methods Softw.*, 2018. DOI: [10.1080/10556788.2018.1504049](#).
- [18] K. T. Arasu and J. F. Dillon: *Perfect ternary arrays*. In *Difference Sets, Sequences and their Correlation Properties*. A. Pott, P. V. Kumar, T. Hellesteth, and D. Jungnickel, editors. Springer Netherlands, Dordrecht, 1999, pages 1–15.
- [19] K. T. Arasu and T. A. Gulliver: Self-dual codes over \mathbb{F}_p and weighing matrices. *IEEE Trans. Inform. Theory*, 47(5):2051–2055, 2001.
- [20] K. T. Arasu and A. J. Gutman: Circulant weighing matrices. *Cryptogr. Commun.*, 2(2):155–171, 2010.
- [21] K. T. Arasu, I. S. Kotsireas, C. Koukouvinos, and J. Seberry: On circulant and two-circulant weighing matrices. *Australas. J. Combin.*, 48:43–51, 2010.
- [22] K. T. Arasu, K. H. Leung, S. L. Ma, A. Nabavi, and D. K. Ray-Chaudhuri: Circulant weighing matrices of weight 2^{2t} . *Des. Codes Cryptogr.*, 41(1):111–123, 2006.
- [23] L. Aronszajn: Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68(3):337–404, 1950.
- [24] J. B. Baillon, R. E. Bruck, and S. Reich: On the asymptotic behavior of nonexpansive mappings and semigroups in Banach spaces. *Houston J. Math.*, 4(1):1–9, 1978.

-
- [25] H. H. Bauschke: The approximation of fixed points of compositions of nonexpansive mappings in Hilbert space. *J. Math. Anal. Appl.*, 202(1):150–159, 1996.
- [26] H. H. Bauschke, J. Y. Bello Cruz, T. T. Nghia, H. M. Phan, and X. Wang: Optimal rates of linear convergence of relaxed alternating projections and generalized Douglas–Rachford methods for two subspaces. *Numer. Algor.*, 73(1):33–76, 2016.
- [27] H. H. Bauschke, J. Y. Bello Cruz, T. T. Nghia, H. M. Phan, and X. Wang: The rate of linear convergence of the Douglas–Rachford algorithm for subspaces is the cosine of the Friedrichs angle. *J. Approx. Theory*, 185:63–79, 2014.
- [28] H. H. Bauschke and J. M. Borwein: Dykstra’s alternating projection algorithm for two sets. *J. Approx. Theory*, 79(3):418–443, 1996.
- [29] H. H. Bauschke and J. M. Borwein: On the convergence of von Neumann’s alternating projection algorithm for two sets. *Set-Valued Anal.*, 1(2):185–212, 1993.
- [30] H. H. Bauschke, J. M. Borwein, and A. S. Lewis: The method of cyclic projections for closed convex sets in Hilbert space. *Contemp. Math.*, 204:1–38, 1997.
- [31] H. H. Bauschke, J. M. Borwein, and P. Tseng: Bounded linear regularity, strong CHIP, and CHIP are distinct properties. *J. Convex Anal.*, 7(2):395–412, 2000.
- [32] H. H. Bauschke, R. S. Burachik, and C. Y. Kaya: Constraint splitting and projection methods for optimal control of double integrator, 2018. arXiv: [1804.03767](https://arxiv.org/abs/1804.03767).
- [33] H. H. Bauschke and P. L. Combettes: A Dykstra-like algorithm for two monotone operators. *Pacific J. Optim.*, 4(3):383–391, 2008.
- [34] H. H. Bauschke and P. L. Combettes: A weak-to-strong convergence principle for Fejér-monotone methods in Hilbert spaces. *Math. Oper. Res.*, 26(2):248–264, 2001.
- [35] H. H. Bauschke and P. L. Combettes: *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, New York, 2nd edition, 2017.
- [36] H. H. Bauschke, P. L. Combettes, and S. G. Kruk: Extrapolation algorithm for affine-convex feasibility problems. *Numer. Algor.*, 41(3):239–274, 2006.
- [37] H. H. Bauschke, P. L. Combettes, and D. R. Luke: A strongly convergent reflection method for finding the projection onto the intersection of two closed convex sets in a Hilbert space. *J. Approx. Theory*, 141(1):63–69, 2006.
- [38] H. H. Bauschke, P. L. Combettes, and D. R. Luke: Finding best approximation pairs relative to two closed convex sets in Hilbert spaces. *J. Approx. Theory*, 127(2):178–192, 2004.

-
- [39] H. H. Bauschke, P. L. Combettes, and D. R. Luke: Phase retrieval, error reduction algorithm, and fiemap variants: a view from convex optimization. *J. Opt. Soc. Am. A*, 19(7):1334–1345, 2002.
- [40] H. H. Bauschke and M. N. Dao: On the finite convergence of the Douglas–Rachford algorithm for solving (not necessarily convex) feasibility problems in Euclidean spaces. *SIAM J. Optim.*, 27(1):507–537, 2017.
- [41] H. H. Bauschke, M. N. Dao, and S. B. Lindstrom: The Douglas–Rachford algorithm for a hyperplane and a doubleton. *SIAM J. Optim.*, 2018. arXiv: [1804.08880](https://arxiv.org/abs/1804.08880).
- [42] H. H. Bauschke, F. Deutsch, H. Hundal, and S.-H. Park: Accelerating the convergence of the method of alternating projections. *Trans. Am. Math. Soc.*, 355(9):3433–3461, 2003.
- [43] H. H. Bauschke and V. R. Koch: Projection methods: Swiss army knives for solving feasibility and best approximation problems with halfspaces. *Contemp. Math.*, 636:1–40, 2015.
- [44] H. H. Bauschke, D. R. Luke, H. M. Phan, and X. Wang: Restricted normal cones and sparsity optimization with affine constraints. *Found. Comput. Math.*, 14(1):63–83, 2014.
- [45] H. H. Bauschke, D. R. Luke, H. M. Phan, and X. Wang: Restricted normal cones and the method of alternating projections: applications. *Set-Valued Var. Anal.*, 21(3):475–501, 2013.
- [46] H. H. Bauschke, D. R. Luke, H. M. Phan, and X. Wang: Restricted normal cones and the method of alternating projections: theory. *Set-Valued Var. Anal.*, 21(3):431–473, Sept. 2013.
- [47] H. H. Bauschke, B. Lukens, and W. M. Moursi: Affine nonexpansive operators, Attouch–Théra duality and the Douglas–Rachford algorithm. *Set-Valued Var. Anal.*, 25(3):481–505, 2017.
- [48] H. H. Bauschke and W. M. Moursi: On the Douglas–Rachford algorithm. *Math. Program., Ser. A*, 164(1–2):263–284, 2017.
- [49] H. H. Bauschke and D. Noll: On the local convergence of the Douglas–Rachford algorithm. *Arch. Math.*, 102(6):589–600, 2014.
- [50] H. H. Bauschke, D. Noll, and H. M. Phan: Linear and strong convergence of algorithms involving averaged nonexpansive operators. *J. Math. Anal. Appl.*, 421(1):1–20, 2015.

- [51] H. H. Bauschke, H. M. Phan, and X. Wang: The method of alternating relaxed projections for two nonconvex sets. *Vietnam J. Math.*, 42(4):421–450, 2014.
- [52] R. Behling, J. Y. Bello Cruz, and L. Santos: Circumcentering the Douglas–Rachford method. *Numer. Algor.*, 78(3):759–776, 2018.
- [53] J. Bell and B. Stevens: A survey of known results and research areas for n-queens. *Discrete Math.*, 309:1–31, 2009.
- [54] J. Benoist: The Douglas–Rachford algorithm for the case of the sphere and the line. *J. Global Optim.*, 63(2):363–380, 2015.
- [55] J. M. Borwein, G. Li, and L. Yao: Analysis of the convergence rate for the cyclic projection algorithm applied to basic semialgebraic convex sets. *SIAM J. Optim.*, 24(1):498–527, 2014.
- [56] J. M. Borwein, S. B. Lindstrom, B. Sims, A. Schneider, and M. P. Skerritt: Dynamics of the Douglas–Rachford method for ellipses and p-spheres. *Set-Valued Var. Anal.*, 26(2):385–403, 2018.
- [57] J. M. Borwein and M. K. Tam: A cyclic Douglas–Rachford iteration scheme. *J. Optim. Theory. Appl.*, 160(1):1–29, 2014.
- [58] J. M. Borwein and M. K. Tam: *Reflection methods for inverse problems with applications to protein conformation determination*. In *Generalized Nash Equilibrium Problems, Bilevel Programming and MPEC*. D. Aussel and C. Lalitha, editors. Springer Singapore, Singapore, 2017, pages 83–100.
- [59] J. M. Borwein and M. K. Tam: The cyclic Douglas–Rachford method for inconsistent feasibility problems. *J. Nonlinear Convex Anal.*, 16(4):573–584, 2015.
- [60] J. M. Borwein and B. Sims: *The Douglas–Rachford algorithm in the absence of convexity*. In *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*. H. H. Bauschke, R. S. Burachik, P. L. Combettes, V. Elser, D. R. Luke, and H. Wolkowicz, editors. Springer-Verlag, New York, 2011, pages 93–109.
- [61] J. P. Boyle and R. L. Dykstra: A method for finding projections onto the intersection of convex sets in Hilbert spaces. In R. Dykstra, T. Robertson, and F. T. Wright, editors, *Advances in Order Restricted Statistical Inference*, volume 37 of *Lecture Notes in Statistics*, pages 28–47. Springer, New York, 1986.
- [62] L. M. Bregman: The method of successive projection for finding a common point of convex sets. *Soviet Math. Dokl.*, 162(3):688–692, 1965.

-
- [63] R. P. Brent: Finding D-optimal design by randomised decomposition and switching. *Australas. J. Combin.*, 55:15–30, 2013.
- [64] W. L. Briggs and V. E. Henson: *The DFT. An Owner's Manual for the Discrete Fourier Transform*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 1995.
- [65] R. S. Burachik and V. Jeyakumar: A simple closure condition for the normal cone intersection formula. *Proc. Amer. Math. Soc.*, 133(6):1741–1748, 2005.
- [66] A. Cegielski: *Iterative Methods for Fixed Point Problems in Hilbert Spaces*, volume 2057 of *Lecture Notes in Mathematics*. Springer, Heidelberg, 2012.
- [67] A. Cegielski and A. Suchocka: Relaxed alternating projection methods. *SIAM J. Optim.*, 19(3):1093–1106, 2008.
- [68] Y. Censor: Iterative methods for convex feasibility problems. *Ann. Discrete Math.*, 20:83–91, 1984.
- [69] Y. Censor and A. Cegielski: Projection methods: an annotated bibliography of books and reviews. *Optimization*, 64(11):2343–2358, 2015.
- [70] G. J. Chaitin: Register allocation and spilling via graph coloring. *SIGPLAN Not.*, 39(4):66–74, 2004.
- [71] G. J. Chaitin, M. A. Auslander, A. K. Chandra, J. Cocke, M. E. Hopkins, and P. W. Markstein: Register allocation via coloring. *Computer Languages*, 6(1):47–57, 1981.
- [72] C. K. Chui, F. Deutsch, and J. D. Ward: Constrained best approximation in Hilbert space. *Constr. Approx.*, 6(1):35–64, 1990.
- [73] C. K. Chui, F. Deutsch, and J. D. Ward: Constrained best approximation in Hilbert space II. *J. Approx. Theory*, 71(2):213–238, 1992.
- [74] G. Cimmino: Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari. *La Ricerca Scientifica, II*, 9:326–333, 1938.
- [75] J. H. E. Cohn: On determinants with elements ± 1 . *Bull. London Math. Soc.*, 21(1):36–42, 1989.
- [76] C. J. Colbourn and J. H. Dinitz, editors: *Handbook of Combinatorial Designs*. Chapman & Hall/CRC, Boca Raton, FL, 2nd edition, 2007.
- [77] P. L. Combettes: Iterative construction of the resolvent of a sum of maximal monotone operators. *J. Convex Anal.*, 16(4):727–748, 2009.

- [78] P. L. Combettes: Proximity for sums of composite functions. *J. Math. Anal. Appl.*, 380(2):680–688, 2011.
- [79] M. N. Dao and M. K. Tam: A Lyapunov-type approach to convergence of the Douglas–Rachford algorithm for a nonconvex setting. *J. Glob. Optim.*:1–30, 2018. DOI: [10.1007/s10898-018-0677-3](https://doi.org/10.1007/s10898-018-0677-3).
- [80] F. Deutsch: Accelerating the convergence of the method of alternating projections via a line search: a brief survey. In D. Butnariu, Y. Censor, and S. Reich, editors, *Inherently Parallel Algorithms in Feasibility and Optimization and their Applications*. Volume 8, Studies in Computational Mathematics, pages 203–217. Elsevier, 2001.
- [81] F. Deutsch: *Best Approximation in Inner Product Spaces*, volume 7 of *CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC*. Springer-Verlag, New York, 2001.
- [82] F. Deutsch: *Rate of convergence of the method of alternating projections*. In *Parametric Optimization and Approximation: Conference Held at the Mathematisches Forschungsinstitut, Oberwolfach, October 16–22, 1983*. B. Brosowski and F. Deutsch, editors. Birkhäuser, Basel, 1985, pages 96–107.
- [83] F. Deutsch and H. Hundal: The rate of convergence for the method of alternating projections, II. *J. Math. Anal. Appl.*, 205(2):381–405, 1997.
- [84] F. Deutsch, W. Li, and J. D. Ward: A dual approach to constrained interpolation from a convex subset of Hilbert space. *J. Approx. Theory*, 90(3):385–414, 1997.
- [85] D. Ž. Đoković and I. S. Kotsireas: Compression of periodic complementary sequences and applications. *Des. Codes Cryptogr.*, 74(2):365–377, 2015.
- [86] D. Ž. Đoković and I. S. Kotsireas: D-optimal matrices of orders 118, 138, 150, 154 and 174. In C. J. Colbourn, editor, *Algebraic Design Theory and Hadamard Matrices*, pages 71–82, Cham. Springer International Publishing, 2015.
- [87] D. Ž. Đoković and I. S. Kotsireas: New results on D-optimal matrices. *J. Combin. Des.*, 20(6):278–289, 2012.
- [88] E. D. Dolan and J. J. Moré: Benchmarking optimization software with performance profiles. *Math. Program. Ser. A*, 91(2):201–213, 2002.
- [89] J. Douglas and H. H. Rachford: On the numerical solution of heat conduction problems in two and three space variables. *Trans. Amer. Math. Soc.*, 82:421–439, 1956.

-
- [90] R. L. Dykstra: An algorithm for restricted least squares regression. *J. Amer. Statist. Assoc.*, 78(384):837–842, 1983.
- [91] J. Eckstein and D. P. Bertsekas: On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Program.*, 55(1):293–318, 1992.
- [92] H. Ehlich: Determinantenabschätzungen für binäre matrizen. *Math. Zeitschr.*, 83:123–132, 1964.
- [93] V. Elser: Phase retrieval by iterated projections. *J. Opt. Soc. Am. A*, 20(1):40–55, 2003.
- [94] V. Elser, I. Rankenburg, and P. Thibault: Searching with iterated maps. *Proc. Natl. Acad. Sci.*, 104(2):418–423, 2007.
- [95] V. Elser: The complexity of bit retrieval. *IEEE Transactions on Information Theory*, 64(1):412–428, 2018.
- [96] P. Erdős and A. Rényi: On random graphs I. *Publ. Math. Debrecen*, 6:290–297, 1959.
- [97] P. Erdős, A. L. Rubin, and H. Taylor: Choosability in graphs. In *Proc. West Coast Conf. on Combinatorics, Graph Theory and Computing, Congressus Numerantium*, volume 26, pages 125–157, 1979.
- [98] R. Escalante and M. Raydan: *Alternating Projection Methods*, volume 8 of *Fundamentals of Algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2011.
- [99] M. Fält and P. Giselsson: Line search for generalized alternating projections. In *2017 American Control Conference (ACC)*, pages 4637–4642. IEEE, 2017.
- [100] M. Fält and P. Giselsson: Optimal convergence rates for generalized alternating projections. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2268–2274. IEEE, 2017.
- [101] S. T. Flammia and S. Severini: Weighing matrices and optical quantum computing. *J. Phys. A*, 42(6):065302, 2009.
- [102] P. Formanowicz and K. Tanaś: A survey of graph coloring - its types, methods and applications. *Found. Comput. Decision Sci.*, 37(3):223–238, 2012.
- [103] C. Franchetti and W. Light: On the von Neumann alternating algorithm in hilbert space. *J. Math. Anal. Appl.*, 114(2):305–314, 1986.

-
- [104] K. Friedrichs: On certain inequalities and characteristic value problems for analytic functions and for functions of two variables. *Trans. Amer. Math. Soc.*, 41(3):321–364, 1937.
- [105] N. Gaffke and R. Mathar: A cyclic projection algorithm via duality. *Metrika*, 36(1):29–54, 1989.
- [106] M. R. Garey, D. S. Johnson, and H. C. So: An application of graph coloring to printed circuit testing. *IEEE Transactions on circuits and systems*, 23(10):591–599, 1976.
- [107] M. R. Garey, D. S. Johnson, and L. Stockmeyer: Some simplified NP-complete graph problems. *Theoret. Comput. Sci.*, 1(3):237–267, 1976.
- [108] W. B. Gearhart and M. Koshy: Acceleration schemes for the method of alternating projections. *J. Comput. Appl. Math.*, 26(3):235–249, 1989.
- [109] S. W. Golomb and G. Gong: *Signal Design for Good Correlation*. Cambridge University Press, New York, 2004.
- [110] L. G. Gubin, B. T. Polyak, and E. V. Raik: The method of projections for finding the common point of convex sets. *USSR Comp. Math. Math. Phys.*, 7(6):1–24, 1967.
- [111] A. J. Gutman: *Circulant weighing matrices*. Master’s Thesis, Wright State University, USA, 2009.
- [112] W. K. Hale: Frequency assignment: theory and applications. *Proceedings of the IEEE*, 68(12):1497–1514, 1980.
- [113] I. Halperin: The product of projection operators. *Acta Sci. Math.*, 23:96–99, 1962.
- [114] B. Halpern: Fixed points of nonexpanding maps. *Bulletin of the AMS*, 73(6):957–961, 1967.
- [115] Y. Haugazeau: *Sur les inequality variationnelles et la minimization de fonctionnelles convexes*. Thèse, Université de Paris, France, 1968.
- [116] R. Hesse and D. R. Luke: Nonconvex notions of regularity and convergence of fundamental algorithms for feasibility problems. *SIAM J. Optim.*, 23(4):2397–2419, 2013.
- [117] R. Hesse, D. R. Luke, and P. Neumann: Alternating projections and Douglas–Rachford for sparse affine feasibility. *IEEE Transactions on Signal Processing*, 62(18):4868–4881, 2014.

-
- [118] K. J. Horadam: *Hadamard Matrices and Their Applications*. Princeton University Press, New Jersey, 2012.
- [119] R. A. Horn and C. R. Johnson: *Matrix Analysis*. Cambridge University Press, Cambridge, 2nd edition, 2013.
- [120] H. S. Hundal: An alternating projection that does not converge in norm. *Nonlinear Anal.*, 57(1):35–61, 2004.
- [121] O.F. Inc.: The on-line encyclopedia of integer sequences (2018). URL: <https://oeis.org/A088202>.
- [122] A. F. Izmailov, M. V. Solodov, and E. T. Uskov: Globalizing stabilized sequential quadratic programming method by smooth primal-dual exact penalty function. *J. Optim. Theor. Appl.*, 169(1):1–31, 2016.
- [123] T.R. Jensen and B. Toft: *Graph Coloring Problems*. John Wiley & Sons, New York, 1995.
- [124] F. Johansson: *mpmath*, version 1.0 (2017). URL: <http://mpmath.org>.
- [125] S. Kaczmarz: Angenaherte auflosung von systemen linearer gleichungen. *Bull. Int. Acad. Sci. Pologne, A*, 35:355–357, 1937. English translation: S. Kaczmarz, Approximate solution of systems of linear equations. *Int. J. Contr.* 57(6):1269–1271, 1993.
- [126] D. Karger, R. Motwani, and M. Sudan: Approximate graph coloring by semidefinite programming. *Journal of the ACM (JACM)*, 45(2):246–265, 1998.
- [127] R.M. Karp: Reducibility among combinatorial problems. In M. R. and T. J., editors, *Complexity of computer computations*, pages 85–103, New York. Plenum Press, 1972.
- [128] S. Kayalar and H. L. Weinert: Error bounds for the method of alternating projections. *Math. Control Signal Systems*, 1(1):43–59, 1988.
- [129] E. Kopecká and S. Reich: A note on the von Neumann alternating projections algorithm. *J. Nonlinear Convex Anal.*, 5(3):379–386, 2004.
- [130] I. S. Kotsireas: *Algorithms and metaheuristics for combinatorial matrices*. In *Handbook of Combinatorial Optimization*. P. M. Pardalos, D.-Z. Du, and R. L. Graham, editors. Springer-Verlag, New York, NY, 2013, pages 283–309.
- [131] I. S. Kotsireas, C. Koukouvinos, and J. Seberry: Hadamard ideals and hadamard matrices with two circulant cores. *European J. Combin.*, 27(5):658–668, 2006.

- [132] A. Y. Kruger, D. R. Luke, and N. H. Thao: Set regularities and feasibility problems. *Math. Program. Ser. B*, 168(1–2):279–311, 2018.
- [133] B. P. Lamichhane, S. B. Lindstrom, and B. Sims: Application of projection algorithms to differential equations: boundary value problems, 2017. arXiv: [1705.11032](#).
- [134] M. L. Lapidus: Generalization of the Trotter–Lie formula. *Integral Equations Operator Theory*, 4(3):366–415, 1981.
- [135] F. T. Leighton: A graph coloring algorithm for large scheduling problems. *J. Res. Nat. Bur. Standard*, 84(6):489–506, 1979.
- [136] A. Levi and H. Stark: Image restoration by the method of generalized projections with application to restoration from magnitude. *J. Opt. Soc. Am. A*, 1(9):932–943, 1984.
- [137] A. S. Lewis, D. R. Luke, and J. Malick: Local linear convergence for alternating and averaged nonconvex projections. *Found. Comput. Math.*, 9(4):485–513, 2009.
- [138] A. S. Lewis and J. Malick: Alternating projections on manifolds. *Math. Oper. Res.*, 33(1):216–234, 2008.
- [139] R. M. R. Lewis: *A Guide to Graph Colouring Algorithms and Applications*. Springer International Publishing.
- [140] P. L. Lions and B. Mercier: Splitting algorithms for the sum of two nonlinear operators. *SIAM J. Numer. Anal.*, 16(6):964–979, 1979.
- [141] P.-L. Lions: Approximation de points fixes de contractions. *CR Acad. Sci. Paris Serie, AB*, 284:1357–1359, 1977.
- [142] D. R. Luke: Finding best approximation pairs relative to a convex and a prox-regular set in a Hilbert space. *SIAM J. Optim.*, 19(2):714–739, 2008.
- [143] E. Matoušková and S. Reich: The Hundal example revisited. *J. Nonlinear Convex Anal.*, 4(3):411–427, 2003.
- [144] C. D. Meyer: *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, 2000.
- [145] G. A. Minty: A theorem on monotone sets in Hilbert spaces. *J. Math. Anal. Appl.*, 14:434–439, 1967.
- [146] T. S. Motzkin and I. J. Schoenberg: The relaxation method for linear inequalities. *Canad. J. Math.*, 6(3):393–404, 1954.

-
- [147] D. Noll and A. Rondepierre: On local convergence of the method of alternating projections. *Found. Comput. Math.*, 16(2):425–455, 2016.
- [148] P. M. Pardalos, T. Mavridou, and J. Xue: *The graph coloring problem: a bibliographic survey*. In *Handbook of Combinatorial Optimization*. D.-Z. Du and P. M. Pardalos, editors. Volume 1–3. Springer US, Boston, MA, 1999, pages 1077–1141.
- [149] A. Pazy: Asymptotic behavior of contractions in Hilbert space. *Israel J. Math.*, 9:235–240, 1971.
- [150] D. W. Peaceman and H. H. Rachford, Jr.: The numerical solution of parabolic and elliptic differential equations. *J. Soc. Indust. Appl. Math.*, 3(1):28–41, 1955.
- [151] H. M. Phan: Linear convergence of the Douglas–Rachford method for two closed sets. *Optim.*, 65(2):369–385, 2016.
- [152] G. Pierra: Decomposition through formalization in a product space. *Math. Program.*, 28:96–115, 1984.
- [153] S. Reich: A limit theorem for projections. *Linear Multilinear Algebra*, 13(3):281–290, 1983.
- [154] S. Reich and R. Zalas: A modular string averaging procedure for solving the common fixed point problem for quasi-nonexpansive mappings in Hilbert space. *Numer. Algor.*, 72(2):297–323, 2016.
- [155] S. Reich and R. Zalas: The optimal error bound for the method of simultaneous projections. *J. Approx. Theory*, 223:96–107, 2017.
- [156] R. T. Rockafellar: Monotone operators and the proximal point algorithm. *SIAM J. Control Optim.*, 14(5):877–898, 1976.
- [157] R. T. Rockafellar: *Convex Analysis*. Princeton University Press, Princeton, N.J., 1970.
- [158] M. Sala, S. Sakata, T. Mora, C. Traverso, and L. Perret, editors: *Gröbner Bases, Coding, and Cryptography*. Springer-Verlag, Berlin Heidelberg, 2009.
- [159] J. Schaad: *Modeling the 8-queens problem and Sudoku using an algorithm based on projections onto nonconvex sets*. Master’s thesis, University of British Columbia, Canada, 2010.
- [160] K. Schwarz: More NP completeness. Lecture notes for Mathematical Foundations of Computing. Stanford University. URL: <http://web.stanford.edu/class/archive/cs/cs103/cs103.1132/lectures/27/Small127.pdf>. Accessed 29 August 2017.

- [161] J. R. Seberry: *Orthogonal Designs: Hadamard Matrices, Quadratic Forms and Algebras*. Springer International Publishing, 2017.
- [162] J. Seberry and M. Yamada: Hadamard matrices, sequences, and block designs. In J. H. Dintz and D. R. Stinson, editors, *Contemporary Design Theory: A Collection of Surveys*, pages 431–560. John Wiley & Sons, 1992.
- [163] K. T. Smith, D. C. Solmon, and S. L. Wagner: Practical and mathematical aspects of the problem of reconstructing objects from radiographs. *Bull. Amer. Math. Soc.*, 83:1227–1270, 1977.
- [164] D. R. Stinson: *Combinatorial Designs*. Springer-Verlag, New York, 2004.
- [165] Y. Strassler: *The classification of circulant weighing matrices of weight 9*. PhD Thesis, Bar-Ilan University, Israel, 1997.
- [166] B. Sturmfels: *Algorithms in Invariant Theory*. Springer-Verlag, Vienna, 2008.
- [167] B. F. Svaiter: On weak convergence of the Douglas–Rachford method. *SIAM J. Control Optim.*, 49(1):280–287, 2011.
- [168] M. K. Tan: Regularity properties of non-negative sparsity sets. *J. Math. Anal. Appl.*, 447(2):758–777, 2017.
- [169] M. M. Tan: Group invariant weighing matrices. *Des. Codes Cryptogr.*, 2018. DOI: [10.1007/s10623-018-0466-5](https://doi.org/10.1007/s10623-018-0466-5).
- [170] M. M. Tan: *Relative difference sets and circulant weighing matrices*. PhD Thesis, Nanyang Technological University, Singapur, 2014.
- [171] N. H. Thao: A convergent relaxation of the Douglas–Rachford algorithm. *Comput. Optim. Appl.*, 70(3):841–863, 2018.
- [172] W. Van Dam: Quantum algorithms for weighing matrices and quadratic residues. *Algorithmica*, 34(4):413–428, 2002.
- [173] V. G. Vizing: Coloring the vertices of a graph in prescribed colors. *Diskret. Analiz.*, 29:3–10, 1976.
- [174] J. von Neumann: *Functional Operators II: The Geometry of Orthogonal Spaces*. Princeton University Press, 1950. (Reprint of mimeographed lecture notes first distributed in 1933).
- [175] R. Wittmann: Approximation of fixed points of nonexpansive mappings. *Arc. Math.*, 58(5):486–491, 1992.

List of Figures

1.1	Examples of projectors and reflectors onto convex and nonconvex sets. . . .	5
2.1	Behavior of the alternating projections method in three possible scenarios.	30
2.2	Failure of the method of alternating projections for solving the best approximation problem for arbitrary convex sets.	30
2.3	Illustration of two possible generalization of AP for finitely many sets. . . .	32
2.4	One iteration of RAP, PRAP and GAP methods.	33
2.5	Geometric interpretation of the Douglas–Rachford iteration.	36
2.6	Behavior of the Douglas–Rachford algorithm in three possible scenarios. . .	38
2.7	Failure of the Douglas–Rachford method for solving the best approximation problem for arbitrary convex sets.	38
2.8	Failure of the 3-sets Douglas–Rachford iteration.	39
2.9	Illustration of two versions of DR for finitely many sets.	40
2.10	One iteration of GDR, RAAR and CDR methods.	42
2.11	AP and DR algorithms applied to a finite set and a halfspace.	47
2.12	Illustration of Dykstra’s algorithm.	48
2.13	Illustration of Haugazeau’s algorithm.	50
2.14	Illustration of Halpern’s algorithm.	51
2.15	Illustration of Combettes’ algorithm.	52
3.1	Geometric interpretation of the modified reflector.	58
3.2	Geometric interpretation of the AAMR operator.	59
3.3	Illustration of the AAMR iterative scheme for two sets.	70
3.4	Behavior of the AAMR algorithm in three possible scenarios.	71
3.5	Illustration of Example 3.21.	73
3.6	Asymptotic behavior of the AAMR iteration.	75

3.7	Illustration of the AAMR iterative scheme for many sets in Theorem 3.26.	78
3.8	Best value of α with respect to the Friedrichs angle for 1000 pairs of random subspaces for AAMR (left) and CM (right). For each β , the average value of the best α is represented by a dashed line.	81
3.9	Number of required iterations with respect to the value of α of GDR and AAMR for three different values of the parameter β	81
3.10	Behavior of the AAMR (in blue) and alternating projections (in red) algorithms when applied to two lines in \mathbb{R}^2 for two different Friedrichs angles. We see that AAMR outperforms AP for small angles, while AP is faster for large angles.	82
3.11	Median of the required number of iterations with respect to the Friedrichs angle of AP, DR, Haugazeau's method, CM and AAMR for six different values of β	83
3.12	Standard deviation of the required number of iterations with respect to the Friedrichs angle of AP, DR, Haugazeau's method, CM and AAMR for six different values of β	84
3.13	Distance to the solution of the 100 first iterations of the monitored sequences of AP, DR, HLWB, Haugazeau's method and AAMR for five different values of the parameter β	85
3.14	Median number of iterations for 10 random starting points required by the AAMR method for different values of the parameter β with respect to the Friedrichs angle.	86
3.15	The three possible scenarios for the function $f_{\alpha,\beta,r}(c)$	92
3.16	Piecewise domain of the function $\Gamma(\alpha, \beta)$ for three different values of the angle.	98
3.17	Comparison of the rates of linear convergence with optimal parameters of AP, SP, RAP, DR, AAMR and GAP.	101
3.18	Graphical representation of Theorem 3.34 for two subspaces (a) and failure of the result for two arbitrary closed and convex sets (b). The sequence $\{x_k + z\}_{k=0}^{\infty}$ is generated by AAMR with $x_0 = 0$, while the sequence $\{z_k\}_{k=0}^{\infty}$ is generated by GAP with $z_0 = z$	104

3.19	Number of iterations required to converge for the AAMR algorithm with $\alpha = 0.8$ and five different values of the parameter β , with respect to the Friedrichs angle. The lines correspond to the approximate upper bounds given by (3.46) and the theoretical rates (3.32).	105
3.20	Median, difference between the maximum and the median, and coefficient of variation of the required number of iterations with respect to the Friedrichs angle of AP, SP, RAP, DR and AAMR for their respective optimal parameters.	106
3.21	Illustration of the AAMR iterative scheme for many sets in Theorem 3.48.	117
4.1	A 3-coloring of Petersen graph.	120
4.2	Gadgets of the variables and colors.	122
4.3	Gadgets of the clauses.	123
4.4	Two different formulations of the 3-SAT problem in (4.1) as a 3-coloring problem. The same solution of the 3-SAT problem is shown for both formulations.	124
4.5	List coloring reduced to graph coloring of a wheel graph of 5 nodes with admissible colors lists $L(1) = L(4) = \{1, 2, 3\}$, $L(2) = \{1\}$, $L(3) = \{3\}$, and $L(5) = \{2, 3\}$. Nodes c_1 , c_2 and c_3 represent colors 1, 2 and 3, respectively.	125
4.6	Graph formulation of a Sudoku, with maximal cliques highlighted.	126
4.7	Unsolved Sudoku puzzle (left) and its graph representation (right): each complete subgraph represents a subgrid and the squared nodes correspond to prefilled cells. The valid 9-coloring of the graph leads to a solution of the Sudoku.	126
4.8	A solution to the 8-queens puzzle (left) and its graph representation (right).	127
4.9	(a) A 16-knights puzzle with 4 colors: a solution will fill the chessboard. (b) A 10-queens puzzle with 3 colors played in a 9×9 chessboard with a hole. (c) Empty ‘ π -zzle’. The goal of this puzzle is to place on the board 8 times each of the 18 letters A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, R and W. Ten cells have been prefilled.	128
4.10	A Hamiltonian path constructed from a coloring of the graph.	129
4.11	Hamiltonian cycle reduced to Hamiltonian path.	130
4.12	A knight’s cycle on a 12×12 chessboard found with DR.	130
4.13	Plot of the windmill graph $Wd(6, 5)$	134

4.14	Number of iterations spent by DR to find a solution of a 3-coloring of Petersen graph for 100,000 random starting points. On average, each solution was found in 0.00533 seconds. Instances were labeled as “Unsolved” after 500 iterations.	138
4.15	Proper colorings of some simple graphs.	139
4.16	Cumulated number of instances solved by DR with respect to the 300 first iterations for some complete, cycle and wheel graphs of different sizes. . . .	139
4.17	Cumulated number of instances solved by DR (out of 10,000 random starting points) to find a solution of the windmill graph $Wd(6, 5)$, with and without maximal clique information, with respect to the iterations.	141
4.18	For 1,000 random starting points, we represent the iteration k in the horizontal axis and $\ Z_k\ $ in the vertical axis for 1,000 iterations of the Douglas–Rachford algorithm.	142
4.19	Performance profiles for the results of the 3-SAT experiment in Table 4.5.	144
4.20	Performance profiles for the results of the Sudoku experiment in Table 4.6.	145
4.21	Time (in \log_{10}) required by DR for finding knight’s paths and cycles on chessboards of different size. For each size, 50 random starting points were chosen. Blue dots represent instances of the DR method applied with the addition of the redundant constraint $\tilde{C}_2 = \mathbb{R}^{n \times n}$, while red crosses represent instances where the method was run without \tilde{C}_2 . The dotted lines were obtained by linear regression. The algorithm was stopped after a maximum time of 5,000 seconds, in which case the instance is not displayed.	146
4.22	Solution to the puzzles in Figure 4.9 computed with DR. For 10 random starting points, the average (maximum) time spent for puzzles (a), (b) and (c) was 0.23, 3.32 and 252.82 seconds (0.35, 11.49 and 424.67 seconds), respectively.	147
4.23	Graphical representation of Example 4.11.	149
4.24	Graphical representation of Example 4.14	153
4.25	Results of the Queens- n^2 experiment for three implementations of GDR. Each marker corresponds to the median of the solved instances among 10 random starting points. At the bottom of each graph, we show the percentage of solved instances for each value of α . Instances were considered as unsolved after 100,000 iterations.	157

4.26	Performance profiles of the Queens- n^2 experiment for three implementations of GDR.	158
4.27	Performance profiles of the Queens- n^2 experiment comparing the implementations $T_{C_1, C_2, 0.375}$, $T_{C_2, C_1, 0.25}$ and $T_{D, C, 0.5}$	158
4.28	Comparison of the number of iterations and the number of digits used in the machine precision for 10 random starting points, when $T_{D, C, 0.5}$ was employed to solve the Queens- 6^2 and the Queens- 7^2 puzzles. For every starting point and every value of the machine precision, the algorithm found a solution to the puzzle.	159
4.29	Comparison of the value of Error in (4.23) and the number of iterations for different number of decimal digits used in the machine precision, when $T_{D, C, 0.5}$ was employed to solve the Queens- 6^2 and the Queens- 7^2 puzzles.	159
4.30	Results of the experiment on m -colorable random graphs for $m = 8, 9, 10$, for GDR implemented with $T_{C_1, C_2, 0.375}$. Each marker corresponds to the median of the solved instances among 10 random starting points, and the lines where obtained by linear regression among all the solved instances.	161
4.31	Results of the experiment on m -colorable random graphs for $m = 8, 9, 10$, for the implementation $T_{C_1, C_2, \alpha}$ of GDR. Each marker corresponds to the median of the solved instances among 10 random starting points. At the bottom of each graph, we show the percentage of solved instances for each value of α . Instances were considered as unsolved after 100,000 iterations.	162
4.32	Performance profiles comparing the binary and the rank matrix formulations for solving 95 Sudoku problems. For each problem, 10 starting points were randomly generated. Instances were considered as unsolved after 300 seconds.	164
4.33	Number of solved instances (right), among 100 random starting points, to find the solution of the ‘nasty’ Sudoku (left) by GDR with the cubic, the binary, and the rank formulations. For each interval of time (in seconds), we show the number of solved instances and the cumulative proportion of solved instances for each formulation. The algorithm was stopped after a maximum of 300 seconds, in which case the problem was labeled as ‘Unsolved’.	165
4.34	Results for CW matrices (10 random initialization, 3600s time limit).	177

List of Tables

3.1	Rates of convergence with optimal parameters of AP, SP, RAP, GAP, GDR and AAMR when they are applied to two subspaces.	101
4.1	Number of iterations spent by DR to find a solution of an n -coloring of a complete graph with n vertices for 10,000 random starting points, with $n = 4, 5, 6$. Each solution was found, on average, in 0.00215 seconds for $n = 4$, 0.00371 seconds for $n = 5$, and 0.00569 seconds for $n = 6$. Instances were labeled as “Unsolved” after 500 iterations.	140
4.2	Number of iterations spent by DR to find a solution of two wheel graphs for 10,000 random starting points. Each solution was found, on average, in 0.00266 seconds for wheel 5, and 0.00455 seconds for wheel 6. Instances were labeled as “Unsolved” after 500 iterations.	140
4.3	Number of iterations spent by DR to find a solution of three cycle graphs for 10,000 random starting points. Each solution was found, on average, in 0.0025 seconds for cycle 10, 0.00561 seconds for cycle 15, and 0.00731 seconds for cycle 20. Instances were labeled as “Unsolved” after 500 iterations.	140
4.4	Comparison of the number of iterations spent by DR to find a solution of the windmill graph $Wd(6, 5)$ for 10,000 random starting points. Complete maximal clique information was used in the right columns. Each solution was found, on average, in 0.13347 seconds without clique information, and 0.02424 seconds with maximal clique information. Instances were labeled as “Unsolved” after 10,000 iterations.	141
4.5	Time spent (in seconds) by DR to find a solution of 50 different 3-SAT problems with 20 variables and 91 clauses. For each problem, 10 random starting points were chosen. After 300 seconds without finding a solution, instances were labeled as “Unsolved”. Two formulations of the gadgets were considered, with 4 and 5 nodes.	143

4.6	Time spent (in seconds) to find the solution of 95 different Sudoku problems by DR with the graph precoloring, the cubic, and the graph coloring formulations. For each problem, 20 starting points were randomly chosen. We stopped the algorithm after a maximum time of 300 seconds, in which case the problem was labeled as “Unsolved”	145
4.7	Number of failed runs in either the cubic or the graph precoloring formulation. Sudokus not listed here were solved by these two formulations for every starting point.	146
4.8	Chromatic number $\chi(n)$ of the Queens n^2 graph.	156
4.9	Summary of the results of GDR for finding proper colorings of windmill graphs. For each formulation, we show the number of solved instances, the averaged time (in seconds) and the averaged number of iterations. Instances were considered as unsolved after 60 seconds.	163
4.10	Summary of the results of the GDR algorithm implemented with $T_{C_1, C_2, 0.375}$ for finding proper colorings of a representative sample of DIMACS benchmark instances. For each problem, we show the number of solved runs, the average time (in seconds) and the average number of iterations. We also include the number of nodes and edges, and the chromatic number of each graph. Runs were considered as unsolved after 3600 seconds.	166
4.11	Experimental results for D-optimal designs with parameters (n, α, β) given by Proposition 4.33 (10 random initialization, 3600s time limit).	177
4.12	Experimental results for DCHM designs with parameters $(n, \alpha = 1, \beta = 1)$ given by Proposition 4.36 (10 random initialization, 3600s time limit).	178
4.13	Detailed results for CW matrices (10 random initialization, 3600s time limit).	182

Notation and Symbols

Basic notation

\mathcal{H}	real Hilbert space
$\langle x, y \rangle$	inner product of the vectors x and y
$\ x\ $	norm of x induced by the inner product, i.e., $\ x\ = \sqrt{\langle x, x \rangle}$
$\mathbb{B}(x, \rho)$	open ball centered at x with radius ρ
$x_n \rightharpoonup x$	the sequence $(x_n)_{n=1}^{\infty}$ converges weakly to the point x
$x_n \rightarrow x$	the sequence $(x_n)_{n=1}^{\infty}$ converges strongly to the point x

Sets

C^{\perp}	orthogonal complement of the set C
\overline{C}	closure of the set C
$\text{aff } C$	affine hull of the set C
$\text{cone } C$	cone generated by the set C
$\text{conv } C$	convex hull of the set C
$\text{core } C$	algebraic interior of the set C
d_C	distance function to the set C
i_C	indicator function of the set C
$\text{int } C$	interior of the set C
P_C	projector onto the set C
R_C	reflector with respect to the set C
$\text{ri } C$	relative interior of the set C
$\text{span } C$	span of the set C
$\text{sri } C$	strong relative interior of the set C
σ_C	support function of the set C

Functions

$f : \mathcal{H} \rightarrow \mathbb{R} \cup \{\pm\infty\}$	extended real-valued function
$\text{dom } f$	domain of the function f
$\text{epi } f$	epigraph of the function f
prox_f	proximity operator of the function f
∂f	subdifferential of the function f

Operators

$T : D \rightrightarrows \mathcal{H}$	set-valued operator from the set D to \mathcal{H}
$T : D \rightarrow \mathcal{H}$	single-valued operator from the set D to \mathcal{H}
T^{-1}	inverse operator of T
J_T	resolvent of the operator T
R_T	reflected resolvent of the operator T
T_w	inner w -perturbation of the operator T
$\text{dom } T$	domain of the operator T
$\text{Fix } T$	set of fixed points of the operator T
$\text{gra } T$	graph of the operator T
Id	identity mapping
$\text{ran } T$	range of the operator T
$\text{zer } T$	set of zeros of the operator T
$\mathcal{F} : \mathbb{C}^n \rightarrow \mathbb{C}^n$	unitary discrete Fourier transform
$\star : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^n$	periodic correlation operator

Subspaces

$\dim U$	dimension of the linear subspace U
$U - U$	parallel linear subspace to the affine subspace U
θ_F	Friedrichs angle between two linear subspaces
$c_F(U, V)$	cosine of the Friedrichs angle between U and V

Matrices

$\mathbb{C}^{n \times m}$	vector space of $n \times m$ complex matrices
$\mathbb{R}^{n \times m}$	vector space of $n \times m$ real matrices
\mathcal{S}^n	subspace of symmetric matrices in $\mathbb{R}^{n \times n}$
\mathcal{S}_+^n	cone of positive semidefinite matrices in $\mathbb{R}^{n \times n}$
I_n	$n \times n$ identity matrix
0_n	$n \times n$ zero matrix
$0_{n \times m}$	$n \times m$ zero matrix
A^T	transpose of the matrix A
A^*	conjugate transpose of the matrix A
A^{-1}	inverse of the matrix A
$\text{circ}(a)$	circulant matrix whose first row is the vector a
$\det A$	determinant of the square matrix A
$\text{diag}(a)$	diagonal matrix whose entries are the elements of the vector a
$\text{Fix } A$	set of fixed points of the matrix A
$\ker A$	kernel of the matrix A
$\text{ran } A$	range of the matrix A
$\text{rank } A$	rank of the matrix A
$\text{tr } A$	trace of the matrix A
$\sigma(A)$	set of all eigenvalues (spectrum) of the matrix A
$\rho(A)$	spectral radius of the matrix A
$\gamma(A)$	modulus of the subdominant eigenvalues of the matrix A
$\ A\ _2$	matrix 2-norm of the matrix A
$\ A\ _F$	Frobenius norm of the matrix A

Index of Topics

- 3-SAT, [121](#)
- 8-queens, [127](#)
- affine hull, [2](#)
- alternating projections, [28](#)
 - averaged, [31](#)
 - cyclic, [31](#)
 - generalized, [33](#)
 - partial relaxed, [33](#)
 - relaxed, [32](#)
- angle
 - Friedrichs, [17](#)
 - principal, [17](#)
- averaged alternating modified reflections (AAMR)
 - method, [56](#)
 - operator, [59](#)
 - splitting algorithm, [107](#)
- averaged mapping, [7](#)
- Banach–Picard iteration, [6](#)
- best approximation problem, [26](#)
- Chebyshev set, [4](#)
- circulant weighing matrix (CW), [172](#)
- clique, [121](#)
- cocoercive mapping, [7](#)
- Combette’s method, [52](#)
- combinatorial design of circulant type, [168](#)
- complementary sequences, [21](#)
- complete graph, [139](#)
- cone, [2](#)
 - generated, [2](#)
 - normal, [5](#)
- convergent matrix, [19](#)
- convex
 - function, [3](#)
 - hull, [2](#)
 - set, [1](#)
- correlation, [20](#)
- cycle graph, [139](#)
- D-optimal design of circulant type, [173](#)
- distance function, [3](#)
- domain
 - of a function, [3](#)
 - of an operator, [2](#)
- Douglas–Rachford
 - algorithm, [36](#)
 - circumcentered, [41](#)
 - cyclic, [40](#)
 - generalized, [41](#)
 - operator, [36](#)
 - relaxed averaged alternating reflections, [41](#)
 - splitting algorithm, [43](#)
- Dykstra’s algorithm, [47](#)

- eigenvalue
 - semisimple, 16
 - subdominant, 16
- epigraph, 3
- feasibility problem, 26
- firmly nonexpansive mapping, 7
- fixed point
 - iteration, *see* Banach–Picard iteration
 - of a matrix, 15
 - of an operator, 2
- Fourier transform, 22
- Frobenius norm, 16
- function, 3
- Gram matrix, 15
- graph, 120
 - coloring problem, 120
 - partial coloring, 127
 - precoloring and list coloring, 124
 - of an operator, 2
- Hadamard matrix with two circulant cores (DCHM), 175
- Hamiltonian path, 129
- Haugazeau’s algorithm, 49
- Hermitian matrix, 15
- HLWB method, 50
- identity operator, 2
- indicator function, 3
- induced norm, 16
- inner perturbation, 13
- interior, 1
 - algebraic, 2
 - relative, 2
 - strong relative, 2
- inverse operator, 2
- kernel of a matrix, 15
- knight’s tour, 130
- Krasnosel’skiĭ–Mann iteration, 9
- lower semicontinuous function, 3
- matrix 2-norm, 16
- maximally monotone operator, 11
- modified
 - reflected resolvent, 108
 - reflector, 58
- monotone operator, 10
- nonexpansive
 - mapping, 7
 - matrix, 15
- orthogonal complement, 2
- performance profiles, 143
- Petersen graph, 120
- positive semidefinite matrix, 15
- product-space reformulation, 27
- projection mapping, 4
- proper
 - coloring, 120
 - function, 3
- proximal-point algorithm, 14
- proximal set, 4
- proximity operator, 3
- range
 - of a matrix, 15
 - of an operator, 2
- rank of a matrix, 15
- reflected resolvent, 12
- reflector, 4

-
- relaxation, [8](#)
 - resolvent, [12](#)
 - set-valued operator, [2](#)
 - single-valued mapping, [2](#)
 - span, [2](#)
 - spectral radius, [16](#)
 - spectrum, [16](#)
 - standard centered regular simplex, [148](#)
 - strengthening of an operator, [108](#)
 - strong CHIP, [6](#)
 - strongly monotone operator, [11](#)
 - subdifferential, [3](#)
 - Sudoku, [125](#)
 - support function, [3](#)
 - unitary matrix, [15](#)
 - wheel graph, [139](#)
 - windmill graph, [134](#)
 - zeros of an operator, [2](#)