ULCO
UCA

École doctorale **SI & ED SPI**
Unité de recherche **LMPA& LAMAI**

Thèse présentée par **Achraf Badahmane**
Soutenue le **27 décembre 2019**
En vue de l'obtention du grade de docteur de l'ULCO et de l'UCA

Discipline **Mathématiques**
Spécialité **Mathématiques Appliquées**

Titre de la thèse

Méthodes de sous espaces de Krylov préconditionnées pour les problèmes de point-selle avec plusieurs seconds membres

Thèse dirigée par	Abdeslem Hafid Bentbib	directeur
	Hassane Sadok	co-directeur

Composition du jury

Rapporteurs	Mohammed Seaid	professeur à l'Université de Durham	
	Souad El Bernoussi	professeur à l'UM5	
	Jilali Abouir	professeur à l'UH2C	
Examinateur	Mohamed El Alaoui Talibi	professeur à l'UCA	président du jury
Directeurs de thèse	Abdeslem Hafid Bentbib	professeur à l'UCA	
	Hassane Sadok	professeur à l'ULCO	

ULCO
UCA

École doctorale **SI & ED SPI**
Unité de recherche **LMPA& LAMAI**

Thèse présentée par **Achraf Badahmane**
Soutenue le **27 décembre 2019**
En vue de l'obtention du grade de docteur de l'ULCO et de l'UCA

Discipline **Mathématiques**
Spécialité **Mathématiques Appliquées**

Titre de la thèse

Méthodes de sous espaces de Krylov préconditionnées pour les problèmes de point-selle avec plusieurs seconds membres

Thèse dirigée par	Abdeslem Hafid Bentbib	directeur
	Hassane Sadok	co-directeur

Composition du jury

Rapporteurs	Mohammed Seaid	professeur à l'Université de Durham	
	Souad El Bernoussi	professeur à l'UM5	
	Jilali Abouir	professeur à l'UH2C	
Examinateur	Mohamed El Alaoui Talibi	professeur à l'UCA	président du jury
Directeurs de thèse	Abdeslem Hafid Bentbib	professeur à l'UCA	
	Hassane Sadok	professeur à l'ULCO	

ULCO
UCA

Doctoral School **SI & ED SPI**
University Department **LMPA& LAMAI**

Thesis defended by **Achraf Badahmane**
Defended on **27th December, 2019**
In order to become Doctor from ULCO and from UCA

Academic Field **Mathematics**
Speciality **Applied Mathematics**

Thesis Title

Preconditioned global Krylov subspace methods for solving saddle point problems with multiple right-hand sides

Thesis supervised by	Abdeslem Hafid Bentbib	Supervisor
	Hassane Sadok	Co-Supervisor

Committee members

Referees	Mohammed Seaid	Professor at Université de Durham	
	Souad El Bernoussi	Professor at UM5	
	Jilali Abouir	Professor at UH2C	
Examiner	Mohamed El Alaoui Talibi	Professor at UCA	Committee President
Supervisors	Abdeslem Hafid Bentbib	Professor at UCA	
	Hassane Sadok	Professor at ULCO	

Cette thèse a été préparée au

LMPA& LAMAI
Maison de la Recherche Blaise Pascal
50, rue Ferdinand Buisson
CS 80699
62228 Calais Cedex
France
UCA, FSTG, BP 549, Marrakech
Maroc

☎	(33)(0)3 21 46 55 86
🖷	(212)(0)5 24 43 34 04
🖂	`secretariat@lmpa.univ-littoral.fr`
Site	`http://www-lmpa.univ-littoral.fr/`

Je dédie ce travail

À mon papa

À ma maman

À mes frères...

Méthodes de sous espaces de Krylov préconditionnées pour les problèmes de point-selle avec plusieurs seconds membres

Résumé

La résolution numérique des problèmes de point-selle a eu une attention particulière ces dernières années. A titre d'exemple, la mécanique des fluides et solides conduit souvent à des problèmes de point-selle. Ces problèmes se présentent généralement par des équations aux dérivées partielles que nous linéarisons et discrétisons. Le problème linéaire obtenu est souvent mal conditionné. Le résoudre par des méthodes itératives standard n'est donc pas approprié. En plus, lorsque la taille du problème est grande, il est nécessaire de procéder par des méthodes de projections. Nous nous intéressons dans ce sujet de thèse à développer des méthodes numériques robustes et efficaces de résolution numérique de problèmes de point-selle. Nous appliquons les méthodes de Krylov avec des techniques de préconditionnement bien adaptées à la résolution de problèmes de point selle. L'efficacité de ces méthodes à été montrée dans les tests numériques.

Mots clés : point-selle, préconditionnement, krylov, produit de kronecker, produit de diamant

Preconditioned global Krylov subspace methods for solving saddle point problems with multiple right-hand sides

Abstract

In these last years there has been a surge of interest in saddle point problems. For example, the mechanics of fluids and solids often lead to saddle point problems. These problems are usually presented by partial differential equations that we linearize and discretize. The linear problem obtained is often ill-conditioned. Solving it by standard iterative methods is not appropriate. In addition, when the size of the problem is large, it is necessary to use the projection methods. We are interested in this thesis topic to develop an efficient numerical methods for solving saddle point problems. We apply the Krylov subspace methods incorporated with suitable preconditioners for solving these types of problems. The effectiveness of these methods is illustrated by the numerical experiments.

Keywords: saddle point , preconditioner, global krylov subspace method, kronecker product

LMPA& LAMAI
Maison de la Recherche Blaise Pascal – 50, rue Ferdinand Buisson – CS 80699 – 62228 Calais Cedex – France – UCA, FSTG, BP 549, Marrakech – Maroc

Remerciements

Ah les remerciements... J'ai bien dû y penser une centaine de fois durant ma thèse. J'avais hâte d'en arriver à ce moment-là et d'écrire les mots justes pour chacun d'entre vous. Et bien finalement, c'est une tâche bien plus difficile à réaliser qu'on ne le croit, aucun article pour s'appuyer, seulement moi et le souvenir de ces trois années de thèse...

Je tiens à exprimer mes plus vifs remerciements à Messieurs Hassane Sadok et Abdeslem Hafid Bentbib qui furent pour moi des directeurs de thèse attentif et disponible malgré leurs nombreuses charges. leurs compétences, leurs rigueurs scientifiques et leurs clairvoyances m'ont beaucoup appris. Ils ont été et resteront des moteurs de mon travail de chercheur.

J'exprime tous mes remerciements à l'ensemble des membres de mon jury : Madame Souad El Bernoussi et Messieurs Mohamed El Alaoui Talibi, Mohammed Seaid et Jilali Abouir.

J'adresse toute ma gratitude à tous mes ami(e)s et à toutes les personnes qui m'ont aidé dans la réalisation de ce travail. Je remercie le directeur du laboratoire ainsi que Khalide Jbilou et Abderrahman Bouhamidi pour m'avoir accueilli dans l'unité de recherche LMPA-UCLO Calais et de m'avoir permis de travailler dans bonnes conditions.

Enfin, les mots les plus simples étant les plus forts, j'adresse toute mon affection à ma famille, et en particulier à mon papa et ma maman qui m'ont fait comprendre que la vie n'est pas faite que de problèmes qu'on pourrait résoudre grâce à des formules mathématiques et des algorithmes. Malgré mon éloignement depuis de (trop) nombreuses années, leur intelligence, leur confiances, leurs tendresses, leur amour me portent et me guident tous les jours. Je vous aime.

Table des matières

Table des figures

Liste des Algorithmes

Première partie

General Introduction

Introduction générale

Sommaire

Selle

La selle est un objet en cuir placé sur le dos d'un cheval et sur lequel le cavalier se place. La première utilisation de selle est généralement attribuée aux Maures en Afrique du Nord. Ces premiers hommes ont commencé à apprendre comment conduire un cheval. Pendant un certain temps la conduite était inconfortable et des selles ont commencé à émerger, formant une barrière rembourrée entre le cavalier et le cheval. Comme avec toutes les inventions, elles ont commencé simplement, mais sont rapidement devenues décorées et plus élaborées.
Nous donnons maintenant la définition de point-selle :

FIGURE 1 – Selle.

Point-selle

Définition : Soient X et Y deux ensembles et $g : X \times Y \to \overline{\mathbb{R}}$ une fonction pouvant prendre les valeurs $\pm\infty$. On dit que $(\bar{x}, \bar{y}) \in X \times Y$ est un point-selle de g sur $X \times Y$ si

$$\forall\, (x,y) \in X \times Y : \quad g(\bar{x}, y) \leqslant g(\bar{x}, \bar{y}) \leqslant g(x, \bar{y}).$$

Dans les conditions ci-dessus, $g(\bar{x}, \bar{y})$ est appelée la valeur-selle de g. Comme application de cette définition nous prenons l'exemple suivant :

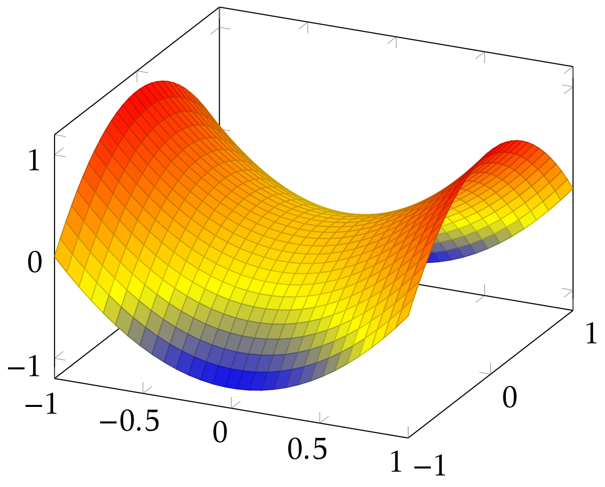
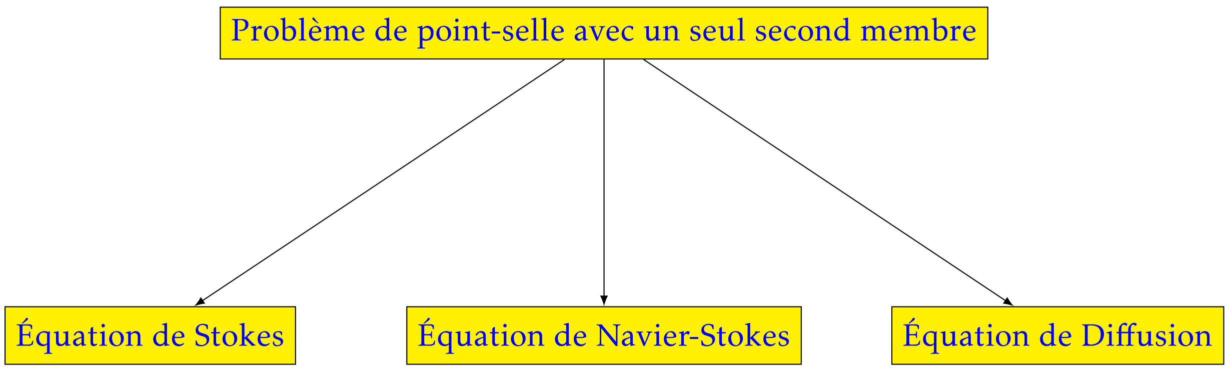


FIGURE 2 – $g(x,y) = x^2 - y^2$

D'après la définition ci-dessus, il est clair que $(0,0,0)$ est un point-selle de la surface.

Les problèmes de point-selle avec un seul second membre

Dans cette section, nous allons présenter les différentes applications de point-selle.


Problème de point-selle avec un seul second membre
Équation de Stokes
Équation de Navier-Stokes
Équation de Diffusion

Équation de Stokes

L'équation de Stokes permet de décrire un fluide visqueux s'écoulant lentement dans un lieu étroit ou autour d'un petit objet. Les effets visqueux dominent sur les effets inertiels. Son écoulement est alors appelé écoulement de Stokes ou écoulement rampant. Il est en effet régi par une version simplifiée de l'équation de Navier-Stokes : l'équation de Stokes, dans laquelle les influences extérieures sont absents. L'écoulement de Stokes correspond ainsi à un faible nombre de Reynolds (beaucoup plus petit que 1). L'équation de Stokes permet en particulier de décrire l'écoulement d'un fluide dans un tyau et aussi l'écoulement d'un fluide visqueux entre deux surfaces dont l'une est en mouvement par rapport à l'autre. Cette équation s'écrit :

$$\left\{ \begin{array}{rcl} -\nabla^2 u + \nabla p & = & f \text{ dans } \Omega, \\ \nabla \cdot u & = & 0 \text{ dans } \Omega, \end{array} \right. \tag{1}$$

u et p représentent respectivement le champ de vitesse et la pression du fluide. On considère le problème posé sur un domaine Ω de dimension 2 ou 3 avec des conditions aux bords définies par

$$u = 0 \ \text{ sur } \partial\Omega.$$

Formulation faible

Nous commençons par définir quelques notations qui seront utilisées dans cette sous-section.

$$L_2(\Omega) := \left\{ u : \Omega \to \mathbb{R} \mid \int_\Omega u^2 < \infty \right\},$$

et l'espace de Sobolev par :

$$H^1(\Omega) := \left\{ u : \Omega \to \mathbb{R} \mid u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y} \in L_2(\Omega) \right\}.$$

Nous définissons les espaces de solutions et tests :

$$H^1_E(\Omega) := \left\{ u \in H^1(\Omega)^2 \mid u = w \text{ sur } \partial\Omega_D \right\},$$

$$H^1_{E_0} := \left\{ v \in H^1(\Omega)^2 \mid v = 0 \text{ sur } \partial\Omega_D \right\}.$$

La formulation faible de (1.1) va être définie sous la forme suivante : trouver un $u \in H^1_{E_0}$ et $p \in L_2(\Omega)$ tels que :

$$\begin{array}{rcl} \int_\Omega \nabla u : \nabla v + \int_\Omega p(\nabla \cdot v) & = & \int_\Omega f \cdot v \ \forall v \in H^1_{E_0}, \\ \int_\Omega q(\nabla \cdot u) & = & 0 \ \ \forall q \in L_2(\Omega). \end{array} \tag{2}$$

Formulation faible discrète

Nous utilisons des espaces de dimension finie. Soient $V^h \subset H^1_{E_0}$ et $Q^h \subset L_2(\Omega)$, la version discrète de (2) devient :

trouver $u_h \in V^h$ et $p \in Q^h$

$$\begin{array}{rcl} \int_\Omega \nabla u_h : \nabla v_h + \int_\Omega p_h(\nabla \cdot v_h) & = & \int_\Omega f \cdot v_h \quad \forall v_h \in V^h, \\ \int_\Omega q_h(\nabla \cdot u_h) & = & 0 \ \ \forall q_h \in Q^h. \end{array} \tag{3}$$

En remplaçant les expressions de u_h et p_h en fonction des bases de V^h et Q^h dans (3), on obtient le système d'équations suivant :

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ 0 \end{pmatrix}, \tag{4}$$

où $A = [a_{i,j}]$, $a_{i,j} = \int_\Omega \nabla\phi_i : \nabla\psi_j$, pour $i = 1,..,n_u$ et $j = 1,..,n_u$.

$B = [b_{k,j}]$, $b_{k,j} = \int_\Omega \psi_k \cdot \nabla\phi_j$, pour $k = 1,..,n_p$ et $j = 1,..,n_u$, (ϕ_i base de V^h et ψ_j base de Q^h). Dans la sous-section suivante, nous allons donner une application de l'équation de Stokes.

Écoulement sur une marche arrière

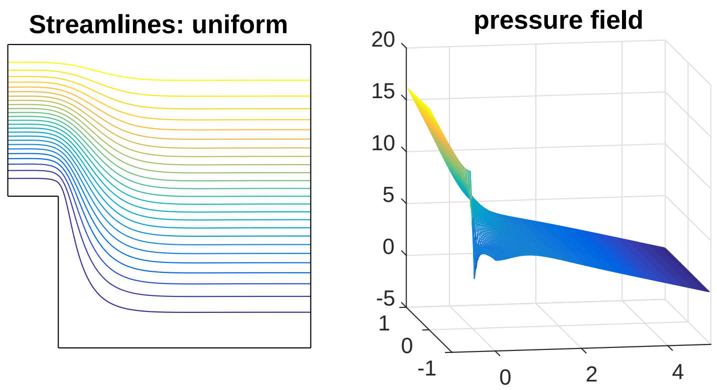


Figure 3 – Solution et rendu de la pression

La figure 3 montre l'écoulement du fluide et le rendu de la pression dans un conduit rectangulaire. On pose les conditions de Dirichlet à l'entrée du conduit rectangulaire et les conditions de Neumann à la sortie du conduit. D'après l'expression de la solution classique du fluide, la vitesse verticale est égale à zéro, ce qui entraîne que les lignes de l'écoulement du fluide se propagent d'une manière parallèle à la paroi du conduit rectangulaire. Le rendu de la pression devient plus proche de zéro à la sortie du conduit. Ensuite nous utilisons le mode de discrétisation $Q_1 - P_0$ (voir la figure 4), pour discrétiser ce problème. Les points noirs correspondent aux nœuds utilisés pour discrétiser le vecteur vitesse et le point en jaune correspond au nœud utilisé pour discrétiser le vecteur pression.

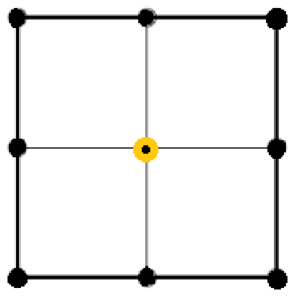


Figure 4 – Variation de la discrétisation

Équation de Navier-Stokes

En mécanique des fluides, les équations de Navier-Stokes sont des équations aux dérivées partielles non linéaires qui décrivent le mouvement des fluides (gaz ou liquide). On conçoit que l'équation de Navier-Stokes concerne beaucoup de choses. Il peut être utilisé pour comprendre le mouvement des courants dans les océans, optimiser les ailes d'avions et modéliser la circulation de sang dans nos artères, etc. La résolution des équations de Navier-Stokes constitue l'un des problèmes du prix du millénaire. Ces équations sont nommées ainsi pour honorer les travaux de deux scientifiques du XIXe siècle : le mathématicien et ingénieur des Ponts Henri Navier, qui le premier a introduit la notion de viscosité dans les équations d'Euler en 1823, et le physicien George Gabriel Stokes, qui a donné sa forme définitive à l'équation de conservation de la quantité de mouvement en 1845. La forme de l'équation de Navier-Stokes dépend des caractéristiques du fluide :

$$\begin{tikzcd}
 & & \boxed{\text{Les caractéristiques du fluide}} \arrow[dll] \arrow[dl] \arrow[d] \arrow[dr] \arrow[drr] & & \\
\boxed{\text{Visqueux}} & \boxed{\text{Stationnaire}} & \boxed{\text{Incompressible}} & \boxed{\text{Instationnaire}} & \boxed{\text{Compressible}}
\end{tikzcd}$$

Ici, on s'intéressera uniquement aux équations stationnaires modélisant des fluides incompressibles. D'abord, on commencera par rappeler les équations de Navier-Stokes avec les conditions aux bords. Puis nous en déduirons une formulations, faible, et avec l'utilisation des techniques de discrétisation nous obtenons le problème de point selle. Les équations de Navier-Stokes s'écrivent sous la forme suivante

$$\begin{cases} -\mathbf{v}\nabla^2 u + u \cdot \nabla u + \nabla p &= f \text{ dans } \Omega \\ \nabla \cdot u &= 0 \text{ dans } \Omega, \end{cases} \tag{5}$$

u et p représentent respectivement le champ de vitesse et la pression du fluide, $\mathbf{v}$ la viscosité du fluide. La première équation modélise la conservation du moment du fluide. La deuxième équation modélise la conservation de la masse. On considère le problème posé sur un domaine Ω de dimension 2 ou 3 avec des conditions aux bords définies par

$$\begin{aligned} u &= w \text{ sur } \partial\Omega_D, \\ v\tfrac{\partial u}{\partial n} - \vec{n}p &= 0 \text{ sur } \partial\Omega_N, \end{aligned} \tag{6}$$

où $\vec{n}$ désigne le vecteur normal et $\partial\Omega = \partial\Omega_D \cup \Omega_N$.

Formulation faible

Nous commençons par définir quelques notations qui seront utilisées dans cette section

$$L_2(\Omega) := \left\{ u : \Omega \to \mathbb{R} \mid \int_\Omega u^2 < \infty \right\},$$

et l'espace de Sobolev par :

$$H^1(\Omega) := \left\{ u : \Omega \to \mathbb{R} \,|u, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y} \in L_2(\Omega) \right\}.$$

Nous définissons les espaces de solutions et tests :

$$H^1_E(\Omega) := \left\{ u \in H^1(\Omega)^2 \,|u = w \text{ sur } \partial\Omega_D \right\},$$

$$H^1_{E_0} := \left\{ v \in H^1(\Omega)^2 \,|v = 0 \text{ sur } \partial\Omega_D \right\}.$$

La formulation faible de (5) va être définie sous la forme suivante : trouver un $u \in H^1_E(\Omega)$ et $p \in L_2(\Omega)$ tels que :

$$\begin{aligned} \mathbf{v}\int_\Omega \nabla u : \nabla v + \int_\Omega (u \cdot \nabla u) v - \int_\Omega p(\nabla \cdot v) &= \int_\Omega f \cdot v \quad \forall v \in H^1_{E_0}, \\ \int_\Omega q(\nabla \cdot u) &= 0 \quad \forall q \in L_2(\Omega). \end{aligned} \tag{7}$$

Linéarisation

Nous posons d'abord, $u = u_k + \delta u_k$ et $p = p_k + \delta p_k$. La forme linéarisée de (3.4) va être définie sous la forme suivante :

trouver $\delta u_k \in H^1_{E_0}$ et $\delta p_k \in L_2(\Omega)$ qui satisfont :

$$\begin{array}{rcl} D(u_k, \delta u_k, v) + \mathbf{v}\int_\Omega \nabla \delta u_k : \nabla v - \int_\Omega \delta p_k(\nabla \cdot v) & = & R_k(v), \\ \int_\Omega q(\nabla \cdot \delta u_k) & = & r_k(q). \end{array} \tag{8}$$

où

$$\begin{array}{rcl} D(u_k, \delta u_k, v) & = & \int_\Omega (\delta u_k \cdot \nabla \delta u_k)\cdot v + \int_\Omega (\delta u_k \cdot \nabla u_k)\cdot v + \int_\Omega (u_k \cdot \nabla \delta u_k)\cdot v. \\ R_k(v) & = & \int_\Omega f \cdot v - \int_\Omega (u_k \cdot \nabla u_k) v - \mathbf{v}\int_\Omega \nabla u_k : \nabla v + \int_\Omega p_k(\nabla \cdot v). \\ r_k(q) & = & -\int_\Omega q(\nabla \cdot u_k). \end{array}$$

$\forall v \in H^1_{E_0}$ et $\forall q \in L_2(\Omega)$.

Formulation faible discrète

Nous utilisons des espaces de dimension finie. Soient $V^h \subset H^1_{E_0}$ et $Q^h \subset L_2(\Omega)$, la version discrète de (8) devient :

trouver $u_h \in V^h$ et $p \in Q^h$

$$\begin{array}{rcl} D(u_h, \delta u_h, v_h) + \mathbf{v}\int_\Omega \nabla \delta u_h \nabla v_h - \int_\Omega \delta p_h \cdot \nabla v_h & = & R_k(v_h), \\ \int_\Omega q_h \cdot \nabla \delta u_h & = & r_k(q_h), \end{array} \tag{9}$$

pour tout $v_h \in V^h$ et $q_h \in Q^h$.

Ensuite nous utilisons l'expression de la solution du problème (9) en fonction des bases de V^h et Q^h, pour obtenir le système d'équations suivant :

$$\begin{pmatrix} \mathbf{v}A + D_k & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} \Delta u^{(k)} \\ \Delta p^{(k)} \end{pmatrix} = \begin{pmatrix} \mathbf{R}_k \\ \mathbf{r}_k \end{pmatrix}, \tag{10}$$

où $\Delta u^{(k)} = u^{(k+1)} - u^{(k)}$, $\Delta p^{(k)} = p^{(k+1)} - p^{(k)}$, A est la matrice représentant le laplacien

$$A = [a_{i,j}], \;\; a_{i,j} = \int_\Omega \nabla \phi_i : \nabla \psi_j, \;\; i,j = 1,..,n_u.$$

B la matrice de divergence

$$B = [b_{k,j}], \;\; b_{k,j} = -\int_\Omega \psi_k \cdot \nabla \phi_j, \;\; j = 1,..,n_u, k = 1,..,n_p.$$

D la matrice des termes non linéaires

$$D = [d_{i,j}], \;\; d_{i,j} = \int_\Omega (u_h \cdot \nabla \phi_j) \cdot \phi_i + \int_\Omega (\phi_j \cdot \nabla u_h) \cdot \phi_i) + \int_\Omega (\phi_j \cdot \nabla \phi_j) \cdot \phi_i,$$

où $i = 1,..,n_u$, $j = 1,..,n_u$ et $k = 1,..,n_p$. Le second membre est défini par

$\mathbf{R} = [R_i]$, $R_i = \int_\Omega f \cdot \phi_i - \int_\Omega (u_h \cdot \nabla u_h) \cdot \phi_i - \mathbf{v} \int_\Omega \nabla u_h \nabla \phi_i + \int_\Omega p_h \cdot \nabla \phi_i$,

$\mathbf{r} = [r_i]$, $r_k = \int_\Omega \psi_k \cdot \nabla u_h$.

Maintenant nous présentons les différentes applications de l'équation de Navier-Stokes.

Fluide s'écoulant dans un tunnel contenant une expansion soudaine

La figure 5 montre l'écoulement du fluide et le rendu de la pression dans un conduit rectangulaire contenant une soudaine expansion. On pose les conditions de Dirichlet à l'entrée du conduit rectangulaire et les conditions de Neumann à la sortie du conduit. La vitesse verticale est égale à zéro, ce qui entraîne que les lignes de l'écoulement du fluide se propagent d'une manière parallèle aux parois du conduit rectangulaire. Le rendu de la pression devient plus proche de zéro à la sortie du conduit.

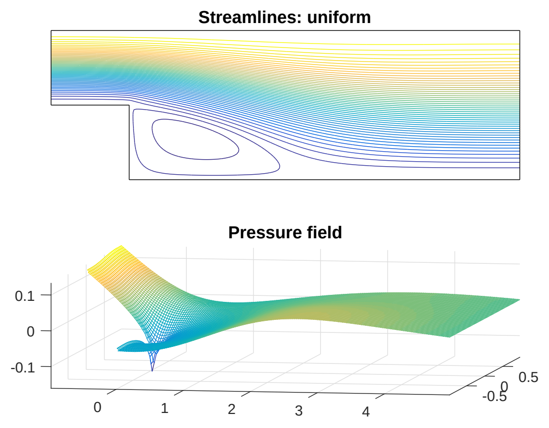


Figure 5 – Solution et rendu 3D de la pression

Ensuite nous choisissons le mode de discrétisation $Q_2 - Q_1$, (voir figure 6). Les points noirs correspondent aux nœuds utilisés pour discrétiser le vecteur vitesse et les points cerclés correspondent aux nœuds utilisés pour discrétiser le vecteur pression.

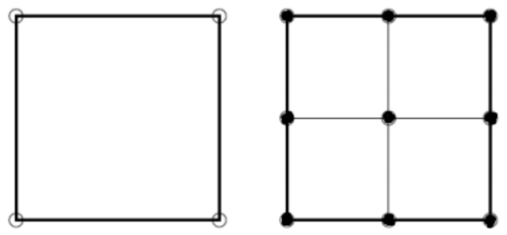


FIGURE 6 – Variation de la discrétisation

Fluide s'écoulant dans un tunnel contenant un obstacle

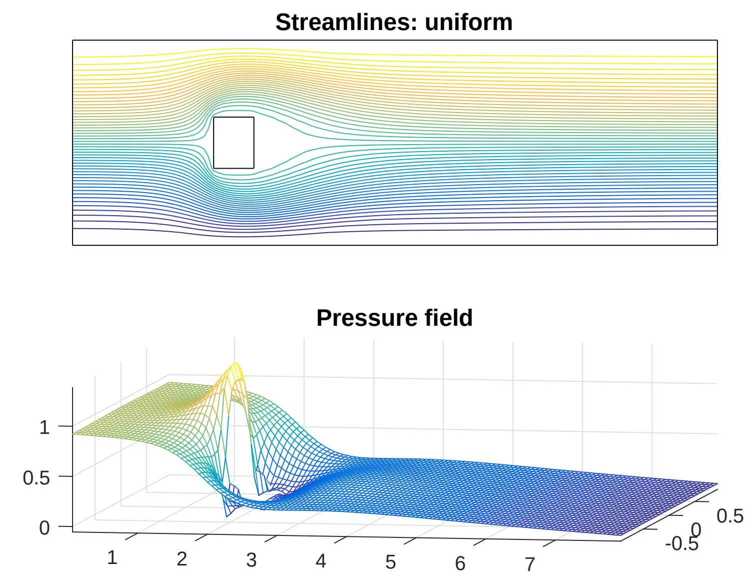


FIGURE 7 – Solution et rendu 3D de la pression

La figure 7 montre l'écoulement du fluide et le rendu de la pression dans un tuyau contenant un obstacle. Comme dans l'exemple précédent, on pose les conditions de Dirichlet à l'entrée du conduit rectangulaire et les conditions de Neumann à la sortie. Ensuite nous choisissons le mode de discrétisation $Q_1 - P_0$. Les points noirs correspondent aux nœuds utilisés pour discrétiser le vecteur vitesse et le point en jaune correspond au nœud utilisé pour discrétiser le vecteur pression.

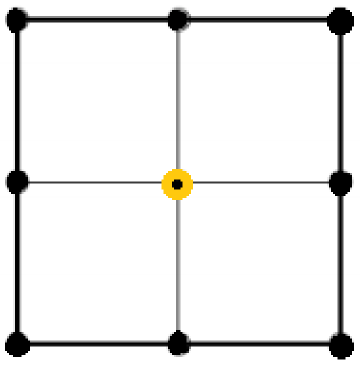


Figure 8 – Variation de la discrétisation

L'optimisation de la conception de la lentille

Figure 9 – Lentille

Nous pouvons trouver les problèmes de type point-selle dans les problèmes de l'optimisation de la lentille. Sachant que la lentille présente des aberrations (défauts) comme une aberration sphérique et une distorsion, ces aberrations ne permettent pas aux lentilles de corriger correctement la vision de quelqu'un. Donc notre problème principal c'est optimiser les aberrations pour mieux trouver la conception idéale que nous pouvons utiliser pour corriger les défauts de vision. La fonction qui décrira ce problème d'optimisation avec les différents types d'aberrations est appelée fonction de mérite. C'est une fonction qui mesure l'accord entre les données et le modèle approprié pour un choix particulier des paramètres. Par convention, la fonction de mérite est faible lorsque l'accord est bon, et la conception de la lentille sera basée sur les paramètres qui donnent la plus petite valeur de la fonction de mérite. La figure ci-dessous illustre l'aberration sphérique.

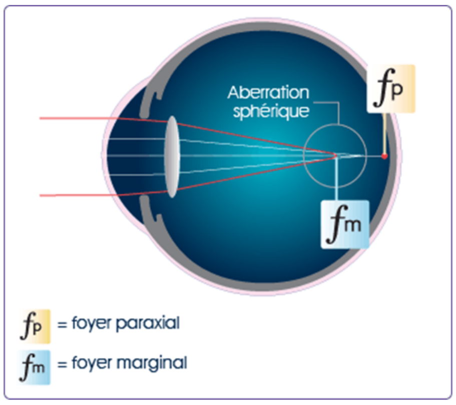


Figure 10 – Aberration sphérique

Nous observons que les rayons parallèles ne se focalisent pas sur un seul point. Voici un exemple de problème d'optimisation de la lentille avec une aberration sphérique

$$\begin{aligned} (A^TA + p^2I)x + B^T\lambda &= A^Tg, \\ Bx &= e. \end{aligned}$$

Où B est une matrice représentant un ensemble de dérivés b_{ij}, e est un vecteur représentant un ensemble de quantités e_j et λ est un vecteur représentant un ensemble de multiplicateurs. Nous transformons le problème ci-dessus en problème de point-selle

$$\begin{pmatrix} A^TA + p^2I & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = \begin{pmatrix} A^Tg \\ e \end{pmatrix}. \tag{11}$$

Point-selle avec plusieurs second membres

$$\begin{tikzcd}
 & \boxed{\text{Problème de point-selle avec plusieurs second membres}} \arrow[dl] \arrow[dr] & \\
\boxed{\text{Conception d'avion}} & & \boxed{\text{Équation de Stokes}}
\end{tikzcd}$$

Équation de Stokes

les problèmes de point-selle avec plusieurs second membres apparaissent lorsque nous appliquons l'approche du contrôle de retour de l'information sur l'équation de Stokes instationnaire suivante

$$\begin{cases} \dfrac{\partial u(t,x)}{\partial t} - \mathbf{v}\Delta u(t,x) + \nabla p(t,x) = & 0 \qquad (0,\infty)\times\Omega, \\ \nabla u(t,x) = & 0 \qquad (0,\infty)\times\Omega, \end{cases} \tag{12}$$

où $t \in (0,\infty)$, $x \in \Omega$, $\mathbf{v} \in \mathbb{R}^+$ la viscosité , u et p sont la velocité et la pression. $\Omega \subset \mathbb{R}^2$ est le domaine de l'espace avec le bord $\partial\Omega$; certaines conditions aux limites de Dirichlet décrivant un écoulement du fluide entrant avec les conditions initiales adéquates, seront trouvées dans [11].

Conception d'avion

Pour créer des nouvelles conceptions aérodynamiques efficaces, nous pouvons utiliser les équations de Navier-Stokes linéarisées. Dans certains cas, plusieurs second membres doivent être résolus en gardant la même matrice de point-selle. Nous pouvons décrire ce problème sous la forme suivante :

$$\underbrace{\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}}_{\mathcal{A}} \underbrace{\begin{pmatrix} X \\ Y \end{pmatrix}}_{\mathcal{X}} = \underbrace{\begin{pmatrix} F \\ G \end{pmatrix}}_{\mathcal{B}}, \tag{13}$$

où $A \in \mathbb{R}^{n\times n}$ est une matrice symétrique définie positive et $B^T \in \mathbb{R}^{n\times m}$ de rang maximal, $X \in \mathbb{R}^{n\times s}$, $Y \in \mathbb{R}^{m\times s}$, $F \in \mathbb{R}^{n\times s}$ et $G \in \mathbb{R}^{m\times s}$.

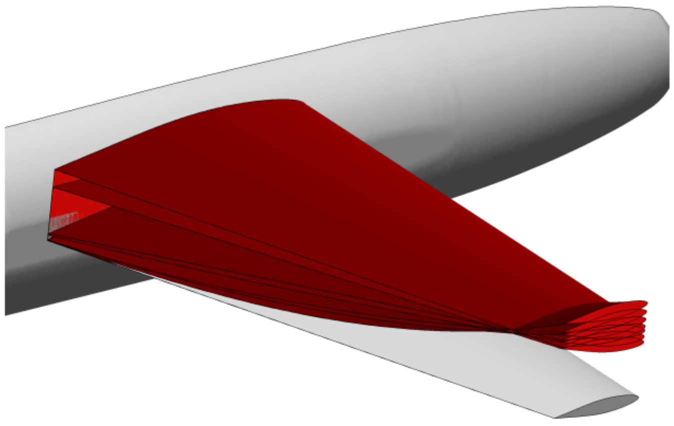


Figure 11 – Les mouvement d'ailes d'avion

Pourquoi (13) est il un problème de point-selle ?

Avec l'utilisation du produit de Frobenius $\langle .,. \rangle_F$, (13) apparaître comme un problème d'optimisation sous contrainte

$$\begin{cases} \min_X J(X) = \min_X \frac{1}{2}\langle AX, X\rangle_F - \langle X, F\rangle_F, \\ \qquad BX = G. \end{cases} \tag{14}$$

Ensuite nous définissons le lagrangien $\mathcal{L}$, pour montrer la liaison entre le problème (13) et le point-selle.

$$\mathcal{L}(X,Y) \;=\; \frac{1}{2}\langle AX, X\rangle_F - \langle X, F\rangle_F + \langle Y, BX - G\rangle_F. \tag{15}$$

La variable Y représente le vecteur des multiplicateurs de Lagrange. La solution (X^*, Y^*) de (13) est un point-selle du lagrangien (15) :

$$\mathcal{L}(X_*, Y) \leq \mathcal{L}(X_*, Y_*) \leq \mathcal{L}(X, Y_*), \text{ for any } X \in \mathbb{R}^{n\times s} \text{ and } Y \in \mathbb{R}^{m\times s}, \tag{16}$$

c'est pour cela qu'on appelle (13) un problème de point-selle.

Deuxième partie

Preconditioned global Krylov subspace method for solving saddle point problem with multiple right-hand sides

Chapitre 1

Preconditioned global Krylov subspace methods

Sommaire

Abstract :

In the present paper, we propose a preconditioned global approach as a new strategy to solve multiple linear systems with several right-hand sides coming from saddle point problems. The preconditioner is obtained by replacing some $(2,2)$ block in the original saddle-point matrix $\mathcal{A}$ by another well chosen block. We apply the global GMRES method to solve this new problem with several right hand sides, and give some convergence results for the global GMRES method. Moreover we analyze the eigenvalue-distribution and the eigenvectors of the proposed preconditioner when the first block is positive definite. We also compare our proposed preconditioned global GMRES algorithm with preconditioned block GMRES algorithm. In this paper, preconditioned global MINRES and preconditioned global FGMRES methods are also introduced for solving linear systems with multiple right-hand sides. Numerical results show that our preconditioned global GMRES method, has a high performance as compared to other preconditioned global subspace Krylov and preconditioned block Krylov subspace methods for solving the saddle point problem with several right hand sides.

Résumé :

Dans ce chapitre, nous proposons une approche globale préconditionnée [31], comme nouvelle stratégie pour la résolution de problèmes de point selle avec plusieurs second membres. Le préconditionneur est obtenu en remplaçant le bloc $(2,2)$ dans la matrice de point selle $\mathcal{A}$ par un autre bloc bien choisi. Nous avons appliqué les méthodes globale GMRES et MINRES préconditionées pour résoudre ce problème avec plusieurs second membres et également nous avons donné des résultats de convergence. Enfin, les résultats numériques montrent l'efficacité de la méthode globale GMRES préconditionnée pour la résolution de problèmes de point selle avec plusieurs second membres.

1.1 Introduction

Many problems in Science and Engineering require the solution of saddle point problems with multiple right-hand sides, (see, for example [11, 15]),

$$\underbrace{\begin{pmatrix} A & B^T \\ \epsilon B & O \end{pmatrix}}_{\mathcal{A}}\underbrace{\begin{pmatrix} X \\ Y \end{pmatrix}}_{\mathcal{X}} = \underbrace{\begin{pmatrix} F \\ \epsilon G \end{pmatrix}}_{\mathcal{B}}. \tag{1.1}$$

Here $A \in \mathbb{R}^{n\times n}$ is a symmetric matrix and $B^T \in \mathbb{R}^{n\times m}$ has full column rank, with $X \in \mathbb{R}^{n\times s}$, $Y \in \mathbb{R}^{m\times s}$, $F \in \mathbb{R}^{n\times s}$, $G \in \mathbb{R}^{m\times s}$ and $\epsilon = \pm 1$. Here B^T denotes the transpose of B. Since the matrices A and B in (1.1) are large and sparse, the solution of (1.1) with $s = 1$ is suited for solution by an iterative method. Many of effective iterative methods have been developed for solving the saddle point problem, such as, the Hermitian and skew-Hermitian splitting (HSS) method [6] and the Regularized HSS iteration method [2]. Instead of applying a standard iterative method to the solution each one of the saddle point problems

$$\mathcal{A}\mathcal{X}^{(i)} = b^{(i)}, \quad \text{for} \quad i = 1, \ldots, s, \tag{1.2}$$

independently, it is often more efficient to apply a global method to (1.1). We use the global version of GMRES, for the solution of nonsymmetric saddle point problems with multiple right-hand sides (1.1). The global version of GMRES has been introduced in [31] and studied in [10]. This method is based on global version of the standard Arnoldi, see for example [1]. However Krylov subspace methods without a good preconditioner converge very slowly when applied to the saddle point problems with multiple right-hand sides. In order to accelerate the convergence of the associated Krylov subspace method, several preconditioners and iterative methods are proposed for (1.1) with a single right-hand [2, 10]. The paper is organized as follows. An example of modeling that leads to this type of system is outlined in Section 2. In Section 3 we give the first-order optimality condition for a Block quadratic problem. In Section 4, we review some properties and definitions of global Krylov subspace methods. In Section 5 we introduce the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$, and we also analyze the eigenvalue-distribution and the eigenvectors of the proposed precondtioner. Numerical experiments are discussed in Section 6. We use the following notation. For two matrices Y and Z in $\mathbb{R}^{(n+m)\times s}$, we define the inner product $\langle Y, Z\rangle_F = \text{trace}(Y^T Z)$. The associated norm is the Frobenius norm denoted by $\|.\|_F$. A system of vectors (matrices) of $\mathbb{R}^{(n+m)\times s}$ is said to be F-orthonormal if it is orthonormal with respect to the scalar product $\langle ., .\rangle_F$. Finally, the Kronecker product of the matrices C and D is given by $C \otimes D = [c_{i,j} D]$.

1.2 The classification of the solution of the saddle point problem (1.1).

We assume that A is a positive definite matrix. With the use of the Frobenius inner product $\langle .,. \rangle_F$, equation (1.1) arises as a quadratic constrained optimization problem

$$\begin{cases} \min_X J(X) = \min_X \frac{1}{2}\langle AX, X\rangle_F - \langle X, F\rangle_F, \\ \qquad BX = G. \end{cases} \tag{1.3}$$

We define the Lagrangian $\mathcal{L}$ as

$$\mathcal{L}(X,Y) \;=\; \frac{1}{2}\langle AX, X\rangle_F - \langle X, F\rangle_F + \langle Y, BX - G\rangle_F. \tag{1.4}$$

Theorem 1.1 *The solution of* (1.1) *is a saddle point of the Lagrangian* (1.4).

Proof :
To show that the solution of (1.1) is a saddle point of (1.4), we start discussing the differentiability of $\mathcal{L}$, and we calculate its differential. As we now the following function

$$\begin{array}{ll} \mathbb{R}^{n\times s} \times \mathbb{R}^{n\times s} & \to \mathbb{R} \\ (X,Y) & \to \mathrm{Trace}(Y^T X). \end{array}$$

is differentiable, then the functional $\mathcal{L}$ is differentiable. Next we calculate the differential of the functional $\mathcal{L}$. Let $H \in \mathbb{R}^{n\times s}$.

$$\begin{aligned} \mathcal{L}(X+H,Y) &= \frac{1}{2}\langle A(X+H), X+H\rangle_F - \langle X+H, F\rangle_F + \langle Y, B(X+H) - G\rangle_F, \\ &= \frac{1}{2}\langle AX, X\rangle_F + \langle AH, X\rangle_F + \frac{1}{2}\langle AH, H\rangle_F - \langle X, F\rangle_F \\ &\quad -\langle H, F\rangle_F + \langle Y, BX - G\rangle_F + \langle Y, BH\rangle_F, \\ &= \frac{1}{2}\langle AX, X\rangle_F - \langle X, F\rangle_F + \langle Y, BX - G\rangle_F + \langle AH, X\rangle_F \\ &\quad + \frac{1}{2}\langle AH, H\rangle_F - \langle H, F\rangle_F + \langle Y, BH\rangle_F, \\ &= \mathcal{L}(X,Y) + \langle AH, X\rangle_F + \frac{1}{2}\langle AH, H\rangle_F - \langle H, F\rangle_F + \langle Y, BH\rangle_F. \end{aligned} \tag{1.5}$$

Which can be written as

$$\begin{aligned}\mathcal{L}(X+H,Y)-\mathcal{L}(X,Y)-\frac{1}{2}\langle AH,H\rangle_F &= \langle AX,H\rangle_F-\langle H,F\rangle_F+\langle Y,BH\rangle_F,\\ &= \langle AX-F+B^TY,H\rangle_F.\end{aligned}$$

Therefore

$$d_X\mathcal{L}(X,Y)=AX-F+B^TY. \tag{1.6}$$

Similarly we obtain

$$d_Y\mathcal{L}(X,Y)=BX-G. \tag{1.7}$$

According to (1.6) and (1.7), we conclude that the solution (X_*,Y_*) of (1.1) is a critical point of the functional $\mathcal{L}$.
To complete the proof, it must be shown that (X_*,Y_*) satisfies

$$\mathcal{L}(X_*,Y)\le\mathcal{L}(X_*,Y_*)\le\mathcal{L}(X,Y_*),\ \text{ for any } X\in\mathbb{R}^{n\times s} \text{ and } Y\in\mathbb{R}^{m\times s}. \tag{1.8}$$

$$\begin{aligned}\mathcal{L}(X,Y) &= \frac{1}{2}\text{Trace}\left(\begin{pmatrix}X\\Y\end{pmatrix}^T\begin{pmatrix}A & B^T\\ \epsilon B & O\end{pmatrix}\begin{pmatrix}X\\Y\end{pmatrix}\right)-\text{Trace}\left(\begin{pmatrix}X\\Y\end{pmatrix}^T\begin{pmatrix}F\\ \epsilon G\end{pmatrix}\right),\\ &= \frac{1}{2}\text{Trace}\left(\begin{pmatrix}X\\Y\end{pmatrix}^T\begin{pmatrix}A & B^T\\ \epsilon B & O\end{pmatrix}\begin{pmatrix}X\\Y\end{pmatrix}\right)-\text{Trace}\left(\begin{pmatrix}X\\Y\end{pmatrix}^T\begin{pmatrix}A & B^T\\ \epsilon B & O\end{pmatrix}\begin{pmatrix}X_*\\Y_*\end{pmatrix}\right).\end{aligned}$$

Then

$$\mathcal{L}(X_*,Y_*) = -\frac{1}{2}\text{Trace}\left(\begin{pmatrix}X_*\\Y_*\end{pmatrix}^T\begin{pmatrix}A & B^T\\ \epsilon B & O\end{pmatrix}\begin{pmatrix}X_*\\Y_*\end{pmatrix}\right).$$

Consequently

$$\mathcal{L}(X,Y)-\mathcal{L}(X_*,Y_*) = \frac{1}{2}\text{Trace}\left(\begin{pmatrix}X-X_*\\Y-Y_*\end{pmatrix}^T\begin{pmatrix}A & B^T\\ \epsilon B & O\end{pmatrix}\begin{pmatrix}X-X_*\\Y-Y_*\end{pmatrix}\right).$$

Case 1 : $X=X_*$

$$\mathcal{L}(X_*,Y)-\mathcal{L}(X_*,Y_*) = 0. \tag{1.9}$$

Case 2 : $Y = Y_*$

$$\mathcal{L}(X, Y_*) - \mathcal{L}(X_*, Y_*) \quad = \quad \mathrm{Trace}((X - X_*)^T A(X - X_*)) > 0. \tag{1.10}$$

We conclude that (X_*, Y_*) is a saddle point of the Lagrangian $\mathcal{L}$. □

In the following section we recall some properties and definitions of the global methods [16, 33].

1.3 Global Krylov subspace methods

Definition 1.1 *The global Krylov subspace $\mathcal{K}_k(\mathcal{A}, V)$ is the subspace spanned by the matrices V, $\mathcal{A}V$, ..., $\mathcal{A}^{k-1}V$.*

Remark 1.3.1 *Let V be a real matrix of dimension $(n+m)\times s$. According to the definition of the subspace $\mathcal{K}_k(\mathcal{A}, V)$, we have $\mathcal{Z} \in \mathcal{K}_k(\mathcal{A}, V) \iff \mathcal{Z} = \sum_{i=1}^{k} \alpha_i \mathcal{A}^{i-1} V, \alpha_i \in \mathbb{R}, i = 1, .., k$.*
In other words, $\mathcal{K}_k(\mathcal{A}, V)$ is the subspace of $\mathbb{R}^{(n+m)\times s}$ which contains all the matrices of dimension $(n+m)\times s$, written as $\mathcal{Z} = P(\mathcal{A})V$, where P is a polynomial of degree at most equal $k-1$.

Definition 1.2 *Diamond Product*
Let $Y = [Y_1, Y_2, \ldots, Y_p]$ and $Z = [Z_1, Z_2, \ldots, Z_l]$ be matrices of dimension $n \times ps$ and $n \times ls$
respectively where Y_i and Z_j $(i = 1, \ldots, p; j = 1, \ldots, l)$ are $n \times s$ matrices. The product $\diamond$ is defined by

$$Y^T \diamond Z = \begin{pmatrix} \langle Y_1, Z_1\rangle_F & \langle Y_1, Z_2\rangle_F & \ldots & \langle Y_1, Z_l\rangle_F \\ \langle Y_2, Z_1\rangle_F & \langle Y_2, Z_2\rangle_F & \ldots & \langle Y_2, Z_l\rangle_F \\ \vdots & \vdots & \ddots & \vdots \\ \langle Y_p, Z_1\rangle_F & \langle Y_p, Z_2\rangle_F & \ldots & \langle Y_p, Z_l\rangle_F \end{pmatrix} \subset \mathbb{R}^{p\times l}.$$

We use the global Arnoldi process for building an F-orthonormal basis of $\mathcal{K}_k(\mathcal{A}, V)$. Let X_0 be the initial approximation of the solution of to (1.1) and $R_0 = \mathcal{B} - \mathcal{A}X_0$ be the corresponding residual. The following global Arnoldi process based on Gram-Schmidt constructs an F-orthonormal basis V_1, V_2, . . ., V_k of $\mathcal{K}_k(\mathcal{A}, R_0)$, in other words $\mathrm{Trace}(V_i^T V_j) = \delta_{ij}$ for $i, j = 1, ..., k$, where δ_{ij} is the Kronecker symbol with $\delta_{ij} \neq 0$ for $i \neq j$ and $\delta_{ii} = 1$.

Algorithme 1 : Global Arnoldi process

```
1 : $V_1 = R_0/\|R_0\|_F$ ;
2 : for $j = 1, 2, ..., k$ do;
3 : $W := \mathcal{A}V_j$ ;
4 : for $i = 1, 2, ..., j$ do;
5 : $H_{i,j} =< W, V_i >_F$ ;
6 : $W = W - H_{ij}V_i$;
7 : end;
8 : $H_{j+1,j} = \|W\|_F$;
9 : $V_{j+1} = W/H_{j+1,j}$;
10 : end;
```

Proposition 1

Assume that $h_{i+1,i} \neq 0$ for $i = 1,..,k$ and let $\mathcal{V}_k$ be the matrix defined by $\mathcal{V}_k = \{V_1,..,V_k\}$, where V_i for $i = 1,..,k$ are the matrices computed by Algorithm 1. We have the following relations :

$$\begin{aligned} \mathcal{A}\mathcal{V}_k &= \mathcal{V}_k \diamond H_k + H_{k+1,k}[O,..,O,V_{k+1}], && (1.11) \\ \mathcal{A}\mathcal{V}_k &= \tilde{\mathcal{V}}_k \diamond \tilde{H}_k, && (1.12) \end{aligned}$$

where $\tilde{\mathcal{V}}_k = \{V_1,..,V_k,V_{k+1}\} \in \mathbb{R}^{(n+m)\times(k+1)s}$ and $\tilde{H}_k \in \mathbb{R}^{(k+1)\times k}$. The global GMRES method constructs, at step k, the approximation X_k satisfying the following two relations

$$X_k - X_0 \in \mathcal{K}_k(\mathcal{A}, R_0) \text{ and } \langle \mathcal{A}^j R_0, R_k \rangle_F = 0, \quad \text{for } j = 1, \ldots, k.$$

The residual $R_k = \mathcal{B} - \mathcal{A}X_k$ satisfies the minimization property

$$\|R_k\|_F = \min_{Z \in \mathcal{K}_k(\mathcal{A}, R_0)} \|R_0 - \mathcal{A}Z\|_F. \tag{1.13}$$

The problem (1.13) is solved by applying the global Arnoldi process.

1.3.1 Convergence analysis of the global GMRES method

In this subsection, we recall some convergence results for the global GMRES method. Let $\mathcal{A} = \mathcal{Z}\mathcal{D}\mathcal{Z}^{-1}$, where $\mathcal{D}$ is the diagonal matrix whose elements are the eigenvalues $\lambda_1, \ldots, \lambda_{n+m}$, and $\mathcal{Z}$ is the eigenvector matrix.

Theorem 1.2 *[10] Let the initial residual R_0 be decomposed as $R_0 = \mathcal{Z}\beta$ where β is an $(n+m)\times s$ matrix whose columns are denoted by $\beta^{(1)},\dots,\beta^{(s)}$. Let $R_k = \mathcal{B} - \mathcal{A}\mathcal{X}_k$ be the kth residual obtained by the global GMRES method when applied to* (1.1). *Then we have*

$$\|R_k\|_F^2 \leq \frac{\|\mathcal{Z}\|_2^2}{e_1^T (V_{k+1}^T \widetilde{\mathcal{D}} V_{k+1})^{-1} e_1}, \tag{1.14}$$

where

$$\widetilde{\mathcal{D}} = \begin{pmatrix} \sum_{i=1}^{s} \mid {\beta_1}^{(i)} \mid^2 & & \\ & \ddots & \\ & & \sum_{i=1}^{s} \mid {\beta_{n+m}}^{(i)} \mid^2 \end{pmatrix} \quad \text{and} \quad V_{k+1} = \begin{pmatrix} 1 & \lambda_1 & \dots & \lambda_1^k \\ \vdots & \vdots & & \vdots \\ 1 & \lambda_{n+m} & \dots & \lambda_{n+m}^k \end{pmatrix}. \tag{1.15}$$

The coefficients $\beta_1^{(i)},\dots,\beta_{n+m}^{(i)}$ are the components of the vector $\beta^{(i)}$ and e_1 is the first unit vector of $\mathbb{R}^{k+1}$.

1.4 Preconditioning

In this section we investigate a regularization preconditioner for solving saddle point problems with multiple right-hand sides (1.1), where $A \in \mathbb{R}^{n\times n}$ is a symmetric and invertible matrix, $B \in \mathbb{R}^{m\times n}$ has a full row rank. A assumed to be large and sparse. The idea of preconditioning is to transform the linear system (1.1) into another one, easier to solve. Left preconditioning (1.1) gives the following new linear system

$$\mathcal{M}^{-1}\mathcal{A}\mathcal{X} = \mathcal{M}^{-1}\mathcal{B}. \tag{1.16}$$

Although right preconditioning can be used in our context, we will only focus on left preconditioning thoughout this work. By exploiting the particulatr block structure of (1.1), severals authors investigate several block preconditionners [13]. We introduce and analyze the matrix $\mathcal{P}_{\epsilon,\alpha,Q}$, a block preconditioner for (1.1).

$$\mathcal{P}_{\epsilon,\alpha,Q} = \begin{pmatrix} A & B^T \\ \epsilon B & \alpha Q \end{pmatrix}, \tag{1.17}$$

with $\alpha > 0$ and Q is a symmetric positive definite matrix (Q can be for example, for the discrete Stokes system, an approximation of the pressure Schur complement $S = BA^{-1}B^T$ (see [26])).

1.4.1 Properties of the Preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$

Now we propose a block factorizations and some properties of the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$.

Proposition 2

The preconditioner (1.17) has the block-triangular factorization

$$\mathcal{P}_{\epsilon,\alpha,Q} = \begin{pmatrix} A & B^T \\ \epsilon B & \alpha Q \end{pmatrix} = \begin{pmatrix} I & O \\ \epsilon BA^{-1} & I \end{pmatrix}\begin{pmatrix} A & O \\ O & \tilde{S} \end{pmatrix}\begin{pmatrix} I & A^{-1}B^T \\ O & I \end{pmatrix}, \quad (1.18)$$

where $\tilde{S} = \left(\alpha Q - \epsilon BA^{-1}B^T\right)$.
If $\epsilon = -1$, or $\epsilon = 1$ and $\alpha\lambda_{\min}(Q) > \lambda_{\max}(BA^{-1}B^T)$, then the block $\tilde{S}$ is a positive definite matrix.

The inverse of the preconditioner matrix $\mathcal{P}_{\epsilon,\alpha,Q}$ is given by

$$\mathcal{P}_{\epsilon,\alpha,Q}^{-1} = \begin{pmatrix} A & B^T \\ \epsilon B & \alpha Q \end{pmatrix}^{-1} = \begin{pmatrix} I & -A^{-1}B^T \\ O & I \end{pmatrix}\begin{pmatrix} A^{-1} & O \\ O & \tilde{S}^{-1} \end{pmatrix}\begin{pmatrix} I & O \\ BA^{-1} & I \end{pmatrix}. \quad (1.19)$$

The eigenvalue and eigenvector distributions of the preconditioned matrix relate closely to the convergence rate of Krylov subspace methods. Therefore, it is meaningful to investigate the spectral properties of the preconditioned matrix $\mathcal{P}_{\epsilon,\alpha,Q}^{-1}\mathcal{A}$. In the following theorem, we will deduce the eigenvalue distribution of the preconditioned matrix $\mathcal{P}_{\epsilon,\alpha,Q}^{-1}\mathcal{A}$.

Theorem 1.3 *Let the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$ be defined as in* (1.17)*, A and Q are positive definite matrices. Then the matrix $\mathcal{P}_{\epsilon,\alpha,Q}^{-1}\mathcal{A}$ has $m_1 + 1$ distincts eigenvalues $\{1, \lambda_1, \ldots, \lambda_{m_1}\}$ and $(n+m)$ linearly independent eigenvectors, where $1 \le m_1 \le m$. Moreover*

1. *1 is an eigenvalue with multiplicity n.*
2. *$\lambda_i \neq 1$, for $i = 1, .., m_1$.*
3. *The matrix $\mathcal{P}_{\epsilon,\alpha,Q}^{-1}\mathcal{A}$ is diagonalizable.*

Proof :
Let λ be an eigenvalue of the preconditioned matrix $\mathcal{P}_{\epsilon,\alpha,Q}^{-1}\mathcal{A}$ and $(x^T, y^T)^T$ be the corresponding eigenvector. To derive the eigenvector distribution, we consider the following generalized eigenvalue problem :

$$\mathcal{P}_{\epsilon,\alpha,Q}^{-1} \quad \mathcal{A}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} I_n & K_1 \\ 0 & K_2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \lambda\begin{pmatrix} x \\ y \end{pmatrix}. \tag{1.20}$$

Where $\tilde{S} = \left(\alpha Q - \epsilon BA^{-1}B^T\right)$, $K_1 = \left(I + \epsilon A^{-1}B^T\tilde{S}^{-1}B\right)A^{-1}B^T$ and $K_2 = -\epsilon\tilde{S}^{-1}BA^{-1}B^T$. Equation (2.30) can be equivalently rewritten as

$$\begin{cases} (1-\lambda)x = -\left(I + \epsilon A^{-1}B^T\tilde{S}^{-1}B\right)A^{-1}B^T y, \\ BA^{-1}B^T y = \dfrac{\alpha\lambda}{\epsilon(\lambda-1)}Qy. \end{cases} \tag{1.21}$$

If $\lambda = 1$ holds true, then from the second equation of (1.21), we can easily get $y = 0$. Thus there are n linearly independent eigenvectors $\begin{pmatrix} u^{(i)} \\ 0 \end{pmatrix}$, $(i = 1,..,n)$ corresponding to the eigenvalue 1, where $u^{(i)}$ are arbitrary linearly independent vectors.
If $\lambda \neq 1$ and $y = 0$, then we obtain from the first equation of (1.21) that $x = 0$. This contradicts the assumption that $(x^T, y^T)^T$ is an eigenvector of the preconditioned matrix $\mathcal{P}_{\epsilon,\alpha,Q}^{-1}\mathcal{A}$ and then $y \neq 0$. If y satisfies the second equation of (1.21), then

$$BA^{-1}B^T y = \mu Q y, \tag{1.22}$$

where

$$\mu = \frac{\epsilon\alpha\lambda}{(\lambda-1)}.$$

We deduce that

$$\lambda = \frac{\mu}{\mu - \alpha\epsilon}.$$

Hence when $\epsilon\alpha < 0$ then $\lambda < 1$.
We know that Q is a positive definite matrix, then Q admits a Cholesky decomposition $Q = RR^T$. Premultiplying (1.22) with R^{-1} results in

$$R^{-1}BA^{-1}B^TR^{-T}z = \mu z \text{ with } z = R^T y \ . \tag{1.23}$$

Since $R^{-1}BA^{-1}B^TR^{-T}$ is a positive definite matrix, then there are m linearly independent eigenvectors, of the form $w^{(i)} = R^{-T}z^{(i)}$, $(i = 1,..,m)$, where $z^{(i)}$, are orthogonal eigenvectors of $R^{-1}BA^{-1}B^TR^{-T}$.

U, V and W are matrices whose columns are $\left(u^{(1)},..,u^{(n)}\right)$, $\left(v^{(1)},..,v^{(m)}\right)$ and $\left(w^{(1)},..,w^{(m)}\right)$, respectively, where $v^{(i)}$ $(i = 1,..,m)$ are given by the following relation

$$v^{(i)} = \frac{\left(I + \epsilon A^{-1}B^T\tilde{S}^{-1}B\right)A^{-1}B^T}{(\lambda - 1)} w^{(i)}. \tag{1.24}$$

Since U and W are nonsingular matrices, we infer that $\det\begin{pmatrix} U & V \\ O & W \end{pmatrix} = \det(U)\det(W) \neq 0$ and $\mathcal{P}^{-1}_{\epsilon,\alpha,Q}\mathcal{A}$ is diagonalizable. □

Remark 1.4.1 *If A is positive definite matrix, $\epsilon = -1$ and μ satisfy the generalized eigenvalue problem* (1.22) *then*

$$\mu = \frac{\lambda\alpha}{(1-\lambda)} \quad \textit{and} \qquad \lambda = \frac{\mu}{\alpha + \mu} \leq 1.$$

Hence when $\alpha \to 0$, $\lambda \to 1$.

For the discrete Stokes system, if γ is the (generalized) inf-sup constant and Γ is a boundedness constant, we have $\gamma^2 \leq \mu \leq \Gamma^2$ (see for example [26]),

$$\lambda \in \left[\frac{\gamma^2}{\gamma^2 + \alpha}, \frac{\Gamma^2}{\Gamma^2 + \alpha}\right].$$

1.4.2 Preconditioned Global Subspaces Krylov Solvers

We consider now the preconditionned global Krylov subspace methods for solving

$$\mathcal{P}^{-1}_{\epsilon,\alpha,Q} \quad \mathcal{A}\mathcal{X} = \mathcal{P}^{-1}_{\epsilon,\alpha,Q}\mathcal{B}.$$

Each global Krylov iteration requires the solution of a linear system of the form

$$\underbrace{\begin{pmatrix} A & B^T \\ \epsilon B & \alpha Q \end{pmatrix}}_{\mathcal{P}_{\epsilon,\alpha,Q}} \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}, \tag{1.25}$$

with $V_1 \in \mathbb{R}^{n\times s}$, $V_2 \in \mathbb{R}^{m\times s}$, $Z_1 \in \mathbb{R}^{n\times s}$ and $Z_2 \in \mathbb{R}^{m\times s}$.

This system can be solved in two stages as :

$$\begin{cases} \underbrace{\left(A - \frac{\epsilon}{\alpha} B^T Q^{-1} B\right)}_{A_\alpha} Z_1 = \underbrace{V_1 - \frac{1}{\alpha} B^T Q^{-1} V_2}_{J}, \\ Z_2 = \frac{1}{\alpha} Q^{-1}(V_2 - \epsilon B Z_1). \end{cases} \tag{1.26}$$

Hence the system $A_\alpha Z_1 = J$ is solvable if the symmetrix matrix A_α is invertible. If $\epsilon = -1$, the matrix A_α is symmetric positive definite. So we have to solve the preconditioned system

$$\mathcal{M}^{-1} A_\alpha = \mathcal{M}^{-1} J. \tag{1.27}$$

Where $\mathcal{M}$ is a given preconditioner.

Algorithme 2 : The preconditioned Global CG

```
1 : Choose $X_0$, compute $R_0 = J - A_\alpha X_0$ and solve $\mathcal{M} Z_0 = R_0$ and set $P_0 = Z_0$ ;
2 : for $k = 0, 1, ..do$ ;
3 : $\alpha_k = \frac{\langle Z_k, R_k \rangle_F}{\langle A_\alpha P_k, P_k \rangle_F}$ ;
4 : $X_{k+1} = X_k + \alpha_k P_k$ ;
5 : $R_{k+1} = R_k - \alpha_k A_\alpha P_k$ ;
6 : Solve $\mathcal{M} Z_{k+1} = R_{k+1}$ ;
7 : $\beta_k = \frac{\langle Z_{k+1}, R_{k+1} \rangle_F}{\langle Z_k, R_k \rangle_F}$ ;
8 : $P_{k+1} = Z_{k+1} + \beta_k P_k$ ;
```

Depending on the value of ϵ, the matrix of the linear system or the preconditioner can be symmetric or nonsymmetric. When the matrix is symmetric positive and definite, we will apply the preconditioned Global CG [32]. If the matrix is symmetric but indefinite, we will use the Global MINRES. When the matrix of the system is not symmetric, several Global subspace Krylov methods can be used. The most useful is the Global GMRES, , computes iterates which minimise the Frobenius norm of the residual. We will define here a 'flexible' version of Global GMRES, which allows for a preconditioner that varies during iterations. The standard Flexible GMRES was defined by Saad [43]. Since the amount of computation and storage is increasing in each iteration for Global GMRES and Global FGMRES, we will also consider the Global BiCGSTAB introduced in [26], since it requires only a fixed amount of work at each iteration.

Algorithme 3 : The preconditioned Global MINRES

```
1 : $V_0 = O,\ W_0 = O,\ W_1 = O,\ \gamma_0 = 0$ ;
2 : Choose $X_0$, compute $V_1 = J - A_\alpha X_0$, and solve $\mathcal{M}Z_1 = V_1$ ;
3 : Set $\gamma_1 = \sqrt{\langle Z_1, V_1\rangle_F}$, $v = \gamma_1$, $s_0 = s_1 = 0$, $c_0 = c_1 = 1$ ;
4 : for j=1 until convergence do ;
5 : $Z_j = Z_j/\gamma_j$ ;
6 : $\delta_j = \langle A_\alpha Z_j, Z_j\rangle_F$ ;
7 : $V_{j+1} = A_\alpha Z_j - (\delta_j/\gamma_j)V_j - (\delta_j/\gamma_{j-1})V_{j-1}$ ;
8 : Solve $\mathcal{M}Z_{j+1} = V_{j+1}$ ;
9 : $\gamma_{j+1} = \sqrt{\langle Z_{j+1}, V_{j+1}\rangle_F}$ ;
10 : $\alpha_0 = c_j\delta_j - c_{j-1}s_j\gamma_j$ ;
11 : $\alpha_1 = \sqrt{\alpha_0^2 + \gamma_{j+1}^2}$ ;
12 : $\alpha_2 = s_j\delta_j + c_{j-1}c_j\gamma_j$ ;
13 : $\alpha_3 = s_{j-1}\gamma_j$ ;
14 : $c_{j+1} = \alpha_0/\alpha_1$ ;
15 : $s_{j+1} = \gamma_{j+1}/\alpha_1$ ;
16 : $W_{j+1} = \big(Z_j - \alpha_3 W_{j-1} - \alpha_2 W_j\big)/\alpha_1$ ;
17 : $U_j = U_{j-1} + c_{j+1} v W_{j+1}$ ;
18 : $v = -s_{j+1}v$ ;
19 : end do;
```

Algorithme 4 : The preconditioned Global GMRES

```
1 : $\mathcal{P}_{\epsilon,\alpha,Q}V_1 = R_0,\ V_1 = V_1/\|V_1\|_F$
2 : for $j = 1, 2, ..., k$ do;
3 : $\mathcal{P}_{\epsilon,\alpha,Q}W := \mathcal{A}V_j$ ;
4 : for $i = 1, 2, ..., j$ do;
5 : $H_{i,j} =< W, V_i >_F$ ;
6 : $W = W - H_{ij}V_i$;
7 : end;
8 : $H_{j+1,j} = \|W\|_F$;
9 : $V_{j+1} = W/H_{j+1,j}$;
10 : Solve the linear system $H_{k,k}y = \beta e_1$ ;
11 : Set $\mathcal{X}_k = \mathcal{X}_0 + V_k \diamond y$ and $R_k = \mathcal{B} - \mathcal{A}\mathcal{X}_k$ ;
12 : end for
```

Algorithme 5 : The preconditioned Global FGMRES

```
1 : $V_1 = R_0$, $V_1 = V_1/\|V_1\|_F$
2 : for $j = 1, 2, ..., k$ do;
3 : $Z := \mathcal{P}^{-1}_{\epsilon,\alpha,Q} V_j$;
4 : $W := \mathcal{A}Z$;
5 : for $i = 1, 2, ..., j$ do;
6 : $H_{i,j} =< W, V_i >_F$;
7 :$W = W - H_{ij} V_i$;
8 : end;
9 : $H_{j+1,j} = \|W\|_F$;
10 : $V_{j+1} = W/H_{j+1,j}$;
11 : Solve the linear system $H_{k,k} y = \beta e_1$;
12 : Set $\mathcal{X}_k = \mathcal{X}_0 + V_k \diamond y$ and $R_k = \mathcal{B} - \mathcal{A}\mathcal{X}_k$;
13 : end for
```

Algorithme 6 : The preconditioned Bl-BiCGSTAB

```
1 : Choose $X_0$, compute $R_0 = J - A_\alpha X_0$, $P_0 = R_0$ and solve $\mathcal{M}\widehat{P_0} = R_0$;
2 : $\widetilde{R}_0$ an arbitrary $n \times s$ matrix;
3 : for $k = 0, 1, ..$do;
4 : $V_k = A_\alpha \widehat{P_k}$;
5 : Solve $\left(\widetilde{R}_0^T V_k\right) \alpha_k = \widetilde{R}_0^T R_k$;
6 : $S_k = R_k - V_k \alpha_k$;
7 : $\mathcal{M}\widehat{S_k} = S_k$;
8 : $T_k = A_\alpha \widehat{S_k}$;
9 : $w_k = \frac{\langle T_k, S_k \rangle_F}{\langle T_k, T_k \rangle_F}$;
10 : $X_{k+1} = X_k + \widehat{P_k}\alpha_k + w_k \widehat{S_k}$;
11 : $R_{k+1} = S_k - w_k T_k$;
12 : Solve $\left(\widetilde{R}_0 V_k\right) \beta_k = -\widetilde{R}_0^T T_k$;
13 : $P_{k+1} = R_{k+1} + (P_k - w_k V_k) \beta_k$;
14 : Solve $\mathcal{M}\widehat{P}_{k+1} = P_{k+1}$;
15 : end.
```

Numerical results

Sommaire

Numerical results

In this section we present the results of numerical experiments aimed at evaluating the convergence behaviour of the preconditioned global GMRES, global MINRES and global FGMRES methods using the proposed preconditioner. All of the reported numerical results were performed on a 64-bit 2.49 GHz core i5 processor and 8.00 GB RAM using MATLAB 2016. In all the experiments, we used ten right-hand sides. CPU times and iteration counts are reported under ”CPU” and ”Iter” in the tables below. In the cases of the preconditioned global GMRES and global FGMRES methods two values are reported under ”Iters”, namely, the number of steps for the preconditioned global GMRES method or for the preconditioned global FGMRES method and in parenthesis the total number of inner preconditioned conjugate gradient iterations. In all the tables, a dagger † shows that the method has not converged in at most 500 iterations. When $\epsilon = -1$, we used the preconditioned global GMRES and global FGMRES methods to solve the nonsymmetric saddle point problems with multiple right-hand sides (1.1). In practice, the preconditioners used for solving (1.1) are $\mathcal{P}_{\epsilon,\alpha,Q}$, $\mathcal{P}_T$ and $\mathcal{P}_D$. Where $\mathcal{P}_{\epsilon,\alpha,Q}$, $\mathcal{P}_T$ and $\mathcal{P}_D$ are given as follows :

$$\mathcal{P}_{\epsilon,\alpha,Q} = \begin{pmatrix} A & B^T \\ -B & \alpha Q \end{pmatrix}, \; \mathcal{P}_T = \begin{pmatrix} A & O \\ -B & S \end{pmatrix} \text{and } \mathcal{P}_D = \begin{pmatrix} A & O \\ O & S \end{pmatrix}. \tag{1.28}$$

When $\epsilon = 1$, we used the preconditioned global MINRES methods to solve the symmetric saddle point problems with multiple right-hand sides (1.1). The preconditioners are given as follows

$$\mathcal{P}_{\epsilon,\alpha,Q} = \begin{pmatrix} A & B^T \\ B & \alpha Q \end{pmatrix} \text{and } \mathcal{P}_D = \begin{pmatrix} A & O \\ O & S \end{pmatrix}. \qquad (1.29)$$

Where S is a sparse approximation of the pressure Schur complement $BA^{-1}B^T$ and Q is one of the matrices I or $\text{diag}(S)$. The parameter of the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioner is chosen as $\alpha \in \left[10^{-5}, 1\right]$. In all the numerical tests below, the initial guess is taken to be the null matrix and the right-hand side $\mathcal{B} \in \mathbb{R}^{(n+m)\times s}$ is chosen such that the exact solution of the saddle point problem (1.1) is a matrix of ones. For the preconditioned global MINRES, global GMRES, and global Flexible GMRES (FGMRES) methods, the iterations were stopped where

$$\frac{\|\mathcal{P}^{-1}\mathcal{B} - \mathcal{P}^{-1}\mathcal{A}\mathcal{X}^{(k)}\|_F}{\|\mathcal{P}^{-1}\mathcal{B}\|_F} < 10^{-12}, \qquad (1.30)$$

where $\mathcal{P}$ is one of the preconditioners $\mathcal{P}_{\epsilon,\alpha,Q}$, $\mathcal{P}_T$, or $\mathcal{P}_D$, $\|\cdot\|_F$ stands for the Frobenius norm and $\mathcal{X}^{(k)} \in \mathbb{R}^{(n+m)\times s}$ denotes the current iterate. When using the inner preconditioned global conjugate gradient for solving the first system of Algorithm 4, the preconditioner used is a drop tolerance-based incomplete Cholesky factorization, computed using the Matlab function "ichol(.,opts)", where

— opts.type = 'ict',
— opts.droptol = 1e-2.

The inner relative residual norm less than ($tol = 10^{-9}$).

Table 1 : The size of the matrices A and B for Lid driven cavity problem on $2^l \times 2^l$.

Lid driven cavity					
l	n	m	size of A	size of B	size of Q
4	578	192	578× 578	578× 192	192× 192
5	2178	768	2178 ×2178	2178 × 766	766× 766
6	8450	3070	8450×8450	8450× 3070	3070× 3070
7	33282	12288	33282 ×33282	33282 × 12288	12288× 12288

We use the IFISS software package developed by Elman et al.[25] to generate the linear systems with multiple right-hand sides corresponding to $l = 4$, $l = 5$, $l = 6$ and $l = 7$. The IFISS software provides the matrices Ast, Bst and Q for the matrices A, B and S, respectively. For the Lid driven cavity problem, Bst is a rank deficient matrix, thus we drop two first rows of Bst to get a full rank matrix. Generic information of the test problems, including n and m, are given in Table 1.

Numerical results for the nonsymmetric saddle point problem with multiple right-hand sides ($\epsilon = -1$) :

Table 2 : Numerical results for the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$ with $Q = I$.

	l	$l = 5$	$l = 6$	$l = 7$
α		Global GMRES	Global GMRES	Global GMRES
10^{-5}	Iter	6(103)	7(155)	9(225)
	CPU	1.17	7.31	89.29
	RES	2.93e-05	2.11e-04	1.39e-07
	ERR	1.65e-03	1.20e-02	1.92e-05
10^{-4}	Iter	8(81)	11(100)	17(116)
	CPU	1.25	7.43	81.65
	RES	3.93e-06	6.47e-07	1.54e-07
	ERR	3.43e-05	1.39e-05	1.09e-05
10^{-3}	Iter	14(47)	21(46)	32(67)
	CPU	1.28	7.79	102.83
	RES	1.17e-07	4.90e-08	7.66e-08
	ERR	1.79e-06	9.18e-07	5.47e-06
10^{-2}	Iter	27(23)	37(33)	43(55)
	CPU	1.50	9.28	94.54
	RES	1.18e-08	1.46e-08	2.30e-08
	ERR	1.69e-07	5.62e-07	2.73e-06
10^{-1}	Iter	40(17)	44(29)	46(50)
	CPU	1.53	9.43	97.17
	RES	5.43e-08	1.97e-08	2.18e-08
	ERR	1.18e-06	2.88e-06	1.87e-06
1	Iter	42(16)	45(26)	46(50)
	CPU	1.52	8.84	101.96
	RES	1.10e-08	2.29e-08	1.01e-08
	ERR	3.37e-07	9.37e-07	2.80e-06

Table 3 : Numerical results for the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$ with $Q = I$.

	l	$l = 5$	$l = 6$	$l = 7$
α		Global FGMRES	Global FGMRES	Global FGMRES
10^{-5}	Iter	6(85)	7(143)	8(206)
	CPU	0.93	6.61	60.67
	RES	5.97e-04	2.09e-04	1.31e-07
	ERR	4.87e-03	4.11e-03	2.87e-05
10^{-4}	Iter	7(69)	10(90)	14(106)
	CPU	0.99	5.59	55.58
	RES	3.77e-06	1.70e-07	8.43e-08
	ERR	3.23e-05	2.55e-06	2.46e-06
10^{-3}	Iter	12(43)	18(42)	27(65)
	CPU	0.98	6.99	68.58
	RES	1.08e-07	6.30e-08	7.05e-08
	ERR	6.68e-07	6.62e-07	1.87e-06
10^{-2}	Iter	24(21)	33(32)	39(53)
	CPU	1.19	7.92	78.40
	RES	6.94e-08	4.80e-08	9.28e-08
	ERR	2.01e-07	4.76e-07	3.57e-06
10^{-1}	Iter	37(16)	41(28)	45(51)
	CPU	1.45	8.84	90.95
	RES	5.54e-08	9.23e-08	1.08e-07
	ERR	1.09e-07	1.12e-06	2.04e-06
1	Iter	43(16)	46(27)	49(49)
	CPU	1.66	9.04	96.55
	RES	7.39e-08	7.92e-08	1.08e-07
	ERR	2.51e-07	1.78e-06	1.27e-06

Table 4 : Numerical results for the three preconditioned global GMRES methods.

l		$\mathcal{P}_{\epsilon,\alpha,Q}(\alpha = 1, Q = I)$	$\mathcal{P}_T$	$\mathcal{P}_D$
$l = 5$	Iter	42(16)	60(15)	122(15)
	CPU	1.52	2.24	4.90
	RES	1.10e-08	1.00e-07	1.04e-06
	ERR	3.37e-07	3.48e-06	6.84e-06
$l = 6$	Iter	45(26)	62(26)	129(25)
	CPU	8.84	11.06	22.70
	RES	2.29e-08	2.04e-07	9.65e-07
	ERR	9.37e-07	6.22e-06	5.57e-05
$l = 7$	Iter	46(50)	64(47)	133(46)
	CPU	101.96	114.56	295.49
	RES	1.01e-08	3.66e-07	2.14e-06
	ERR	3.37e-07	3.04e-05	1.21e-04

Table 5 : Numerical results for the three preconditioned global FGMRES methods.

l		$\mathcal{P}_{\epsilon,\alpha,Q}(\alpha = 10^{-4}, Q = I)$	$\mathcal{P}_T$	$\mathcal{P}_D$
$l = 5$	Iter	7(69)	58(16)	115(16)
	CPU	0.99	2.24	3.78
	RES	3.77e-06	1.65e-06	8.30e-07
	ERR	3.23e-05	1.11e-05	1.41e-06
$l = 6$	Iter	10(90)	59(27)	119(27)
	CPU	5.59	11.62	22.99
	RES	1.70e-07	1.05e-07	5.30e-08
	ERR	2.55e-06	4.36e-06	1.16e-06
$l = 7$	Iter	14(106)	61(49)	121(50)
	CPU	55.58	92.52	181.65
	RES	8.43e-08	1.36e-07	9.32e-08
	ERR	2.46e-06	1.75e-05	1.00e-06

In Tables 2, 3, 4 and 5 we report the results for the preconditioned global GMRES and global FGMRES iterative methods. From numerical results listed in Tables, we can conclude that the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global GMRES and global FGMRES methods require less iterations and has faster CPU times than $\mathcal{P}_T$ and $\mathcal{P}_D$ in all trials.

Table 6 : Numerical results for the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$ with $l = 5$.

α	Q	$Q = I$	$Q = \text{diag}(S)$	$Q = S$
		Global GMRES	Global GMRES	Global GMRES
10^{-5}	Iter	6(103)	9(211)	14(215)
	CPU	1.17	3.45	6.05
	RES	1.25e-04	8.11e-09	1.62e-07
	ERR	6.06e-03	6.26e-08	9.49e-07
10^{-4}	Iter	8(81)	8(190)	15(209)
	CPU	1.25	2.92	6.19
	RES	3.93e-06	3.63e-09	1.72e-07
	ERR	3.43e-05	2.59e-08	9.91e-07
10^{-3}	Iter	14(47)	11(162)	22(173)
	CPU	1.28	3.62	7.51
	RES	1.17e-07	4.16e-09	8.08e-09
	ERR	1.79e-06	3.76e-08	5.51e-08
10^{-2}	Iter	27(23)	14(123)	33(123)
	CPU	1.50	4.37	8.87
	RES	1.18e-08	2.80e-08	9.72e-08
	ERR	1.69e-07	1.77e-07	6.21e-07
10^{-1}	Iter	40(17)	23(65)	115(65)
	CPU	1.53	3.64	18.63
	RES	5.43e-08	1.21e-07	8.72e-08
	ERR	1.18e-06	8.72e-07	5.36e-07
1	Iter	42(16)	27(26)	†
	CPU	1.52	1.73	†
	RES	1.10e-08	1.43e-07	†
	ERR	3.37e-07	3.03e-06	†

Table 6 indicate, that the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioner with $Q = I$ leads to much better numerical results than the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioner with $Q = \text{diag}(S)$ and $Q = S$ as the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned GMRES method with $Q = I$, need less CPU times in all trials and iteration steps in some values of α comparing with the other choices of the matrix Q.

Numerical results for symmetric saddle point problem with multiple right-hand sides ($\epsilon = 1$) :

Table 7 : Numerical results for the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$ with $Q = I$.

		$l = 5$	$l = 6$	$l = 7$
α		Global MINRES	Global MINRES	Global MINRES
10^{-1}	Iter	38	38	34
	CPU	0.41	2.58	45.60
	RES	3.11e-10	6.95e-10	3.07e-08
	ERR	1.91e-07	5.16e-06	7.02e-04
1	Iter	37	38	37
	CPU	0.36	2.41	46.18
	RES	1.00e-09	1.90e-09	5.46e-08
	ERR	1.59e-06	8.33e-05	2.68e-03
10	Iter	39	38	38
	CPU	0.54	2.79	49.52
	RES	2.11e-09	7.51e-09	1.45e-07
	ERR	8.05e-07	9.25e-06	1.70e-02

Table 8 : Numerical results for global MINRES with $\mathcal{P}_{\epsilon,\alpha,Q}$ and $\mathcal{P}_D$.

l		$\mathcal{P}_{\epsilon,\alpha,Q}(\alpha = 1, Q = I)$	$\mathcal{P}_D$
$l = 5$	Iter	37	104
	CPU	0.36	0.97
	RES	1.00e-09	8.75e-10
	ERR	1.59e-06	1.26e-06
$l = 6$	Iter	38	110
	CPU	2.41	5.22
	RES	1.90e-09	5.68e-10
	ERR	8.33e-05	1.24e-04
$l = 7$	Iter	37	†
	CPU	46.18	†
	RES	5.46e-08	†
	ERR	2.68e-03	†

As observed in Table 7 and 8, the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global MINRES method with the proper parameter α has a better performance than the $\mathcal{P}_D$ preconditioned global MINRES method, in terms of the iterations and CPU times.

Numerical results for nonsymmetric saddle point problem with multiple right-hand sides ($\epsilon = -1$) and when the discretization is non-uniform.

Table 9 : Numerical results for the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$ with $Q = I$ and $l = 5$.

α		Global GMRES(inner PCG)	Global GMRES(inner PBl-BiCGSTAB)	Global FGMRES(inner PCG)
10^{-3}	Iter	37(76)	44(67)	33(79)
	CPU	5.43	21.96	4.67
	RES	3.49e-06	1.47e-05	1.78e-07
	ERR	1.04e-05	6.37e-05	6.95e-07
10^{-2}	Iter	99(32)	118(19)	93(32)
	CPU	7.17	16.90	6.11
	RES	4.81e-07	7.31e-05	1.25e-07
	ERR	2.71e-06	6.01e-04	2.91e-07
10^{-1}	Iter	192(18)	192(9)	178(18)
	CPU	12.19	16.81	9.20
	RES	2.88e-08	5.42e-07	9.77e-08
	ERR	5.12e-07	1.54e-05	2.17e-07
1	Iter	271(4)	251(9)	227(3)
	CPU	12.29	19.34	10.42
	RES	2.59e-12	3.55e-09	1.30e-11
	ERR	1.63e-06	9.42e-07	3.54e-08

For $\alpha = 10^{-5}$ and $\alpha = 10^{-4}$, the $\mathcal{P}_{\epsilon,\alpha,Q}$(inner PBl-BiCGSTAB) preconditioned global GMRES method diverge while the $\mathcal{P}_{\epsilon,\alpha,Q}$(inner PCG) preconditioned global GMRES and the $\mathcal{P}_{\epsilon,\alpha,Q}$(inner PCG) preconditioned global FGMRES converge. For the following experiments we will show the good behavior of the preconditioned global GMRES methods even for a particular right-hand sides.

Numerical result for the preconditioned global GMRES methods when the right-hand sides $\mathcal{B} = \begin{pmatrix} F \\ O \end{pmatrix}$.

Table 10 : Numerical results for the three preconditioned global GMRES methods.

l		$\mathcal{P}_{\epsilon,\alpha,Q}(\alpha = 10^{-5}, Q = I)$	$\mathcal{P}_T$	$\mathcal{P}_D$
$l = 5$	Iter	6(92)	62(15)	134(15)
	CPU	1.00	2.04	4.11
	RES	4.46e-08	6.18e-08	3.56e-08
	ERR	3.10e-07	6.05e-07	2.67e-07
$l = 6$	Iter	8(132)	64(26)	142(26)
	CPU	7.04	10.35	23.00
	RES	7.23e-07	2.44e-07	8.58e-08
	ERR	1.10e-05	6.02e-06	1.9e-06
$l = 7$	Iter	10(199)	66(47)	148(46)
	CPU	69.69	127.88	297.82
	RES	4.59e-07	4.09e-07	6.54e-08
	ERR	8.96e-05	2.10e-05	7.35e-06

From the numerical results shown in Table 10, we see that the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioner is superior to the $\mathcal{P}_T$ and the $\mathcal{P}_D$ preconditioners in terms of the IT and CPU times.

Table 11 : Numerical results for preconditioned global GMRES and preconditioned block GMRES methods with $l = 5$ and $Q = I$.

α		$\mathcal{P}_{\epsilon,\alpha,Q}$ global GMRES	$\mathcal{P}_{\epsilon,\alpha,Q}$ block GMRES
10^{-5}	Iter	6(104)	164
	CPU	1.93	20.56
	RES	2.93e-05	8.78e-01
	ERR	1.65e-03	2.17e+01
10^{-4}	Iter	8(81)	191
	CPU	1.25	24.60
	RES	3.93e-06	1.20e-03
	ERR	1 3.41e-05	7.50e-03
10^{-3}	Iter	14(47)	164
	CPU	1.30	15.77
	RES	1.17e-07	1.49e-06
	ERR	1.79e-06	6.68e-06
10^{-2}	Iter	27(23)	132
	CPU	1.44	9.66
	RES	1.18e-08	2.10e-08
	ERR	1.69e-07	1.36e-07
10^{-1}	Iter	40(17)	171
	CPU	1.54	18.17
	RES	5.43e-08	4.74e-09
	ERR	1.18e-06	4.68e-08
1	Iter	42(16)	236(42)
	CPU	1.66	41.95
	RES	1.10e-08	3.87e-09
	ERR	3.37e-07	3.41e-07

The preconditioned global GMRES and preconditioned block GMRES methods incorporated with the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioners are listed in Tables 11 for different values of α. From numerical results listed in the preceding Table, we can conclude that the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global GMRES method requires less iterations and has faster CPU times than $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned block GMRES method in all trials. In each case when all two methods produce solutions, $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global GMRES method, gives smaller relative residual and error than $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned block GMRES method.

Table 12 : Numerical results for preconditioned global GMRES with $Q = I$ and $l = 5$.

α		$\mathcal{P}_{\epsilon,\alpha,Q}$(inner PGCG)	$\mathcal{P}_{\epsilon,\alpha,Q}$ (inner PBl-CG)
10^{-5}	Iter	6(104)	6(279)
	CPU	1.93	6.95
	RES	2.93e-05	7.31e-04
	ERR	1.65e-03	7.08e-03
10^{-4}	Iter	8(81)	8(203)
	CPU	1.25	6.74
	RES	3.93e-06	4.02e-06
	ERR	1 3.41e-05	3.50e-05
10^{-3}	Iter	14(47)	14(49)
	CPU	1.30	4.59
	RES	1.17e-07	1.17e-07
	ERR	1.79e-06	1.80e-06
10^{-2}	Iter	27(23)	27(24)
	CPU	1.44	3.37
	RES	1.18e-08	3.93e-08
	ERR	1.69e-07	4.26e-07
10^{-1}	Iter	40(17)	40(18)
	CPU	1.54	4.05
	RES	5.43e-08	3.85e-08
	ERR	1.18e-06	7.50e-07
1	Iter	42(16)	42(17)
	CPU	1.66	3.48
	RES	1.10e-08	2.01e-08
	ERR	3.37e-07	2.82e-07

Table 12 reports the Iter, CPU times, residual (RES) and error (ERR) of the tested $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global GMRES methods with respect to different values of α. From Table 12, we observe that the $\mathcal{P}_{\epsilon,\alpha,Q}$(inner PGCG) preconditioned global GMRES method outperforms the $\mathcal{P}_{\epsilon,\alpha,Q}$(inner PBl-CG) preconditioned global GMRES method in terms of the Iter and CPU times.

We will present a numerical results for some examples of Navier-Stokes equations. In the following figures, we present a streamline plot for the velocity solution, and a plot of the pressure solution of the Flow in a symmetric step channel, over a plate and over a backward facing step.

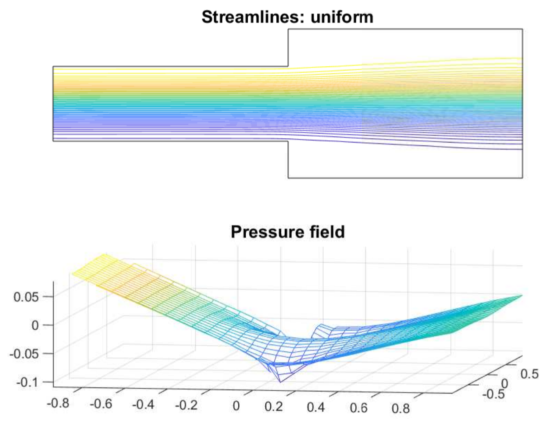


FIGURE 1.1 – Streamline plot and pressure plot of Flow in a symmetric step channel.

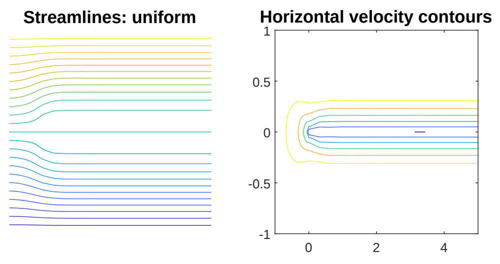


FIGURE 1.2 – Streamline plot and pressure plot of Flow over a plate problem.

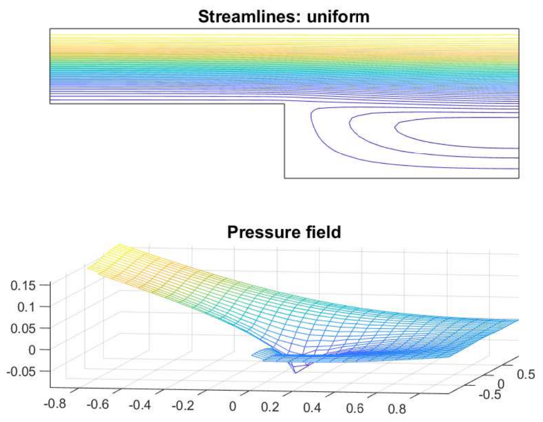


FIGURE 1.3 – Streamline plot and pressure plot of Flow over a backward facing step.

Table 13 : Numerical results for the three preconditioned global GMRES methods with $l = 5$.

Problem		$\mathcal{P}_{\epsilon,\alpha,Q}(\alpha = 10^{-1}, Q = I)$	$\mathcal{P}_T$	$\mathcal{P}_D$
Flow over a plate	Iter	34(103)	†	†
	CPU	13.51	†	†
	RES	1.51e-05	†	†
	ERR	3.07e-05	†	†
Flow in a symmetric step channel	Iter	22(65)	†	†
	CPU	5.21	†	†
	RES	1.41e-05	†	†
	ERR	4.12e-05	†	†
Flow over a backward facing step	Iter	22(55)	†	†
	CPU	2.80	†	†
	RES	2.56e-04	†	†
	ERR	3.19e-03	†	†

Table 14 : Numerical results for the three preconditioned global FGMRES methods with $l = 5$.

Problem		$\mathcal{P}_{\epsilon,\alpha,Q}(\alpha = 10^{-1}, Q = I)$	$\mathcal{P}_T$	$\mathcal{P}_D$
Flow over a plate	Iter	42	†	†
	CPU	0.43	†	†
	RES	4.81e-08	†	†
	ERR	1.29e-06	†	†
Flow in a symmetric step channel	Iter	25	†	†
	CPU	0.17	†	†
	RES	3.32e-08	†	†
	ERR	7.05e-07	†	†
Flow over a backward facing step	Iter	26	†	†
	CPU	0.16	†	†
	RES	2.27e-08	†	†
	ERR	9.48e-07	†	†

From numerical results listed in Tables, we can conclude that the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global GMRES and global FGMRES methods requires less iterations and has faster CPU times.

Conclusion

We have presented the global approach, as a new strategy to solve the saddle point with multiple right-hand sides. In addition, we introduced and studied the preconditioner $\mathcal{P}_{\epsilon,\alpha,Q}$. Tables 2, 3, 4, 5, 6 and 9, illustrate that the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioner with suitable choices of the parameter α, has a better performance as compared to $\mathcal{P}_T$ and $\mathcal{P}_D$ in all trials. The Tables 7 and 8, reveal the efficiency of the $\mathcal{P}_{\epsilon,\alpha,Q}$ preconditioned global MINRES method for solving the symmetric saddle point problems with several right-hand sides. We conclude that our preconditioned global Krylov subspace methods are very powerful for solving the saddle point problems with multiple right-hand sides (1.1).

Annexe

MATLAB program of the preconditioned Global Krylov subspace method for solving saddle point problem with multiple right-hand sides

```
%% Element fini Q2-P1
m=size(Q,1);
n=size(A,1);
BB=Bst(3:m,1:n);
[l,nm]=size(BB);

%% Mass matrix
Q1=Q(1:l,1:l);

%% Choose alpha
alpha=1e-3;

%% Choose type of the problem
epsilon=-1;

%% Create artificial rhs by choosing the exact solution
%% Fix the right-hand side number
nrhs=10;
Exact_Sol=ones(n+l,nrhs);

%% Choose the initial solution
 x0=zeros(n+l,nrhs);

%% Compute the right hand side
rhs=AAx(epsilon,Ast,BB,Exact_Sol);

I=speye(l,l);
%% ------------ paramaters for KRYLOV SUBSPACE method
%% im=200; maxits=2000; tolIts=1e-12; tol0=10^-8;
```

```
im=200; maxits=500; tolIts=1e-12; tol0=10^-8;

%% Preconditioner of system intern
L=ichol(Ast,struct('type','ict','droptol',1e-2));

tic
%% Left preconditioning Pr
[sol,res,its,it]=PGlbGMRES(epsilon,Ast,BB,I,L,L',alpha,rhs
   ,x0,im,maxits,tolIts);
toc


Res=norm(rhs-AAx(epsilon,Ast,BB,sol),'fro');
fprintf('\n Iteration  =%3g      ',its);
fprintf('\n Res =%12.4e       ',Res);
fprintf('\n Err   =%12.4e      ', norm(sol-Exact_Sol,'fro')
   );
fprintf('\n Iteration intern  =%3g      ',it/its);
```

```

function [y,it]=Precfunct(epsilon,A,B,QQ,L,U,alpha,x)
%--------------------------------------------------
% y=precfun(A,B,QQ,alpha,r)
%  Preconditioner matrix times vector x
% A, B  = matrices from the Saddle system
%  QQ : diagonal matrix
%
%  (A   B^T)
%  (       ) x= y
%  (-B    0 )
%
%   im      = krylov subspace dimension
%   rhs     = right-hand side
%   x0      = initial guess
%   tol     = tolerance for stopping criterion
%   maxits = max number of matrix - vector products
%    allowed (= max tot. # of global gmres steps)
%--------------------------------------------------
 [m,n]=size(B); alphainv=1/alpha;
 temp = alphainv * x(n+1:n+m,:);
 u=x(1:n,:)-B'*temp;

 [y1]=Gcg01(A,B,QQ,alphainv,L,u) ;
 y2=temp-epsilon*(QQ\(B*alphainv*y1));
 y=[y1;y2];
end
```

```
function [x,its] = Gcg01(A,B,QQ,beta,L,U,rhs)


%% solves by using Global CG the linear system :
%(A + beta  B' *  QQ^(-1) *  B ) * x = rhs

[n,m]=size(rhs);
x0=zeros(n,m);
maxit=n;tol=1e-8;
alpha=(1/beta);


%%------------------------------------------

 OutputG=1;
 n=size(A,1);
 x = x0;
 r=rhs-(A  * x + B' * (QQ\( B*(beta *x)) ));


 z = U\(L\r);
 p = z ;
 ro1 = z(:)' * r(:);
 tol1 = tol*tol*ro1;
 its = 0 ;
 while (its < maxit && ro1 > tol1)

         its = its+1;
     res(its)=ro1;
         ro = ro1;
         ap=  A  * p + B' * (QQ\( B*(beta *p)));

         alp = ro / ( ap(:)'* p(:)) ;
         x = x + alp * p ;
         r = r - alp * ap;
```

```
    z =U\(L\ r);
    ro1= z(:)'*r(:);
        bet = ro1 / ro ;

        p = z + bet * p;


end
  if(OutputG)
    fprintf(' %4.0f %22.14e \n', its,sqrt(ro1))
    end

end
```

```
function [sol,res,its,it]=PGlbGMRES(epsilon,A,B,Q,L,U,
   alpha,rhs,x0,im,maxits,tol);
%-------------------------------------------------
% [res,Res,sol] =PGlbGMRES(A,B,Q,alpha,rhs,x0,im,maxits,
   tolIts)
% restarted Preconditioned Global gmres with Krylov
   subspace of dim = im.
% A , B  = matrices from the Saddle system
%  Q is a diagonal matrix introduced for the
   preconditioner
%
%      (A    B^T)^(-1)      (A    B^T)        (A    B^T)^(-1)
%      (         )        *  (         ) * X=  (         )
   *  Rhs
%      (-B    ?Q)            (-B     0 )      (-B    ?Q)
%   im     = krylov subspace dimension
%   rhs    = right-hand side
%   x0     = initial guess
%   tol    = tolerance for stopping criterion
%   maxits = max number of matrix - vector products
%          allowed (= max tot. # of global gmres steps)
%-------------------------------------------------
      tolmac = eps;
      its = 1 ;
      sol = x0;
      outputG=0;
      [m,n]=size(B);
      it=0;

%%
      while (its < maxits)
      r=rhs-AAx(epsilon,A,B,sol);
      vv{1}=Precfunct(epsilon,A,B,Q,L,U,alpha,r);

      ro = norm(vv{1},'fro') ;
      res(its) = ro;
      if (its  == 1)
        tol1=tol*ro ;
      end
      if (ro < tol1 |its > maxits)
        return
      end
      t = 1.0/ ro ;
      vv{1} = vv{1} * t ;
%%---initialize 1-st term of rhs of hess.system..
      rs(1) = ro ;
      i = 0 ;
 while (i < im & (ro > tol1)& its < maxits )
      %% print its/residual info
      if (outputG)
       fprintf(1,' its %d  res %e \n',its,ro)
      end
      i=i+1 ;
      its = its + 1 ;
      i1 = i + 1 ;
      z = AAx(epsilon,A,B,vv{i});
      vv{i1}=Precfunct(epsilon,A,B,Q,L,U,alpha,z);
%---------- modified GS ;
      for j=1:i

         t = (vv{j}(:))'*(vv{i1}(:)) ;
         hh(j,i) = t ;
           vv{i1} =      vv{i1} - t*vv{j} ;
      end
      t = norm(vv{i1},'fro') ;
      hh(i1,i) = t ;
      if (t  ~= 0.0)
```

```
        t = 1.0 / t ;
        vv{i1} = vv{i1}*t ;
      end
%%
    if (i ~= 1)
%---------- apply previous transformations on i-th column
   of h ;
       for k=2:i
         k1 = k-1 ;
         t = hh(k1,i) ;
         hh(k1,i) = c(k1)*t + s(k1)*hh(k,i) ;
         hh(k,i) = -s(k1)*t + c(k1)*hh(k,i) ;
       end
     end  %% IF
%%
     gam = sqrt(hh(i,i)^2 + hh(i1,i)^2) ;
     gam = max(tolmac,gam);
%---------- determine plane rotation & update rhs of LS Pb
     c(i) = hh(i,i)/gam ;
     s(i) = hh(i1,i)/gam ;
     rs(i1) = -s(i)*rs(i) ;
     rs(i) =  c(i)*rs(i) ;
%---------- determine res. norm. and test for convergence
     hh(i,i) = c(i)*hh(i,i) + s(i)*hh(i1,i) ;
     ro = abs(rs(i1)) ;
     res(its) = ro ;
   end
%---------- compute solution. solve upper triangular
   system
   rs(i) = rs(i)/hh(i,i) ;
   for  k=i-1:-1:1
      t=rs(k) ;
      for j=k+1:i
         t = t-hh(k,j)*rs(j) ;
      end
      rs(k) = t/hh(k,k) ;
   end
%---------- now form linear combination to get solution
   for j=1:i
      sol = sol +rs(j)* vv{j} ;
   end
     if ((ro  <=  tol1) | (its >= maxits)) return; end;
 end
```

```
function [sol,res,its]=BlBiCGSTABEX(epsilon,A,B,Q,L,U,alpha,rhs,x0,im,maxits,tol);
 %-------------------------------------------------
% [sol,res,its] =BlBiCGSTABEX(epsilon,A,B,Q,L,U,alpha,rhs,x0,im,maxits,tol)
% restarted Preconditioned Bl-BICGSTAB with Krylov subspace of dim = im.
% A , B  = matrices from the Saddle system
%  Q is a diagonal matrix introduced for the preconditioner
%
%      (A   B^T)^(-1)      (A   B^T)        (A   B^T)^(-1)
%      (        )       *  (        ) * X=  (        )  *  Rhs
%      (-B   ?Q)           (-B    0 )       (-B   ?Q)
%   im     = krylov subspace dimension
%   rhs    = right-hand side
%   x0     = initial guess
%   tol    = tolerance for stopping criterion
%   maxits = max number of matrix - vector products
%          allowed (= max tot. # of global gmres steps)
%-------------------------------------------------

          [n,m]=size(rhs);
          tol=1e-8;
          OutputG=1;
          n=size(A,1);
          sol = x0;


          r=rhs-AAx(epsilon,A,B,sol);
          r_hat=r;
          p=r_hat;

          p_hat=Precfunct(epsilon,A,B,Q,L,U,alpha,r);
          ro1=r_hat(:)'*r(:);

          roo=r(:)' *p_hat(:);
          tol1=tol*tol*roo;

          its=0 ;
          res1=r(:)'*p_hat(:);
          while (its < maxit && sqrt(res1) > tol)

          its = its+1;
          res(its)=ro1;
          ro = ro1;

          ap=AAx(epsilon,A,B,p_hat);
          [alp]=PGMRESS(ap,r,r_hat,zeros(m,m),im,maxit,1e
             -6);

          s=r-ap*alp;
          s_hat=Precfunct(epsilon,A,B,Q,L,U,alpha,s);
          as=AAx(epsilon,A,B,s_hat);

          ro2=as(:)'*s(:);
          w=ro2/( as(:)'* as(:));

          sol=sol+p_hat*alp+ w*s_hat;
          r=s-w*as;

          ro1= r_hat(:)'*r(:);
          res1=r(:)'*r(:);
          [bet]=PGMRESS(ap,-as,r_hat,zeros(m,m),im,maxit,1
             e-6);

          p=r+(p-w*ap)*bet;
          p_hat=Precfunct(epsilon,A,B,Q,L,U,alpha,p);
          if(OutputG)
          fprintf(' %4.0f %22.14e \n', its,sqrt(res1))
          end

          end

          end
```

Troisième partie

Preconditioned Krylov subspace and GMRHSS iteration methods

Chapitre 2

Preconditioned Krylov subspace and GMRHSS iteration methods

Sommaire

Abstract :

In this chapter, we propose a separate approach as a new strategy to solve the saddle point problem arising from the stochastic Galerkin finite element discretization of Stokes problems. The preconditioner is obtained by replacing the $(1,1)$ and $(1,2)$ blocks in the RHSS preconditioner by others well chosen and the parameter α in $(2,2)$–block of the RHSS preconditioner by another parameter β. The proposed preconditioner can be used as a preconditioner corresponding to the stationary itearative method or to accelerate the convergence of the Generalized Minimal Residual method (GMRES). The convergence properties of the GMRHSS iteration method are derived. Meanwhile, we analyzed the eigenvalue-distribution and the eigenvectors of the preconditioned matrix. Finally, numerical results show the effectiveness of the proposed preconditioner as compared to other preconditioners.

Résumé :

Dans ce chapitre, nous proposons une approche séparée comme nouvelle stratégie pour résoudre le problème de point selle obtenu de la discrétisation stochastique de problèmes de Stokes. Le préconditionneur s'obtient en remplaçant les blocs $(1,1)$ et $(1,2)$ du préconditionneur RHSS par d'autres bien choisis. Le préconditionneur proposé peut être utilisé comme préconditionneur correspondant à la méthode itérative stationnaire ou pour accélérer la convergence de la méthode GMRES. Les propriétés de convergence de la méthode d'itération GMRHSS sont données. Parallèlement, nous avons analysé la distribution des valeurs propres et les vecteurs propres de la matrice préconditionnée. Enfin, les résultats numériques montrent l'efficacité du préconditionneur proposé par rapport aux autres préconditionneurs.

2.1 Introduction

Firslty, we recall the definition of the Kronecker product :

Definition 2.1 *Let $C \in \mathbb{R}^{p\times q}$ and $D \in \mathbb{R}^{r\times d}$. Then the Kronecker product of C and D has the following form*

$$C \otimes D = \begin{pmatrix} C_{11}D & \dots & C_{1q}D \\ \vdots & \ddots & \vdots \\ C_{p1}D & \dots & C_{pq}D \end{pmatrix} \in \mathbb{R}^{pr\times qd}.$$

Now we consider the following nonsymmetric saddle point problem :

$$\left[G_0 \otimes \begin{pmatrix} A & B^T \\ -B & 0 \end{pmatrix}\right]\left[z \otimes \begin{pmatrix} x \\ y \end{pmatrix}\right] = c \otimes b, \tag{2.1}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric positive definite matrix, $B^T \in \mathbb{R}^{n\times m}$ has full column rank, $G_0 \in \mathbb{R}^{s\times s}$ is invertible matrix, O is zero matrix, $c \in \mathbb{R}^{s\times 1}$ and $b \in \mathbb{R}^{(n+m)\times 1}$ are given vectors, with $m \leq n$. Here B^T denotes the transpose of B. The above assumptions guarantee the existence and uniqueness of the solution of the saddle-point problem (2.1). This type of problem is important and arose in scientific and engineering application, for more details we recommend the readers to see [11, 15]. Since the matrices A and B in (2.1) are large and sparse, the solution of (2.1) is suited for being solved by the iterative methods. Many of effective iteration methods have been developed for solving the saddle point problem, for example, the Hermitian and skew-Hermitian splitting (HSS) method [6] and the Regularized HSS iteration method [2] . Also the Krylov subspace methods, which are introduced in [1] and studied in [48] , are very efficent methods for solving the large linear systems of equations. However Krylov subspace methods without good preconditioner converge very slowly, when they are applied to the corresponding saddle point problems. In order to accelerate the convergence rate of the associated Krylov subspace, several preconditioners [6, 2, 10] are proposed for (2.1). The paper is organized as follows. In section 1 we review some properties and definitions of Kronecker product. Section 2 introduce the separate approach. The convergence of the GMRHSS iteration will be analysed in Section 3, the eigenvalue-distribution and the eigenvectors of the preconditioned matrix will be investigated in Section 4. In Section 5, we gave the implementation of the GMRHSS preconditioner. Throughout this paper, I will denote identity.

Model problem :

As discussed in [27, 41], diffusion problems with random data, groundwater flow modelling, steady state Navier-Stokes, Stokes flow and deterministic second-order elliptic partial differential equations (PDEs), lead to the saddle point problems (2.1).

$$\begin{cases} -v\nabla^2 u + \nabla p = f, & \Omega, \\ \nabla \cdot u = 0, & \Omega. \end{cases} \tag{2.2}$$

Where $v > 0$ is a given constant associated with the kinematic viscosity. The variable u represents the velocity of the fluid and p represents the pressure. $\Omega \subset \mathbb{R}^2$ is domain of space with boundary $\partial\Omega$, some Dirichlet boundary conditions which describe an inflow outflow problem with the adequate initial conditions, will be found in [13]. By using the discretization with the stochastics finite element [27, 41] , we obtain the following saddle point problem :

$$\begin{pmatrix} G_0 \otimes A_0 + \sum_{i=1}^{l} G_i \otimes A_i & G_0 \otimes B^T \\ G_0 \otimes B & O \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \mathbf{f} \otimes \mathbf{g} \\ O \end{pmatrix}. \tag{2.3}$$

Where $A_0, A_1, ..., A_l$, B are finite element matrices and $G_0, .., G_l$ are Galerking matrices. In order to obtain the saddle point problem (2.1), using the orthonormal basis of stochastics finite element for more details, we refer the reader to see. [27]. In the following section we recall some definitions and properties of the Kronecker product. It's not difficult to show the following properties satisfied by the product $\otimes$.

2.2 Properties of the Kronecker product

The following properties have been shown for the product $\otimes$. Let $H, O \in \mathbb{R}^{p\times q}$, $J, K \in \mathbb{R}^{r\times d}$, $L \in \mathbb{R}^{q\times k}$, $N \in \mathbb{R}^{d\times l}$ and $\alpha \in \mathbb{R}$. Then we have

$$\begin{aligned} (\alpha H) \otimes J &= H \otimes (\alpha J) = \alpha (H \otimes J). \\ (H \otimes J)^T &= H^T \otimes J^T. \\ (H + O) \otimes K &= H \otimes K + O \otimes K. && (2.4) \\ H \otimes (J + K) &= H \otimes J + H \otimes K. && (2.5) \\ (H \otimes J)(L \otimes N) &= HL \otimes JN. && (2.6) \end{aligned}$$

If Z and S are two invertible square matrices, then the inverse of their Kronecker product is

$$(Z \otimes S)^{-1} = Z^{-1} \otimes S^{-1}. \tag{2.7}$$

In the following section we introduce a separation of solution approach as a new startegy to solve (2.1).

2.3 Separate solution

Since the right side of system (2.1) has a particular form, its solution is given in the following theorem.

Theorem 3

If z is a solution of $G_0 z = c$ and $\begin{pmatrix} u \\ v \end{pmatrix}$ is a solution of the following problem

$$\underbrace{\begin{pmatrix} A & B^T \\ -B & 0 \end{pmatrix}}_{\mathcal{A}} \begin{pmatrix} u \\ v \end{pmatrix} = b. \tag{2.8}$$

Then $z \otimes \begin{pmatrix} u \\ v \end{pmatrix}$ is a solution of (2.1).

Proof : Invoking distributive properties (2.4) and (2.5), we obtain

$$(G_0 \otimes \mathcal{A})\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right) = G_0 z \otimes \mathcal{A}\begin{pmatrix} u \\ v \end{pmatrix}. \tag{2.9}$$

Finally, the relation (2.9) and the hypothesis of the theorem imply

$$(G_0 \otimes \mathcal{A})\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right) = c \otimes b. \tag{2.10}$$

Which shows the result. □

2.4 The generalized modified of the RHSS preconditioner

Since the convergence rate of the Krylov subspace methods dependent on the spectral distribution of the preconditioned matrix, that is expected that the eigenvalue distribution of the preconditioned matrix are well-clustered. Moreover, to obtain a better preconditioner for the nonsymmetric saddle point matrix, we use the following coefficient matrix splitting.

2.4.1 The RHSS splitting

Recently, Zhong-Zhi Bai and Michele Benzi [2] , proposed an efficient splitting (RHSS splitting) for the nonsymmetric saddle point matrix $\mathcal{A} \in \mathbb{R}^{(n+m)\times(n+m)}$. For a given symmetric positive semidefinite matrix $Q \in \mathbb{R}^{m\times m}$, the RHSS splitting is given as follow :

$$\begin{aligned}\mathcal{A} &= \begin{pmatrix} A & 0 \\ 0 & Q \end{pmatrix} + \begin{pmatrix} 0 & B^T \\ -B & -Q \end{pmatrix} = H_+ + S_-, \\ &= \begin{pmatrix} 0 & B^T \\ -B & Q \end{pmatrix} + \begin{pmatrix} A & 0 \\ 0 & -Q \end{pmatrix} = S_+ + H_-. \end{aligned} \tag{2.11}$$

Using the RHSS splitting of the matrix $\mathcal{A}$, we get the RHSS iteration method [2] into two systems of fixed-point equations :

$$\begin{cases} (\alpha I + H_+)\begin{pmatrix} u \\ v \end{pmatrix}^{(k+\frac{1}{2})} = (\alpha I - S_-)\begin{pmatrix} u \\ v \end{pmatrix}^{(k)} + \begin{pmatrix} f \\ g \end{pmatrix}, \\ (\alpha I + S_+)\begin{pmatrix} u \\ v \end{pmatrix}^{(k+1)} = (\alpha I - H_-)\begin{pmatrix} u \\ v \end{pmatrix}^{(k+\frac{1}{2})} + \begin{pmatrix} f \\ g \end{pmatrix}. \end{cases}$$

With $\alpha > 0$ and I is the identity matrix.
In the following subsection we redefine the RHSS iteration method for (2.1) and we consider the case where Q is a positive definite matrix.

2.4.2 Redefinition of the RHSS iteration method

Invoking the RHSS splitting, we obtain

$$\begin{cases} G_0 \otimes \mathcal{A} = G_0 \otimes \left[\begin{pmatrix} A & 0 \\ 0 & Q \end{pmatrix} + \begin{pmatrix} 0 & B^T \\ -B & -Q \end{pmatrix}\right] = G_0 \otimes (H_+ + S_-), \\ \qquad\quad = G_0 \otimes \left[\begin{pmatrix} 0 & B^T \\ -B & Q \end{pmatrix} + \begin{pmatrix} A & 0 \\ 0 & -Q \end{pmatrix}\right] = G_0 \otimes (S_+ + H_-). \end{cases}$$

The RHSS splitting of the matrix $G_0 \otimes \mathcal{A}$ naturally gives rise to equivalent reformulations of the saddle-point problem (2.1) into two systems of fixed-point equations as :

$$\begin{cases} (G_0 \otimes (\alpha I + H_+))\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k+\frac{1}{2})} = (G_0 \otimes (\alpha I - S_-))\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k)} + c \otimes b, \\ (G_0 \otimes (\alpha I + S_+))\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k+1)} = (G_0 \otimes (\alpha I - H_-))\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k+\frac{1}{2})} + c \otimes b. \end{cases} \tag{2.12}$$

From the preceding iteration scheme, Bai and Benzi [2], proposed the regularized Hermitian and skew-Hermitian splitting (RHSS) preconditioner as follows :

$$\frac{1}{2}\left[G_0 \otimes \left[\underbrace{\underbrace{\begin{pmatrix} \frac{1}{\alpha} I & 0 \\ 0 & (\alpha I + Q)^{-1} \end{pmatrix}}_{\mathcal{U}} \begin{pmatrix} \alpha I + A & 0 \\ 0 & (\alpha I + Q) \end{pmatrix} \begin{pmatrix} \alpha I & B^T \\ -B & (\alpha I + Q) \end{pmatrix}}_{\mathcal{P}_{RHSS}}\right]\right]. \tag{2.13}$$

$\mathcal{P}_{RHSS}$ is the RHSS preconditioner. For more details see [2].

2.5 The generalized modified RHSS iteration method

In this section, motivated by the ideas of (2.12), we propose a new splitting called the generalized modified RHSS (GMRHSS) for the matrix $G_0 \otimes \mathcal{A}$ as follows :

$$G_0 \otimes \mathcal{A} = G_0 \otimes (\Omega + \mathcal{H}_+) - G_0 \otimes (\Omega - \mathcal{S}_-) = G_0 \otimes (\Omega + \mathcal{S}_+) - G_0 \otimes (\Omega - \mathcal{H}_-), \tag{2.14}$$

where $\mathcal{H}_+$, $\mathcal{S}_+$ and $\mathcal{H}_-$, $\mathcal{S}_-$ are defined as in (2.11) and

$$\Omega = \begin{pmatrix} \alpha I & 0 \\ 0 & \beta\Lambda \end{pmatrix}, \tag{2.15}$$

with α, $\beta > 0$ and Λ is symmetric positive definite matrix. Using the same strategy utilized in (2.12), we propose the following splitting iteration method :

$$\begin{cases} (G_0 \otimes (\Omega + H_+)) \left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix} \right)^{\left(k+\frac{1}{2}\right)} = (G_0 \otimes (\Omega - S_-)) \left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix} \right)^{(k)} + c \otimes b, \\ (G_0 \otimes (\Omega + S_+)) \left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix} \right)^{(k+1)} = (G_0 \otimes (\Omega - H_-)) \left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix} \right)^{\left(k+\frac{1}{2}\right)} + c \otimes b. \end{cases}$$

Note that the above iteration method can be obtained from the matrix splitting

$$G_0 \otimes \mathcal{A} = G_0 \otimes \mathcal{M} - G_0 \otimes \mathcal{N},$$

where $\mathcal{M} = \frac{1}{2}\mathcal{U}(\Omega + \mathcal{H}_+)(\Omega + \mathcal{S}_+)$ and $\mathcal{N} = \frac{1}{2}\mathcal{U}(\Omega + \mathcal{H}_-)(\Omega + \mathcal{S}_-)$. After an easy handling, we have

$$\mathcal{M} = G_0 \otimes \frac{1}{2}\begin{pmatrix} \frac{1}{\alpha}I & 0 \\ 0 & (\beta\Lambda + Q)^{-1} \end{pmatrix}\begin{pmatrix} \alpha I + A & 0 \\ 0 & \beta\Lambda + Q \end{pmatrix}\begin{pmatrix} \alpha I & B^T \\ -B & \beta\Lambda + Q \end{pmatrix}. \quad (2.16)$$

To obtain a better approximation of the matrix $G_0 \otimes \mathcal{A}$. We replace the block $\alpha I + A$ by A and change the factor $\frac{1}{2}$ to 1, then a new preconditioner which is referred to as the GMRHSS preconditioner is given as follows :

$$\mathcal{P} = G_0 \otimes \underbrace{\begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix}}_{\mathcal{P}_{GMRHSS}}, \quad (2.17)$$

where $\alpha > 0$ and $\beta \geq 0$ are two constants. Then we can split the matrix $G_0 \otimes \mathcal{A}$ as follow :

$$G_0 \otimes \mathcal{A} = \mathcal{P} - \mathcal{R} = G_0 \otimes \begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix} - G_0 \otimes \begin{pmatrix} 0 & \frac{1}{\alpha}AB^T - B^T \\ 0 & \beta\Lambda + Q \end{pmatrix}.$$

Based on the above splitting, we propose a new iterative method, called the GMRHSS iteration method.

The GMRHSS iteration method : Let $\alpha > 0$ and $\beta \geq 0$ be two given constants. Given an initial guess $(x^{(0)^T}, y^{(0)^T})^T$. For $k = 0, 1, 2, ...,$ until $(x^{(k)^T}, y^{(k)^T})^T$ converges, compute

$$\left(G_0 \otimes \begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix}\right)\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k+1)} = \left(G_0 \otimes \begin{pmatrix} 0 & \frac{1}{\alpha}AB^T - B^T \\ 0 & \beta\Lambda + Q \end{pmatrix}\right)\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k)} + c \otimes b,$$

which can be rewritten as the follow

$$\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k+1)} = \mathcal{T}(\alpha, \beta)\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)^{(k)} + \left[G_0^{-1} \otimes \begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix}^{-1}\right](c \otimes b),$$

where

$$\mathcal{T}(\alpha, \beta) = \left[I \otimes \left[\begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix}^{-1}\begin{pmatrix} 0 & \frac{1}{\alpha}AB^T - B^T \\ 0 & \beta\Lambda + Q \end{pmatrix}\right]\right], \quad (2.18)$$

is the iteration matrix of the GMRHSS iteration method.

2.5.1 Convergence of the GMRHSS iteration method

In this section we analyse the convergence properties of the (GMRHSS) iteration method, by studying the spectral properties of the iteration matrix $\mathcal{T}$.

Proposition 4

The preconditioner (2.17) has the following factorization

$$\mathcal{P} = G_0 \otimes \begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix} = G_0 \otimes \left[\begin{pmatrix} I & 0 \\ -BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & \tilde{A} \end{pmatrix} \begin{pmatrix} I & \frac{1}{\alpha}B^T \\ 0 & I \end{pmatrix} \right],$$

where $\tilde{A} = \left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)$. Then the inverse of the preconditioner $\mathcal{P}$ is given by the following equality

$$G_0^{-1} \otimes \begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix}^{-1} = G_0^{-1} \otimes \left[\begin{pmatrix} I & -\frac{1}{\alpha}B^T \\ 0 & I \end{pmatrix} \begin{pmatrix} A^{-1} & 0 \\ 0 & \tilde{A}^{-1} \end{pmatrix} \begin{pmatrix} I & 0 \\ BA^{-1} & I \end{pmatrix} \right].$$

Theorem 5

If λ is an eigenvalue of the preconditioned matrix $\mathcal{P}^{-1}_{GMRHSS}\mathcal{A}$ and $\begin{pmatrix} u \\ v \end{pmatrix}$ be the corresponding eigenvector. Then λ is an eigenvalues of the preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ and $\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)$ the corresponding eigenvector.

Proof : Let λ be an eigenvalue of the preconditioned matrix $\mathcal{P}^{-1}_{GMRHSS}\mathcal{A}$ and $\begin{pmatrix} u \\ v \end{pmatrix} \in \mathbb{R}^{n+m}$ be the corresponding eigenvector. Invoking the relation (2.5), we obtain

$$\left(I \otimes \mathcal{P}^{-1}_{GMRHSS}\mathcal{A}\right)\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right) - \lambda\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right) = z \otimes \left[\mathcal{P}^{-1}_{GMRHSS}\mathcal{A}\begin{pmatrix} u \\ v \end{pmatrix} - \lambda\begin{pmatrix} u \\ v \end{pmatrix}\right]. \tag{2.19}$$

Then from the relation (2.19) and the fact that λ is an eigenvalue of the preconditioned matrix $\mathcal{P}^{-1}_{GMRHSS}\mathcal{A}$, it follows that λ is an eigenvalue of the preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ and $\left(z \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right)$ the corresponding eigenvector, which shows the result. □

Theorem 6

If $\begin{pmatrix} u_i \\ v_i \end{pmatrix} \in \mathbb{R}^{n+m}$, $(i = 1,..,k)$ are linear independent vectors and $z \neq 0$, then

$$\left(z \otimes \begin{pmatrix} u_i \\ v_i \end{pmatrix}\right),$$

are linear independent vectors.

Proof : Let $a = (a_1, a_2, a_3, .., a_k) \in \mathbb{R}^k$. Then we need to prove

$$\left[z \otimes \begin{pmatrix} u_1 \\ v_1 \end{pmatrix} \quad \cdots \quad z \otimes \begin{pmatrix} u_k \\ v_k \end{pmatrix} \right] \begin{pmatrix} a_1 \\ \vdots \\ a_k \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}, \tag{2.20}$$

holds if and only if a is a zero vector. (2.20) can be equivalently written as

$$\begin{cases} a_1 z_1 u_1 + \cdots + a_k z_1 u_1 = 0, \\ \vdots \\ a_1 z_s v_1 + \cdots + a_k z_s v_k = 0. \end{cases}$$

Since $z \neq 0$ and using the fact that $\begin{pmatrix} u_i \\ v_i \end{pmatrix}$, $(i = 1,..,k)$ are linear independent vectors, then we have $a_i = 0$ for all $i = 1,..,k$, which shows the result. □

Theorem 7

Let $A \in \mathbb{R}^{n\times n}$ be a positive definite matrix and $B \in \mathbb{R}^{m\times n}$ be of full row rank, then the GMRHSS iteration method is convergent if and only if the parameters α and β satisfy

$$\alpha > 0, \quad \beta > \max\left\{\frac{1}{\mu}\left(\frac{c}{2} - q - \frac{1}{\alpha}e\right), 0\right\}.$$

Where $e = \frac{x^T B^T B x}{x^T x} > 0$, $q = \frac{x^T Q x}{x^T x} \geq 0$, $c = \frac{x^T B^T A^{-1} B x}{x^T x} > 0$ and $\mu = \frac{x^T \Lambda x}{x^T x} \geq 0$. x is an eigenvector corresponding to an eigenvalue of the matrix $K_2 = \tilde{A}^{-1}\left(\frac{1}{\alpha}B^T B - B^T A^{-1} B + \beta\Lambda + Q\right)$.

Proof : The iteration matrix (2.18) can be rewritten as follow

$$I \otimes \left[\begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix}^{-1} \begin{pmatrix} 0 & \frac{1}{\alpha}AB^T - B^T \\ 0 & \beta\Lambda + Q \end{pmatrix}\right] = I \otimes \begin{pmatrix} 0 & K_1 \\ 0 & K_2 \end{pmatrix}, \tag{2.21}$$

where $K_1 = \frac{1}{\alpha}B^T - A^{-1}B^T - \frac{1}{\alpha}B^T K_2$, $K_2 = \tilde{A}^{-1}\left(\frac{1}{\alpha}B^T B - B^T A^{-1} B + \beta\Lambda + Q\right)$ and $\tilde{A} = \left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)$. If λ is an eigenvalue of the matrix $\mathcal{T}(\alpha,\beta)$ which is defined as in (2.21), then $\lambda = 0$ or λ is an eigenvalue of K_2, which mean there exist a vector $x \neq 0$ satisfies the following equality

$$K_2 x = \tilde{A}^{-1}\left(\frac{1}{\alpha}B^T B - B^T A^{-1} B + \beta\Lambda + Q\right)x = \lambda x, \tag{2.22}$$

which can be written as

$$\left(\frac{1}{\alpha}B^T B - B^T A^{-1} B + \beta\Lambda + Q\right)x = \lambda\left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)x. \tag{2.23}$$

As we know the vector $x \neq 0$, premultiplying (2.23) from both sides by $\frac{x^T}{x^T x}$ lead to

$$\lambda = \frac{\beta\mu + q - c + \frac{1}{\alpha}e}{\beta\mu + q + \frac{1}{\alpha}e}. \tag{2.24}$$

And therefore the convergence of GMRHSS iteration method, it must satisfies the following inequality

$$|\lambda| = \frac{|\beta\mu + q - c + \frac{1}{\alpha}e|}{\beta\mu + q + \frac{1}{\alpha}e} < 1. \tag{2.25}$$

After some easy handling, we obtain that the parameters α and β satisfy

$$\alpha > 0, \ \ \beta > \max\left\{\frac{1}{\mu}\left(\frac{c}{2} - q - \frac{1}{\alpha}e\right), 0\right\}.$$

Conversely, if such α and β are chosen then this guarantees that $|\lambda| < 1$. □

2.6 Spectral analysis of the GMRHSS preconditioned matrix

In this section, we will investigate the spectral properties of the preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$. We also determine the number of linearly independent eigenvectors of the preconditioned matrix.

Theorem 8

let λ be an eigenvalue of the preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ and $z \otimes \begin{pmatrix} u \\ v \end{pmatrix}$ being the corresponding eigenvector. Thus $\lambda = 1$ or $\lambda = \frac{e}{\beta\mu + q + \frac{1}{\alpha}c}$ with
$c = \frac{u^T BB^T u}{u^T u}$, $e = \frac{u^T B^T A^{-1} Bu}{u^T u}$ and $q = \frac{u^T Qu}{u^T u}$.
If Q is the mass matrix and if ($\beta \longrightarrow 0$ and $\alpha \longrightarrow \infty$) then

$$\lambda \in [\gamma^2, \Gamma^2].$$

Where γ is the (generalized) inf-sup constant and Γ is a boundedness constant.

Proof : The preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ can be rewritten as follow

$$\mathcal{P}^{-1}(G_0 \otimes \mathcal{A}) = I \otimes \underbrace{\begin{pmatrix} I & M_1 \\ 0 & M_2 \end{pmatrix}}_{\mathcal{P}^{-1}_{GMRHSS}\mathcal{A}}, \tag{2.26}$$

where $\tilde{A} = \left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)$, $M_1 = \left(I - \frac{1}{\alpha}B^T\tilde{A}^{-1}B\right)A^{-1}B^T$ and $M_2 = \tilde{A}^{-1}BA^{-1}B^T$. Invoking Theorem 7.2 and using (2.26), the eigenvalues of $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ are $\lambda = 1$ or λ is an eigenvalue of M_2, which mean there exist a vector $(x \neq 0) \in \mathbb{R}^m$ satisfies the following equality

$$M_2 x = \left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)^{-1} BA^{-1}B^T x = \lambda x, \tag{2.27}$$

which can be written as

$$BA^{-1}B^T x = \lambda\left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)x. \tag{2.28}$$

As we know that the vector $x \neq 0$. Premultiplying (2.28) from the left by $\frac{x^T}{x^T x}$ yields

$$\lambda = \frac{e}{\beta\mu + q + \frac{1}{\alpha}c}. \tag{2.29}$$

Invoking (2.29) and using the fact that if $\beta \longrightarrow 0$ and $\alpha \longrightarrow \infty$, the discrete inf-sup stability and boundedness imply $\gamma^2 \leq \lambda \leq \Gamma^2$, see for example [26]. Which shows the result. $\square$

Theorem 9

Let the preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ be defined as in (2.26). Then the preconditioned matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ has $m_1 + 1$ distincts eigenvalues $\{1, \lambda_1, \ldots, \lambda_{m_1}\}$ and $(n+m)$ linearly independent eigenvectors, where $1 \leq m_1 \leq m$. Moreover.

1. 1 is an eigenvalue with multiplicity n.
2. $\lambda_i \neq 1$, for $i = 1, \ldots, m_1$.
3. The matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ is diagonalizable.

Proof : Let λ be an eigenvalue of the preconditioned matrix $\mathcal{P}^{-1}_{GMRHSS}\mathcal{A}$ and $(x^T, y^T)^T$ be the corresponding eigenvector. To derive the eigenvector distribution, we consider the following generalized eigenvalue problem :

$$\underbrace{\begin{pmatrix} I & M_1 \\ 0 & M_2 \end{pmatrix}}_{\mathcal{P}_{GMRHSS}^{-1}\mathcal{A}} \begin{pmatrix} x \\ y \end{pmatrix} = \lambda \begin{pmatrix} x \\ y \end{pmatrix}. \tag{2.30}$$

Where $\tilde{A} = \left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)$, $M_1 = \left(I - \frac{1}{\alpha}B^T\tilde{A}^{-1}B\right)A^{-1}B^T$ and $M_2 = \tilde{A}^{-1}BA^{-1}B^T$. Equation (2.30) can be equivalently rewritten as

$$\begin{cases} (1-\lambda)x + \left(I - \dfrac{1}{\alpha}B^T\tilde{A}^{-1}B\right)A^{-1}B^Ty = 0, \\ BA^{-1}B^Ty = \lambda\tilde{A}y. \end{cases} \tag{2.31}$$

If $\lambda = 1$ holds true, then from the first equation of (2.31) we can easily get $\left(I - \frac{1}{\alpha}B^T\tilde{A}^{-1}B\right)A^{-1}B^Ty = 0$. When $y = 0$, equation (2.31) is always true for the case of $\lambda = 1$. Hence, there are n linearly independent eigenvectors $\begin{pmatrix} u_i \\ 0 \end{pmatrix}$ $(i = 1,..,n)$, corresponding to the eigenvalue 1, where u_i are arbitrary linearly independent vectors.

If $\lambda \neq 1$ and $y = 0$, then we obtain from the first equation of (2.31) that $x = 0$. This contradicts to the assumption that $(x^T, y^T)^T$ is an eigenvector of the preconditioned matrix $\mathcal{P}_{GMRHSS}^{-1}\mathcal{A}$. Hence $y \neq 0$, if y satisfies the second equation of (2.31), then

$$BA^{-1}B^Ty = \lambda\tilde{A}y. \tag{2.32}$$

We know that $\tilde{A}$ is a positive definite matrix, then $\tilde{A}$ admits a Cholesky decomposition $\tilde{A} = RR^T$. Premultiplying (2.32) with R^{-1} results in

$$R^{-1}BA^{-1}B^TR^{-T}z = \lambda z \;\text{ with }\; z = R^Ty\;. \tag{2.33}$$

Since $R^{-1}BA^{-1}B^TR^{-T}$ is a positive definite matrix, then there are m linearly independent eigenvectors of the form $w_i = R^{-T}z_i$ $(i = 1,..,m)$, where z_i are orthogonal eigenvectors of $R^{-1}BA^{-1}B^TR^{-T}$.
U, V and W are matrices whose columns are $(u_1,..,u_n)$, $(v_1,..,v_m)$ and $(w_1,..,w_m)$, respectively, where v_i $(i = 1,..,m)$ are given by the following equation

$$v_i = \frac{1}{(\lambda-1)}\left(I - \frac{1}{\alpha}B^T\tilde{A}^{-1}B\right)A^{-1}B^Tw_i.$$

Since U and W are nonsingular matrices, we infer that $\det\begin{pmatrix} U & V \\ O & W \end{pmatrix} = \det(U)\det(W) \neq 0$ and $\mathcal{P}_{GMRHSS}^{-1}\mathcal{A}$ is diagonalizable. Then the spectral decomposition of $\mathcal{P}_{GMRHSS}^{-1}\mathcal{A}$ is given by $\mathcal{P}_{GMRHSS}^{-1}\mathcal{A} = P\Lambda P^{-1}$, where $\Lambda = \text{diag}(1, \lambda_1.., \lambda_{m_1})$ and $1 \leq m_1 \leq m$. It is left to show that $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ is diagonalizable. By making use of (2.6) and (2.7), we get the following equalities

$$\begin{aligned} \mathcal{P}^{-1}(G_0 \otimes \mathcal{A}) &= I \otimes \mathcal{P}_{GMRHSS}^{-1}\mathcal{A}, \\ &= I \otimes (P\Lambda P^{-1}), \\ &= (I \otimes P)(I \otimes \Lambda)(I \otimes P^{-1}), \\ &= (I \otimes P)(I \otimes \Lambda)(I \otimes P)^{-1}. \end{aligned} \tag{2.34}$$

Consequently, the equality (2.34) implies that the matrix $\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})$ is diagonalizable.□

2.7 Implementation of GMRHSS

In our implementation, we need to solve the following preconditioned saddle point problem :

$$\underbrace{(I \otimes \mathcal{P}_{GMRHSS}^{-1}\mathcal{A})}_{\mathcal{P}^{-1}(G_0 \otimes \mathcal{A})}\left(\tilde{z} \otimes \begin{pmatrix} u \\ v \end{pmatrix}\right) = G_0^{-1}c \otimes \mathcal{P}_{GMRHSS}^{-1}b, \tag{2.35}$$

We will solve the two following systems independently :

$$\begin{cases} G_0\tilde{z} = c, \\ \mathcal{P}_{GMRHSS}^{-1}\mathcal{A}\begin{pmatrix} u \\ v \end{pmatrix} = \mathcal{P}_{GMRHSS}^{-1}b. \end{cases} \tag{2.36}$$

For solving the first system of (2.36), we use the restarted version of the GMRES (denoted by GMRES(k)) or the preconditioned GMRES(k), with k is the number of restart. We use GMRES incorporated with the preconditioner $\mathcal{P}_{GMRHSS}$ to solve the second system of (2.36). At each step of applying the preconditioner $\mathcal{P}_{GMRHSS}$ inside the GMRES(k) algorithm, we need to compute a vector of the form

$$\begin{pmatrix} A & \frac{1}{\alpha}AB^T \\ -B & \beta\Lambda + Q \end{pmatrix} z = r. \tag{2.37}$$

$$\begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} I & 0 \\ -BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & \tilde{A} \end{pmatrix} \begin{pmatrix} I & \frac{1}{\alpha}B^T \\ 0 & I \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}. \tag{2.38}$$

Then, the following algorithmic version of the GMRHSS iteration method can be defined as follow :

Algorithme 7 : Implementation of GMRHSS preconditioner

```
1 : Solve $At = r_1$ ;
2 : Solve $\left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)z_2 = r_2 + Bt$ ;
3 : Compute $z_1 = t - \frac{1}{\alpha}Bz_2$ ;
```

The matrices A and $\left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)$ are symmetric positive definite. Therefore, we can solve the systems with the coefficient matrices A and $\left(\beta\Lambda + Q + \frac{1}{\alpha}BB^T\right)$ by the preconditioned CG method or by the preconditioned GMRES method.

Numerical results

Sommaire

Numerical results

In this section we present some numerical examples of Stokes problem (Lid driven cavity) discretized by the finite element $(Q_2 - P_1)$ [26] . The software IFISS [25] provides the matrices Ast, Bst and G for the matrices A, B and G_0. The matrix Bst has a rank difficent, thus we dorp two first rows of Bst to obtain a full rank matrix. We take $Q = \text{diag}(BB^T)$, where $\text{diag}(BB^T)$ denote the diagonal part of the matrix BB^T. $+$ means that the method has not converged after 500 iterations. The numerical examples show the effectiveness of the GMRHSS preconditioned GMRES method, for solving the non symmetric saddle point problem (2.1) over the HSS and RHSS preconditioned GMRES methods, in terms of the iterations and CPU times. We have also tested the HSS, RHSS, and GMRHSS iteration methods, but their convergence are very slowly and that's why we no longer show their numerical results. All numerical procedures are performed in Matlab 2016 on a computer with Intel(R) Core (TM) i5 4300U CPU 2.49 GHZ, 8 GB Computer RAM, and Windows 10 operating system. In all the tables "IT" stands for the number of iterations and "CPU" denotes the elapsed CPU time of the convergence. In the numerical test, the preconditioners used for solving (2.1) are $\mathcal{P}_{GMRHSS}$, $\mathcal{P}_{HSS}$ [6] and $\mathcal{P}_{RHSS}$ [2] . We choose $\Lambda = I$, because this case help the preconditioner $\mathcal{P}_{GMRHSS}$ to make a better performance more than $\Lambda = Q$ and Λ equal to the mass matrix [25] . When using PGMRES for solving the first system of (2.36), the preconditioner used is a drop tolerance-based incomplete LU factorization computed using the matlab function

$$\text{ilu}(G_0, \text{setup}),$$

setup.type = ’crout’,
setup.milu = ’row’,
setup.droptol = 0.1.
In all the tests, the initial vector $x^{(0)}$ is set to be a zero vector and the right-hand side vector b, is chosen such that the exact solution of the saddle point problem (2.1) is a vector of all ones. The iterations are terminated as soon as the current iterate $z^{(k)} \otimes X^{(k)}$ satisfies

$$\frac{||c - G_0 z^{(k)}||}{||c||} < 10^{-12} \text{ and } \frac{||b - \mathcal{A}X^{(k)}||}{||b||} < 10^{-12}.$$

Table 1 : The size of the matrices A and B for lid driven cavity on $2^r \times 2^r$.

Lid driven cavity				
r	n	m	size of A	size of B
4	578	192	578× 578	578× 192
5	2178	768	2178 ×2178	2178 × 766
6	8450	3070	8450×8450	8450× 3070

We start by presenting the numerical results for the first system of (2.36) :

$$G_0 \tilde{z} \quad = c,$$

Table 2 : Numerical results for the preconditioned GMRES.

	$r = 4$	$r = 5$	$r = 6$
IT	2	4	8
CPU	0.001	0.009	0.075
Res	1.24e-14	7.72e-14	2.71e-12
Err	1.24e-14	7.72e-14	2.71e-12

Numerical results for the second system of (2.36) :

$$\mathcal{P}^{-1}_{GMRHSS} \mathcal{A}X \quad = \mathcal{P}^{-1}_{GMRHSS} b,$$

Table 3 : Numerical results for the preconditioned GMRES with $r = 5$.

	$\beta = 10^{-4}$		$\beta = 10^{-2}$		$\beta = 1$		$\beta = 10^{2}$	
$\alpha = 10^{-2}$	IT	80	IT	80	IT	60	IT	48
	CPU	1.50	CPU	1.43	CPU	0.90	CPU	0.52
	RES	2.92e-06	RES	2.18e-06	RES	2.10e-06	RES	4.40e-07
	ERR	3.93e-04	ERR	9.08e-05	ERR	3.49e-05	ERR	1.37e-04
$\alpha = 1$	IT	42	IT	41	IT	47	IT	47
	CPU	0.50	CPU	0.46	CPU	0.51	CPU	0.43
	RES	5.61e-07	RES	6.68e-07	RES	8.15e-07	RES	1.05e-06
	ERR	9.75e-05	ERR	2.40e-04	ERR	1.23e-04	ERR	1.06e-05
$\alpha = 10^{2}$	IT	35	IT	37	IT	45	IT	47
	CPU	0.35	CPU	0.37	CPU	0.42	CPU	0.44
	RES	3.51e-07	RES	4.50e-07	RES	7.98e-07	RES	1.05e-06
	ERR	6.32e-06	ERR	3.81e-06	ERR	7.94e-06	ERR	4.92e-05
$\alpha = 10^{4}$	IT	35	IT	37	IT	45	IT	47
	CPU	0.32	CPU	0.34	CPU	0.43	CPU	0.33
	RES	3.51e-07	RES	4.52e-07	RES	7.51e-07	RES	1.05e-06
	ERR	3.95e-06	ERR	3.44e-06	ERR	4.47e-05	ERR	1.03e-05

In Tables 3, we list the results of the GMRHSS preconditioned GMRES method in terms of IT and CPU. From these results we conclude that the GMRHSS preconditioned GMRES, become very efficent when β decrease and α increase.

Table 4 : Numerical results for the preconditioned GMRES with $r = 5$.

	$\beta = 10^{-1}$		$\beta = 1$		$\beta = 10$		$\beta = 10^{2}$	
$\alpha = 10^{-1}$	IT	63	IT	44	IT	38	IT	36
	CPU	0.24	CPU	0.12	CPU	0.09	CPU	0.12
	RES	2.64e-10	RES	2.64e-10	RES	5.85e-10	RES	4.45e-09
	ERR	1.58e-08	ERR	1.58e-08	ERR	1.31e-07	ERR	4.06e-05
$\alpha = 1$	IT	43	IT	39	IT	38	IT	36
	CPU	0.12	CPU	0.10	CPU	0.11	CPU	0.10
	RES	2.31e-11	RES	3.81e-11	RES	3.65e-10	RES	4.46e-09
	ERR	1.72e-09	ERR	8.46e-09	ERR	9.78e-08	ERR	1.46e-05
$\alpha = 10$	IT	40	IT	39	IT	38	IT	36
	CPU	0.10	CPU	0.37	CPU	0.09	CPU	0.09
	RES	5.14e-12	RES	2.49e-11	RES	3.61e-10	RES	4.46e-09
	ERR	1.12e-09	ERR	6.37e-09	ERR	9.66e-08	ERR	1.46e-06
$\alpha = 10^{2}$	IT	41	IT	39	IT	38	IT	36
	CPU	0.09	CPU	0.13	CPU	0.09	CPU	0.08
	RES	5.04e-12	RES	2.68e-11	RES	3.62e-10	RES	4.46e-09
	ERR	1.09e-09	ERR	6.75e-09	ERR	9.69e-08	ERR	1.46e-06

In the table 4, we use the sparse Cholesky factorization with SYMAMD reordering, for solving linear systems with coefficient A and $\beta\Lambda + Q + \frac{1}{\alpha}BB^T$ see Algorithm 10. We present the numerical results of the error norm for the following problem

$$(G_0 \otimes \mathcal{A})(z \otimes X) \quad = c \otimes b.$$

Table 5 : Numerical results for the preconditioned GMRES with $r = 5$.

	$\beta = 10^{-1}$	$\beta = 1$	$\beta = 10$	$\beta = 100$
$\alpha = 10^{-1}$	Err 4.38e-07	Err 3.02e-07	Err 3.62e-06	Err 4.06e-05
$\alpha = 1$	Err 4.78e-08	Err 2.34e-07	Err 2.70e-06	Err 4.05e-05
$\alpha = 10$	Err 3.12e-08	Err 1.76e-07	Err 2.67e-08	Err 4.05e-05
$\alpha = 100$	Err 3.02e-08	Err 1.86e-07	Err 2.68e-06	Err 4.05e-05

Table 6 : Numerical results for the three preconditioned GMRES with $r = 5$.

α		$\mathcal{P}_{GMRHSS}$		$\mathcal{P}_{HSS}$		$\mathcal{P}_{RHSS}$
10^{-2}	IT	78	IT	119	IT	124
	CPU	1.07	CPU	1.80	CPU	1.61
	RES	2.00e-09	RES	7.13e-08	RES	5.23e-08
	ERR	5.96e-07	ERR	1.69e-06	ERR	7.92e-07
1	IT	42	IT	336	IT	337
	CPU	0.26	CPU	2.49	CPU	2.43
	RES	3.15e-10	RES	1.68e-11	RES	1.54e-11
	ERR	1.64e-06	ERR	5.67e-08	ERR	5.12e-08
100	IT	44	IT	†	IT	†
	CPU	0.21	CPU	†	CPU	†
	RES	2.43e-11	RES	†	RES	†
	ERR	6.14e-09	ERR	†	ERR	†

The preconditioned GMRES methods incorporated with the $\mathcal{P}_{GMRHSS}$, $\mathcal{P}_{HSS}$ and $\mathcal{P}_{RHSS}$ preconditioners are listed in Tables 6 for different values of α. From numerical results listed in Tables, we can conclude that the $\mathcal{P}_{GMRHSS}$ preconditioned GMRES method requires less iterations and has faster CPU times than $\mathcal{P}_{HSS}$ and $\mathcal{P}_{RHSS}$ in all trials. It also give solutions for $\alpha = 100$ when the other two methods fails to converge. In each case when all three methods produce solutions, $\mathcal{P}_{GMRHSS}$, gives smaller relative residual and error than $\mathcal{P}_{HSS}$ and $\mathcal{P}_{RHSS}$.

Conclusion

We have presented a separation of solution approach, as a new strategy to solve the saddle point problem (2.1). In addition, we have introduced and investigated the preconditioner $\mathcal{P}_{GMRHSS}$. The tables 3 and 4, illustrate that the $\mathcal{P}_{GMRHSS}$ preconditioned GMRES method with suitable choices of the parameters α and β, has better performance than the $\mathcal{P}_{HSS}$ and $\mathcal{P}_{RHSS}$ preconditioned GMRES methods, in terms of the iterations and CPU times. We conclude that our preconditioned Krylov subspace method is very powerful for solving the nonsymmetric saddle point problems (2.1).

Annexe

MATLAB program of the preconditioned GMRES method for solving (2.9)

```
   %% Element fini Q2-P1
     m=size(Q,1);
     n=size(A,1);
     BB=Bst(3:m, 1:n);
     [l,mm]=size(BB);
  %% Choose alpha
   alpha=1e+6;
   beta=1e-4;
   %% Choose type of the problem
   epsilon=-1;
  %% Create artificial rhs by choosing the exact solution
  %% Fix the right handside number
  nrhs=1;
  %% Choose the exact solution
  Exact_Sol=ones(n+l,nrhs);
  x0=zeros(n+l,nrhs);
  %% Compute the right hand side
  rhs=AAx(epsilon,Ast,BB,Exact_Sol);
  I=speye(l,l);
  Q1=Q(1:l,1:l);
%% ------------ paramaters for KRYLOV SUBSPACE method
%% im=200; maxits=2000; tolIts=1e-12; tol0=10^-8;
im=200; maxits=400; tolIts=1e-12; tol0=10^-8;

QQ=diag(diag(BB*BB'));
DD=beta*I+QQ+(1/alpha)*BB*BB';
%% Preconditioner of system intern of  PGMRHSS
L=ichol(Ast,struct('type','ict','droptol',1e-2));
U=ichol(DD,struct('type','ict','droptol',1e-2));

```

```
setup.type = 'crout';
setup.milu = 'row';
setup.droptol = 1e-2;
[L1,U1]=ilu(G,setup);
rhss=G*ones(n,1);

tic
[sol1,ro,its1]=PGMRES(G,rhss,L1,U1,zeros(n,1),im,maxits,1e
   -12);
%% Left Preconditioning PGMRHSS
[sol2,res,its2]=PGlbGMRES5(epsilon,Ast,BB,QQ,L,U,alpha,
   beta,rhs,x0,im,maxits,tolIts);
toc


Res=norm(rhs-AAx(epsilon,Ast,BB,sol),'fro');
fprintf('\n Iteration  =%3g     ',its2);
fprintf('\n Res =%12.4e       ',Res);
fprintf('\n Err    =%12.4e      ', norm(sol2-Exact_Sol,'fro'
   ));
```

```
function y=Precfunctr(A,B,QQ,L,G,alpha,beta,x)

%--------------------------------------------------
%  Preconditioner matrix times vector x
% A , B  = matrices from the Saddle system
%  QQ : diagonal matrix
%
%  (A    B^T)
%  (        ) x= y
%  (-B    0 )
%
%   im      = krylov subspace dimension
%   rhs     = right-hand side
%   x0      = initial guess
%   tol     = tolerance for stopping criterioncl
%   maxits = max number of matrix - vector products
%           allowed (= max tot. # of global gmres steps)
%--------------------------------------------------
     [m,n]=size(B); alphainv=1/alpha;
     [t] = Gcg03(A,L,x(1:n,:)) ;
     y2=Gcg010(B,QQ,beta,alphainv,G,x(n+1:n+m,:)+B*t);
     y1=t-alphainv*B'*y2;
     y=[y1;y2];
end
```

```
function [x,its] = Gcg03(A,L,rhs)
%
%% solves by using Global CG the linear system :
%
%(A + beta  B' *  QQ^(-1) *  B ) * x = rhs
%
[n,m]=size(rhs);
x0=zeros(n,m);
maxit=n;tol=1e-6;
%%------------------------------------------
OutputG=0;
 n = size(A,1);
 x = x0;
 r = rhs - A  * x ;
 z = L'\(L\r);
 p = z ;
 ro1 = z(:)' * r(:);
 tol1 = tol*tol*ro1;
%%
 its = 0 ;
 while (its < maxit && ro1 > tol1)
         its = its+1;
     res(its)=ro1;
         ro = ro1;
         ap =  A  * p  ;
         alp = ro / ( ap(:)'* p(:)) ;
         x = x + alp * p ;
         r = r - alp * ap;
     z =L'\(L\ r);
         ro1= z(:)'*r(:);
         bet = ro1 / ro ;

         p = z + bet * p;
      if(OutputG)
     fprintf(' %4.0f %22.14e \n', its,sqrt(ro1))
     end

 end


 end
```

```
function [x,its] = Gcg010(B,QQ,beta,alpha,L,rhs)

%% solves by using Global CG the linear system :
%% (A + beta  B' *  QQ^(-1) *  B ) * x = rhs
[n,m]=size(rhs);
x0=zeros(n,m);
maxit=n;tol=1e-8;
I=speye(m,m);
%% ------------------------------------------
OutputG=1;
%% n = size(A,1);
 x = x0;
 r = rhs - (beta* x + QQ*x+B*B'*(alpha*x) );
 z = L'\(L\r);
 p = z ;
 ro1 = z(:)' * r(:);
 tol1 = tol*tol*ro1;
%%
 its = 0 ;
 while (its < maxit && ro1 > tol1)
         its = its+1;
     res(its)=ro1;
         ro = ro1;
        ap=beta * p + QQ*p+B*B'*(alpha*p);
         alp = ro / ( ap(:)'* p(:)) ;
         x = x + alp * p ;
         r = r - alp * ap;
         z =L'\(L\ r);
         ro1= z(:)'*r(:);
         bet = ro1 / ro ;
         p = z + bet * p;
      if(OutputG)
     fprintf(' %4.0f %22.14e \n', its,sqrt(ro1))
     end

 end


 end
```

```
function [sol,res,its]=PGlbGMRES5(epsilon,A,B,Q,L,G,alpha,
   beta,rhs,x0,im,maxits,tol);
%-----------------------------------------------------
% [res,Res,sol] =PGlbGMRES(A,B,Q,alpha,rhs,x0,im,maxits,
   tolIts)
% restarted Preconditioned Global gmres with Krylov
   subspace of dim = im.
% A , B  = matrices from the Saddle system
%  Q is a diagonal matrix introduced for the
   preconditioner
%
%      (A   B^T)^(-1)      (A   B^T)       (A   B^T)^(-1)
%      (        )        * (        ) * X=  (        )
   *  Rhs
%      (-B   ?Q)            (-B    0 )      (-B   ?Q)
%   im     = krylov subspace dimension
%   rhs    = right-hand side
%   x0     = initial guess
%   tol    = tolerance for stopping criterion
%   maxits = max number of matrix - vector products
%          allowed (= max tot. # of global gmres steps)
%-----------------------------------------------------
      tolmac = eps;
      its = 1 ;
      sol = x0;
      outputG=0;
      [m,n]=size(B);
  while (its < maxits)
      r=rhs-AAx(epsilon,A,B,sol);
      vv{1}=Precfunctr(A,B,Q,L,G,alpha,beta,r);
      ro = norm(vv{1},'fro');
      res(its) = ro;
      if (its  == 1)
          tol1=tol*ro ;
      end
      if (ro < tol1 | its > maxits)
```

```
        return
      end
      t = 1.0/ ro ;
      vv{1} = vv{1} * t ;
%% ---initialize 1-st term of rhs of hess.system..
       rs(1) = ro ;
       i = 0 ;
 while (i < im & (ro > tol1)& its < maxits)
      %% print its/residual info
      if (outputG)
      fprintf(1,' its %d  res %e \n',its,ro)
      end
     i=i+1 ;
     its = its + 1 ;
     i1 = i + 1 ;
     z = AAx(epsilon,A,B,vv{i});
     vv{i1}=Precfunctr(A,B,Q,L,G,alpha,beta,z);

%---------- modified GS ;
      for j=1:i
       t = (vv{j}(:))'*(vv{i1}(:)) ;
       hh(j,i) = t ;
           vv{i1} =       vv{i1} - t*vv{j} ;
      end
      t = norm(vv{i1},'fro') ;
      hh(i1,i) = t ;
      if (t  ~= 0.0)
        t = 1.0 / t ;
        vv{i1} = vv{i1}*t ;
      end
%%
    if (i ~= 1)
%---------- apply previous transformations on i-th column
   of h ;
       for k=2:i
         k1 = k-1 ;
         t = hh(k1,i) ;
         hh(k1,i) = c(k1)*t + s(k1)*hh(k,i) ;
         hh(k,i) = -s(k1)*t + c(k1)*hh(k,i) ;
       end
     end  %% IF
%%
     gam = sqrt(hh(i,i)^2 + hh(i1,i)^2) ;
     gam = max(tolmac,gam);
%---------- determine plane rotation & update rhs of LS Pb
     c(i) = hh(i,i)/gam ;
     s(i) = hh(i1,i)/gam ;
     rs(i1) = -s(i)*rs(i) ;
     rs(i) =  c(i)*rs(i) ;
%---------- determine res. norm. and test for convergence
     hh(i,i) = c(i)*hh(i,i) + s(i)*hh(i1,i) ;
     ro = abs(rs(i1)) ;
     res(its) = ro ;
   end
%---------- compute solution. solve upper triangular
   system
   rs(i) = rs(i)/hh(i,i) ;
   for  k=i-1:-1:1
      t=rs(k) ;
      for j=k+1:i
         t = t-hh(k,j)*rs(j) ;
      end
      rs(k) = t/hh(k,k) ;
   end
%---------- now form linear combination to get solution
   for j=1:i
      sol = sol +rs(j)* vv{j} ;
   end
    if ((ro  <=  tol1) | (its >= maxits)) return; end;
  end
end
```

```
function [sol,ro,its]=PGMRES(A,rhs,L,U,x0,im,maxits,tol);
%--------------------------------------------------
% [res,Res,sol] =PGlbGMRES(A,B,Q,alpha,rhs,x0,im,maxits,
   tolIts)
% restarted Preconditioned Global gmres with Krylov
   subspace of dim = im.
% A , B  = matrices from the Saddle system
%  Q is a diagonal matrix introduced for the
   preconditioner
```

```
%
%       (A    B^T)^(-1)        (A    B^T)         (A    B^T)^(-1)
%       (         )         *  (         ) * X=   (         )
   *  Rhs
%       (-B    ?Q)             (-B     0 )        (-B    ?Q)
%    im      = krylov subspace dimension
%    rhs     = right-hand side
%    x0      = initial guess
%    tol     = tolerance for stopping criterion
%    maxits = max number of matrix - vector products
%           allowed (= max tot. # of global gmres steps)
%-------------------------------------------------
      tolmac = eps;
      its = 1 ;
      sol = x0;
      outputG=0;
      [n,n]=size(A);

%%
  while (its < maxits)
      r=rhs-A*sol;
      vv{1}=U\(L\r);
      ro = norm(vv{1},'fro');
      res(its) = ro;
      if (its  == 1)
        tol1=tol*ro ;
      end
      if (ro < tol1 | its > maxits)
        return
      end
      t = 1.0/ ro ;
      vv{1} = vv{1} * t ;
%%---initialize 1-st term of rhs of hess.system..
      rs(1) = ro ;
      i = 0 ;
 while (i < im & (ro > tol1) & its < maxits)
      %% print its/residual info
      if (outputG)
         fprintf(1,' its %d  res %e \n',its,ro)
      end
      i=i+1 ;
      its = its + 1 ;
      i1 = i + 1 ;
      z=A*vv{i};
     vv{i1}=U\(L\z);
%---------- modified GS ;
      for j=1:i

         t = (vv{j}(:))'*(vv{i1}(:)) ;
         hh(j,i) = t ;
           vv{i1} =      vv{i1} - t*vv{j} ;
      end
      t = norm(vv{i1},'fro') ;
      hh(i1,i) = t ;
      if (t  ~= 0.0)
        t = 1.0 / t ;
        vv{i1} = vv{i1}*t ;
      end
%%
    if (i ~= 1)
%---------- apply previous transformations on i-th column
   of h ;
       for k=2:i
         k1 = k-1 ;
         t = hh(k1,i) ;
         hh(k1,i) = c(k1)*t + s(k1)*hh(k,i) ;
         hh(k,i) = -s(k1)*t + c(k1)*hh(k,i) ;
       end
     end  %% IF
%%
     gam = sqrt(hh(i,i)^2 + hh(i1,i)^2) ;
     gam = max(tolmac,gam);
%---------- determine plane rotation & update rhs of LS Pb
     c(i) = hh(i,i)/gam ;
     s(i) = hh(i1,i)/gam ;
     rs(i1) = -s(i)*rs(i) ;
     rs(i) =  c(i)*rs(i) ;
%---------- determine res. norm. and test for convergence
     hh(i,i) = c(i)*hh(i,i) + s(i)*hh(i1,i) ;
     ro = abs(rs(i1)) ;
     res(its) = ro ;
   end
%---------- compute solution. solve upper triangular
   system
   rs(i) = rs(i)/hh(i,i) ;
   for  k=i-1:-1:1
      t=rs(k) ;
      for j=k+1:i
         t = t-hh(k,j)*rs(j) ;
      end
      rs(k) = t/hh(k,k) ;
   end
%---------- now form linear combination to get solution
   for j=1:i
      sol = sol +rs(j)* vv{j} ;
   end
    if ((ro  <=  tol1) | (its >= maxits)) return; end;
 end
```

Quatrième partie

Regularized PGMRES and the regularized iteration method for solving saddle point problem

Chapitre 3

Regularized PGMRES and the regularized iteration method

Sommaire

Abstract :

In this chapter we propose a new preconditioner for solving the saddle point problem. The preconditioner is obtained by replacing the $(1,2)$ and $(2,2)$ blocks in the original saddle-point matrix $\mathcal{A}$ by another well chosen block. The proposed preconditioner can be used as a preconditioner corresponding to the stationary itearative method or to accelerate the convergence of the generalized minimal residual method (GMRES). The convergent and semi-convergent analyses of the regularized iteration method are presented. Meanwhile, we analyzed the eigenvalue-distribution and the forms of the eigenvectors of the preconditioned matrix. Finally, numerical results show the effectiveness of the proposed preconditioner as compared to other preconditioners.

Résumé :

Dans ce chapitre, nous proposons un nouveau préconditionneur pour résoudre le problème de point selle. Le préconditionneur est obtenu en remplaçant les blocs $(1,2)$ et $(2,2)$ de la matrice de point de selle originale $\mathcal{A}$ par des autres blocs bien choisis. Le préconditionneur proposé peut être utilisé comme préconditionneur correspondant à la méthode itérative stationnaire ou pour accélérer la convergence de la méthode GMRES. Les analyses de la convergente et la semi-convergentes de la méthode d'itération régularisée sont présentées. Parallèlement, nous avons analysé en détail la distribution des valeurs propres et les formes des vecteurs propres de la matrice préconditionnée. Enfin, les résultats numériques montrent l'efficacité du préconditionneur proposé par rapport aux autres préconditionneurs.

3.1 Introduction

In this paper, we are interesting of the following saddle-point form :

$$\mathcal{A}X = \begin{pmatrix} A & B \\ -B^T & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \underbrace{\begin{pmatrix} f \\ -g \end{pmatrix}}_{d}, \tag{3.1}$$

where $A \in \mathbb{R}^{n\times n}$ is a positive definite matrix and maybe not Hermitian, $B \in \mathbb{R}^{n\times m}$ $(m < n)$ maybe a rank deficient matrix with $\text{rank}(B) = r < m$, $f \in \mathbb{C}^n$ and $g \in \mathbb{C}^m$ are given vectors. This type of problems is important and arose in scientific and engineering application, for more details we recommend the readers to see [11, 15]. Since the matrices A and B in (3.1) are large and sparse, the solution of (3.1) is suited for being solved by the iterative methods. Many effective iteration methods have been developed for solving the saddle point problem such as, the Hermitian and skew-Hermitian splitting (HSS) method [6] and the Regularized HSS iteration method [2]. Also the Krylov subspace methods, which are introduced and studied in [1] are very efficent methods for solving the large linear systems of equations. However Krylov subspace methods without a good preconditioner converge very slowly when they are applied to the corresponding saddle point problem. In order to accelerate the convergence rate of the associated Krylov subspace, several preconditioners [6, 2, 10] are proposed for (3.1). The paper is organized as follows. In the section 2 we introduce the regularized iteration method and the regularized preconditioner $\mathcal{P}_r$. The convergence and semi convergence of the regularized iteration method will be analysed in subsection 2.1 and 2.2. In subsection 2.3, we gave the implementation of the regularized preconditioner. The numerical experiments is discussed in section 3.

3.2 New regularized iteration method

In this section, motivated by the idea of [30] , we develop a new splitting for the non symmetric saddle point matrix $\mathcal{A}$. We split the saddle-point matrix $\mathcal{A}$ as follows :

$$\mathcal{A} = \mathcal{P}_r - \mathcal{R} = \begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix} - \begin{pmatrix} 0 & B \\ 0 & \alpha Q \end{pmatrix}, \tag{3.2}$$

where $\alpha > 0$ is a constant and Q is is $m \times m$ symmetric positive definite matrix. Then, the new regularized iteration method based on the splitting (3.2) can be derived as follows : let $\alpha > 0$ be a given constant and $\left(x^{(0)^T}, y^{(0)^T}\right)^T$ be a given guess. For $k = 0, 1, 2, ..,$ until
$\left(x^{(k)^T}, y^{(k)^T}\right)^T$ converges, compute

$$\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix} \begin{pmatrix} x^{(k+1)} \\ y^{(k+1)} \end{pmatrix} = \begin{pmatrix} 0 & B \\ 0 & \alpha Q \end{pmatrix} \begin{pmatrix} x^{(k)} \\ y^{(k)} \end{pmatrix} + \begin{pmatrix} f \\ -g \end{pmatrix}, \tag{3.3}$$

which can be rewritten as the following fixed point form

$$\begin{pmatrix} x^{(k+1)} \\ y^{(k+1)} \end{pmatrix} = \mathcal{T}_r(\alpha) \begin{pmatrix} x^{(k)} \\ y^{(k)} \end{pmatrix} + \begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}^{-1} \begin{pmatrix} f \\ -g \end{pmatrix},$$

where

$$\mathcal{T}_r(\alpha)=\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}^{-1}\begin{pmatrix} 0 & B \\ 0 & \alpha Q \end{pmatrix}, \tag{3.4}$$

is the iteration matrix of (3.3). The splitting preconditioner corresponding to the iteration method (3.3) is given by

$$\mathcal{P}_r=\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}, \tag{3.5}$$

which is called the regularized preconditioner for the nonsymmetric saddle point matrix $\mathcal{A}$. In the following subsection, we study the convergence properties of the iteration method (3.3).

3.2.1 Convergence for nonsingular saddle point problem

Proposition 10

The preconditioner $\mathcal{P}_r$ (3.5), has the following factorization

$$\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}=\begin{pmatrix} I & 0 \\ -B^TA^{-1} & I \end{pmatrix}\begin{pmatrix} A & 0 \\ 0 & \tilde{A} \end{pmatrix}\begin{pmatrix} I & 2A^{-1}B \\ 0 & I \end{pmatrix}, \tag{3.6}$$

where $\tilde{A}=\alpha Q+2B^TA^{-1}B$. Then the inverse of the regularized preconditioner is given by the following equality

$$\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}^{-1}=\begin{pmatrix} I & -2A^{-1}B \\ 0 & I \end{pmatrix}\begin{pmatrix} A^{-1} & 0 \\ 0 & \tilde{A}^{-1} \end{pmatrix}\begin{pmatrix} I & 0 \\ B^TA^{-1} & I \end{pmatrix}. \tag{3.7}$$

Lemma 3.2.1 *Assume that $A\in\mathbb{R}^{n\times n}$ is a positive definite matrix. If λ is an eigenvalue of the iteration matrix $\mathcal{T}_r$ see* (3.4)*, then $|\lambda|\le 1$.*

Proof : The iteration matrix $\mathcal{T}_r$ can be rewritten as follow

$$\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}^{-1}\begin{pmatrix} 0 & B \\ 0 & \alpha Q \end{pmatrix}=\begin{pmatrix} 0 & K_1 \\ 0 & K_2 \end{pmatrix}, \tag{3.8}$$

where $K_1=A^{-1}B(I-2K_2)$, $K_2=\tilde{A}^{-1}\left(B^TA^{-1}B+\alpha Q\right)$. If λ is an eigenvalue of the matrix $\mathcal{T}_r(\alpha)$, then $\lambda=0$ or λ is an eigenvalue of K_2, which mean there exists a vector $y \neq 0$ satisfies the following equality

$$K_2 y = \tilde{A}^{-1}\left(B^T A^{-1} B + \alpha Q\right) y = \lambda y, \tag{3.9}$$

(3.9) is equivalent to

$$\begin{aligned} \tilde{A}\tilde{A}^{-1}\left(B^T A^{-1} B + \alpha Q\right) y &= \lambda \tilde{A} y, \\ \left(B^T A^{-1} B + \alpha Q\right) y &= \lambda\left(2B^T A^{-1} B + \alpha Q\right) y. \end{aligned} \tag{3.10}$$

Premultiplying (3.10) from both sides by $\frac{y^*}{y^* y}$ lead to

$$\lambda = \frac{a + \alpha q + ib}{2a + \alpha q + 2ib}, \tag{3.11}$$

where i is the imaginary complex ($i^2 = -1$), a and b are the real part and the imaginary part of $\frac{y^* B A^{-1} B^T y}{y^* y}$ and $q = \frac{y^* Q y}{y^* y}$. Since A, Q are positive definite matrices, then $q > 0$ and $a \geq 0$. After some handling, we obtain the following inequality

$$|\lambda| = \frac{|a + \alpha q + ib|}{|2a + \alpha q + 2ib|} \leq 1. \tag{3.12}$$

□

Lemma 3.2.2 *Assume that the condition of Lemma 3.2.1 is satisfied, B has a full column rank and λ is an eigenvalue of the iteration matrix $\mathcal{T}_r$, then $|\lambda| < 1$.*

Proof : If λ is an eigenvalue of the iteration matrix $\mathcal{T}_r$, then $\lambda = 0$ (3.8) or $\lambda = \frac{a + \alpha q + ib}{2a + \alpha q + 2ib}$, see (3.11). Since A, Q are positive definite matrices, and B has a full column rank, then $q > 0$ and $a > 0$, see [29]. After an easy manipulation, λ must hold the following inequality

$$|\lambda| = \frac{|a + \alpha q + ib|}{|2a + \alpha q + 2ib|} < 1. \tag{3.13}$$

□

The following theorem readily follows from Lemmas 3.2.1 − 3.2.2.

Theorem 11

Let $A \in \mathbb{R}^{n\times n}$ be a positive definite matrix and $B \in \mathbb{R}^{n\times m}$ be of full column rank, then the regularized iteration method (3.3) is convergent for all $\alpha > 0$.

3.2.2 Semi-convergence for singular saddle point problem

In this subsection, we assume that the block $(1,2)$ in the saddle-point matrix $\mathcal{A}$ (3.1) is rank deficient and discuss the semi convergence of the regularized iteration method (3.3). To demonstrate the semi-convergence of the regularized iteration method we use the conditions of [33].
The iteration scheme (3.3) is semi-convergent if and only if the following two conditions are satisfied :
(i) $\text{rank}((I-\mathcal{T}_r)^2) = \text{rank}(I-\mathcal{T}_r)$, where $\mathcal{T}_r$ is the iteration matrix.
(ii) The pseudo-spectral radius of $\mathcal{T}_r$ is less than one

$$\gamma(\mathcal{T}_r) = \max\{|\lambda| : \lambda \in \sigma(\mathcal{T}_r), \lambda \neq 1\} < 1,$$

where $\sigma(\mathcal{T}_r)$ is the spectrum of $\mathcal{T}_r$.

Lemma 3.2.3 *$rank((I-\mathcal{T}_r)^2) = rank(I-\mathcal{T}_r)$ if and only if, for any $(Y \neq 0) \in range(\mathcal{A}\mathcal{P}_r^{-1})$, $Y \notin null(\mathcal{A}\mathcal{P}_r^{-1})$.*

Proof : The proof strategy is similar to that used in [29].

Lemma 3.2.4 *Assume that A is a positive definite matrix, B is a rank deficient matrix and $\alpha > 0$, then $rank((I-\mathcal{T}_r)^2) = rank(I-\mathcal{T}_r)$.*

Proof : From Lemma 3.2.3 it's sufficient to prove that for any $(Y \neq 0) \in \text{range}(\mathcal{A})$, $Y \notin \text{null}(\mathcal{A}\mathcal{P}_r^{-1})$. Let $X = (x^T, y^T)^T$, where $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, such that

$$Y = \mathcal{A}X = \begin{pmatrix} Ax + By \\ -B^T x \end{pmatrix} \neq 0, \tag{3.14}$$

Suppose $(\mathcal{A}\mathcal{P}_r^{-1})Y = 0$ by contradiction, which means $\mathcal{P}_r^{-1}Y \in \text{null}(\mathcal{A})$, then there exists a vector $\epsilon = (0^T, \eta^T)^T$, such that $B\eta = 0$. Thus

$$Y = \mathcal{P}_r\epsilon = \begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix}\begin{pmatrix} 0 \\ \eta \end{pmatrix} = \begin{pmatrix} 0 \\ \alpha Q\eta \end{pmatrix}, \tag{3.15}$$

by combining with (3.15) and (3.14), we get

$$-B^T x = \alpha Q\eta \Longrightarrow \eta = -\frac{1}{\alpha}Q^{-1}B^T x \Longrightarrow B\eta = -\frac{1}{\alpha}BQ^{-1}B^T x = 0,$$

then we have $B^T x = 0$. Thus $Y = 0$, contradiction (because $Y \neq 0$). □
Next, we analyze the condition (ii).

Theorem 3.1 *Assume that the condition of Lemma* 3.2.1 *is satisfied, B is rank deficient, and $\lambda \neq 1$ is an eigenvalue of $\mathcal{T}_r$, then $|\lambda| < 1$.*

From the results above, we obtain the following semi-convergence result for the regularized iteration method (3.3).

Theorem 3.2 *Let $A \in \mathbb{R}^{n\times n}$ be a positive definite matrix and $B \in \mathbb{R}^{n\times m}$ be rank deficient , then the regularized iteration method* (3.3) *is semi-convergent for all $\alpha > 0$.*

3.2.3 Implementation of the regularized preconditioner

From (3.3), the regularized iteration method computes the approximate solutions of (3.1) iteratively by

$$\begin{pmatrix} A & 2B \\ -B^T & \alpha Q \end{pmatrix} X^{(k+1)} = \begin{pmatrix} 0 & B \\ 0 & \alpha Q \end{pmatrix} X^{(k)} + d, \tag{3.16}$$

where $X^{(k)} = \begin{pmatrix} x^{(k)} \\ y^{(k)} \end{pmatrix} \in \mathbb{R}^{n+m}$. The equation (3.16) can be equivalently rewritten as

$$\mathcal{P}_r z_k = r_k, \tag{3.17}$$

where $r_k = d - \mathcal{A}X^{(k)}$ is the k-th residual vector and $z_k = X^{(k+1)} - X^{(k)}$. At each step of the regularized iteration (3.16) or applying the regularized preconditioner $\mathcal{P}_r$ within a Krylov subspace method, then (3.17) need to be solved at each iteration step. To this end we use the following algorithm

Algorithme 8 : Implementation of the regularized preconditioner

```
1 : Solve $\underbrace{\left(A + \frac{2}{\alpha}BQ^{-1}B^T\right)}_{A_\alpha} z_1 = \underbrace{r_1 - \frac{2}{\alpha}BQ^{-1}r_2}_{c}$.
2 : Compute $z_2 = \frac{1}{\alpha}Q^{-1}\left(r_2 + B^T z_1\right)$, where $Q$ is a diagonal matrix.
```

Note that, for the sublinear system arising in Algorithm 8, is solved by preconditioned GMRES or preconditioned CG.

Convergence result for the preconditioned conjugate gradient method

In this subsection we use the following symmetric bilinear form on $\mathbb{R}^n$ defined by

$$\langle x, y\rangle_{\mathcal{P}^{-1}} = y^T\mathcal{P}^{-1}x, \tag{3.18}$$

where $\mathcal{P}$ is symmetric positive definite matrix, which corresponding to the preconditioner. The associated norm $||.||_{\mathcal{P}^{-1}}$ is given

$$||x||_{\mathcal{P}^{-1}} = \sqrt{x^T\mathcal{P}^{-1}x}, \text{ for } x \in \mathbb{R}^n. \tag{3.19}$$

The preconditioned conjugate gradient method is derived by using that $\mathcal{P}$ admits the Cholesky decomposition $\mathcal{P} = LL^T$. Thus one could use the central preconditioning for preserving symmetry of A_α

$$L^{-1}A_\alpha L^{-T}y = L^{-1}c \text{ with } y = L^T x \ .$$

The matrix $L^{-1}A_\alpha L^{-T}$ is still symmetric. Moreover, note that the preconditioned residual satisfies $||L^{-1}r_k|| = ||r_k||_{\mathcal{P}^{-1}}$. Then we have the following equality

$$\mathcal{K}_k(L^{-1}A_\alpha L^{-T}, L^{-1}r_0) = L^{-1}\mathcal{K}_k(A_\alpha\mathcal{P}^{-1}, r_0), \tag{3.20}$$

where $\mathcal{K}_k(L^{-1}A_\alpha L^{-T}, L^{-1}r_0) = \text{span}\{L^{-1}r_0, (L^{-1}A_\alpha L^{-T})L^{-1}r_0, .., (L^{-1}A_\alpha L^{-T})^{k-1}L^{-1}r_0\}$ and $\mathcal{K}_k(A_\alpha\mathcal{P}^{-1}, r_0) = \text{span}\{r_0, (A_\alpha\mathcal{P}^{-1})r_0, .., (A_\alpha\mathcal{P}^{-1})^{k-1}r_0\}$ are Krylov subspaces. The preconditioned conjugate gradient method constructs at step k, the approximation x_k satisfying the following two relations

$$x_k - x_0 \in \mathcal{K}_k(A_\alpha\mathcal{P}^{-1}, r_0) \text{ and } r_k \perp \mathcal{K}_k(A_\alpha\mathcal{P}^{-1}, r_0). \tag{3.21}$$

By using (3.21), the error norm $||e_k||_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}}$ and the residual norm $||r_k||_{\mathcal{P}^{-1}}$ are given as follows

$$\begin{aligned}
||e_k||^2_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}} &= \frac{1}{e_1^T\left(K_{k+1}^TA_\alpha^{-1}K_{k+1}\right)^{-1}e_1}, &(3.22)\\
||e_k||^2_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}} &= \frac{\det\left(K_{k+1}^TA_\alpha^{-1}K_{k+1}\right)}{\det(K_k^T\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}K_k)}, &(3.23)\\
||r_k||^2_{\mathcal{P}^{-1}} &= \frac{\det\left(K_k^T\mathcal{P}^{-1}K_k\right)\det\left(K_{k+1}^T\mathcal{P}^{-1}K_{k+1}\right)}{\det\left(K_k^T\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}K_k\right)^2}, &(3.24)
\end{aligned}$$

where $K_{k+1} = \left[r_0, \left(A_\alpha \mathcal{P}^{-1}\right) r_0, \ldots, \left(A_\alpha \mathcal{P}^{-1}\right)^k r_0\right]$. The matrix A_α is symmetric positive definite, so A_α admits a Cholesky decomposition $A_\alpha = GG^T$. Since $G^{-1}\left(A_\alpha \mathcal{P}^{-1}\right) G$ is symmetric, then it's spectral decomposition is given by
$G^{-1}(A_\alpha \mathcal{P}^{-1})G = S\Lambda S^T$, where $\Lambda = \text{diag}(\lambda_1, .., \lambda_n)$. In the following theorem, we use the eigenvalues $\lambda_1, ..\lambda_n$, to give a new estimations of the error norm and the residual norm where Λ and $\mathcal{S}$ are the eigenvalues and the eigenvectors matrices of $G^{-1}\left(A_\alpha \mathcal{P}^{-1}\right) G$, with $\mathcal{S}^T\mathcal{S} = I$.

Theorem 12

Let the initial preconditioned residual r_0 be decomposed as $r_0 = G\mathcal{S}\beta$ where β is vector whose components are denoted by $\beta_1, .., \beta_n$. Then we have

$$(1) \quad \|e_k\|^2_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}} = \frac{1}{e_1^T(V_{k+1}^T \widetilde{D} V_{k+1})^{-1} e_1},$$

$$(2) \quad \|r_k\|^2_{\mathcal{P}^{-1}} \leq \frac{1}{e_1^T(V_{k+1}^T \widetilde{D} V_{k+1})^{-1} e_1} \left(\frac{\tilde{\mu}^{k+1}}{\mu^k}\right),$$

where

$$\widetilde{D} = \begin{pmatrix} \beta_1^2 & & \\ & \ddots & \\ & & \beta_n^2 \end{pmatrix} \quad \text{and} \quad V_{k+1} = \begin{pmatrix} 1 & \lambda_1 & \ldots & \lambda_1^k \\ \vdots & \vdots & & \vdots \\ 1 & \lambda_n & \ldots & \lambda_n^k \end{pmatrix}, \qquad (3.25)$$

e_1 is the first unit vector of $\mathbb{R}^{k+1}$, $\widetilde{\mu}$ and μ are the maximum and the minimum eigenvalues respectively of the tridiagonal matrices $\widetilde{T}_{k+1} = \left(W_{k+1}^T A_\alpha^{-1} W_{k+1}\right)^{-1}$ and $T_k = U_k^T \mathcal{P}^{-1} A_\alpha \mathcal{P}^{-1} U_k$, obtained by Lanczos.

Proof : $G^{-1}K_{k+1}$ can be written as

$$G^{-1}K_{k+1} = [\mathcal{S}\beta, \mathcal{S}\Lambda\beta, \ldots, \mathcal{S}\Lambda^k\beta], \qquad (3.26)$$

which is equivalent to

$$G^{-1}K_{k+1} = \mathcal{S}D_\beta V_{k+1}, \qquad (3.27)$$

where D_β

$$D_\beta = \begin{pmatrix} \beta_1 & & \\ & \ddots & \\ & & \beta_n \end{pmatrix}. \qquad (3.28)$$

From (3.27), it follows that

$$(G^{-1}K_{k+1})^T(G^{-1}K_{k+1}) = K_{k+1}^T A_\alpha^{-1} K_{k+1} = V_{k+1}^T D_\beta^T D_\beta V_{k+1}. \tag{3.29}$$

Using (3.22), (3.29) and the fact that $\widetilde{D} = D_\beta^T D_\beta$, we obtain the following equality

$$\|e_k\|^2_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}} = \frac{1}{e_1^T(V_{k+1}^T\widetilde{D}V_{k+1})^{-1}e_1}. \tag{3.30}$$

The proof of (1) is completed. Let's now prove (2). By using (3.23) and (3.24), we get the following equality

$$\|r_k\|^2_{\mathcal{P}^{-1}} = \|e_k\|^2_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}} \frac{\det\left(K_k^T\mathcal{P}^{-1}K_k\right)\det\left(K_{k+1}^T\mathcal{P}^{-1}K_{k+1}\right)}{\det\left(K_k^T\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}K_k\right)\det\left(K_{k+1}^T A_\alpha^{-1}K_{k+1}\right)}, \tag{3.31}$$

substituting the QR factorization of $K_k = U_kR_k$ and $K_{k+1} = W_{k+1}\widetilde{R}_{k+1}$ in (3.31), we obtain

$$\|r_k\|^2_{\mathcal{P}^{-1}} = \|e_k\|^2_{\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}} \frac{\det(\widetilde{T}_{k+1})}{\det(T_k)}, \tag{3.32}$$

where $T_k = U_k^T\mathcal{P}^{-1}A_\alpha\mathcal{P}^{-1}U_k$ and $\widetilde{T}_{k+1} = (W_{k+1}^T A_\alpha^{-1} W_{k+1})^{-1}$. From (3.30), we obtain

$$\|r_k\|^2_{\mathcal{P}^{-1}} = \frac{1}{e_1^T(V_{k+1}^T\widetilde{D}V_{k+1})^{-1}e_1} \frac{\det(\widetilde{T}_{k+1})}{\det(T_k)}, \tag{3.33}$$

then, it follows from (3.33) that

$$\|r_k\|^2_{\mathcal{P}^{-1}} \leq \frac{1}{e_1^T(V_{k+1}^T\widetilde{D}V_{k+1})^{-1}e_1} \left(\frac{\tilde{\mu}^{k+1}}{\mu^k}\right), \tag{3.34}$$

where $\widetilde{\mu}$ and μ are the maximal and the minimal eigenvalues respectively of $\widetilde{T}_{k+1}$ and T_k. □

Numerical results

Sommaire

Numerical results

In this section, we use a numerical example to further illustrate the feasibility and effectiveness of the regularized preconditioned GMRES method over the RHSS [2], HSS [5] , GSOR [7] , PHSS [6], AHSS [3] and LSS [4] preconditioned GMRES methods from the point of view of both the number iterations and CPU times. All numerical procedures are performed in Matlab 2016 on a computer with Intel(R) Core (TM) i5 4300U CPU 2.49 GHZ, 8 GB RAM, and Windows 10 operating system. In all the tables 'IT' stands for the number of iterations, 'CPU' denotes the elapsed CPU time of the convergence, norm of absolute residual vector (denoted by RES). Here, 'RES' is defined as RES $= \|d - \mathcal{A}X\|_2$ and 'ERR' denote the error norm, is defined as ERR $= \|X - X^*\|_2$, where X^* is the exact solution of (3.1). The optimal parameter is very difficult to be obtained. In this paper, the parameter α in all preconditioners are taken as $\alpha = \left\{10^{-3}, 10^{-2}, 1, 10\right\}$ and the matrix Q in the regularized preconditioner is chosen as $Q = I$, because this case help the preconditioner $\mathcal{P}_r$ to make a better performance. When using PGMRES or PCG for solving the system at step (1) in Algorithm 1, the preconditioners used are the incomplete LU and Cholesky factorizations. In all the tests, the initial vector $X^{(0)}$ is set to be a zero vector and the right-hand side vector d is chosen such that the exact solution of the saddle point problem (3.1) is the vector of all ones. The iterations are terminated as soon as the current iterate $X^{(k)}$ satisfies

$$\frac{\|\mathcal{P}_r^{-1} d - \mathcal{P}_r^{-1} \mathcal{A} X^{(k)}\|}{\|\mathcal{P}_r^{-1} d\|} < 10^{-9}.$$

Example 1 : Consider the Stokes problem : find u and p such that

$$\left\{\begin{array}{rcl} -\nabla^2 u + \nabla p & = & f \text{ in } \Omega, \\ \nabla \cdot u & = & 0 \text{ in } \Omega, \end{array}\right. \tag{3.35}$$

where $\Omega \subset \mathbb{R}^2$ is a domain of space. $u : \Omega \to \mathbb{R}^2$ and $p : \Omega \to \mathbb{R}$ are respectively, the velocity and the pressure. By discretizing (3.35) with upwind scheme [6], we obtain the nonsingular saddle-point problem, whose the $(1,1)$ and $(1,2)$ blocks of $\mathcal{A}$ are given as follows

$$A = \begin{pmatrix} I \otimes T + T \otimes I & 0 \\ 0 & I \otimes T + T \otimes I \end{pmatrix} \in \mathbb{R}^{2l^2 \times 2l^2}, \tag{3.36}$$

note that A is symmetric.

$$B = \begin{pmatrix} I \otimes F \\ F \otimes I \end{pmatrix} \in \mathbb{R}^{2l^2 \times l^2}, \tag{3.37}$$

$T = \frac{v}{h^2}\text{tridiag}(-1,2,-1) \in \mathbb{R}^{l \times l}$, $F = \frac{1}{h}\text{tridiag}(-1,1,0) \in \mathbb{R}^{l \times l}$, $\otimes$ is the Kronecker product and $h = \frac{1}{l+1}$ represents the discretization mesh size. Numerical result of the preconditioned GMRES methods incorporated with the regularized, RHSS, HSS, PHSS, LSS and GSOR preconditioners are listed in Table 1, 2, 3, 4 and 5 for $v = 1$ and $v = 0.1$ on different uniform grids.

Table 1 : Numerical results of the preconditioned GMRES with $v = 0.1$ and $l = 32$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	5	47	47	18	7	91
	CPU	0.19	1.19	0.63	2.95	0.27	0.54
	RES	3.48e-02	8.35e-01	9.18e-01	2.86e-05	1.12e-05	3.18e-03
	ERR	3.29e-03	1.20e-02	1.33e-02	1.57e-06	1.25e-06	4.17e-05
10^{-2}	IT	6	55	61	5	7	93
	CPU	0.23	1.03	0.63	1.86	0.73	0.59
	RES	1.23e-02	1.38e-01	1.31e-01	1.04e-02	9.42e-04	2.93e-03
	ERR	1.10e-03	1.77e-03	1.66e-03	1.95e-04	1.18e-04	3.85e-05
1	IT	8	40	82	8	15	50
	CPU	0.19	0.24	0.84	0.25	1.07	0.77
	RES	2.58e-06	5.99e-04	1.72e-03	2.04e-06	5.12e-06	1.21e-04
	ERR	8.87e-08	2.38e-05	2.06e-05	6.56e-08	7.25e-07	9.72e-07
10	IT	12	69	75	13	90	85
	CPU	0.18	0.53	0.52	0.26	26.51	0.64
	RES	1.02e-06	5.49e-05	8.29e-05	3.82e-06	1.48e-03	9.64e-03
	ERR	5.99e-08	7.88e-06	1.12e-06	1.40e-07	2.17e-05	7.73e-04

Table 2 : Numerical results of the preconditioned GMRES with $v = 0.1$ and $l = 64$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	5	63	63	8	†	136
	CPU	3.46	17.57	14.64	84.78	†	9.41
	RES	9.42e-01	1.18e+01	1.18e+01	1.35e+00	†	8.51e-02
	ERR	2.28e-02	1.04e-01	1.04e-01	1.94e-02	†	5.49e-04
10^{-2}	IT	6	83	83	8	†	138
	CPU	4.64	20.39	21.08	21.21	†	9.53
	RES	3.52e-02	1.37e+00	1.37e+00	7.80e-02	†	7.71e-02
	ERR	1.81e-03	1.00e-02	1.00e-02	9.61e-04	†	5.01e-04
1	IT	9	119	119	17	†	76
	CPU	2.28	24.37	28.48	8.67	†	4.91
	RES	3.17e-05	2.50e-02	2.35e-02	5.54e-04	†	1.57e-03
	ERR	5.87e-07	1.55e-04	1.45e-04	1.09e-05	†	3.59e-06
10	IT	13	120	105	25	†	132
	CPU	2.29	17.72	26.93	6.66	†	8.77
	RES	2.79e-06	1.30e-03	7.56e-03	1.64e-04	†	3.57e-01
	ERR	1.35e-07	8.39e-06	4.89e-05	2.69e-06	†	2.27e-02

Table 3 : Numerical results of the preconditioned GMRES with $v = 1$ and $l = 32$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	6	35	35	5	7	81
	CPU	0.20	0.73	0.43	1.13	0.54	0.58
	RES	9.31e-02	6.46e+00	7.63e+00	1.12e-01	1.12e-04	2.81e-01
	ERR	3.77e-03	1.10e-01	1.33e-01	2.11e-03	4.76e-06	3.5e-03
10^{-2}	IT	7	42	47	6	7	83
	CPU	0.21	0.74	0.58	1.88	0.30	0.54
	RES	3.57e-06	8.08e-01	9.18e-01	3.37e-03	1.55e-04	2.71e-01
	ERR	5.29e-07	1.19e-02	1.33e-02	5.49e-05	1.24e-04	3.41e-03
1	IT	12	31	71	13	15	49
	CPU	0.18	0.25	0.80	0.30	1.19	0.31
	RES	8.83e-07	2.97e-03	1.76e-02	4.94e-06	5.49e-06	2.12e-04
	ERR	5.45e-08	1.51e-04	2.14e-04	1.68e-07	3.86e-06	3.41e-06
10	IT	16	46	75	17	81	76
	CPU	0.22	0.37	0.64	0.24	27.95	0.50
	RES	4.70e-06	1.27e-04	1.53e-03	2.42e-06	1.38e-01	1.21e+00
	ERR	2.08e-07	6.01e-05	1.96e-05	2.10e-07	1.92e-03	1.91e-01

Table 4 : Numerical results of the preconditioned GMRES with $v = 1$ and $l = 64$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	6	43	43	43	†	117
	CPU	3.46	3.70	3.70	3.83	†	7.45
	RES	3.91e-01	9.59e+01	9.58e+01	9.95e+01	†	7.47e+00
	ERR	2.13e-02	1.47e+00	1.47e+00	1.47e+00	†	4.93e-02
10^{-2}	IT	7	46	65	6	†	120
	CPU	3.05	5.77	4.46	17.48	†	8.28
	RES	1.04e-01	1.08e+01	9.09e+00	4.72e-02	†	8.18e+00
	ERR	3.25e-03	9.84e-02	7.81e-02	4.34e-04	†	5.36e-02
1	IT	13	34	101	14	†	76
	CPU	1.71	3.02	6.34	4.49	†	4.86
	RES	6.32e-06	2.38e-02	2.54e-01	1.97e-05	†	2.30e-03
	ERR	1.62e-07	1.45e-03	1.64e-03	2.41e-06	†	2.13e-05
10	IT	18	49	113	18	†	117
	CPU	1.95	2.43	7.60	2.92	†	7.80
	RES	5.40e-06	4.02e-03	2.13e-02	2.86e-05	†	9.40e+00
	ERR	6.54e-07	2.03e-03	1.43e-04	1.87e-06	†	1.15e+00

Table 5 : Numerical results of the preconditioned GMRES with $v = 1$ and $l = 128$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	6	40	40	5	†	163
	CPU	27.39	56.99	49.13	44.49	†	97.49
	RES	4.98e+00	8.34e+02	8e+02	3.10e+00	†	2.56e+02
	ERR	8.30e-2	1.59e+01	1.50e+01	1.78e+02	†	9.05e-01
10^{-2}	IT	7	44	81	171	†	6
	CPU	32.22	34.86	49.03	84.87	†	88.64
	RES	5.96e-01	1.50e+02	1.58e+02	7.63e-02	†	2.58e+02
	ERR	8.99e-03	1.23e+00	1.26e+00	2.30e-04	†	9.11e-01
1	IT	13	33	141	15	†	116
	CPU	11.97	15.86	81.05	10.77	†	66.18
	RES	1.65e-04	6.78e-01	2.21e+00	1.58e-04	†	1.12e-02
	ERR	2.26e-06	1.98e-02	8.18e-03	1.42e-06	†	1.42e-06
10	IT	17	17	49	19	†	180
	CPU	11.28	11.03	22.23	13.42	†	106.21
	RES	6.39e-02	1.11e-01	4.64e-02	1.28e-04	†	8.15e+02
	ERR	2.70e-01	2.60e-01	1.23e-02	3.54e-06	†	1.47e+02

From numerical results listed in Tables 1, 2, 3, 4 and 5 we can conclude some observations. Tables 1, 2, 3, 4 and 5 show that for different parameters, the regularized preconditioned GMRES method requires less IT and CPU times than the RHSS, HSS, PHSS, LSS and GSOR preconditioned GMRES methods, which means that the proposed regularized preconditioned GMRES method is more efficent than RHSS, HSS, PHSS, LSS and GSOR preconditioned GMRES methods in accelerating the convergence of the GMRES method for solving the nonsingular saddle point problem in Example 1. The PHSS and the GSOR preconditioned GMRES methods have worse convergence behaviors when the size grows. † means that the method has not converged after 500 iterations and RIM denote the regularized iteration method. We have also tested the RPDPSS, RHSS and HSS iteration methods, but their performance are poor. That's why we don't show the corresponding numerical results. In Tables 7, 11 and 14, to implement the regularized preconditioner effeciently, we need to choose the parameters α appropriately since the analytic determination of the parameters which results in the fastest convergence of the preconditioned GMRES iteration appears to be quite a difficult problem. In the regularized preconditioner, the parameter α is taken as $\alpha = ||B||_2^2/||A||_2$, which balances the matrices A and B^TB in the Euclidean norm. The parametres α chosen for testing the RHSS, HSS, LSS, PHSS and GSOR preconditioners in Tables 7, 11 and 14 of numerical experiments are the optimal

Table 6 : Numerical results for the four iteration methods with $v = 1$ and $l = 32$

α		RIM
10^{-3}	IT	25
	CPU	1.62
	RES	5.29e-10
	ERR	1.21e-06
10^{-2}	IT	27
	CPU	1.42
	RES	8.21e-10
	ERR	2.39e-06
1	IT	283
	CPU	5.84
	RES	9.59e-10
	ERR	4.87e-05
10	IT	†
	CPU	†
	RES	†
	ERR	†

Table 7 : Numerical results of the preconditioned GMRES with $v = 0.1$

Grid		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
32×32	IT	12	69	75	26	17	93
	CPU	0.25	0.27	0.88	0.42	0.96	0.89
	RES	1.68e-06	5.49e-05	8.11e-05	1.79e-05	4.36e-05	2.85e-03
	ERR	6.71e-08	7.90e-06	1.19e-06	1.14e-06	2.04e-06	3.74e-05
	α_{exp}	9.9993	10.0676	10.0676	9.9993	0.5128	0.0117
64×64	IT	13	81	120	29	18	138
	CPU	1.05	1.73	5.22	2.73	17.88	12.83
	RES	3.33e-05	4.01e-04	1.28e-03	1.38e-04	1.78e-04	9.26e-02
	ERR	5.84e-07	7.43e-05	8.14e-06	8.23e-06	3.95e-06	5.91e-04
	α_{exp}	9.9993	10.0676	10.0676	9.9993	0.5128	0.0117
128×128	IT	13	88	184	31	†	†
	CPU	5.91	11.81	66.41	27.23	†	†
	RES	4.30e-04	3.19e-03	2.11e-02	1.38e-03	†	†
	ERR	2.98e-06	7.41e-04	6.64e-05	2.63e-05	†	†
	α_{exp}	9.9993	10.0676	10.0676	9.9993	0.5128	0.0117

iteration parameters α^*, see [4, 5, 6, 3, 7]. The iteration counts and the elapsed CPU times of the regularized preconditioned GMRES method are much smaller than those of the RHSS, HSS, PHSS, LSS and GSOR preconditioned GMRES methods. Both iteration counts and elapsed CPU times show that the regularized preconditioner is much efficient than the RHSS, HSS, PHSS, LSS and GSOR preconditioners.
Example 2 : Consider the Navier-Stokes problem : find u and p such that

$$\begin{cases} -v\nabla^2 u + u\cdot\nabla u + p &= f \text{ in } \Omega, \\ \nabla\cdot u &= 0 \text{ in } \Omega, \end{cases} \tag{3.38}$$

where $\Omega \subset \mathbb{R}^2$ is a domain of space with boundary $\partial\Omega$. $u : \Omega \to \mathbb{R}^2$ and $p : \Omega \to \mathbb{R}$ are respectively, the velocity and the pressure. By discretizing (3.38) with upwind scheme [6], we obtain the nonsingular saddle-point problem, whose the $(1,1)$ and $(1,2)$ blocks of $\mathcal{A}$ are given as follows

$$A = \begin{pmatrix} I\otimes T + T\otimes I & 0 \\ 0 & I\otimes T + T\otimes I \end{pmatrix} \in \mathbb{R}^{2l^2\times 2l^2}, \tag{3.39}$$

note that A is nonsymmetric.

$$B = \begin{pmatrix} I\otimes F \\ F\otimes I \end{pmatrix} \in \mathbb{R}^{2l^2\times l^2}, \tag{3.40}$$

$T = \dfrac{v}{h^2}\text{tridiag}(-1,2,-1) + \dfrac{1}{h^2}\text{tridiag}(-1,0,1) \in \mathbb{R}^{l\times l}$, $F = \dfrac{1}{h}\text{tridiag}(-1,1,0) \in \mathbb{R}^{l\times l}$, $\otimes$ is the Kronecker product and $h = \dfrac{1}{l+1}$ represents the discretization mesh size. In Tables 8, 9 and 10, we list the numerical results with respect to IT and CPU, ERR and RES for the preconditioned GMRES methods. From these tables, we can conclude some observations as follows. The regularized preconditioned GMRES method leads to much better performance than the RHSS, HSS, PHSS, LSS, GSOR methods.

Table 8 : Numerical results of the preconditioned GMRES with $v = 0.1$ and $l = 32$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	5	69	72	7	8	91
	CPU	0.97	2.45	2.70	0.38	8.32	4.04
	RES	1.08e-04	9.43e-01	9.60e-01	1.53e-04	3.10e-02	3.21e-03
	ERR	1.78e-06	1.85e-02	1.83e-02	4.30e-06	7.79e-03	4.22e-05
10^{-2}	IT	6	70	99	10	5	93
	CPU	0.75	2.48	3.38	27.79	6.20	4.15
	RES	6.14e-07	1.63e-01	1.24e-01	4.03e-06	3.45e-03	2.92e-03
	ERR	8.51e-09	2.28e-03	1.68e-03	1.03e-05	4.84e-04	3.71e-05
1	IT	8	69	141	15	8	50
	CPU	0.69	1.12	4.54	1.63	1.55	1.99
	RES	1.49e-05	1.23e-04	1.54e-03	5.01e-06	1.64e-05	1.21e-04
	ERR	2.62e-07	1.31e-04	1.91e-05	5.01e-06	7.42e-07	1.93e-06
10	IT	12	273	96	91	13	85
	CPU	0.29	3.05	2.21	39.49	1.05	3.73
	RES	5.70e-06	6.67e-05	1.01e-04	1.31e-03	1.32e-05	9.71e-03
	ERR	9.93e-08	1.20e-04	1.19e-06	1.87e-05	2.97e-07	7.82e-04

Table 9 : Numerical results of the preconditioned GMRES with $v = 0.1$ and $l = 64$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	6	43	43	†	8	135
	CPU	5.75	11.66	51.97	†	8.32	118.83
	RES	8.22e-05	7.57e+01	9.55e+01	†	3.10e-02	8.79e-02
	ERR	7.51e-07	1.06e+00	1.46e+00	†	7.79e-03	5.68e-04
10^{-2}	IT	7	46	65	†	5	138
	CPU	4.80	9.05	75.67	†	6.20	145.41
	RES	3.43e-06	1.08e+01	8.52e+00	†	3.45e-03	8.88e-02
	ERR	1.90e-08	9.83e-02	7.28e-02	†	4.84e-04	5.77e-04
1	IT	13	33	101	†	8	76
	CPU	1.37	15.48	91.35	†	1.55	58.27
	RES	3.41e-05	2.36e-02	2.54e-01	†	1.64e-05	1.58e-03
	ERR	2.53e-07	1.48e-03	1.64e-03	†	7.42e-07	6.00e-06
10	IT	18	49	112	†	13	132
	CPU	1.19	20.49	74.04	†	1.05	144.46
	RES	1.94e-05	2.11e-03	2.06e-02	†	1.32e-05	3.60e-01
	ERR	5.56e-07	8.75e-04	1.75e-04	†	2.97e-07	2.29e-02

Table 10 : Numerical results of the preconditioned GMRES with $v = 0.1$ and $l = 128$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	5	†	†	†	†	†
	CPU	27.87	†	†	†	†	†
	RES	3.78e-03	†	†	†	†	†
	ERR	1.85e-05	†	†	†	†	†
10^{-2}	IT	6	†	†	†	†	†
	CPU	29.47	†	†	†	†	†
	RES	1.76e-04	†	†	†	†	†
	ERR	8.46e-07	†	†	†	†	†
1	IT	9	†	†	†	†	†
	CPU	13.51	†	†	†	†	†
	RES	2.06e-04	†	†	†	†	†
	ERR	1.29e-06	†	†	†	†	†
10	IT	13	†	†	†	†	†
	CPU	5.68	†	†	†	†	†
	RES	5.78e-05	†	†	†	†	†
	ERR	6.39e-07	†	†	†	†	†

Table 11 : Numerical results of the preconditioned GMRES with $v = 0.1$

Grid		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
32×32	IT	12	67	70	5	17	93
	CPU	0.10	0.73	1.74	0.43	1.02	0.98
	RES	5.70e-06	6.82e-05	1.00e-04	6.16e-05	4.39e-05	2.86e-03
	ERR	9.94e-08	9.38e-06	2.18e-06	1.58e-06	2.30e-06	3.75e-05
	α_{exp}	9.9993	10.067	10.067	9.9993	0.5128	0.0117
64×64	IT	13	78	116	15	18	139
	CPU	1.33	13.55	37.27	6.61	12.12	13.65
	RES	1.57e-05	4.97e-04	1.41e-03	1.18e-04	1.92e-04	8.44e-02
	ERR	1.54e-07	9.12e-05	1.44e-05	6.16e-06	4.85e-06	5.43e-04
	α_{exp}	9.9993	10.067	10.067	9.9993	0.5128	0.0117
128×128	IT	13	†	†	†	†	†
	CPU	7.64	†	†	†	†	†
	RES	5.78e-05	†	†	†	†	†
	ERR	6.59e-07	†	†	†	†	†
	α_{exp}	9.9993	10.067	10.067	9.9993	0.5128	0.0117

By comparing the results in Table 8, 9 and 10 it can be seen that the regularized preconditioned GMRES methods succeed in producing high-quality approximate solutions in all cases, while the regularized preconditioned GMRES method outperforms the RHSS, HSS, PHSS, GSOR and LSS preconditioned GMRES methods in terms of IT, CPU times, ERR and RES. The RHSS, HSS, GSOR and LSS preconditioned GMRES methods have worse convergence behaviors when the size grows. From Table 11, it can be observed that the regularized preconditioned GMRES requires the least IT and CPU times, which implies that the regularized preconditioned GMRES is superior to the other methods in terms of computing efficiency.

Example 3 : Consider the nonsymmetric singular saddle point problem structured as (3.1) with the following coefficient sub-matrices

$$A = \begin{pmatrix} I\otimes T + T\otimes I & 0 \\ 0 & I\otimes T + T\otimes I \end{pmatrix} \in \mathbb{R}^{2l^2\times 2l^2}, \tag{3.41}$$

note that A is nonsymmetric.

$$B = \begin{pmatrix} \hat{B} & b_1 & b_2 \end{pmatrix} \in \mathbb{R}^{2l^2\times(l^2+2)}, \tag{3.42}$$

$T = \dfrac{v}{h^2}\text{tridiag}(-1,2,-1) + \dfrac{1}{2h}\text{tridiag}(-1,0,1) \in \mathbb{R}^{l\times l}$, $B = \begin{pmatrix} I\otimes F \\ F\otimes I \end{pmatrix} \in \mathbb{R}^{2l^2\times l^2}$, $b_1 = \hat{B}\begin{pmatrix} e \\ 0 \end{pmatrix}$, $b_2 = \hat{B}\begin{pmatrix} 0 \\ e \end{pmatrix}$, $e = (1,1,\ldots,1) \in \mathbb{R}^{l^2}$. $F = \dfrac{1}{h}\text{tridiag}(-1,1,0) \in \mathbb{R}^{l\times l}$, $\otimes$ is the Kronecker product and $h = \dfrac{1}{l+1}$ represents the discretization mesh size. We list the numerical results of the RHSS, HSS, PHSS, LSS and GSOR preconditioned GMRES methods with two different values of v ($v = 1$ and $v = 0.1$) in Tables 12 and 13.

Table 12 : Numerical results of the preconditioned GMRES with $v = 0.1$ and $l = 64$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	6	70	73	†	60	144
	CPU	3.91	5.11	4.89	†	51.19	171.72
	RES	1.90e+01	2.01e+01	1.29e+01	†	1.36e+01	2.75e-02
	ERR	6.39e+01	6.32e+01	1.42e+00	†	6.39e+01	6.39e+01
10^{-2}	IT	6	72	93	†	69	146
	CPU	3.05	4.85	10.56	†	70.28	133.75
	RES	4.16e-01	2.079e+00	1.46e+00	†	2.00e+00	2.82e-02
	ERR	6.39e+01	6.32e+01	7.36e+01	†	6.39e+01	6.39e+01
1	IT	7	36	222	†	80	81
	CPU	1.24	4.54	20.77	†	67.52	80.96
	RES	9.94e-05	7.73e-03	3.44e-02	†	1.47e-03	3.00e-02
	ERR	6.39e+01	6.32e+01	2.29e+01	†	6.39e+01	6.39e+01
10	IT	9	60	125	†	74	134
	CPU	0.65	7.48	74.04	†	49.42	127.09
	RES	2.80e-05	5.93e-04	1.55e-03	†	2.59e-03	3.24e-01
	ERR	6.39e+01	6.32e+01	6.39e+01	†	6.39e+01	6.39e+01

Table 13 : Numerical results of the preconditioned GMRES with $v = 1$ and $l = 64$

α		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{GSOR}$
10^{-3}	IT	6	41	45	†	60	144
	CPU	3.91	37.60	16.10	†	51.19	171.72
	RES	1.90e+01	6.91e+02	6.71e+02	†	1.36e+01	2.75e-02
	ERR	6.39e+01	6.35e+01	6.35e+01	†	6.39e+01	6.39e+01
10^{-2}	IT	7	54	73	†	69	146
	CPU	5.79	40.77	23.18	†	70.28	133.75
	RES	3.56e-02	1.76e+01	1.92e+01	†	2.00e+00	2.82e-02
	ERR	6.39e+01	6.32e+01	6.32e+01	†	6.39e+01	6.39e+01
1	IT	9	25	25	†	80	81
	CPU	0.67	12.13	1.13	†	67.52	80.96
	RES	6.91e-05	5.13e-02	5.14e-02	†	1.47e-03	3.00e-02
	ERR	6.39e+01	6.32e+01	6.32e+01	†	6.39e+01	6.39e+01
10	IT	12	33	28	†	74	134
	CPU	0.86	14.97	0.87	†	49.42	127.09
	RES	1.74e-05	3.21e-03	2.56e-02	†	2.59e-03	3.24e-01
	ERR	6.39e+01	6.32e+01	6.32e+01	†	6.39e+01	6.39e+01

From Tables 12 and 13, we can conclude some observations as follows. The regularized preconditioned GMRES method returns better numerical results than the other preconditioned GMRES methods in terms of IT, CPU times, ERR and RES.

Table 14 : Numerical results of the preconditioned GMRES with $v = 0.1$

Grid		$\mathcal{P}_r$	$\mathcal{P}_{RHSS}$	$\mathcal{P}_{HSS}$	$\mathcal{P}_{LSS}$	$\mathcal{P}_{PHSS}$	$\mathcal{P}_{GSOR}$
32×32	IT	12	67	70	5	17	93
	CPU	0.10	0.73	1.74	0.43	0.95	1.45
	RES	5.70e-06	6.82e-05	1.00e-04	6.16e-05	4.39e-05	3.05e-03
	ERR	9.94e-08	9.38e-06	2.18e-06	1.58e-06	2.30e-06	4.02e-05
	α_{exp}	9.9993	10.067	10.067	9.9993	0.5128	0.0117
64×64	IT	13	78	116	15	18	139
	CPU	1.33	13.55	37.27	6.61	12.12	13.89
	RES	1.57e-05	4.97e-04	1.41e-03	1.18e-04	1.92e-04	8.44e-02
	ERR	1.54e-07	9.12e-05	1.44e-05	6.16e-06	4.85e-06	5.43e-04
	α_{exp}	9.9993	10.067	10.067	9.9993	0.5128	0.0117
128×128	IT	13	†	†	†	†	†
	CPU	7.64	†	†	†	†	†
	RES	5.78e-05	†	†	†	†	†
	ERR	6.59e-07	†	†	†	†	†
	α_{exp}	9.9993	10.067	10.067	9.9993	0.5128	0.0117

From Table 14, if the regularized preconditioner were used, the regularized preconditioned GMRES converges very fast. Furthermore, it can be observed the iteration counts and the elapsed CPU times of the regularized preconditioned GMRES method are much smaller than those of the RHSS, HSS, LSS, PHSS and GSOR preconditioned GMRES methods.

Conclusion

In this paper, we proposed a regularized preconditioner for nonsymmetric saddle point problem (3.1). This new preconditioner improve the convergence rate of GMRES method. The unconditional convergence and semi-convergence analysis of the regularized iteration method for solving nonsingular and singular saddle point problems, respectively, are presented. In addition, we developed in this paper a new convergence results for preconditioned conjugate gradient method. Numerical experiments are presented to show the advantages of the regularized preconditioner over HSS, RHSS, GSOR, PHSS and LSS preconditioners for solving nonsingular and singular saddle point problems. However, how to choose the optimal parameters α for the regularized preconditioner is a very practical and interesting problem that needs to be further in-depth studied in the future.

Annexe

MATLAB program of the preconditioned GMRES for solving (3.1)

```
%%  The size of matrix
   m=32;
%% The discretization meshsize
   h=1/(m+1);
%%  The vilosity
   v=1;
%%
   T=(v/h^2)*(gallery('tridiag',m,-1,2,-1))+(1/2*h)*(
      gallery('tridiag',m,-1,0,1));
%% T=(v/h^2)*(gallery('tridiag',m,-1,2,-1));
   F=(1/h)*(gallery('tridiag',m,-1,1,0));
   I=speye(m,m);
%%  The construction of the matrices
A=[kron(I,T)+kron(T,I)   zeros(m^2,m^2);zeros(m^2,m^2)
       kron(I,T)+kron(T,I)];
B=[kron(I,F);kron(F,I)];
B=B';
[l,n]=size(B);

%% Choose alpha
alpha=1e+2;
%% Choose type of the problem
epsilon=-1;
%% Create artificial rhs by choosing the exact solution
%% Fix the right handside number
nrhs=1;

%% Choose the exact solution
Exact_Sol=ones(n+l,nrhs);
%%  Compute the right hand side
```

```
  rhs=AAx(epsilon,A,B,Exact_Sol);
  r=rhs;
  I=speye(l,l);

%% ------------ paramaters for KRYLOV SUBSPACE method
%% im=200; maxits=2000; tolIts=1e-12; tol0=10^-8;
im=200; maxits=400; tolIts=1e-9; tol0=10^-8;x0=zeros(n+l,
   nrhs);

%Preconditioner of system intern of  Pr
L=ichol(A,struct('type','ict','droptol',10^-2));

tic
% Right Preconditioning Pr
[sol,res,its]=PGlbGMRES(epsilon,A,B,I,L,U,alpha,rhs,x0,im,
   maxits,tolIts);
toc
```

```

function [y,it]=Precfunct(epsilon,A,B,QQ,L,U,alpha,x)
%-------------------------------------------------
% y=precfun(A,B,QQ,alpha,r)
%  Preconditioner matrix times vector x
% A, B  = matrices from the Saddle system
%  QQ : diagonal matrix
%
%  (A   B^T)
%  (        ) x= y
%  (-B    0 )
%
%   im     = krylov subspace dimension
%   rhs    = right-hand side
%   x0     = initial guess
%   tol    = tolerance for stopping criterion
%   maxits = max number of matrix - vector products
%    allowed (= max tot. # of global gmres steps)
%-------------------------------------------------
 [m,n]=size(B); alphainv=1/alpha;
 temp = alphainv * x(n+1:n+m,:);
 u=x(1:n,:)-B'*temp;

 [y1]=Gcg01(A,B,QQ,alphainv,L,u) ;
 y2=temp-epsilon*(QQ\(B*alphainv*y1));
 y=[y1;y2];
end
```

```
function [x,its] = Gcg01(A,B,QQ,beta,L,U,rhs)


%% solves by using Global CG the linear system :
%(A + beta  B' *  QQ^(-1) *  B ) * x = rhs

[n,m]=size(rhs);
x0=zeros(n,m);
maxit=n;tol=1e-8;
alpha=(1/beta);


%%--------------------------------------------

 OutputG=1;
 n=size(A,1);
 x = x0;
 r=rhs-(A  * x + B' * (QQ\( B*(beta *x)) ));


 z = U\(L\r);
 p = z ;
 ro1 = z(:)' * r(:);
 tol1 = tol*tol*ro1;
 its = 0 ;
 while (its < maxit && ro1 > tol1)

         its = its+1;
     res(its)=ro1;
         ro = ro1;
         ap=  A  * p + B' * (QQ\( B*(beta *p)));

         alp = ro / ( ap(:)'* p(:)) ;
         x = x + alp * p ;
         r = r - alp * ap;
     z =U\(L\ r);
```

```
    ro1= z(:)'*r(:);
        bet = ro1 / ro ;

        p = z + bet * p;


end
  if(OutputG)
    fprintf(' %4.0f %22.14e \n', its,sqrt(ro1))
    end

end
```

```
function [sol,res,its,it]=PGlbGMRES(epsilon,A,B,Q,L,U,
   alpha,rhs,x0,im,maxits,tol);
%-------------------------------------------------
% [res,Res,sol] =PGlbGMRES(A,B,Q,alpha,rhs,x0,im,maxits,
   tolIts)
% restarted Preconditioned Global gmres with Krylov
   subspace of dim = im.
% A , B  = matrices from the Saddle system
%  Q is a diagonal matrix introduced for the
   preconditioner
%
%      (A    B^T)^(-1)      (A    B^T)        (A    B^T)^(-1)
%      (         )        *  (         ) * X=  (         )
   *  Rhs
%      (-B    ?Q)            (-B     0  )      (-B    ?Q)
%   im      = krylov subspace dimension
%   rhs     = right-hand side
%   x0      = initial guess
%   tol     = tolerance for stopping criterion
%   maxits = max number of matrix - vector products
%          allowed (= max tot. # of global gmres steps)
%-------------------------------------------------
      tolmac = eps;
      its = 1 ;
      sol = x0;
      outputG=0;
      [m,n]=size(B);
      it=0;
```

```

%%
      while (its < maxits)
      r=rhs-AAx(epsilon,A,B,sol);
      vv{1}=Precfunct(epsilon,A,B,Q,L,U,alpha,r);

      ro = norm(vv{1},'fro') ;
      res(its) = ro;
      if (its  == 1)
        tol1=tol*ro ;
      end
      if (ro < tol1 |its > maxits)
        return
      end
      t = 1.0/ ro ;
      vv{1} = vv{1} * t ;
%%---initialize 1-st term of rhs of hess.system..
      rs(1) = ro ;
      i = 0 ;
 while (i < im & (ro > tol1)& its < maxits )
      %% print its/residual info
      if (outputG)
       fprintf(1,' its %d  res %e \n',its,ro)
      end
      i=i+1 ;
      its = its + 1 ;
      i1 = i + 1 ;
      z = AAx(epsilon,A,B,vv{i});
      vv{i1}=Precfunct(epsilon,A,B,Q,L,U,alpha,z);
%---------- modified GS ;
      for j=1:i

         t = (vv{j}(:))'*(vv{i1}(:)) ;
         hh(j,i) = t ;
           vv{i1} =      vv{i1} - t*vv{j} ;
      end
      t = norm(vv{i1},'fro') ;
      hh(i1,i) = t ;
      if (t  ~= 0.0)
        t = 1.0 / t ;
        vv{i1} = vv{i1}*t ;
      end
%%
    if (i ~= 1)
%---------- apply previous transformations on i-th column of h ;
       for k=2:i
         k1 = k-1 ;
         t = hh(k1,i) ;
         hh(k1,i) = c(k1)*t + s(k1)*hh(k,i) ;
         hh(k,i) = -s(k1)*t + c(k1)*hh(k,i) ;
       end
     end  %% IF
%%
     gam = sqrt(hh(i,i)^2 + hh(i1,i)^2) ;
     gam = max(tolmac,gam);
%---------- determine plane rotation & update rhs of LS Pb
     c(i) = hh(i,i)/gam ;
     s(i) = hh(i1,i)/gam ;
     rs(i1) = -s(i)*rs(i) ;
     rs(i) =  c(i)*rs(i) ;
%---------- determine res. norm. and test for convergence
     hh(i,i) = c(i)*hh(i,i) + s(i)*hh(i1,i) ;
     ro = abs(rs(i1)) ;
     res(its) = ro ;
   end
%---------- compute solution. solve upper triangular system
   rs(i) = rs(i)/hh(i,i) ;
   for  k=i-1:-1:1
      t=rs(k) ;
      for j=k+1:i
         t = t-hh(k,j)*rs(j) ;
      end
      rs(k) = t/hh(k,k) ;
   end
%---------- now form linear combination to get solution
   for j=1:i
      sol = sol +rs(j)* vv{j} ;
   end
```

```
    if ((ro  <=  tol1) | (its >= maxits)) return; end;
end
```

Conclusions and perspectives

In this chapter, we provide a brief review of the main results on saddle point problems with multiple right-hand sides described in the previous chapters. Moreover we give some suggestions on future works that can develop new techniques of preconditionning for solving saddle point problems.

Summary of result

In this thesis, we are interested in solving the saddle point problems, using the preconditioned Krylov subspace methods. The different techniques of preconditionning in this thesis are summarized as the following

1. In chapter 1 we proposed a preconditioned global approach as a new strategy to solve saddle point problems with multiple right-hand sides. This approach help us to devellope a new convergence results of the preconditoned global Krylov subspace methods and create an effecient algorithm for solving this kind of problem.

2. In chapter 2 we are interested in solving saddle point problem arising from the stochastic Galerking finite element discretization of Stokes, Navier-Stokes and Diffusion equations. For solving this type of problem, we have proposed the preconditioned separate approach that based on the Kronecker product. This strategy allows us to create a new technique of preconditionning and the numerical results shows its effectiveness.

3. In chapter 3 we have used the regularized preconditionner to accelerate the convergence rate of the iteration and the Krylov subspace methods. By using the Shur complement we gave a new convergence results of the preconditionned Conjugate Gradient.

Perspectives

Following the work done in this dissertation, we will be interested in developing new approaches for solving the following saddle point problems

$$\underbrace{\begin{pmatrix} A_i & B^T \\ \pm B & 0 \end{pmatrix}}_{\mathcal{A}_i} \underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{X} = \underbrace{\begin{pmatrix} f \\ \pm g \end{pmatrix}}_{b}, \tag{3.43}$$

where $A_i \in \mathbb{R}^{n\times n}$ is non symmetric matrix, $i = 1, 2.., k$, and $B^T \in \mathbb{R}^{n\times m}$ has full column rank (and maybe has a diffecent rank). And

$$\underbrace{\begin{pmatrix} A & B^T & C^T \\ B & 0 & 0 \\ C & 0 & -D \end{pmatrix}}_{\mathcal{A}} \underbrace{\begin{pmatrix} x \\ y \\ z \end{pmatrix}}_{X} = \underbrace{\begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}}_{b}, \tag{3.44}$$

where $A \in \mathbb{R}^{n\times n}$ is symmetric positive definite (SPD) and $B \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{p\times n}$, and $D \in \mathbb{R}^{p\times p}$ are symmetric positive semidefinite (SPS) and possibly zero.

Bibliographie

[1] W.E. Arnoldi. « The principle of minimized iterations in the solution of the matrix eigenvalue problem. » In : *Quarter. Appl. Math.*, 9 (1951), p. 17–29.

[2] Z.-Z. Bai et M. Benzi. « Regularized HSS iteration methods for saddle-point. » In : *BIT Numer. Math.*, 57 (2017), p. 287–311.

[3] Z.-Z. Bai et G.H. Golub. « Accelerated Hermitian and skew-Hermitian splitting iteration methods for saddle-point problems. » In : *IMA J. Numer. Anal.*, 27 (2007), p. 1–23.

[4] Z.-Z. Bai et G.H. Golub. « Shift-splitting preconditioners for saddle point problems. » In : *J. Comput. Appl. Math.*, 272 (2014), p. 239–250.

[5] Z.-Z. Bai, G.H. Golub et M.K. Ng. « Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems. » In : *SIAM J. Matrix Anal. Appl.*, 24 (2003), p. 603–626.

[6] Z.-Z. Bai, G.H. Golub et J.-Y. Pan. « Preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite linear systems. » In : *Numer. Math.*, 98 (2004), p. 1–32.

[7] Z.-Z. Bai, B.N. Parlett et Z.-Q. Wang. « On generalized successive overrelaxation methods for augmented linear systems. » In : *Numer. Math.*, 102 (2005), p. 1–38.

[8] Z.-Z. Bai et Z.-Q. Wang. « Preconditioned Hermitian and skew-Hermitian splitting methods for non-Hermitian positive semidefinite linear systems. » In : *Numer. Math.*, 428 (2008), p. 2900–2932.

[9] Z.-Z. Bai et al. « Block triangular and skew-Hermitian splitting methods for positive-definite linear systems. » In : *SIAM J. Sci. Comput.*, 26 (2004), p. 844–863.

[10] M. Bellalij, K. Jbilou et H. Sadok. « New convergence results on the global GMRES method for diagonalizable matrices. » In : *J. Comput. Appl. Math.*, 219 (2008), p. 350–358.

[11] P. Benner et al. « Efficient solution of large-scale saddle point systems arising in riccati-based boundary feedback stabilization of incompressible stokes flow. » In : *SIAM J. Sci. Comput.*, 35 (2013), p. 150–170.

[12] M. Benzi. « Preconditioning techniques for large linear systems : A survey . » In : *J. Comput. Phys.*, 477 (2002), p. 418–477.

[13] M. Benzi, G.H. Golub et J. Liesen. « Numerical solution of saddle point problems. » In : *Acta Numerica.*, 14 (2005), p. 1–137.

[14] M. Benzi et V. Simoncini. « On the eigenvalues of a class of saddle point matrices. » In : *Numer. Math.*, 103 (2006), p. 173–196.

[15] A. Bissuel et al. *Linear Systems with multiple right-hand sides with GMRES, an application to aircraft design.* Rapp. tech. ECCOMAS congress., 2016.

[16] R. Bouyouli et al. « Convergence proprieties of some block Krylov subspace methods for multiple linear systems. » In : *J. Comput. Appl. Math.*, 196 (2016), p. 498–511.

[17] Y. Cao, J. Du et Q. Niu. « Shift-splitting preconditioners for saddle point problems. » In : *J. Comput. Appl Math.*, 272 (2014), p. 239–250.

[18] Z.-H. Cao. « Augmentation block preconditioners for saddle point-type matrices with singular (1,1) blocks. » In : *Linear Algebra Appl.*, 15 (2008), p. 515–533.

[19] H.S. Dollar et A.J. Wathen. « Approximate factorization constraint preconditioners for saddle point matrices. » In : *SIAM J. Sci. Comput.*, 27 (2006), p. 1555–1572.

[20] L. Elbouyahyaoui, A. Messaoudi et H. Sadok. « Algebraic properties of the block GMRES and block arnoldi methods. » In : *Electron. Trans. Numer. Anal.*, 33 (2009), p. 207–220.

[21] H.C. Elman. « Preconditioners for saddle point problems arising in computational fluid dynamics. » In : *Appl. Numer. Math.*, 43 (2002), p. 75–89.

[22] H.C. Elman et G.H. Golub. « Inexact and preconditioned Uzawa algorithms for saddle point problems. » In : *Appl. Numer. Math.*, 31 (1994), p. 1645–1661.

[23] H.C. Elman et G.H. Golub. « Inexact and preconditioned Uzawa algorithms for saddle point problems. » In : *Appl. Math. Comput.*, 31 (1994), p. 1645–1661.

[24] H.C. Elman et G.H. Golub. « On choices of iteration parameter in HSS method. Appl. » In : *Appl. Numer. Math.*, 31 (1994), p. 1645–1661.

[25] H.C. Elman, A. Ramage et D.J. Silvester. « Algorithm 866 : IFISS, a Matlab toolbox for modelling incompressible flow. » In : *ACM Trans. Math. Softw.*, 33 (2007), Article 14.

[26] H.C. Elman, D.J. Silvester et A.J. Wathen. « Finite elements and fast iterative solvers, with applications in incompressible fluid dynamics ». In : *Oxford university press.*, (2003).

[27] O.G. Ernst et al. « Efficient Solvers for a Linear Stochastic Galerkin Mixed Formulation of Diffusion Problems with Random Data. » In : *SIAM J. Sci. Comput.*, 31 (2009), p. 1424–1447.

[28] N. Gould, D. Orban et T. Rees. « Projected krylov methods for saddle-point system. » In : *SIAM J. Matrix Anal. Appl.*, 35 (2014), p. 1329–1343.

[29] Z.-G Huang, Z. Xu G.-L. Wang et J.-J. Cui. « A modified generalized shift-splitting preconditioner for nonsymmetric saddle point problems., » in : *Numer. Algor.*, 78 (2017), p. 297–331.

[30] Z.-G. Huang, G.-L. Wang et Z. Xu. « A generalized variant of the deteriorated PSS preconditioner for nonsymmetric saddle point problems. » In : *Numer. Algor.*, 75 (2017), p. 1161–1191.

[31] K. Jbilou, A. Messaoudi et H. Sadok. « Global FOM and GMRES algorithms for matrix equation. » In : *Appl. Numer. Math.*, 75 (1999), p. 49–43.

[32] K. Jbilou, H. Sadok et A. Tinzefte. « Oblique projection methods for linear systems with multiple right-hand sides. » In : *Electron. Trans. Numer. Anal.*, 20 (2005), p. 119–138.

[33] M.-Q. Jiang, Y. Cao et L-.Q. Yao. « On parametrized block triangular preconditioners for generalized saddle point problems. » In : *Appl. Math. Comput.*, 216 (2010), p. 1777–1789.

[34] M.Benzi et A.J. Wathen. « Some Preconditioning Techniques for Saddle Point Problems. » In : *Model Order Reduction : Theory, Research Aspects and Applications.*, 13 (2004), p. 195–211.

[35] S. Mercier et al. « A new preconditioner update strategy for the solution of sequences of linear systems in structural mechanics : application to saddle point problems in elasticity . » In : *Computational Mechanics.*, 60 (2017), p. 969–982.

[36] M.F. Murphy, G.H. Golub et A.J. Wathen. « A note on preconditioning for indefinite linear systems. » In : *SIAM J. Sci. Comput.*, 21 (2000), p. 1969–1972.

[37] J. Pestana et A.J. Wathen. « A preconditioned MINRES method for non-symmetric toeplitz matrices. » In : *SIAM J. Matrix Anal. Appl.*, 36 (2015), p. 273–288.

[38] J. Pestana et A.J Wathen. « Combination preconditioning of saddle point systems for positive definiteness. » In : *Linear Algebra Appl.*, 20 (2012), p. 785–808.

[39] J. Pestana et A.J. Wathen. « Natural preconditioning and iterative methods for saddle point systems. » In : *SIAM Rev.*, 57 (2015), p. 71–91.

[40] J. Pestana et A.J. Wathen. « On the choice of preconditioner for minimum residual methods for non-Hermitian matrices. » In : *J. Comput. Appl. Math.*, 249 (2013), p. 57–68.

[41] C.E. Powell et D.J. Silvester. « Preconditioning steady-state Navier-Stokes equations with random data. » In : *SIAM J. Sci. Comput.*, 34 (2012), p. 482–506.

[42] M. Rozloznik et V. Simoncini. « Krylov subspace methods for saddle point problems with indefinite preconditioning. » In : *SIAM J. Matrix Anal. Appl.*, 24 (2002), p. 368–391.

[43] Y. Saad. « A Flexible Inner-Outer Preconditioned GMRES Algorithm. » In : *SIAM J. Sci. Comput.*, 14 (1993), p. 461–469.

[44] Y. Saad et M. Schultz. « GMRES : A Generalised minimal residual algorithm for solving nonsymmetric linear systems. » In : *SIAM J. Sci. statist. Comput.*, 7 (1986), p. 856–869.

[45] H. Sadok. « Analysis of the convergence of the minimal and the orthogonal residual methods. » In : *Numer. Algor.*, 40 (2005), p. 101–115.

[46] D.K. Salkuyeh et M. Masoudi. « A new relaxed HSS preconditioner for saddle point problems. » In : *Numer. Algor.*, 74 (2017), p. 781–795.

[47] J. Schöberl et W. Zulehner. « Symmetric indefinite preconditioners for saddle point problems with applications to PDE-constrained optimization problems. » In : *Numer. Algor.*, 29 (2007), p. 752–773.

[48] M. Stoll et A. Wathen. « Combination preconditioning and the bramble-pasciak preconditioner. » In : *SIAM J. Matrix Anal. Appl.*, 30 (2008), p. 582–608.

[49] D.E. Sturler et J. Liesen. « Block-diagonal and constraint preconditioners for nonsymmetric indefinite linear systems. » In : *SIAM J. Sci. Comput.*, 26 (2005), p. 1598–1619.

[50] X. Wu, B.P.B. Silva et J.Y. Yuan. « Conjugate gradient method for rank deficient saddle point problems. » In : *Numer. Algor.*, 35 (2004), p. 139–154.

[51] L.-J. Zhang et Q.-C. Gu. « A variant of the deteriorated PSS preconditioner for nonsymmetric saddle point problems. » In : *BIT Numer. Math.*, 56 (2016), p. 587–604.

Méthodes de sous espaces de Krylov préconditionnées pour les problèmes de point-selle avec plusieurs seconds membres

Résumé

La résolution numérique des problèmes de point-selle a eu une attention particulière ces dernières années. A titre d'exemple, la mécanique des fluides et solides conduit souvent à des problèmes de point-selle. Ces problèmes se présentent généralement par des équations aux dérivées partielles que nous linéarisons et discrétisons. Le problème linéaire obtenu est souvent mal conditionné. Le résoudre par des méthodes itératives standard n'est donc pas approprié. En plus, lorsque la taille du problème est grande, il est nécessaire de procéder par des méthodes de projections. Nous nous intéressons dans ce sujet de thèse à développer des méthodes numériques robustes et efficaces de résolution numérique de problèmes de point-selle. Nous appliquons les méthodes de Krylov avec des techniques de préconditionnement bien adaptées à la résolution de problèmes de point selle. L'efficacité de ces méthodes à été montrée dans les tests numériques.

Mots clés : point-selle, préconditionnement, krylov, produit de kronecker, produit de diamant

Preconditioned global Krylov subspace methods for solving saddle point problems with multiple right-hand sides

Abstract

In these last years there has been a surge of interest in saddle point problems. For example, the mechanics of fluids and solids often lead to saddle point problems. These problems are usually presented by partial differential equations that we linearize and discretize. The linear problem obtained is often ill-conditioned. Solving it by standard iterative methods is not appropriate. In addition, when the size of the problem is large, it is necessary to use the projection methods. We are interested in this thesis topic to develop an efficient numerical methods for solving saddle point problems. We apply the Krylov subspace methods incorporated with suitable preconditioners for solving these types of problems. The effectiveness of these methods is illustrated by the numerical experiments.

Keywords: saddle point , preconditioner, global krylov subspace method, kronecker product

LMPA& LAMAI
Maison de la Recherche Blaise Pascal – 50, rue Ferdinand Buisson – CS 80699 – 62228 Calais Cedex – France – UCA, FSTG, BP 549, Marrakech – Maroc