Demystifying the characterization of SDP matrices in mathematical programming

Daniel Porumbel[1]

CEDRIC-CNAM, 292 rue Saint Martin, Paris France
daniel.porumbel@cnam.fr

1 Présentation du travail

Cette communication s'adresse aux chercheurs (en programmation linéaire (en nombre entiers), meta-heuristiques ou d'autres branches de la RO) qui voudraient élargir leurs connaissances pour utiliser des techniques d'optimisation sémi-définie positive (SDP) dans leur travaux. [1] Plus exactement, je présente un travail d'environ 100 pages [1] qui permettra au lecteur d'acquérir les bases de ce domaine. J'ai réalisé ce travail principalement parce que je n'ai pas trouvé d'autre introduction à l'optimisation SDP qui cible le même public (voir Section 2).

Le manuscrit proposé devrait être accessible à tous ceux qui ne détestent pas les mathématiques, qui savent ce qu'est une dérivée et peuvent accepter sans preuve des résultats très basiques comme $\det(AB) = \det(A)\det(B)$. Si vous êtes dans ce cas, je pense vous pouvez comprendre ce manuscrit sans acheter d'autres livres. L'objectif primordial n'est pas de rappeler ou d'énumérer une liste de résultats, mais de faire le lecteur examiner (les preuves de) ces résultats, pour qu'il puisse développer une compréhension profonde du sujet.

Pour se lancer dans l'optimisation SDP, on pourrait naturellement être tenté de commencer par un livre consacré ou un « handbook of SDP ... ». Mais si le lecteur ne possède pas déjà des connaissances assez fraiches en mémoire sur les matrices SDP, je pense que cette voie, bien que basée sur d'excellentes intentions, risque d'être inaccessible. Un pas crucial pour comprendre la programmation SDP est la décomposition d'une matrice SDP en éléments propres ; c'est un concept que le néophyte ne devrait pas prendre à la légère, c.à.d, une lecture rapide d'attention moyenne serait insuffisante. Il suffit de regarder comment ces livres/handbooks présentent cette décomposition pour comprendre qu'ils ciblent un public tout simplement très différent (voir des détails en Section 2).

Ce travail ne s'adresse pas non plus à ceux qui peuvent dire « Je me rappelle un peu la décomposition en éléments propres, car je l'ai fait en L3/M1 il y quelques $n \geq 5$ ans ; pas besoin de preuve. » Je crains qu'une telle attitude ne vise que l'information, sans donner aucun moyen pour pénétrer dans l'essence des choses. De toute façon, mon cerveau fonctionne de la manière la plus opposée : j'aime tout apprendre en vérifiant les preuves par moi-même, dans le moindre détail si possible. Ainsi, les seuls résultats non prouvés dans ce manuscrit sont le théorème fondamental de l'algèbre et deux théorèmes dans section 5.3.2.3. Mais je fournis des preuves complètes pour, par exemple, la décomposition de Cholesky, le théorème de forte dualité pour la programmation linéaire conique (y compris SDP), six formulations équivalentes du nombre de Lovász, la formulation copositive du problème du stable, des résultats de convexification pour les programmes quadratiques et beaucoup d'autres.

J'ai essayé de tout prouver par moi-même, de sorte que certaines preuves sont originales bien que ce manuscrit n'a pas été conçu pour être un travail de recherche. Bien évidement, j'ai eu recours plusieurs fois à l'Internet, à des articles et à des livres, les références étant

1. Je n'ai pas soumis ce travail à un track de programmation non-linéaire, parce que cette communication ne s'adresse pas principalement aux chercheurs consacrés en programmation non-linéaire, convexe ou polynomiale, mais aux chercheurs/étudiants qui veulent découvrir le sujet pour la première fois, e.g., pour trouver des relaxations plus fortes que par relaxation linéaire.

indiquées sous forme de citations de notes de bas de page. J'ai essayé de tout simplifier autant que possible. Les briques de base sont présentées dans la première partie du manuscrit. Il est nécessaire de vraiment maîtriser les résultats de cette première partie avant de se lancer sur la deuxième partie ; en fait, ces briques de base pourraient être utiles si vous essayez de lire d'autres introductions plus avancées à la SDP. Le but de ce manuscrit est de vous donner tous les moyens pour pouvoir passer à un niveau supérieur et effectuer des travaux de recherche.

Une autre différence par rapport à la littérature consacrée officiellement est que ce travail n'est pas écrit par un expert de classe mondiale en programmation SDP, mais il sort tout simplement d'un cerveau qui a du lutter pour comprendre. Cela peut sembler n'être qu'une faiblesse, mais, paradoxalement, c'est à la fois une faiblesse et un avantage. Un avantage vient du fait que de nombreux experts tentent d'oublier les difficultés des débutants, ce qui, je l'espère, ne m'est pas arrivé. D'autres experts s'efforcent de rendre toutes les preuves aussi courtes que possible et de diminuer l'importance de certains résultats qu'ils ont vus des milliers de fois dans leur carrière. J'ai clairement évité cela ; en fait, bien que j'aie raccourci quelques preuves lorsque j'ai révisé ce manuscrit deux ans après la première esquisse, j'ai gardé d'autres preuves un peu trop longues pour viser une compréhension plus profonde du sujet.

Last but not least : si je n'essaie pas d'assumer un rôle d'expert mondial en programmation SDP c'est aussi pour éviter d'engager le lecteur dans une relation (quelquefois aride) de type professeur→étudiant ; au contraire, j'essaie de réduire la distance auteur−lecteur au minimum (au point même d'envisager la possibilité d'une certaine empathie réciproque).

2 La décomposition en éléments propres dans d'autres travaux

Comme déjà indiqué, il suffit de regarder comment d'autres introductions au domaine abordent la décomposition en éléments propres pour voir qu'ils ciblent un public assez différent.

Les introductions de différentes « Handbook of SDP ... » sont plutôt courtes (pas plus d'une vingtaine de pages) et elles rappellent diverses résultats avec des références vers d'autres livres pour obtenir les preuves. Ainsi, dans le « Handbook of Semidefinite Programming Theory, Algorithms, and Applications » par H. Wolkowicz, R. Saigal et L. Vandenberghe, la décomposition en éléments propres (appelée «spectral theorem») est présentée/rappelée sans preuve dans le chapitre 2. L'introduction du "Handbook on Semidefinite, Conic and Polynomial Optimization" par M. Anjost et J.B. Lasserre renvoie le lecteur au livre "Matrix analysis" par Horn and Johnson pour cette même décomposition.

Dans la thèse de HDR « Semidefinite Programming for Combinatorial Optimization », par C. Helmberg, la décomposition est présentée en annexe et redirige le lecteur vers le même livre "Matrix analysis" par Horn et Johnson. Le livre « Convex Optimization» par S. Boyd et L. Vandenberghe utilise des matrices SDP dès le début (e.g., pour définir des ellipsoïdes dans la Section 2.2.2) sans présenter le concept de matrice SDP, même pas en annexe.

J'ai aussi parcouru divers cours de master/PhD, mais certains cours ne sont pas « self-contained » au sens fort, prenant des fois comme pré-requis certains résultats que j'ai préféré prouver dans mon manuscrit. Les slides du cours "Programmation linéaire et optimisation combinatoire" de F. Roupin pour le Master Parisien de Recherche Opérationnelle (MPRO) fournit de nombreux résultats que j'ai intégré dans mon manuscrit, mais j'ai souvent eu besoin de fournir les preuves moi-même. Le cours « Introduction to Semidefinite Programming » by R. Freund à MIT ne donne même pas la définition d'une matrice SDP avant de commencer.

Finalement, mon exposé oral va donner une ou deux preuves de cette décomposition en éléments propres, car, je répète, elle est essentielle pour comprendre les bases de l'optimisation SDP. En suivant ces preuves, les auditeurs pourraient comprendre assez vite s'ils sont attirés par ce domaine ou pas.

Références

[1] D. Porumbel. Demystifying the characterization of SDP matrices in mathematical programming *Rapport technique CEDRIC-CNAM.* `cedric.cnam.fr/~porumbed/papers/sdp.pdf`