ITERATIVE SOLUTIONS OF LARGE-SCALE SOIL STRUCTURE PROBLEMS

TRAN HUU HUYEN TRAN

NATIONAL UNIVERSITY OF SINGAPORE

2014

ITERATIVE SOLUTIONS OF LARGE-SCALE SOIL STRUCTURE PROBLEMS

TRAN HUU HUYEN TRAN

(B.Eng., NUS, Singapore)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF CIVIL AND ENVIRONMENTAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2014

Declaration page

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

ACKNOWLEGEMENTS

First of all, I would like to thank my supervisor, Professor Phoon Kok Kwang. Without his encouragement, his guidance, his advices on several topics, I would not be able to finish this thesis. I, too, greatly appreciate my co-supervisor, Professor Toh Kim Chuan for his patience and gentle guidance. . I am also thankful to the members of my thesis committee, Dr. Chew Soon Hoe, Professor Lee Fook Hou and Associate Professor Tan Sew Ann, for their valuable reviews and advices on my thesis.

I thank my parents, my sister and my big extended family for always supporting and trusting me. A lot of thanks to my close friends Thi, Liên, Bằng, Khôi, Hải, for keep updating me with your travelling activities while I locked myself in the office writing thesis, for cheering me up with all kinds of gossip. Finally I sincerely thank my boyfriend, Chek Khoon, simply for your presence.

My four years of Ph.D would be insufferable and miserable without friends in Geotech Research group. It is the first time in my life I have the feeling of belonging to a group in school. So yes, thanks a lot for your presence, for discussing with me, chatting with me, giving peer pressure to me (well!). So here you are, thanks to: Dr. Cheng Yonggang, Dr. Chen Xi, Dr. Anastiasia Santoso, Dr. Krishna Bahdur Chauhary, Dr. Sindhu Tjahyono, Mr. Lu Yitan, Mr. Tang Chong, Miss Ho Jiahui, Miss Cheng Zongrui, Mr. Zhang Lei, Mr. Chin Bo, Dr. Sun Jie, Ms Ji Jiaming, Dr. Andy Tan, Dr. Tan Chzia Ykaw, Miss Thiri Su, Dr. Ye Feijian, Dr. Chen Jian, Mr. Zhao Ben, Dr. Wu Jun, Miss Li Yuping, Mr. Liu Yong, Mr. Yang Yu.

Table of Contents

Summary

Finite element (FE) method has become extremely popular numerical method in geotechnical engineering. Soil is the main material in geotechnical engineering and very often shows nonlinear and plastic behaviour. Mohr-Coulomb model is a simple, popular and effective constitutive model to simulate the plastic behaviour of soil. When the Mohr-Coulomb model is used in numerical simulation, it is essential to adopt a non-associated flow rule to obtain realistic results. The global stiffness matrix in FE analysis, which is often large in size and highly sparse, becomes nonsymmetric. Little discussion has been focused on the preconditioners for this class of nonsymmetric linear system.

This thesis applies the Induced Dimension Reduction Method (IDR(s)) to solve the large-scale nonsymmetric linear system. This IDR(s) method is shown to be more effective than the current default method, Bi-CGSTAB. In drained analysis, the global stiffness matrix is in form of 1-by-1 block matrix. Incomplete LU factorization with zero fill-in (ILU0) is shown numerically to be the most efficient preconditioner for this matrix among Jacobi, SSOR and ILUT(ρ, τ). In consolidation analysis, the global stiffness matrix is in form of 2-by-2 block matrix. A diagonal block preconditioner is shown to be the most efficient block preconditioner. This diagonal block preconditioner uses ILU0 as the approximation of the soil stiffness matrix and a simple diagonal matrix as the approximation of the Schur complement of the 2-by-2 block matrix

For non-associated MC, nonlinear FE analysis is required and a sequence of large-scale nonsymmetric linear systems has to be solved continuoustly. Two techniques to save the total simulation time in dealing with sequence of nonsymmetric linear systems are recommended for both 1-by-1 and 2-by-2 block matrix as following: 1) Forming the elastoplastic global stiffness matrix implicitly by forming the elastic global stiffness matrix once and update the low-rank matrix at every NR iteration; 2) Updating the preconditioner one time at the beginning of the simulation or updating preconditioners at the beginning of each load steps. When these two techniques are used concurrently, the total simulation time of 1-by-1 block matrix can be reduced 60 percent compared with the default procedure.

List of tables

List of figures

List of symbols

$(.)^{\mathrm{T}}$	transpose of a function
$\lvert . \rvert$	absolute value or modulus of a number
$\lVert . \rVert$	norm of a function
1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
a	real variable
a_{ij}	entry of matrix A
A	general matrix; matrix variable
$\tilde{A}$	preconditioned matrix A
b	right hand side vector
$\tilde{b}$	preconditioned right hand side vector
B	element strain-displacement matrix (Eq.(2.1)); or displacement-pore pressure coupling matrix (Eq.(2.8)); or the width of the shallow foundation
BE	boundary element
Bi-CG	biconjugate gradient
Bi-CGSTAB	biconjugate gradient stabilized
c'	effective cohesion
c_u	Undrained/total cohesion
C	fluid stiffness matrix
CG	conjugate gradient
CGS	conjugate gradient square
CPU	central processing unit
CSC	compressed sparse column
CSR	compressed sparse row

det(.)	determinant of a function
diag(.)	diagonal matrix consisting of leading diagonal entries in argument
D	embedment depth of shallow foundation
D_e	elastic effective stress-strain matrix
D_{ep}	elastoplastic effective stress-strain matrix
$\tilde{D}$	diagonal matrix variable
DOFs	degrees of freedom
E’	effective Young’s modulus
E_u	total Young’s modulus
f	yield surface function
F	applied load, righ-hand-side of the linear system
$\tilde{F}$	righ-hand-side vector modified by preconditioner
FE	finite element
FEM	finite element
g	plastic potential function
G	shear modulus
G_j	nested Krylov subspaces in IDR(s) method
GMRES	generalized minimal residual
H	matrix variable
i, j, k	integer variables
i_tol	tolerance for relative residual
I	identity matrix
I(.)	identity matrix of the size of the argument
I_1	first stress invariant
IDR(s)	induced dimension reduction method
ILU	incomplete LU decomposition

ILU0	incomplete LU factorization with no fill-in
ILU0(.)	ILU0 factorization of the matrix in the argument
ILUT(ρ, τ)	incomplete LU factorization with dual control parameters for fill-in
J_2	second stress invariant
K	global stiffness matrix
$\hat{K}$	approximation of K_{ep}
K_e	elastic global stiffness matrix
K_{ep}	elastoplastic global stiffness matrix
K_w	bulk modulus of water
$\tilde{K}$	preconditioned matrix K
$K(u)$	stress stiffness matrix as a function of u
LOC	lightly over-consolidated
L_A	strictly lower triangular part of A
max(.)	maximum value of a function
min(.)	minimum value of the function
M	(in Figure 1.4) slope of the critical state line
M	preconditioner
M_c	block constrained preconditioner
M_d	diagonal block preconditioner
M_J	Jacobi preconditioner
M_L	left preconditioner
M_R	right preconditioner
$M_{t\text{-}L}$	triangular block preconditioner for left preconditioning
$M_{t\text{-}R}$	triangular block preconditioner for right preconditioning
matvec	matrix-vector multiplication
MC	Mohr-Coulomb model

n	porosity of soil
N	the dimension of total linear system, e.g. the dimension of matrix A
N_γ, N_c, N_q	bearing capacity factors
NC	normally consolidated clay
NCL	normally consolidated line
NK	Newton-Krylov method
NR	Newton-Raphson iterative method
NR_*tol*	Relative tolerance of Newton-Raphson iteration
P	shadow space in IDR(s) method
p^{ex}	excess pore pressure vector
p_0'	reference size of yield locus of Cam-clay model
p_i'	mean effective stress
PC(s)	personal computer(s)
PCG	preconditioned conjugate gradient
q	deviator stress
q_{max}	maximum applied load
q_f	ultimate bearing capacity
QMR	quasi-minimal residual
QMR-GSTAB	Quasi-minimal residual variant of the Bi-CGSTAB algorithm
r	residual vector
$r^{(i)}$	residual vector at the i iteration step
s	dimension of the shadow space in IDR(s) method
S	Schur complement
$\hat{S}$	approximation of S
SSOR	symmetric successive over-relaxation

t_0	maximum loading time
u	displacement vector
$\tilde{u}$	displacement vector modified by preconditioner
u_0	trial displacement vector
x	local spatial coordinate; vector variable
$x^{(0)}$	initial guess of solution
$x^{(i)}$	solution vector at i-th iteration
X	matrix variable
U_A	strictly upper triangular part of A
v	vector variable
w_p	penalty number
y	vector variable
z	integer variable
α	scaling parameter in GJ preconditioner or the increment of percentage of yielded Gauss point
Δ	low-rank matrix
$\Delta(.)$	incremental form of the argument
Δt	time step
$\Delta\lambda$	plastic multiplier
$\Delta\lambda_1$	plastic multiplier from first yield surface in two-vectored return method
$\Delta\lambda_2$	plastic multiplier from second yield surface in two-vectored return method
ε_q	deviator strain
θ	Lode angle or time stepping factor
κ	the swelling or recompression index
$\kappa(.)$	condition number of the argument matrix
λ	(in Figure 1.4) slope of normal compression line in v:

	lnp' plane
λ	eigenvalue
λ_{max}	maximum eigenvalue
λ_{min}	minimum eigenvalue
$\|\lambda\|$	modulus of eigenvalue
$\|\lambda\|_{max}$	maximum modulus of eigenvalue
$\|\lambda\|_{min}$	minimum modulus of eigenvalue
σ_x, σ_y, σ_z	normal stress in X-, Y- and Z-directions, respectively
τ_{xy}, τ_{yz}, τ_{xz}	shear stress in XY-, YZ-, and XZ-directions, respectively
ϕ	friction angle
ψ	dilation angle
ω	input parameter in IDR(s) or relaxation parameter in SSOR
$\partial(.)$	partial derivative of a function
$\sum_i^n .$	summation of a function over the range of index i to n
$\mathbb{C}$	set of complex numbers
$\mathbb{R}$	set of real numbers
$\mathbb{R}^N$	vector space of real N-vectors
$\mathbb{R}^{N\times N}$	vector space of real N-by-N matrices

CHAPTER 1 INTRODUCTION

1.1 Introduction

1.1.1 Three-dimensional finite element analysis and iterative methods

Finite element method (FEM) is an extremely popular numerical method in geotechnical engineering for the last thirty years (Potts & Zdravkovic[137], 1999). Analyses of geotechnical problems using FEM are performed countlessly in research and practice (Migliazza *et al.*[112], 2009; Almeida e Sousa *et al.*[5], 2011; Hashash *et al.*[77], 2011; Lee *et al.*[99], 2011; Hata *et al.*[78], 2012). Several finite element (FE) packages are developed for research purpose such as ICFEP (Potts & Zdravkovic[137], 1999), PECPLAS (Shahrour[153], 1992), SNAC (Abbo & Sloan[4], 2000) and commercial purpose such as GeoFEA[67] (2006), GeoStudio[68] (2012), PLAXIS 2D[133] and 3D[134] (2012).

With the development of underground construction and the computational ability of modern computers, three-dimensional (3D) FE analyses are in great demand to simulate realistic soil structure interactions. Although real geotechnical problems are three-dimensional (3D) in nature, simplified two-dimensional (2D) plane strain or axisymmetric models are preferable in the past due to the lack of graphical interpretation for 3D models and slow computational ability (Augarde & Burd[10], 1995). Now even personal computers (PC) can process 3D models smoothly hence graphical interpretation is not a hindrance. Moreover, certain geotechnical problems cannot be simplified into plane strain or axisymmetric models and require full 3D analyses such as pile-soil interaction (Kahyaoglu *et al.*[91], 2009; Peng *et al.*[127], 2010; Kelesoglu & Springman[93], 2011), deep excavation (Faheem et al.[56], 2004; Zdravkovic *et al.*[186], 2005; Hashash *et al.*[77], 2011; Lee *et al.*[99], 2011), and tunneling process (Mroueh & Shahrour[116, 117], 2003, 2008; Migliazza *et al.*[112], 2009).

FE discretization results in a linear system of the form,

$$Ku = F \tag{1.1}$$

with N is the number of unknown degrees of freedom, $K \in \Re^{N\times N}$ is the stiffness matrix, $u \in \Re^{N}$ is the unknown vector, $F \in \Re^{N}$ is the applied force vector. 3D models are well-known for containing hundreds of thousand unknowns (Lee *et al*.[99], 2011; Hata *et al*.[78], 2012) and the stiffness matrix K is normally large but highly sparse. The large number of unknowns results in long computation time and this is the very hindrance of 3D FEM analysis. This thesis is motivated to reduce this computation time by certain computational techniques.

Theoretically, the exact solution of Eq.(1.1) is

$$u = K^{-1}F \tag{1.2}$$

with K^{-1} denotes the inverse matrix of K. Direct methods can find this exact solution after a fixed number of operations in exact arithmetic (Quarteroni *et al*.[139], 2007). Preferable direct methods are Gauss elimination and its modified forms, which require $O(N^3)$ flops (Isaacson & Keller[84], 1994; Quarteroni *et al*.[139], 2007). When N is in the order of hundreds of thousand as in 3D FE model of geotechnical problems, direct methods are not suitable for solving Eq.(1.2) due to prohibitively expensive computational cost and memory requirement.

Iterative methods and specifically Krylov subspace iterative methods are recommended to efficiently solve large and sparse linear systems (Barrett *et al*.[16], 1994; Saad[144], 2003). Iterative methods aim to generate a series of approximate solution, $x^{(i)}$, that converges to the exact solution (1.2) with any initial guess, $x^{(0)}$. Iterative methods access the linear system through matrix-vector multiplication (matvec) and this operation can be done efficiently when the matrix K is highly sparse as in the case of Eq.(1.1). The iteration process can be stopped when the approximate solution is within some desired accuracy level. This feature is very useful in geotechnical engineering since the system need not be solved to high accuracy because soil is inherently variable hence there are uncertainties in soil properties and soil models (Whitman[176], 2000; Phoon[129], 2008).

Krylov subspace iterative methods are the most popular choice in the 20th century (Saad & Vorst[146], 2000; Gutknecht[75], 2007). Commercial FE softwares like PLAXIS[134] (2012) and ABAQUS[1] (2010) use Krylov iterative methods as linear system solvers. The advantage of Krylov iterative methods over classical stationary methods is that Krylov iterative methods converge to the exact solution in at most N iterations in exact arithmetic (Gurknecht[75], 2007) and normally converge earlier than that. However, N iterations are still expensive when N is in order of hundreds of thousands and with the presence of rounding errors, Krylov methods may require more than N iterations to converge.

Preconditioning is the main technique to accelerate the convergence of Krylov iterative methods (Freund *et al.*[63], 1992; Saad & Vorst[146], 2000; Ferronato[58], 2012). Preconditioning technique is the process of modifying the matrix K to a new matrix $\tilde{K}$ such that the later possesses spectral properties for faster convergence of Krylov iterative methods. It is well known that preconditioners are important in improving the convergence and efficiency of Krylov iterative methods. In geotechnical engineering, preconditioners have only been developed recently for specific geotechnical problems like Biot's consolidation (Chan *et al.*[36], 2001; Phoon *et al.*[130], 2004; Chen *et al.*[42], 2006; Bergamaschi *et al.*[22], 2007; Ferronato *et al.*[59], 2010) and soil-structure interactions (Chauhary[37], 2010). These discussions have been focused on linear elastic material and symmetric linear systems. However, from the practical point of view, linear elastic model is not sufficient to simulate the full range of realistic behaviour of soil. For example, in deep excavations with wall in cantilever mode, many discussions highlight that plastic strain of the soil is generated at very small wall displacement (Jardine *et al.*[85], 1986; Whittle *et al.*[177], 1993; Ou & Kung[122], 2004; Plumey *et al.*[135], 2010). Another example is laterally loaded piles in which plastic zones form at the top of the piles even at relatively low working loads (Liu & Meyerhof[104], 1987; Brown & Shie[31], 1990; Yang & Jeremic[183], 2002; Motta[114], 2013). Besides, it is well-known that soil does fail under certain stress states (Terzaghi[164], 1948; Schofield & Wroth[151], 1968) and this failure definitely cannot be modelled with linear elastic material (Duncan[51], 1994). Hence, it is critical to be aware that the deformation pattern from linear elastic model may not only be "*quantitatively but also qualitatively incorrect*" (Schweiger[152], 2008).

1.1.2 Non-associated plasticity in geotechnical engineering

Linear elastic model gives acceptable solutions only when the strain is small or the safety factor of the system is large enough (Jardine *et al.*[85], 1986; Hicher[82], 1996; Pott & Zdravkovic[138], 2001, p. 169; Leung *et al.*[101], 2010). Nevertheless, soil does not always behave elastically at small strain. Based on the Cam-clay theoretical framework (Roscoe *et al.*[143], 1963; Roscoe & Burland[142], 1968), loading and unloading (swelling) lines of clay are not the same (Figure 1.1) therefore there is plastic strain (irrecoverable deformation) generated during the loading procedure.

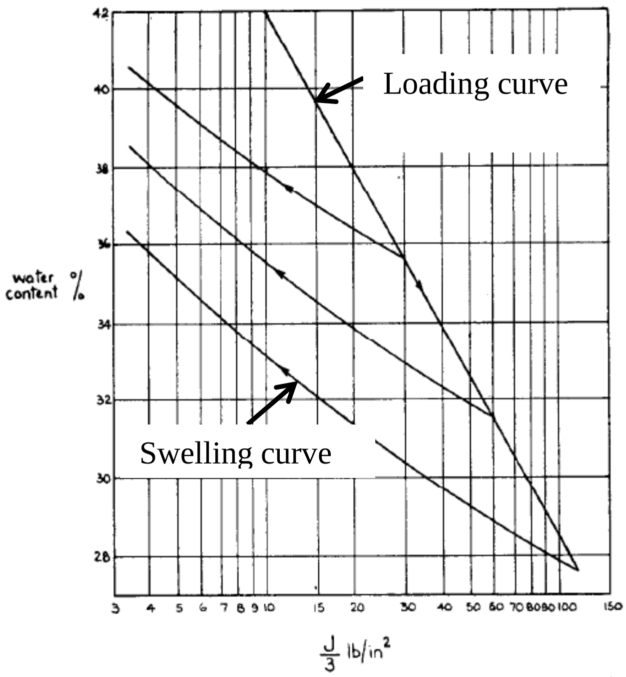


Figure 1.1: Isotropic Consolidation (loading) and Swelling Curves for London Clay (Henkel[79], 1959)

For normally consolidated (NC) clay of which initial stress state lies on normally consolidated line (NCL), Roscoe and others[143] (1963) show that it yields immediately at the initial stress state and does not generate elastic strain during further loading (Figure 1.2). For lightly over-consolidated (LOC) clay,

Figure 1.3 shows there is elastic part in its stress-strain curve but this part is very minor and the elastic strain is very small. Figure 1.4 shows that when the overconsolidation ratio increases, elastic part in the stress-strain curve increases but overall, the elastic strain is very minimal compared to plastic strain.

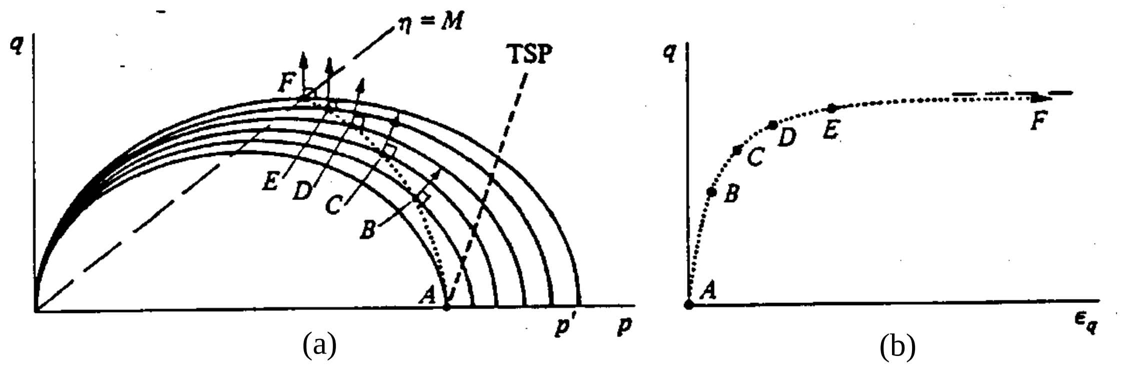


Figure 1.2: Conventional undrained triaxial compression test on NC soil: (a) *p'*: *q* effective stress plane; (b) *q*: ε_q stress: strain plot. (Wood[179], 1991, p.131)

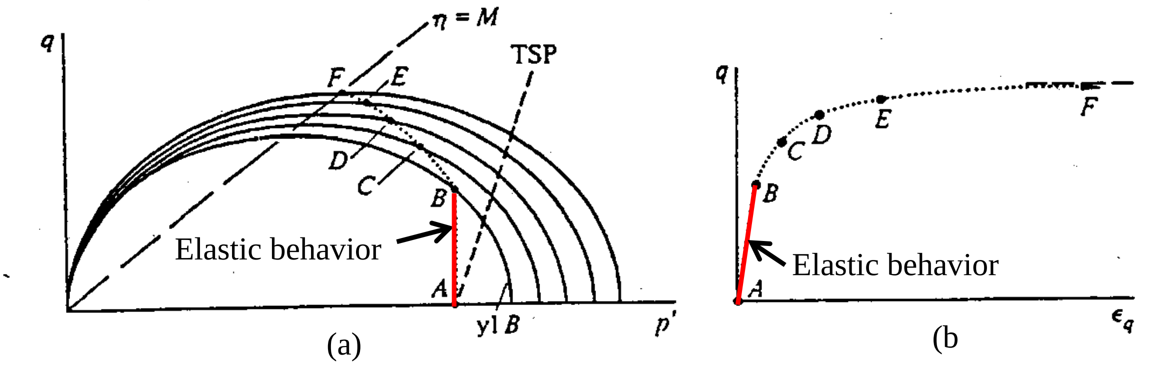


Figure 1.3: Conventional undrained triaxial compression test on LOC soil: (a) *p'*: *q* effective stress plane; (b) *q*: ε_q stress: strain plot. (Wood[179], 1991, p. 132)

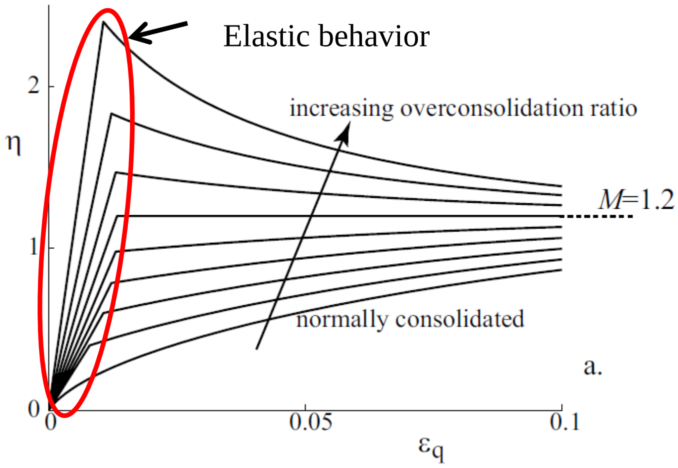


Figure 1.4: Numerical result of Cam clay model: *q*: ε_q stress:strain in drained triaxial compression tests with constant mean stress ($\delta p_0 = 0$) ($\kappa = 0.05$, G = 1500kPa, $\lambda = 0.25$, $M = 1.2$) (overconsolidation ratio p'_0/p'_i in range 1-5, p_0 = 100kPa (Wood[180], 2004, p. 160)

To get more realistic behaviour of soil, models other than linear elastic should be used and Mohr-Coulomb (MC) model is one of the most popular choices. Terzaghi[164] (1948) proposed the use of MC model with two parameters: cohesion and friction angle, to predict the shear resistance of soil. Several experiments were performed to support MC model (Bishop[25], 1966; Parry[125], 1968). MC model is able to give reasonably close results to experimental data or field data for geotechnical problems like piles (Gose *et al*.[72], 1997; Johnson *et al*.[89], 2001; Kahyaoglu *et al*.[91], 2009), deep excavation (Yong *et al*.[185], 1989; Smith & Ho[159], 1992; Bruyn *et al*.[34], 1994; Pakbaz & Zolfagharian[124], 2005; Zvanut *et al*.[188], 2005), and tunnelling (Lee & Rowe[100], 1990; Oettl *et al*.[120], 1998). This model is also used to postulate the failure mechanism of geotechnical systems (Yong *et al*.[185], 1989; Schweiger[152], 2008). Although there are limitations in the model, MC model is popular due to its simplicity and the ease in determining its parameters.

Non-associated flow rule is often used and actually is essential for MC model. This implies that the dilation angle which controls the change in soil volume during shearing is different from the friction angle. Non-associated MC model has been used to re-evaluate failure loads for classic problems like bearing capacity of footing (Manoharan & Dasgupta[107, 108], 1995 1997; Yin *et al*.[184], 2001; Erickson & Drescher[55], 2002; Loukidis & Salgado[105], 2009) and slope stability (Griffiths & Lane[74], 1999; Manzari & Nour[110], 2000; Kumar[97], 2004; Conte *et al*.[45], 2010). For dense sands and overly-consolidated clays which tend to increase volume during shearing (Figure 1.5), experimental data show that their dilation angles are much smaller than the friction angles (Hettler & Vardoulakis[81], 1984; Vermeer & De Borst[168], 1984; Bolton[27], 1986; Houlsby[83], 1991; Schanz & Vermeer[149], 1996). For loose sands which tend to contract during shearing (Figure 1.6), associated flow rule would predict an increase of volumetric strain, which is completely opposite to that produced by real soil behaviour. Besides, Nova[119] (2004) argued that associated flow rule is not suitable for MC due to thermodynamic reasons e.g. no plastic work is dissipated during shearing of soil wedge behind retaining walls. To sum up, non-associated flow rule should be applied when MC model is used to simulate soil behaviour.

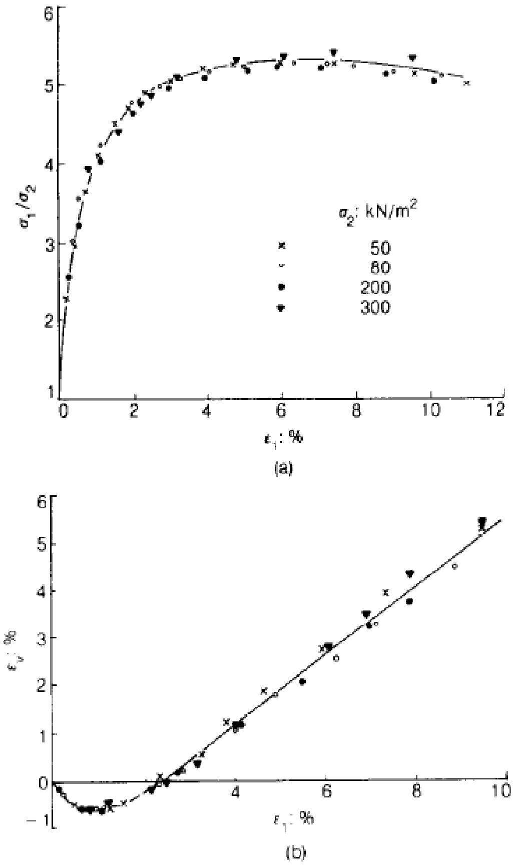


Figure 1.5: Results of triaxial drained test on dense sand (Hettler & Vardoulakis[81], 1984)

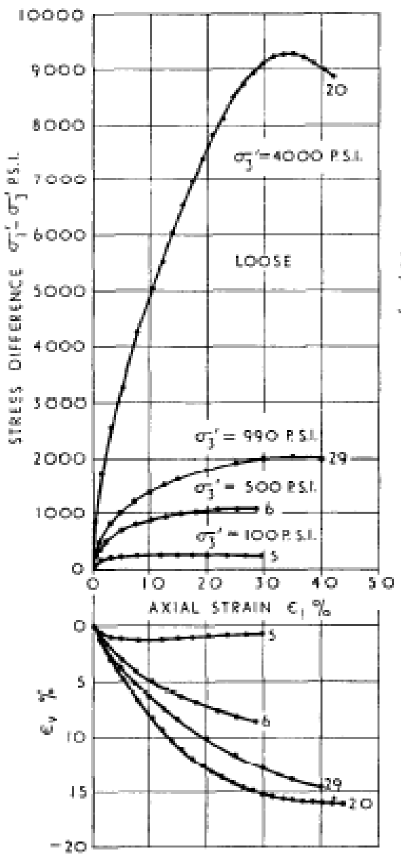


Figure 1.6: Results of triaxial drained tests on saturated Ham River loose sand (Bishop[25], 1966)

1.1.3 Iterative solvers for nonsymmetric linear systems

When the non-associated flow rule is applied, the tangent global stiffness matrix in nonlinear FE analysis, nested within full Newton-Raphson (NR) method, becomes non-symmetric (Owen & Hinton[123], 1980; Potts & Zdravkovic[137], 1999) since the continuum stress-strain matrix D_{ep} in Eq.(1.3) below is nonsymmetric,

$$D_{ep} = D_e - \frac{D_e \left(\frac{\partial g}{\partial \sigma} \right) \left(\frac{\partial f}{\partial \sigma} \right)^T D_e}{\left(\frac{\partial f}{\partial \sigma} \right)^T D_e \left(\frac{\partial g}{\partial \sigma} \right)} \tag{1.3}$$

in which D_e is the elastic stress-strain matrix,

$$\frac{\partial g}{\partial \sigma} = \left\{ \frac{\partial g}{\partial \sigma_x} \quad \frac{\partial g}{\partial \sigma_y} \quad \frac{\partial g}{\partial \sigma_z} \quad \frac{\partial g}{\partial \tau_{xy}} \quad \frac{\partial g}{\partial \tau_{yz}} \quad \frac{\partial g}{\partial \tau_{zx}} \right\}^T$$

$$\frac{\partial f}{\partial \sigma} = \left\{ \frac{\partial f}{\partial \sigma_x} \quad \frac{\partial f}{\partial \sigma_y} \quad \frac{\partial f}{\partial \sigma_z} \quad \frac{\partial f}{\partial \tau_{xy}} \quad \frac{\partial f}{\partial \tau_{yz}} \quad \frac{\partial f}{\partial \tau_{zx}} \right\}^T$$

are the gradients to the plastic potential g and the yield surface f, respectively, and $\sigma = \{\sigma_x \quad \sigma_y \quad \sigma_z \quad \tau_{xy} \quad \tau_{yz} \quad \tau_{xz}\}^T$ is a vector of stress component.

This leads to a non-symmetric global stiffness matrix K_{ep} with the dimension of $N \times N$ in finite element analysis and Eq.(1.1) becomes the following non-symmetric linear system

$$K_{ep} u = F \tag{1.4}$$

in which with $u \in \Re^N$ is an unknown vector and $F \in \Re^N$ is the applied force vector.

This nonsymmetric system can be avoided by switching to a modified NR method (or initial stress method in engineering term). This method uses the same symmetric stiffness matrix for every NR iteration and therefore the global stiffness matrix is only computed once. However, such a modified NR method has convergence difficulty for strongly non-linear problems (Bathe & Cimento[17], 1980; Bonet & Wood[28], 2008; Crisfield[47], 1998; Lewis & Schrefler[102], 1998; Wriggers[181], 2008; ABAQUS theory manual[1], 2010) which appear frequently in geotechnical engineering (Jardine *et al.*[85], 1986; Zdravkovic *et al.*[186], 2005). Hence, full NR is still a recommended method for nonlinear FE (Bonet & Wood[28], 2008; Lewis & Schrefler[102], 1998). Since the non-associated flow rule is essential in MC model, solving the sparse nonsymmetric linear system is unavoidable.

As mentioned in Section 1.1.1, recent discussions all have focused on solving sparse symmetric linear system using Krylov subspace iterative methods. When the linear system is nonsymmetric, the difficulty is not only that the storage memory is doubled but more critically, current iterative solvers and preconditioners that are developed for symmetric systems are no longer optimal or – worst – no longer suitable. The apparent impact of the nonsymmetry on iterative solvers is two matrix-vector multiplications are required in each iteration because the symmetry can no longer be exploited. This leads to the total iteration time is at least doubled because matrix-vector multiplication is the most time-consuming operation. The impact of the nonsymmetry on preconditioners is rather less apparent. Preconditioners aim to accelerate the convergence of iterative solvers hence aim to modify the convergence governing parameters. The convergence of iterative solvers depends on the eigenvalue distribution of the coefficient matrix. When the coefficient matrix is symmetric, the eigenvalues are all real numbers and the convergence is mostly governed by the spectral radius which is the ratio of the maximum eigenvalue over the minimum eigenvalue. The available preconditioners were designed to minimize this spectral radius. However when the coefficient matrix is nonsymmetric, some eigenvalues are complex numbers and the spectral radius becomes less meaningful. This point will be re-establish in Section 1.1.4 and Section 2.4.1.

Preconditioned conjugate gradient (PCG) is one of the most effective iterative solvers for symmetric positive-definite (SPD) linear system. This method can be used to solve the non-symmetric system $K_{ep}u = F$ by solving $K_{ep}^T K_{ep} u = K_{ep}^T F$ instead (Eisenstat *et al.*[54], 1983; Barrett *et al.*[16], 1994). However, this technique is memory and computational expensive since not only the matrix-vector multiplication (matvec), $K_{ep}v$ but also the transpose –vector multiplication, $K_{ep}^T v$ is required at each iteration. Moreover, the convergence of PCG can be very slow (Eisenstat *et al.*[54], 1983; Barrett *et al.*[16], 1994; Kelley[94], 1995) since the condition number of the matrix $K_{ep}^T K_{ep}$ is the square of the condition number of K_{ep} (Kelley[94], 1995) and the eigenvalues of $K_{ep}^T K_{ep}$ can be more scattered than those of K_{ep} (Weiss[173], 1995). Nevertheless, Freund and others[63] (1992) noted that solving $K_{ep}^T K_{ep} u = K_{ep}^T F$ is optimal for skew-symmetric or shifted skew-symmetric matrices but K_{ep} matrix from FE discretization does not belong to these classes. Hence this method is not optimal and is not considered in this thesis. Besides, it may be tempted to use PCG to solve $K_{ep}u = F$ directly when K_{ep} is a weakly non-symmetric matrix. Borja[29] (1991) applied this technique and achieved convergence on his systems. However PCG is strictly developed for SPD linear system and there is no theoretical guarantee that it will converge for weakly non-symmetric matrix.

There are Krylov iterative methods specifically developed to solve nonsymmetric linear systems. The popular ones are GMRES (Saad & Schultz[145], 1986), Bi-CG (Fletcher[61], 1976), CGS (Sonneveld[160], 1989), QMR (Freund & Nachtigal[62], 1991) and Bi-CGSTAB (Vorst[170], 1992). Among these, GMRES and Bi-CGSTAB are the most prominent methods (Pillis[132], 1998; Sonneveld & Gijzen[162], 2008; Ferronato[58], 2012). GMRES is a very efficient method which finds the minimum residual norm over the Krylov subspace spanned, and hence it offers the "lower bound" solution for all Krylov iterative methods (Kelley[94], 1995). Although GMRES is mathematically elegant, it is practically expensive since a new set of orthogonal vectors has to be formed and stored at every iteration (Barrett *et al*.[16], 1994; Saad[144], 2003). Therefore GMRES is not suitable for large-scale problems. Currently, Bi-CGSTAB is the most practical method to solve large sparse nonsymmetric linear systems.

Induced Dimension Reduction (IDR(s)) is a recently developed method based on IDR theorem and is consider competitive with Bi-CGSTAB on some simple test problems done by Sonneveld and Gijzen[162, 70] (2008, 2010). The parameter s is the number of columns of the shadow matrix $P^{N\times s}$ and the upper bound of dimension reduction (refer to Section 2.1.1for the detail elaboration). It is known that in exact arithmetic, IDR(1) and Bi-CGSTAB are mathematically equivalent while IDR(s) with $s > 1$ often converges faster than Bi-CGSTAB does. Bi-CGSTAB has been shown to be related to IDR(s) method and actually its algorithm can be expressed in the way similar to IDR(s) (Sleijpen *et al*.[170], 2010). More importantly, in exact arithmetic, IDR(s) can compute the solution of an $N \times N$ nonsymmetric linear system in $N\left(1+\frac{1}{s}\right)$ matvec (at the expense of forming and solving an $s \times s$ linear system in each iteration) in contrast to the $2N$ matrix-vector multiplications required by the Bi-CGSTAB method. Comparisons of IDR(s) versus Bi-CGSTAB and GMRES have been done on some large-scale nonsymmetric linear systems resulted from finite difference discretization of quantum mechanics equation (Jing *et al*.[88], 2010), of Helmholtz equations (Umetani *et* *al.*[167], 2009; Knibbe *et al.*[95], 2011), and boundary element (BE) discretization of elastodynamics (Xiao *et al.*[182], 2012). These comparisons conclude that: a) the convergence behavior of IDR(*s*) is similar to that of GMRES while the former requires less memory; b) with effective preconditioner like incomplete LU (ILU), IDR ($s > 1$) converges faster than Bi-CGSTAB; and c) more importantly, there are cases where IDR converges well while Bi-CGSTAB does not converge. From all the above, it is of interest for us to investigate whether the IDR(*s*) method has any substantial competitive advantage over the default Bi-CGSTAB solver on large-scale geotechnical problems.

1.1.4 Preconditioners for nonsymmetric linear systems

Section 1.1.1 has noted that preconditioning is the crucial technique to keep Krylov iterative methods converge in a practical span of time. Preconditioners transform the linear system (1.4) into (1.5),

$$\tilde{K}\tilde{u} = \tilde{F} \tag{1.5}$$

in which $\tilde{K}$ is the preconditioned K_{ep}, $\tilde{u}$ and $\tilde{F}$ are modified versions u and F respectively by the preconditioner M. With $M = M_L M_R$, there are three different ways to precondition K_{ep}: left preconditioning, right preconditioning and left-right preconditioning as presented in Eq.(1.6), (1.7), and (1.8) respectively. Right preconditioning, Eq. (1.7), is often preferred because the right-hand-side F does not require modification.

$$\left(M^{-1}K_{ep}\right)u = M^{-1}F \tag{1.6}$$

$$\left(K_{ep}M^{-1}\right)\left(Mu\right) = F \tag{1.7}$$

$$\left(M_L^{-1}K_{ep}M_R^{-1}\right)\left(M_R u\right) = M_L^{-1}F \tag{1.8}$$

Solving Eq.(1.5) with Krylov iterative methods involves the matrix-vector multiplication $\tilde{K}v$ hence, involves solving $Mu = \tilde{u}$. An efficient preconditioner is a balance between the two conflicting criteria: it should, first, approximate matrix K_{ep} well enough so that Krylov iterative methods converge in less iterations, and second, be simple enough so that $Mu = \tilde{u}$ can be solve quickly (Freund *et al.*[63], 1992). This makes the search for an efficient preconditioner challenging especially with the lack of theoretical results (Ferronato[58], 2012).

Section 1.1.1 also noted that current available preconditioners for geotechnical problems are developed from the symmetric linear system arising when the soil follows a linear elastic model. Preconditioners are also developed for the 2-by-2 block symmetric linear system from Biot's consolidation analysis. Phoon and co-workers[131, 130] (2002, 2004) exploited the structure of this block matrix and introduced several preconditioners like Generalized Jacobi (GJ), Modified Symmetric Successive Over-Relaxation (MSSOR) and block preconditioners (Toh *et al*.[166], 2004; Chauhary[37], 2010). While Gambolati and co-workers[64, 65, 66] (2001, 2002, 2003) discussed the use of incomplete LU decomposition (ILU) and incomplete Cholesky decomposition (IC) type preconditioners. However, the optimal ILU or IC preconditioners depend on fill-in parameters while these parameters are not known a priori. Nevertheless, it is of interest to apply ILU preconditioners on the nonsymmetric linear systems Eq.(1.3).

The convergence of Krylov iterative methods for symmetric positive definite linear systems is primarily governed by the condition number, which is equal to the ratio of the maximum eigenvalue $\lambda_{\max}$ over the minimum eigenvalue $\lambda_{\min}$, of the symmetric matrix (Saad[144], 2003). Hence the objective of preconditioning is only to reduce the condition number by making the eigenvalues cluster at some points. Whereas the convergence of Krylov iterative methods for nonsymmetric linear systems is more complicated and governed by quantities that cannot be computed explicitly for general case (Freund et al.[63], 1992; Driscoll *et al*.[50], 1998; Saad[144], 2003). Therefore the process of developing an efficient preconditioner for nonsymmetric linear systems is rather empirical (Ferronato[58], 2012). When the soil follows a linear elastic model, the symmetric global stiffness matrix is constant, and hence the preconditioner can be fixed for a certain problem. But when the soil follows the non-associated MC model, the nonsymmetric global stiffness matrix changes with the increase of the number of yielded Gauss points and preconditioners have to be redesigned to accommodate these changes.

Discussion of preconditioners for nonsymmetric linear system in geotechnical problems is mostly limited to 1-by-1 block matrix from drained analysis. Traditional preconditioners like Jacobi, SSOR and ILU are often used

(Almeida & Paiva[6], 2004; Wieners *et al*.[178], 2005; Ribeiro & Ferreira[141], 2007; Jeremic & Jie[86], 2008). Mroueh and Sharour[115] (1999) did survey on BiCG, Bi-CGSTAB and QMR-CGSTAB methods to solve non-symmetric linear systems arising from shallow foundation, laterally loaded pile and tunnelling process when the soil follows a non-associated MC model. The study used Jacobi and SSOR preconditioners and recommends the use of SSOR as a left preconditioner. Payer and Mang[126] (1997) used CGS, GMRES, and Bi-CGSTAB method with SSOR and ILU preconditioners for the coupling 3D BE-FE analysis of tunnel driving problem. The soil followed a hardening capped model developed from Druker-Prager model. Numerical experiments showed that GMRES and BiCGSTAB are competitive solvers.

White and Borja[175] (2011) have recently applied the block preconditioner proposed by Toh *et al*.[166](2004) in solving the nonsymmetric 2-by-2 block linear system resulted from the study of fluid flow through porous media. The nonsymmetry is due to the non-associated Drucker-Prager model of the porous media. Chen and Phoon[41] (2012) have also given an extended discussion on the application of MSSOR preconditioner to Biot's consolidation problem when the soil follows a non-associated MC model.

1.2 Objective and Scope of the study

The specific objectives of this study can be summarized as follows.

1. To compare the efficiency of IDR(s) and Bi-CGSTAB method with different preconditioners in solving the drained shallow foundation.
2. To investigate the efficiency of preconditioners on drained analysis and show that the total solution time can be greatly reduced by forming the global stiffness matrix implicitly, where K_e is formed only once, and the second term (denoted as Δ) is computed and stored separately from K_e in each NR iteration.
3. To investigate the efficiency of block preconditioners on Biot's consolidation analysis.
4. To evaluate the effectiveness of the proposed preconditioners in the context of realistic large-scale soil-structure interaction problems.

This thesis only discusses the preconditioner related to the assembled global stiffness matrix, often known as "global preconditioner". There is a class of preconditioner call element-by-element (EBE) preconditioner which preconditions the matrix-free analysis. This type of preconditioner is more suitable to parallel simulation while this thesis focuses on PC simulation hence EBE is not discussed in this thesis. Sparse approximate inverse is another type of preconditioner which has recently been popular. This preconditioner is designed and often used with GMRES method, which is not a very practical method for 3D geotechnical problems as discussed in Section 1.1.3, hence is also not discussed here.

1.3 Computer hardware and software

All the numerical experiments in this report are carried out on a DELL Intel Core i7 CPU, 3.4GHz PC with 16GB of RAM running on a Windows 7 operating systems.

The FORTRAN source codes for 3D FEM drained problem with Mohr-Coulomb soil model are based on the 2D version given by Smith and Griffiths[158] (2004). The FORTRAN source codes for 3D FEM Biot's consolidation problems are based on research work by Chen[39] (2005) and Chauhary[37] (2010). The FORTRAN codes are programmed with Intel Visual FORTRAN Compiler 10.1, Professional Edition.

1.4 Thesis outline

This thesis is divided into following chapters. Chapter 2 provides a brief overview of iterative methods used in this thesis and review of various preconditioners for 1-by-1 block matrix and 2-by-2 block matrix as well as the convergence criteria of Krylov iterative methods. Chapter 3 compares the performance of recently developed IDR(*s*) and Bi-CGSTAB method with various traditional preconditioners to recommend the most optimal preconditioner for the 1-by-1 block nonsymmetric linear system coming from the non-associated MC model. Chapter 4 discusses the techniques to exploit the structure of the elastoplastic stiffness K_{ep} and scheme to update preconditioners for 1-by-1 block matrix with examples from drained analysis and undrained analysis. Chapter 5 compares the performance of existing block preconditioners on Biot's consolidation analysis of which elastoplastic stiffness matrix is a 2-by-2 block matrix. The application of these preconditioners on practical examples is demonstrated in Chapter 6. Finally, Chapter 7 offers some general conclusion with recommendations for the further study.

CHAPTER 2 LITERATURE REVIEW

2.1 Induced Dimension Reduction (IDR) method

2.1.1 Overview of IDR(*s*) method

IDR(*s*) method was proposed by Sonneveld and Gijzen[162] in 2008 based on IDR theorem (Wesseling & Sonneveld[174], 1980). IDR theorem is given in Figure 2.1 and its proof can be found in the paper by Sonneveld and Gijzen[162] (2008). This theorem defines a sequence of subspaces $\{G_j\}_{j=0}^{N}$ with two properties: (i) these subspaces are nested; and (ii) when *j* increases, there is either a reduction in dimension of G_j or $G_j = \{0\}$.

Let A be any matrix in $C^{N\times N}$, let v_0 be any nonzero vector in C^N, and let G_0 be the full Krylov space $K^N(A, v_0)$. Let S denote any (proper) subspace of C^N such that S and G_0 do not share a nontrivial invariant subspace of A, and define the sequence G_j, $j = 1, 2, \ldots$, as $$G_j = (I - \omega_j A)(G_{j-1} \cap S)$$ where the ω_j's are nonzero scalers. Then the following hold: (i) $G_j \subset G_{j-1}\ \forall\, j > 0$ (ii) $G_j = \{0\}$ for some $j \le N$

Figure 2.1: IDR theorem (Sonneveld & Gijzen[162], 2008)

For solving a linear system of equations $Ax = b$ with an $N \times N$ coefficient matrix A, the IDR(*s*) method works by projecting residuals into a sequence of nested subspaces $\{G_j\}_{j=0}^{N}$ of reducing dimensions, with $G_0 = \text{span}(r_0, Ar_0, \ldots, A^N r_0)$ being the full dimensional Krylov subspace associated with the initial residual r_0. According to IDR theorem, these nested subspaces are constructed as $G_j = (I - w_j A)(G_{j-1} \cap P^{\perp})$ where $P^{\perp}$ is the orthogonal complement of the range of a fixed $N\times s$ matrix P, often known as shadow space, and w_j is a nonzero scalar. Sonneveld and Gijzen[162] (2008) proved that *s* is the upper bound of the dimension reduction of Gj when *j* increases. This leads to the observation that in exact arithmetic, IDR(*s*) can compute the solution of an *N*

$\times$ N nonsymmetric linear system in $N\left(1+\frac{1}{s}\right)$ matvec (at the expense of forming and solving an $s \times s$ linear system in each iteration). Figure 2.2 presents the pseudo-code of the preconditioned IDR(s) method following Gijzen and Sonneveld[70] (2010).

IDR(1) is mathematically equivalent to Bi-CGSTAB of which pseudo-code is presented in Figure 2.3 (Sleijpen *et al*.[156], 2010). IDR(s) with $s > 1$ is more efficient than Bi-CGSTAB in some examples shown by Sonneveld and Gijzen[162, 70] (2008, 2010) when comparing both matvec count and total iteration time. Jing and others[88] (2010) performed detailed comparisons of IDR(s) with s = 1, 2, 4, 6, 8 and other Krylov iterative methods: CGS, Bi-CGSTAB, full GMRES, restarted GMRES(m) with m = 50, 100, 200. These methods were used to solve the nonsymmetric linear system resulted from finite difference discretization of a three-body problem in quantum mechanics. IDR(4) was shown to require the least time to converge. Umetani *et al*.[167] (2009) and Knibbe *et al*.[95] (2011) compared IDR(2), IDR(4) and Bi-CGSTAB in solving the nonsymmetric linear system resulted from finite difference discretization of the two-dimensional (2D) Helmholtz equation. Multigrid preconditioner was used with IDR(4) and Bi-CGSTAB. Both discussions found that the time IDR(4) requires to converge is marginally less than that required by Bi-CGSTAB. Xiao and other[182] (2012) compared IDR(s) with s = 8, 10, 20 with full GMRES and restarted GMRES(50) in solving the nonsymmetric linear system resulted from boundary element (BE) discretization of elastodynamics problem. The numerical results shown that IDR(s) required less storing memory but more iterations to converge than full GMRES and restarted GMRES did. Because more iterations were required, IDR(s) consumed more time than full GMRES in the tested problems but the differences were marginal. This may be because the linear system resulted from BEM is dense so it is time consuming to compute one matvec, which may not be the case for FE discretization considered in this thesis.

Compute $r^{(0)} = b - Ax^{(0)}$
$P \in C^{N\times s}$; $g_i = u_i = 0 \in C^N$, $i = 1,\ldots s$; $B = I \in C^{s\times s}$; $\omega = 1$
while $\|r\| > tol$
    $f = P^H r, (\phi_1, \ldots, \phi_s)^T = f$
    for $k = 1, \ldots, s$
        Solve $Bc = f, (\gamma_1, \ldots, \gamma_s)^T = c$
        $v = r - \sum_{i=k}^{s} \gamma_i g_i$
        $v = M^{-1} v$
        $u_k = \omega v + \sum_{i=k}^{s} \gamma_i u_i$
        $g_k = A u_k$
        for $i = 1,\ldots,k-1$
            $\alpha = p_i^H g_k / \mu_{i,i}$
            $g_k = g_k - \alpha g_i$
            $u_k = u_k - \alpha u_i$
        end for
        $\mu_{i,k} = p_i^H g_k, B_{i,k} = \mu_{i,k}, i = \overline{k,s}$
        $\beta = \phi_k / \mu_{k,k}$
        $r = r - \beta g_k$
        $x = x + \beta g_k$
        if $k + 1 \le s$
            $\phi_i = 0, i = \overline{1,k}$
            $\phi_i = \phi_i - \beta \mu_{i,k}, i = \overline{k+1,s}$
        end if
    end for
    $v = M^{-1} r$
    $t = Av$
    Calculation of ω using "maintaining the convergence" strategy
    $\omega = t^H r / t^H t$
    $\rho = t^H r / (\|t\| \|r\|)$
    if $|\rho| \le \kappa$
        $\omega = \omega\kappa / |\rho|$
    end if
    $r = r - \omega t$
    $x = x + \omega t$
end while

Figure 2.2: Preconditioned IDR(s)-biortho with preconditioner M (Gijzen & Sonneveld[70], 2010)

Compute $r^{(0)} = b - Ax^{(0)}$
Choose $\tilde{r}^{(0)} = r^{(0)}$
for $i = 1,2 \ldots maxit$
    $\rho_{i-1} = \tilde{r}^T r^{(i-1)}$
    if $\rho_{i-1} = 0$ **method fails**
    if $i = 1$
        $p^{(i)} = r^{(i-1)}$
    else
        $\beta_{i-1} = (\rho_{i-1}/\rho_{i-2})(\alpha_{i-1}/\omega_{i-1})$
        $p^{(i)} = r^{(i-1)} + \beta_{i-1}(p^{(i-1)} - \omega_{i-1}v^{(i-1)})$
    end if
    Solve $M\hat{p} = p^{(i)}$
    $v^{(i)} = A\hat{p}$
    $\alpha_i = \rho_{i-1}/\tilde{r}^T v^{(i)}$
    $s = r^{(i-1)} - \alpha_i v^{(i)}$
    Check norm of s; if small enough: set $x^{(i)} = x^{(i-1)} + \alpha_i \hat{p}$ and STOP
    Solve $M\hat{s} = s$
    $t = A\hat{s}$
    $\omega_i = t^T s / t^T t$
    $x^{(i)} = x^{(i-1)} + \alpha_i \hat{p} + \omega_i \hat{s}$
    $r^{(i)} = s - \omega_i t$
    Check convergence; continue if necessary
    For continuation it is necessary that $\omega_i \neq 0$
end for

Figure 2.3: Preconditioned BiCGSTAB method with preconditioner M (Barrett *et al.*[16], 1994)

2.1.2 Implementation of IDR(s)

From Figure 2.2, an important input of IDR(s) is the shadow matrix $P^{N \times s}$ containing s shadow vectors. Sonneveld and Gijzen[162] (2008) recommended the use of random matrix with orthorgonalized columns. They noted that using matrix P in relation to the problem does not improve the convergence of IDR(s) but even worsen its performance. The better choice of P besides random matrix has not yet been found as noted by Sonneveld[161] in 2012. This thesis follows this recommendation and employs the random matrix P of which entries are random number uniformly distributed from 0 to 1. However, the orthogonalization process such as Gram-Schmidt (Saad[144], 2003, pp. 10-15) is time consuming and numerical experiments in this thesis show that with an efficient preconditioner, IDR(s) converges well without this extra process. Sonneveld and Gijzen[162] (2008) noticed that a random matrix P with complex numbers is a good mitigation when IDR(s) convergence is poor. This option is expensive for the problems studied in this thesis where all data are real numbers hence is not implemented and the convergence of IDR(s) is accelerated by preconditioners, which is the main objective of this thesis.

Note that the dimension (s) of P certainly affects the convergence of IDR(s). As mentioned in Section 2.1.1, in exact arithmetic, IDR(s) converges to the exact solution in at most $N\left(1+\frac{1}{s}\right)$ matvec. Thus IDR(s) is expected to converge faster when s increases but at the cost of solving a larger $s \times s$ linear system in each iteration. Hence, the optimal value of s should compromise both the convergence rate and the overhead time. Sonneveld and Gijzen[162, 70] (2008, 2010) recommended s = 4 based on their numerical experiments. However numerical experiments in this thesis show that s = 6 is more optimal for the nonsymmetric linear system arising from the non-associated MC model.

2.2 Preconditioners for 1-by-1 nonsymmetric block matrix

2.2.1 Nonsymmetric linear systems resulted from drained and undrained analysis

Section 1.1.2 has introduced that this thesis considers the nonsymmetric linear system arising from geotechnical problems where the soil following the non-associated Mohr-Coulomb model. The linear systems due to FE discretization in drained and undrained analysis are often handled as a 1-by-1 block matrix, in contrast with the 2-by-2 block matrix in Biot's consolidation discussed in Section 2.3. The FE discretization of drained analysis is

$$K_{ep}u = \sum_{\text{element}} \left[\int_V B^T D_{ep} B dV \right] u = F \tag{2.1}$$

in which V is the elementary volume body, B is the element strain-displacement matrix, and D_{ep} is the elastoplastic stress-strain matrix given in Eq.(1.3). While the FE discretization of undrained analysis using effective stress parameters is

$$K_{ep}u = \sum_{\text{element}} \left[\int_V B^T \left(D_{ep} + \frac{K_w}{n} mm^T \right) BdV \right] u = F \tag{2.2}$$

in which K_w is the bulk modulus of water, n is the porosity of soil and $m^T = \{1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0\}$ for 3D analysis. Figure 2.4 plots the sparsity pattern of these matrixes. In geotechnical engineering, few discussions are available on preconditioners for the nonsymmetric linear systems in Eq.(2.1) and (2.2). When preconditioner is required, Jacobi or SSOR or sometimes ILU is used. In general discussion about preconditioner, Jacobi, SSOR and ILU are still the most popular preconditioners for the 1-by-1 block matrix.

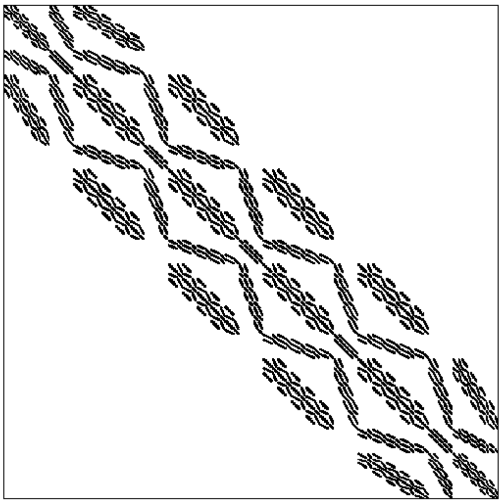


Figure 2.4: Sparsity pattern of 1-by-1 block matrix

2.2.2 Jacobi and SSOR Preconditioners

Jacobi preconditioner is the diagonal matrix containing diagonal entries of K_{ep} (Eq.(2.3)). This is the cheapest preconditioner because it is easy to form, requires less memory and $M_J u = \tilde{u}$ is easy to solve.

$$M_J = D_K \tag{2.3}$$

Jacobi preconditioner performs diagonal scaling of the original matrix which is often quite effective in reducing the condition number $\kappa(M^{-1}K_{ep})$. Jacobi preconditioner is the cheapest but also the crudest approximation of the matrix, hence possesses the lowest efficiency. Jacobi preconditioner is often resorted as a quick and cheap tool to accelerate as well as, hopefully, preserve convergence of Krylov iterative methods when solving the nonsymmetric linear systems is not the main but an unavoidable process, or when preconditioner is not the main objective of discussion (Jiang *et al.*[87], 1994; Sheu *et al.*[154], 1999; Almeida & Paiva[6], 2004; Araujo *et al.*[7], 2004; Pontaza & Reddy[136], 2004)

Jacobi preconditioner is an acceptable approximation when the matrix is diagonally dominant. There is a scaling form of Jacobi preconditioner when the matrix is not diagonally dominant or the diagonal entries are of different scale such as the case of Biot's consolidation equations. Scaling is introduced for this case and called Generalized Jacobi (GJ) preconditioner, which is discussed in detailed in Section 2.3.3.

SSOR preconditioner is a better approximation of K_{ep} than Jacobi preconditioner and takes the following form,

$$M_{SSOR} = \frac{1}{2-\omega}\left(\frac{1}{\omega}D_K + L_K\right)\left(\frac{1}{\omega}D_K\right)^{-1}\left(\frac{1}{\omega}D_K + U_K\right) \tag{2.4}$$

in which D_K is the diagonal matrix containing diagonal entry of K_{ep}, L_K is the strictly lower triangular matrix of K_{ep} and U_K is the strictly upper triangular matrix of K_{ep} and ω is the relaxation parameter which is real a number between (0; 2). SSOR is also regarded as an incomplete LU factorization (Eisenstat[53], 1981; Bank & Douglas[15], 1985; Saad[144], 2003, pp. 285-287) but it is not as efficient as the incomplete LU factorization, ILU (arising from Gaussian elimination), discussed in Section 2.2.3. This is because SSOR does not approximate K_{ep} as well as ILU: the error matrix $K_{ep} - M_{SSOR}$ is generally larger than $K_{ep} - M_{ILU}$.

Being a better approximation of K_{ep}, SSOR is often more efficient than Jacobi preconditioner. SSOR is popular because it is easy and fast to apply and consumes little memory when compared with the ILU preconditioner. Chen and others[38] (2004) recommended the use of SSOR for GMRES to solve the nonsymmetric linear system from FE discretization of waveguide discontinuities with anisotropic dielectric. Stute and others[163] (2013) recently have used SSOR with GMRES and Bi-CGSTAB to demonstrate the superiority of Krylov iterative methods over the direct solvers.

There is an optimal value of ω with which SSOR works best but this optimal value depends on the eigenspectrum of K_{ep}, which is expensive to compute in advance (Barrett *et al.*[16], 1994). Hence the optimal value of ω is practically unknown at the start of the simulation. Payer and Mang[126] (1997) used SSOR as a quick-to-use preconditioner to demonstrate the application of Krylov iterative methods in solving the nonsymmetric linear system from 3D FE-BE analysis. Their numerical experiments showed that the convergence of Krylov iterative methods was rather insensitive to the value of ω in the SSOR preconditioner. Bruaset[33] (1997) also noted that SSOR as a preconditioner is not as sensitively affected by the value of ω as SSOR as an iterative method. For all the above reasons, this thesis chooses $\omega = 1$ for SSOR preconditioner, often known as simple SSOR.

SSOR is often exploited with Eisenstat's trick (Eisenstat[53], 1981) as a left-right preconditioning technique and denoted as SSOR-LR (Eq.(2.5)) in this thesis. Procedure to compute $t = \tilde{K}v$ for SSOR-LR is presented in Eq.(2.6).

$$\begin{gathered} M_L = (L_K + D_K);\ M_R = D_K^{-1}(U_K + D_K) \\ \tilde{K} = (L_K + D_K)^{-1} K (D_K + U_K)^{-1} D_K \end{gathered} \tag{2.5}$$

$$\begin{aligned} &f = (U_K + D_K)^{-1} w \text{ where } w = D_K v \\ &g = D_K f + w \\ &h = (L_K + D_K)^{-1} g \\ &t = f + h \end{aligned} \tag{2.6}$$

SSOR is sometimes used as a left preconditioner (Mroueh & Shahrouh[115], 1999), denoted as SSOR-L in Eq.(2.7).

$$\begin{gathered} M_L = (L_K + D_K) D_K^{-1} (U_K + D_K) \\ \tilde{K} = (D_K + U_K)^{-1} D_K (L_K + D_K)^{-1} K \end{gathered} \tag{2.7}$$

2.2.3 Incomplete factorization preconditioners

Section 2.2.2 has mentioned that the incomplete LU factorization preconditioner, ILU, arising from Gaussian elimination is more efficient than SSOR because the error matrix $K_{ep} - M_{ILU}$ is generally smaller than $K_{ep} - M_{SSOR}$. ILU is considered the most popular class of preconditioners (Saad & Vorst[146], 2000; Benzi[18], 2002; Vorst[171], 2002). But ILU is sometimes impractical because the preconditioner can be expensive to construct: it required more forming time and more storing memory than Jacobi and SSOR preconditioner (Fischer *et al.*[60], 1996; Payer & Mang[126], 1997; Chen *et al.*[38], 2004). However, storing memory has become a less critical problem with the memory capacity of modern computers. Though ILU indeed may require substantially more time to form than Jacobi and SSOR, the total iteration time (including the time to form ILU and the iteration time of Krylov iterative methods) is quite often less than that required by Jacobi and SSOR due to the reduction in the number of iterations needed for convergence. ILU has been successfully applied in large-scale nonsymmetric linear systems resulted from popular problems such as Navier-Stokes equations, in which ILU is often involved in block preconditioners discussed later in Section 2.3.3 (Dahl & Wille[48], 1992; Persson & Peraire[128], 2008; Rehman *et al.*[140], 2008; Diosady & Darmofal[49], 2009), Helmholtz equation (Schneider & Marburg[150], 2003; Kechroud *et al.*[92], 2004; Osei-Kuffuor & Saad[121], 2010), and BE discretization (Fata & Gray[57], 2010; Kacimi & Laghrouche[90], 2011).

The lower and upper triangular matrices computed from Gaussian elimination of a sparse matrix are often less sparse than the original matrix because of fill-ins. ILU preconditioner is formed by dropping off some or all of these fill-ins based on some drop-off criteria. Figure 2.5 shows the pseudo-code of this process. There are two dropping criteria often imposed on ILU factorization: dropping off by the level of fill or when the number of fill-ins exceeds the tolerance value, and dropping off when the absolute numerical values of fill-ins are smaller than the tolerance value (Saad[169], 2003, pp. 288-320).

```
for i = 1,…, n
    w = a_i*
    for k = 1,…, i – 1 and when w_k ≠ 0
        w_k = w_k/a_kk
        Apply a dropping rule to w_k
        if w_k ≠ 0 then
            w = w – w_k u_k*
        end if
    end for
    Apply a dropping rule to row w
    l_i,j = w_j for j = 1,…, i – 1
    u_i,j = w_j for j = i,…, n
    w = 0
end for
```

Figure 2.5: Pseudo-code for ILUT (Saad[144], 2003, pp. 307)

ILU0 is a popular special case of ILU for which the first dropping criterion is imposed: all the fill-ins are dropped off and ILU0 contains the same number of nonzero entries as the original matrix. Hence the storing memory of ILU0 is quantified before the factorization, unlike the ILUT discussed in the following paragraph. Benzi[18] (2002) and Chow and Saad[43] (1997) noted that ILU0 is effective when the matrix is M-matrix or diagonally dominant matrix. Nevertheless, ILU0 has found its use in other classes of matrices because it is simple and inexpensive to implement (Lan & Liang[98], 1997; Dutto & Habashi[52], 1999; Malas & Gurel[106], 2007; Diosady & Darmofal[49], 2009).

Saad[169] (2003) proposed the ILUT(ρ, τ) preconditioner based on the dual threshold strategy at each step of the factorization: fill-ins are dropped off when their absolute values are smaller than τ times the 2-norm of the current row, and at most ρ largest fill-ins are kept in the current row. Both of the dropping criteria mentioned above are used to form this ILU. Benzi[18] (2002) commented that ILUT(ρ, τ) is an powerful preconditioner. The storing memory for ILUT(ρ, τ) is limited by the upper bound of fill-ins in each row, ρ but is still undetermined before the factorization. The main practical drawback of ILUT(ρ, τ) is the optimal values of ρ and τ are priorly unknown and are problem dependent. Saad[169] (2003) and Benzi[18] (2002) observed that ILUT(ρ, τ) worked well with the choice of small τ (from 10^{-5} to 10^{-2}) and/or large ρ (from 20). ILUT(ρ, τ) is more expensive than ILU0 but is expected to be more efficient than ILU0 because fill-ins are allows hence it approximates the original matrix better and the error matrix is smaller. In comparison with ILU0, ILUT(ρ, τ) often requires more time to factorize and the preconditioning step $Mu = \tilde{u}$ requires more time to solve as well; however the reduction in matvec count due to ILUT(ρ, τ) (with the proper choice of ρ and τ!) can help to reduce the total iteration time. Benzi[18] (2002) demonstrated through the convection-diffusion problem that the total iteration time can be reduced by half when ILUT(ρ, τ) is used. Gambolati and co-workers[64, 65, 66] (2001, 2002, 2003) have successfully used ILUT(ρ, τ) as preconditioner for the nonsymmetric form of FE discretization of Biot's consolidation equations. This is elaborated more in Section 2.3.2.

However, ILUT is not as time efficient as ILU0 for the nonsymmetric linear system arising from the non-associated MC model for the examples examined in this thesis (detailed discussion can be found in Section 3.5). Chauhary[161] (2010) found that ILUT did not perform well for the symmetric linear system arising from FE discretization when soil follows a linear elastic model. The reduction in iteration time cannot make up for the time spent to form ILUT(ρ, τ) and to solve the preconditioning step $Mu = \tilde{u}$. ILU0 has been found to be more reliable and time efficient than ILUT(ρ, τ) in several other problems such as coupled structural-acoustic problems (Lin & Grosh[103], 2003), DP_N acceleration equation in transport scheme (Santandrea & Sanchez[148],2005), wave scattering phenomena in computational electromagnetics (Malas & Gurel[106], 2007).

In spite of the popularity of ILU preconditioner, researchers in the scientific computing community often caution that ILU should not be used as a black-box especially for nonsymmetric matrices (Chow & Saad[43], 1997; Benzi[18], 2002). Chow and Saad[43] (1997) were aware that ILU could be unstable due to the four main reasons: inaccuracy due to very small pivots, unstable triangular solves, inaccuracy due to dropping and zero pivots.

Table 2.1 Statistics that can be used to evaluate an incomplete factorization (Chow & Saad[43], 1997)

Statistic	Meaning
condest	$\lVert(\bar{L}\bar{U})^{-1}e\rVert_\infty$, $e = (1,1,\ldots,1)^T$.
1/*pivot*	Size of reciprocal of the smallest pivot
$\max(\bar{L} + \bar{U})$	Size of the largest element in $\bar{L}$ and $\bar{U}$ factors

With $e = (1,1,\ldots,1)^T$
Solve $\bar{L}\bar{U}u = e$
condest = max(u_i), i = 1,…N

Figure 2.6: Pseudo-code to compute *condest* of ILU preconditioner

They recommended three statistics: *condest*, 1/*pivot* and max($\bar{L} + \bar{U}$) presented in Table 2.1 to evaluate an ILU preconditioner. When the values of these three statistics are very large, on the order of 10^{15} recommended by Chow and Saad[43] (1997), ILU is considered unstable and can fail if the values are extremely large. If *condest* and 1/*pivot* are about the same size, the instability of ILU comes from very small pivot. If *condest* is much larger than 1/*pivot*, the instability comes from the triangular solves. Large max($\bar{L} + \bar{U}$) indicates the inaccurate factorization.

Chow and Saad[43] (1997) were aware that there were cases when these values are small but ILU still fail and they commented that the failure may come from the inaccuracy due to dropping. When ILU does not help or even fail the iteration, Benzi and others[20] (1999) recommended ordering the original matrix with reverse Cuthill-McKee (RCM) before performing ILU factorization especially when the original matrix is strongly nonsymmetric.

2.3 Preconditioners for 2-by-2 nonsymmetric block matrix

2.3.1 Nonsymmetric linear systems resulted from Biot's consolidation equations

The 2-by-2 block global stiffness matrix in this thesis comes from FE discretization Biot's consolidation equations. The increment form of this discretization with fully implicit Crank-Nicolson time stepping (θ =1) is given in Eq.(2.8) (Smith & Griffith[158], 2004),

$$A = \begin{bmatrix} K_{ep} & B \\ B^T & -C \end{bmatrix} \begin{Bmatrix} \Delta u \\ \Delta p^{ex} \end{Bmatrix} = \begin{Bmatrix} \Delta f \\ C p^{ex} \end{Bmatrix} \tag{2.8}$$

in which $K_{ep} \in \Re^{nd \times nd}$ is the soil stiffness matrix and is nonsymmetric when the non-associated MC model is applied, $C \in \Re^{np \times np}$ is the fluid stiffness matrix (symmetric positive semi-definite) and $B \in \Re^{nd \times np}$ is the displacement-pore pressure coupling matrix. These submatrices are given in Eq. (2.9), (2.10) and (2.11) respectively. In these equations, V is the volume of the considered body; B_u is the soil element strain-displacement matrix; N_p is the fluid element shape function vector; B_p is the gradient matrix of N_p; $[k]$ is the permeability matrix; and γ_w is the unit weight of pore water taken as 10 kN/m^3 in this thesis. The 2-by2 block matrix in Eq.(2.8) is nonsymmetric solely because K_{ep} is nonsymmetric. Figure 2.7 plots the sparsity pattern of this 2-by-2 block matrix.

$$K_{ep} = \sum_{\text{element}} \left[\int_V B_u^T D_{ep} B_u dV \right] \tag{2.9}$$

$$B = \sum_{\text{element}} \left[\int_V B_u^T 1 N_p dV \right] \tag{2.10}$$

$$C = \theta \Delta t H = \theta \Delta t \sum_{\text{element}} \left[\int_V B_u^T \frac{[k]}{\gamma_w} B_p dV \right] \tag{2.11}$$

From Eq. (2.12), the submatrix C is a function of time step Δt and matrix H, the 2-by-2 block global stiffness matrix in Eq.(2.8) can be written in the following form (Toh & Phoon[165], 2007)

$$\begin{bmatrix} K_{ep} & B \\ -\dfrac{B^T}{\Delta t} & \dfrac{C}{\Delta t} \end{bmatrix} \begin{Bmatrix} \Delta u \\ \Delta p^{ex} \end{Bmatrix} = \begin{Bmatrix} \Delta f \\ -\dfrac{C p^{ex}}{\Delta t} \end{Bmatrix} \tag{2.12}$$

When the soil follows the linear elastic model, Eq.(2.8) is symmetric but Eq.(2.12) is nonsymmetric. Toh and Phoon[165] (2007) compared these two forms and concluded that the symmetric form was preferable because the symmetry can be taken advantage of. However when the soil follows the non-associated MC model, both forms are nonsymmetric hence it is interesting to review this conclusion in this new context.

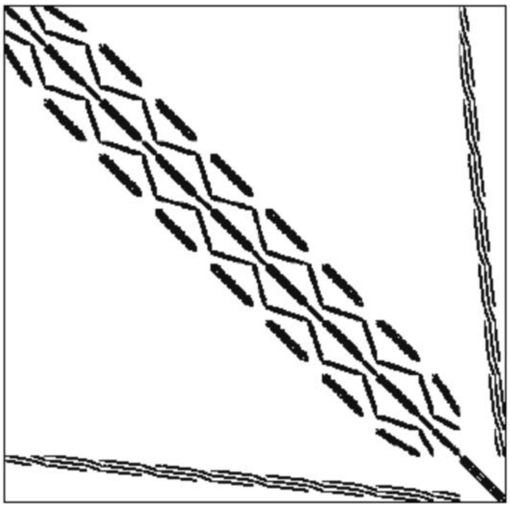


Figure 2.7: Sparsity pattern of 2-by-2 block matrix

This 2-by-2 block matrix can be treated as a 1-by-1 block matrix and the preconditioners in Section 2.2 are applicable for this nonsymmetric linear system, especially the ILU preconditioner. Block preconditioners are more popular for this 2-by-2 block matrix because they can exploit the block structure and the spectral properties of the block matrix. The 2-by-2 block matrix arising from Biot's consolidation equations belongs to the class of saddle point problems hence preconditioners developed for this class are also suitable to this special case.

2.3.2 ILU and MSSOR preconditioner

As noted in Section 2.3.1, the preconditioners discussed in Section 2.2 are still applicable when the 2-by-2 block matrix in Eq.(2.8) is considered as a 1-by-1 block matrix. When the soil follows the linear elastic model, ILU and MSSOR are popular preconditioners for Biot's consolidation analysis. Chen[39] (2005) proposed the Modified SSOR (MSSOR) preconditioner in Eq.(2.13) which is derived from the standard SSOR preconditioner by using the Generalized Jacobi (GJ) preconditioner, which will be discussed in detail in Section 2.3.3, instead of Jacobi preconditioner as SSOR in Eq.(2.4).

$$M_{MSSOR} = \left(L_A + \frac{\hat{D}}{\omega} \right) \left(\frac{\hat{D}}{\omega} \right)^{-1} \left(U_A + \frac{\hat{D}}{\omega} \right) \tag{2.13}$$

in which L_A is the strictly lower triangular matrix of A, U_A is the strictly upper triangular matrix of A, and $\hat{D} = M_{GJ}$. Chen and Phoon[41] have recently applied this MSSOR to Eq.(2.8) when the soil followed the non-associated MC model. They compared the efficiency in solving this nonsymmetric linear system by QMR against solving the equivalent symmetrized linear system by SQMR and found that the latter was preferred because the symmetry could be exploited.

Chauhary[37] (2010) performed comparison of ILU0 and MSSOR when the soil follows a linear elastic model. He found that MSSOR is more robust than ILU0 because SQMR with MSSOR converged over a wide range of parameter values while SQMR with ILU0 did not. He noted that nodal ordering significantly affects the performance of ILU0. With a suitable nodal ordering of the global stiffness matrix, SQMR with ILU0 converges faster than SQMR with MSSOR.

Gambolati and co-workers[64, 65, 66] (2001, 2002, 2003) have long been interested in the use of ILUT(ρ, τ) in solving the nonsymmetric form Eq.(2.12) of Biot's consolidation equation. They concluded that ILUT(ρ, τ) could be very efficient if the proper values of ρ and τ were used. However they did not recommend the range of proper values of ρ and τ for Biot's consolidation problem, which is justifiable because ρ and τ are significantly problem dependent as noted in Section 2.2.3.

2.3.3 Block preconditioners

Similar to preconditioners for 1-by-1 block matrices, block preconditioners should approximate the 2-by-2 block matrices as close as possible. Block preconditioners for Eq.(2.8) are often derived from the following block factorization including the lower triangular block, diagonal block and upper triangular block,

$$A = \begin{bmatrix} K_{ep} & B \\ B^T & -C \end{bmatrix} = \begin{bmatrix} I & 0 \\ B^T K_{ep}^{-1} & I \end{bmatrix} \begin{bmatrix} K_{ep} & 0 \\ 0 & -S \end{bmatrix} \begin{bmatrix} I & K_{ep}^{-1} B \\ 0 & I \end{bmatrix} \tag{2.14}$$

$$S = B^T K_{ep}^{-1} B + C \tag{2.15}$$

in which S is the Schur complement for A. Block preconditioners are often categorized into three types: diagonal block, triangular block and constrained block preconditioner. The efficiency of block preconditioners based on this block factorization relies on how close K_{ep} and S are approximated. Axelsson and co-workers[12, 13] (2010, 2012) have analysed theoretically the spectral properties of the preconditioned system to determine the efficiency of block preconditioners. However their recommended parameters are still too expensive to compute before the iteration process especially when large-scale linear systems are considered. Numerical experiments are still required to determine the optimal block preconditioners for each problem.

2.3.3.1 Diagonal block preconditioner

Diagonal block preconditioner approximates the diagonal block in the factorization of Eq.(2.14). As mentioned in section 2.2, diagonal preconditioner, Jacobi, is the cheapest preconditioner and scaling is often introduced when the diagonal entries are of significantly different scales. This is the case for Biot's consolidation equations in Eq.(2.8) where K_{ep} is a function of Young's modulus E', having order as large as 10^6 and C is a function of permeability k, having order as small as 10^{-10}. Phoon and co-workers[131] (2002) proposed the following scaling form of Jacobi preconditioner called Generalized Jacobi (GJ),

$$M_{GJ} = \begin{bmatrix} \mathrm{diag}(K_{ep}) & 0 \\ 0 & \alpha\,\mathrm{diag}(\hat{S}) \end{bmatrix} \tag{2.16}$$

in which $\hat{S} = C + B^T \mathrm{diag}(K_{ep})^{-1} B$, a cheap approximation of S, and α is a real scaling factor. Toh and others[171] (2004) commented that this preconditioner was memory efficient but did not always possess good convergence time. M_{GJ} is the special form of the diagonal block preconditioner in Eq.(2.17),

$$M_d = \begin{bmatrix} \hat{K} & 0 \\ 0 & \alpha\hat{S} \end{bmatrix} \tag{2.17}$$

$$M_d^{-1} = \begin{bmatrix} \hat{K}^{-1} & 0 \\ 0 & (1/\alpha)\hat{S}^{-1} \end{bmatrix} \tag{2.18}$$

with $\hat{K}$ is an approximation of K_{ep}, α is a real scalar and has the same meaning as in GJ, and $\hat{S} = C + B^T \hat{K}^{-1} B$ is an approximation of S. Figure 2.8 shows the pseudo-code to compute the preconditioning step $M_d^{-1}[u;v]$.

Compute $w = \hat{K}^{-1}u$
Compute $z = \hat{S}^{-1}v$
Set $M_d^{-1}[u;v] = [w;z]$

Figure 2.8: Pseudo-code to compute preconditioning step $M_d^{-1}[u;v]$ (Toh *et al.*[166], 2004)

Phoon and co-workers[131] (2002) have proved the following theorem. They also showed numerically that α = –4 is optimal for many cases and recommended that α should be a negative scalar in general. Although the symmetric 2-by-2 block matrix A was used in their discussion, the theorem and the proof does not require the submatrix block (1,1) in A to be symmetric. Hence when block (1, 1) is nonsymmetric, the recommendations from Phoon *et al.* [131] (2002) on the range of α are still applicable. This thesis first uses α = –4 in the numerical experiments to compare the efficiency of several approximations of K_{ep} and S.

Theorem *Let* $M = \begin{bmatrix} K_{ep} & 0 \\ 0 & \alpha S \end{bmatrix}$ *and* $W = M^{-1}A$. *Then*

$$W(W - I)\left(W^2 - W - \frac{1}{\alpha}I\right) = O\left(\left\|S^{-1}C\right\|\right) \tag{2.19}$$

where $O\left(\left\|S^{-1}C\right\|\right)$ *denotes a matrix whose norm is of order* $\left\|S^{-1}C\right\|$. *Thus if A is non-singular (hence W is non-singular), then W has three distinct clusters of eigenvalues at 1 and* $\left(1 \pm \sqrt{1 + 4/\alpha}\right)/2$ *, each with a diameter of the order* $\left\|S^{-1}C\right\|$. *In particular, when α = -4, W only has two distinct clusters of eigenvalues at* ½ *and* 1.

Proof: From Phoon *et al.*[131] (2002),

$$W^2 - W = \frac{1}{\alpha}\begin{bmatrix} K_{ep}^{-1}BS^{-1}B^T & 0 \\ 0 & I \end{bmatrix} + F \tag{2.19a}$$

where

$$F = \frac{1}{\alpha^2}\begin{bmatrix} 0 & -\alpha K_{ep}^{-1}BS^{-1}C \\ -S^{-1}CS^{-1}B^T & \left(S^{-1}C\right)^2 \end{bmatrix} \tag{2.19b}$$

With $\left(K_{ep}^{-1}BS^{-1}B^{T}\right)^{2} = K_{ep}^{-1}BS^{-1}\left(S-C\right)S^{-1}B^{T} = K_{ep}^{-1}BS^{-1}B^{T} - K_{ep}^{-1}BS^{-1}CS^{-1}B^{T}$, we have

$$\left(W^{2}-W\right)^{2} = \frac{1}{\alpha}\left(W^{2}-W\right) + F^{2} - \frac{1}{\alpha^{3}}\begin{bmatrix} \alpha K_{ep}^{-1}BS^{-1}CS^{-1}B^{T} & \alpha K_{ep}^{-1}BS^{-1}C\left(2I-S^{-1}C\right) \\ \left(2I-S^{-1}C\right)S^{-1}CS^{-1}B^{T} & -2\left(S^{-1}C\right)^{2} \end{bmatrix} \tag{2.20}$$

Thus

$$(W^{2}-W)^{2} = \frac{1}{\alpha}(W^{2}-W) + O\left(\left\|S^{-1}C\right\|\right) \tag{2.21}$$

2.3.3.2 Block constrained preconditioners

Block constrained preconditioners in Eq.(2.22) are better approximations of Eq.(2.14) than the block diagonal preconditioners in Section 2.3.3.1 and is expected to be more efficient than block diagonal preconditioners. However because this preconditioner is more complicated than block diagonal preconditioners, more time is expected to spent on forming this preconditioner as well as on the preconditioning step. Figure 2.9 shows the pseudo-code to compute the preconditioning step $M_c^{-1}[u;v]$.

$$M_{C} = \begin{bmatrix} \hat{K} & B \\ B^{T} & -C \end{bmatrix} \tag{2.22}$$

$$M_{C}^{-1} = \begin{bmatrix} \hat{K}^{-1} - \hat{K}^{-1}B\hat{S}^{-1}B^{T}\hat{K}^{-1} & \hat{K}^{-1}B\hat{S}^{-1} \\ \hat{S}^{-1}B^{T}\hat{K}^{-1} & -\hat{S}^{-1} \end{bmatrix} \tag{2.23}$$

This class of preconditioners is called 'constrained' because they have the same block structure as the native coefficient matrix, but one or more blocks are approximated or 'constrained'.

Compute $w = \hat{K}^{-1}u$
Compute $z = \hat{S}^{-1}\left(B^{T}w - v\right)$
Compute $M_{c}^{-1}[u;v] = \left[\hat{K}^{-1}\left(u - Bz\right); z\right]$

Figure 2.9: Pseudo-code to compute preconditioning step $M_c^{-1}[u;v]$ (Toh *et al.*[166], 2004)

When soil follows a linear elastic model and Eq.(2.8) is symmetric, $\hat{K}$ is often taken as Incomplete Cholesky (IC) factorization with different levels of fill-ins and $\hat{S} = C + B^T \hat{K}^{-1} B$ is also factorized into an IC form. Toh and others[171] (2004) showed that the finest approximation of *S*, was not always useful because the computational time was mostly spent to form that approximation. Bergamaschi and others[22, 23, 59] (2007, 2008, Ferronato *et al.*, 2010) concluded that block constrained preconditioners were better than ILU-based preconditioner especially when the time step Δt was small. However, the efficiency of the IC factorization relies much on the level of fill-ins and this parameter is often determined through trial-and error.

When Eq.(2.8) is nonsymmetric, Botchev and Golub[30] (2006) recommended the use of SSOR (Eq.(2.4)) for $\hat{K}$ and discussed theoretically the optimal value of ω. They applied this preconditioner on the Navier-Stokes equation and noted that the preconditioner was still robust when ω was not optimal. This thesis implements this preconditioner for the square footing problem in Section 5.3.1.

2.3.3.3 Block triangular preconditioners

When Eq.(2.8) is symmetric, Toh and others[171] (2004) studied the block triangular preconditioners taking the following forms:

$$M_{t-L} = \begin{bmatrix} \hat{K} & 0 \\ B^T & -\hat{S} \end{bmatrix} \text{for left preconditioning} \tag{2.24}$$

$$M_{t-R} = \begin{bmatrix} \hat{K} & B \\ 0 & -\hat{S} \end{bmatrix} \text{for right preconditioning} \tag{2.25}$$

$$M_{t-L}^{-1} = \begin{bmatrix} \hat{K}^{-1} & 0 \\ \hat{S}^{-1} B^T \hat{K}^{-1} & -\hat{S}^{-1} \end{bmatrix}; \; M_{t-R}^{-1} = \begin{bmatrix} \hat{K}^{-1} & \hat{K}^{-1} B \hat{S}^{-1} \\ 0 & -\hat{S}^{-1} \end{bmatrix} \tag{2.26}$$

Figure 2.10 shows the pseudo-code to compute preconditioning step $M_{t-L}^{-1}[u;v]$ and $M_{t-R}^{-1}[u;v]$. Their numerical experiments showed that this preconditioner did not offer better convergence time than diagonal block and constrained block preconditioner.

Compute $w = \hat{K}^{-1}u$ Compute $z = \hat{S}^{-1}(B^T w - v)$ Compute $M_{t-L}^{-1}[u;v] = [w;z]$	Compute $w = -\hat{S}^{-1}u$ Compute $z = \hat{K}^{-1}(u - Bw)$ Compute $M_{t-R}^{-1}[u;v] = [z;w]$

Figure 2.10: Pseudo-code to compute preconditioning step $M_{t\text{-}L}^{-1}[u;v]$ and $M_{t\text{-}R}^{-1}[u;v]$ (Toh *et al.*[166], 2004)

2.4 Convergence criteria

2.4.1 Effect of spectral properties

The convergence properties of Krylov iterative methods depend on spectral properties of the coefficient matrix of the linear system (Freund *et al.*[63], 1992; Barrett *et al.*[16], 1994; Saad & Vorst[146], 2000; Golub & Vorst[71], 2001; Saad[144], 2003). Section 1.1.1 has introduced that Krylov iterative methods converge to the exact solution in at most N iterations in exact arithmetic but they normally converge earlier than that although there are cases of breakdown and divergence due to rounding errors. If K_{ep} is diagonalizable so that

$$K_{ep} = X\Lambda X^{-1} \tag{2.27}$$

in which X is a non-singular matrix containing eigenvectors of K_{ep} and Λ = diag($\lambda_1,\ldots,\lambda_n$) is a diagonal matrix containing corresponding eigenvalues of K_{ep}, the residual $r^{(i)}$ at the i iteration step has the following upper bound

$$r^{(i)} = F - K_{ep}u^{(i)} \tag{2.28}$$

$$\left\| r^{(i)} \right\| \le \|X\| \left\| X^{-1} \right\| \max_{k=1,\ldots,n} \left| p_i(\lambda_k) \right| \left\| r^{(0)} \right\| \tag{2.29}$$

in which $p_i(\lambda_k)$ is a polynomial of degree at most i with $p(0) = 1$.

For GMRES, a special case of Krylov iterative methods, which minimizes the 2-norm of the residual, the bound in Eq.(2.29) is further narrowed as

$$\left\| r^{(i)} \right\| \le \|X\| \left\| X^{-1} \right\| \min_{p_i \in P_i} \max_{k=1,\ldots,n} \left| p_i(\lambda_k) \right| \left\| r^{(0)} \right\| \tag{2.30}$$

Eq.(2.29) and (2.30) show that the convergence is mainly governed by the condition number of X matrix and the polynomial of the eigenvalues of K_{ep}. The polynomial of the eigenvalues is not easy to determine explicitly so the convergence criteria can only be observed qualitatively through the distribution of eigenvalues. Graphically speaking, the smaller the ellipse (with suitable normalization) circumscribes all the eigenvalues is, the faster Krylov iterative methods converge. This ellipse is demonstrated in Figure 2.11.

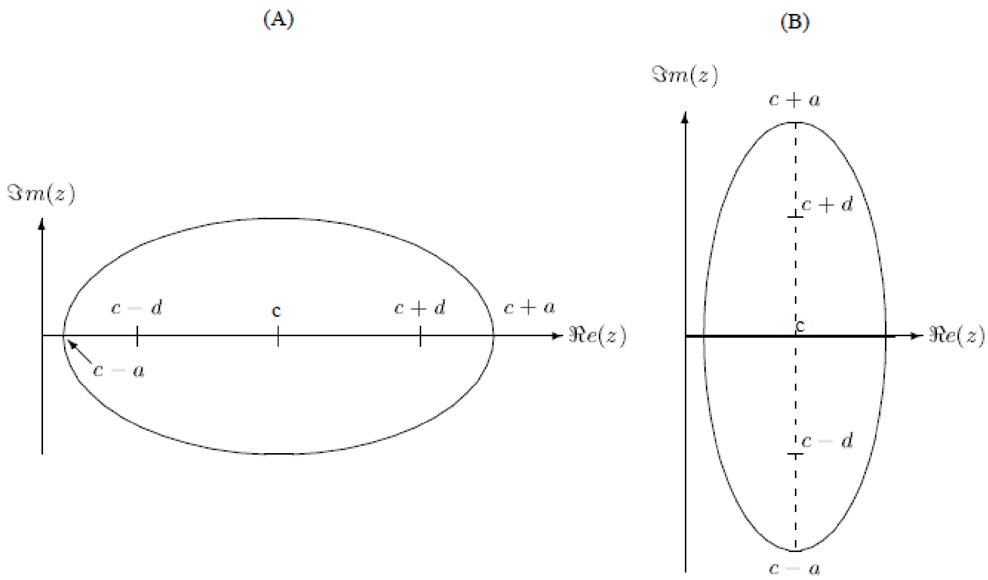


Figure 2.11: Ellipses containing the spectrum of A. (A): real eigenvalues; (B) Purely imaginary eigenvalues (Saad[144], 2003, pp. 195)

Eq.(2.29) and (2.30) are inequality equations and indicate the maximum bound. The maximum bound is the worst approximation of residual at step *i*. The real residual $r^{(i)}$ can be much smaller from the maximum bound because the condition number of matrix *X* can be large for highly non-normal matrices. It is a good situation when $r^{(i)}$ is much smaller than the maximum bound because the Krylov iterative methods will converge faster than expected. But on the other hand, it shows that the bound is too crude to predict the convergence of the methods. However, for the general linear system, Eq.(2.29) is the best convergence criterion for Krylov iterative methods in the current state of the art. This criterion applies to both Bi-CGSTAB and IDR(*s*) method.

2.4.2 Stopping criteria and tolerance of error

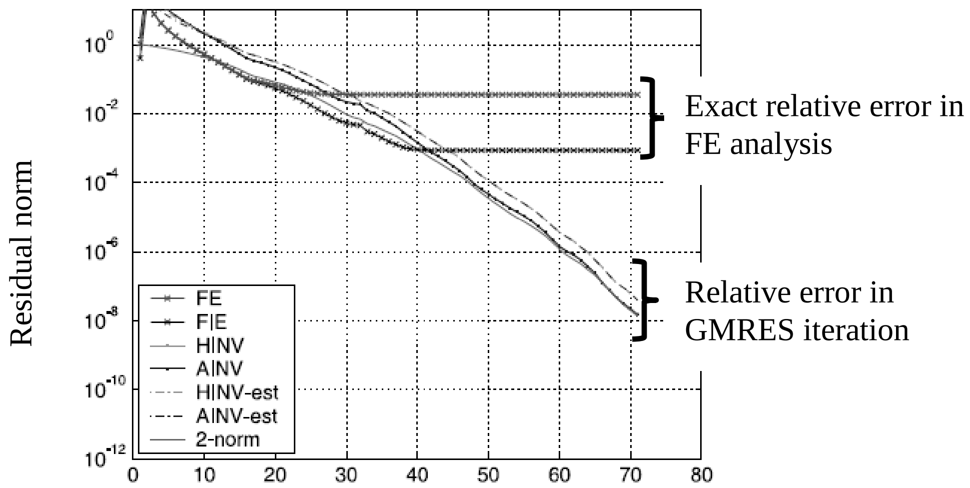
Section 1.1.1 has mentioned the advantage of iterative solvers is that they can be stopped whenever the error satisfies a desired tolerance. The exact error is the difference of exact solution and iterative solution (Eq.(2.31)) and is impractical to compute explicitly hence residual $r^{(i)}$ in Eq.(2.28) is used in most of the cases (Barrett *et al.*[16], 1994).

$$e^{(i)} = x - x^{(i)} \tag{2.31}$$

Relative residual related to 2-norm of the residual vector in Eq. (2.32) is often used in numerical experiments with *i_tol* is the user-defined tolerance. This relative residual is used in this thesis.

$$\frac{\left\| r^{(i)} \right\|_2}{\left\| r^{(0)} \right\|_2} \le i\_tol \tag{2.32}$$

This *i_tol* is problem dependent: the results can be unreliable if *i_tol* is too large, but a too stringent *i_tol* can require much resource for little improvement in the results. Section 1.1.1 has argued that *i_tol* is generally large for geotechnical problem because there are uncertainties in soil properties and soil models. The tolerance *i_tol* = 10^{-6}, which is used throughout this thesis, is often considered too stringent for geotechnical problem. However, the definition of “relaxed tolerance” and “stringent tolerance” is rather subjective and in this thesis, it mostly depends in the FE discretization. This point is elaborated in Section 3.6. The influence of FE discretization has been demonstrated in the discussion of Arioli and others[9] (2005). Figure 2.12 extracts the numerical results of Arioli *et al*.[9] (2005).



(a) $\nu = 1$

Figure 2.12: Comparison of stopping criteria when GMRES is used to solve the linear system from FE discretization of 2D advection-diffusion problem. ν is the diffusion parameter. (Arioli *et al*.[9], 2005)

This figure shows the comparison of stopping criteria when GMRES is used to solve the linear system from FE discretization of 2D advection-diffusion problem. The exact solution of this problem was known hence the exact relative error of FE could be determined and indicated in Figure 2.12. This error reaches a stable level while GMRES iteration count increases and the relative residual error of GMRES reduces. Arioli *et al*.[9] (2005) recommended that GMRES should be stopped at the start of this stable level because the solution could not be improved. This recommendation is useful but impractical since the finite element error, which is unknown at the beginning of the analysis, is required as an input. From Figure 2.12, it can be seen that if FE error is smaller (i.e. the mesh is denser), the tolerance of relative residual of iterative solvers should be smaller to obtain meaningful and reliable results. Table 2.2 presents the *i_tol* values for various numerical experiments in literatures. Small *i_tol* is often used when the discussion on numerical methods while larger *i_tol* is used for practical problems. *i_tol* = 10^{-6} is quite popularly chosen regardless of problems.

Table 2.2: Tolerance values for various iterative methods used in literatures

Author(s)	Max problem size	Tolerance value *i_tol*	Problem description
Almeida & Paiva[6] (2004)	4431	10^{-9}	Layered soil-supperstructure interaction Soil follows linear elastic model
Araujo *et al.*[8] (2006)	10,383	10^{-5}	Rigid foundation (BE analysis) Soil follows linear elastic model
Benzi & Golub[19] (2004)	1,362,480	10^{-10}	Navier-Stokes equations
Bergamaschi *et al.*[21] (2012)	145,114	10^{-4}	Two phase flow equations in porous media
Birken *et al.*[24] (2013)	2,912,000	10^{-3}	Navier-Stokes equations
Chaillat *et al.*[35] (2009)	215,058	10^{-3}	Seismic wave propagation and amplification in complex geological structures
Chen & Phoon[41] (2012)	107,180	10^{-3}	Shallow foundation and pile raft system Soil follows non-associated MC model
Hartmann *et al.*[76] (2009)	100,520	10^{-6}	3D plate with hole Material follows a viscoplasticity model
Kechroud *et al.*[92] (2004)	113,060	10^{-8}	Helmholtz equations
Lin & Grosh[103] (2003)	25,012	10^{-8}	3D high frequency response of fluid-loaded structures
Mroueh & Shahrour[115] (1999)	39,526	10^{-5}	Shallow foundation and laterally loaded pile Soil follows non-associated MC model
Osei-Kuffuor & Saad[121] (2010)	29,241	10^{-8}	Helmholtz equations
Rehman *et al.*[140] (2008)	47,468	10^{-6}	Navier-Stokes equations
White & Borja[175] (2011)	455.3 million	10^{-8}	Fully coupled flow and Geomechanics Soil follows non-associated MC model

2.5 Summary

This chapter review the characteristic of IDR(s), the preconditioners for 1-by-1 block matrix coming from drained/undrained analysis, 2-by-2 block matrix coming from Biot's consolidation analysis. IDR(s) is a promising Krylov iterative solver for nonsymmetric linear systems and will be used in this thesis. The discussed preconditioners will be implemented with IDR(s) to inspect their performance.

CHAPTER 3 ITERATIVE SOLVERS FOR NONSYMMETRIC LINEAR SYSTEMS

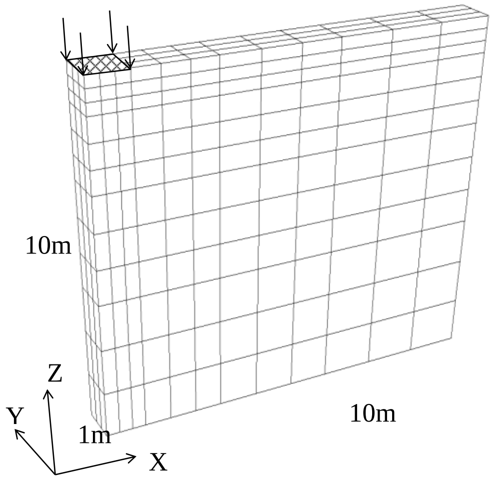
3.1 Introduction

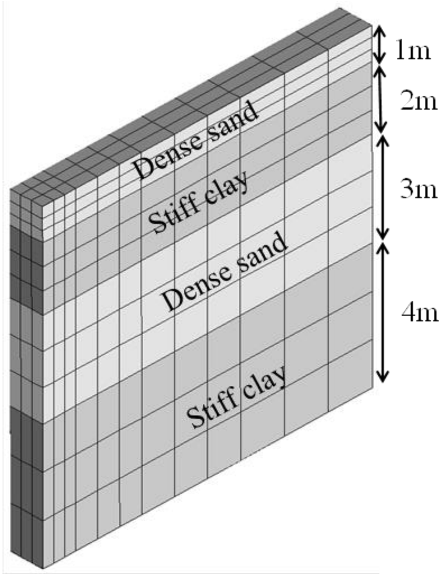
This chapter performs the comparison between IDR(*s*) and Bi-CGSTAB method. Matrix vector multiplication (matvec) and total iteration time (time spent by the Krylov iterative methods plus overhead time required to form preconditioners) are used as comparison indicators. Matvec pertains to the theoretical efficiency of the Krylov iterative methods and preconditioners while total iteration time pertains to the practical efficiency. The overall objective of this thesis is to optimize the solution time of the nonsymmetric linear system hence the practical efficiency is the most concerned goal.

As mentioned in Section 1.1.3 and Section 2.1.1, IDR(1) is mathematically equivalent to Bi-CGSTAB and the efficiency of IDR(*s*) improves when *s* increases but up to certain limit value. Numerical experiments in this section aim at the following four objectives: first, IDR(1) and Bi-CGSTAB are compared to prove that they are more or less equivalent in the presence of rounding errors; second, various values of *s* are used to find the optimal value; third, IDR(*s*) with optimal value of *s* is compared with Bi-CGSTAB to show that IDR(*s*) is more efficient than Bi-CGSTAB in term of matvec and total iteration time. Jacobi and ILU0 are used as right preconditioners. SSOR is used as left-right preconditioner and left preconditioner respectively denoted as SSOR-S and SSOR-L.

3.2 Problem description and theoretical background

All the numerical experiments in this chapter are performed with the plane strain strip footing example. Drained analysis is considered. Figure 3.1a shows the 3D mesh of the strip footing subjected to uniform vertical pressure, *q*. The base of the mesh is fixed in all directions. Side faces are fixed in transverse direction and free in in-plane directions. Top surface is free in all directions. The mesh spans 10 meters in X- and Z-directions, 1 meter in Y-direction. Three cases of soil profile are considered: soil profile 1 is a homogeneous stiff clay layer; soil profile 2 is a homogeneous dense sand layer; and soil profile 3 is a heterogenous soil layer consisting of alternate dense sand and stiff clay as shown in Figure 3.1b. The soil is assumed to be weightless and to follow the non-associated Mohr-Coulomb model with the properties in Table 3.1.





(a) (b)

Figure 3.1: (a) 3D FE mesh of strip footing; (b) Soil profile 3: Heterogenous soil consisting of alternate dense sand and stiff clay

Table 3.1: Parameters of Mohr-Coulomb yield criterion

	Young's modulus, E' (MPa)	Poisson's ratio, v'	Cohesion, c' (kPa)	Friction angle, ϕ' (degree)	Dilation angle, ψ (degree)
Stiff clay	60	0.3	20	20	0
Dense sand	105	0.3	1	30	5

Ultimate bearing capacity of the strip footing resting on the homogeneous soil profile 1 and 2 can be estimated with Terzaghi's formula as Eq. (3.1),

$$q_f = \frac{1}{2}\gamma B N_\gamma + c N_c + \gamma D N_q \tag{3.1}$$

in which γ is the unit weight of soil, B is the width of the shallow foundation, c is the cohesion, D is the embedment depth, N_γ, N_c and N_q are bearing capacity factors and are function of the friction angle ϕ. In this chapter, the soil is assumed weightless and the strip footing rests on the ground surface so the first and third term in Eq.(3.1) are zero and N_c is the main governing parameter. N_c is calculated by Eq.(3.2) and (3.3). The weightless soil is used in this section so that the applied load can be increased closed to the theoretical failure load predicted by Eq.(3.1) and so that a wide range of physical behaviour of the problem can be observed.

$$N_c = \left(N_q - 1\right)\cot\phi \tag{3.2}$$

$$N_q = e^{\pi \tan\phi} \tan^2\left(45^o + \phi/2\right) \tag{3.3}$$

The predicted failure load of soil profile 1 and 2 and the maximum applied load for each soil profile are reported in Table 3.2. It is worth to note that Eq.(3.1) does not take into account the non-associated flow rule, which is considered in this whole thesis, hence the predicted values are only used for reference.

Table 3.2: Ultimate bearing capacity of the strip footing and square footing on the homogenous soil layer and the maximum applied pressure used in numerical experiments

	N_q	N_c	$q_{f\text{-}strip} = c'N_c$ (kPa)	$q_{max\text{-}strip}$ (kPa)
Soil profile 1	6.4	14.83	296.69	280 (= 94%$q_{f\text{-}strip}$)
Soil profile 2	18.40	30.14	30.14	27 (= 90%$q_{f\text{-}strip}$)
Soil profile 3	–	–	–	40

It is known there is numerical difficulty to compute collapse load using FE analysis when the friction angle of the MC model is high (Vermeer & Langen[169], 1989; Manoharan & Dasgupta[109], 1997). Besides, this thesis encounters a numerical difficulty caused by the fact that many Gauss points are forced to lie on the apex of the MC surface (refer to the return mapping procedure in A.2), which is spurious and can be slightly mitigated by increasing the number of load steps. This numerical difficulty was also observed by Clausen and Krabbenhoft[44] (2008) when they studied 2D meshes of footing and bi-axial test problem. The author is not clear if these two numerical difficulties are related or caused by the same reason. For dense sand with friction angle of 30 degree, although increasing the number of load steps helps to load the system close to the theoretical failure load, the number of load steps required grows larger when the mesh is denser (there are more Gauss points hence there are more chance for Gauss points to be returned to the apex). Because the objective of this thesis is to study the performance of preconditioners rather than to address numerical limitation of the MC model, the above difficulties are circumvented by applying the following expedien strategies. The dense sand systems are only loaded to certain percentage of the ultimate failure load and when the mesh is denser, smaller value of load is used.

3.3 Computational procedure

The nonlinear system resulting from FE analysis is solved by the full NR method. In each NR iteration, the linear system is solved by Krylov iterative methods. IDR(*s*) and Bi-CGSTAB are used in this thesis. IDR(*s*) and Bi-CGSTAB are stopped when the relative residual norm or the number of matrix-vector multiplications (matvec) satisfies Eq.(3.4),

$$\begin{aligned} &(1) \quad \frac{\|r^{(i)}\|_2}{\|r_0\|_2} \le i\_tol = 10^{-6} \\ &\text{or } (2) \quad i \ge 5000 \end{aligned} \tag{3.4}$$

with $r^{(i)} = K_{ep}u^{(i)} - F$ and $r_0 = F$.

The first stopping criterion is to ensure the results are within the tolerable accuracy and the second is to ensure the iteration process is within a practical length of time. The first stopping criterion is naturally indispensable while the second is used in this thesis for the practical purpose: to prevent the iterative solvers from running "forever" in cases of slow convergence or no convergence of the iteration process"

Besides, the second criterion also effectively put a limit on the total iteration time. With the presence of preconditioner *M*, the matrix-vector multiplication is performed as followings,

```
Compute u = AM^{-1}v
(1) Solve Mw = v
(2) Compute u = Aw
```

Figure 3.2: Pseudo-code to compute matrix-vector multiplication with a preconditioned matrix

From Figure 3.2, the time spent in one matvec is the sum of the time spent in step (1) and (2). The time spent in step (2) depends on the size of the matrix and in step (1) depends on both the size of the matrix and the preconditioner *M*.

So for a certain matrix *A* and preconditioner *M*, the time spent in one matvec is ideally a predetermined number only depending on the computer configuration. Matrix-vector multiplication is the most time- consuming operation in one iteration hence limiting the number of matvec is limiting the total iteration time and this amount of time is ideally fixed for a certain *A* and *M*. 5000 matvec is chosen in this thesis and this limit appears to be appropriate for the sizes of studied matrices, meaning the recommended preconditioners always converge with less than 5000 matvecs. For the practical examples in Chapter 6, the limit of matvec is chosen to be 50,000 because the size of the matrix *A* is significantly larger.

It is desirable when both of the above stopping criteria are satisfied such that a preconditioner is able to accelerate the Krylov solver to obtain the acceptable results within acceptable span of time. However, limiting the number of matvec can raise the question on disfavouring the cheap preconditioners because cheap preconditioners takes less time to form, more matvec to converge but the total iteration time can still be within the practical range. Table 3.4 to Table 3.6 in the following section show that this is not the case in this chapter because the number of matvecs to converge is often so huge that the total iteration time becomes absurdly large. For instance, Krylov solvers with Jacobi preconditioner converge in several times more than 5000 matvecs and always consumes much more time than other preconditioners. In Table 5.4 and Table 5.5 at Chapter 5, there are cases that the cheap preconditioners converge with more matvecs and less time than those more expensive preconditioners but they all converge within 5000 matvecs. When more than 5000 matvecs are required, the iteration process either consumes much more time or does not converge at all.

The values of matvec and total iteration time, which includes the time spent to form preconditioner and the time spent by Krylov iterative methods are average values over all the NR iterations in each load step. At each load step, the NR iteration is stopped when the relative residual norm satisfies Eq.(3.5).

$$\frac{\left\| F - K_{ep}u \right\|_2}{\left\| F \right\|_2} \le 10^{-6} \tag{3.5}$$

For Bi-CGSTAB following the pseudo-code in Figure 2.3, the input vector $\tilde{r}^{(0)}$ is chosen as the initial residual, r_0. For IDR(s), as discussed in Section 2.1.2, the shadow matrix $P^{N\times s}$ is a random matrix containing uniformly distributed random number from 0 to 1. In the FORTRAN code, constant seeds for generating random number are used so that with the same N and s, the same random matrix P is generated every time P is required. Exception is set for the case of $s = 1$ when IDR(1) is used to compared with Bi-CGSTAB to show numerically that IDR(1) is equivalent to Bi-CGSTAB. When $s = 1$, matrix P reduces to a vector of dimension N and is set to be equal to the initial residual r_0. IDR(s) pseudo-code in Figure 2.2 also requires the input of limiting value of ω to "maintain the convergence." This limiting value is set to the default value of 0.7 as recommended by Sonneveld and Gijzen[162] (2008) when s > 1. For $s = 1$, this limiting value is taken as 0 to get equivalent parameter in Bi-CGSTAB pseudo-code.

Preconditioners used are: Jacobi, Symmetric Successive Over Relaxation (SSOR) and ILU0. Jacobi and ILU0 are used as right preconditioner. SSOR-L is left preconditioner and SSOR-LR is left-right preconditioner as presented in Section 2.2.2. For SSOR-L and SSOR-LR, only the diagonal D_K is required to form explicitly and this is the Jacobi preconditioner hence the time required to form the preconditioner is reported the same for Jacobi, SSOR-LR and SSOR-L preconditioner.

3.4 Comparison of IDR(*s*) and Bi-CGSTAB

The characteristics of the 3D meshes are presented in Table 3.3. Three problem sizes $12\times3\times12$, $24\times6\times24$ and $32\times8\times32$ are adopted. These three meshes produce the small, medium and large-scale stiffness matrices respectively. When the mesh is denser, not only the number of unknowns increases but the elastic matrix also becomes sparser. When the number of yielded Gauss points increases, the number of non-zero entries in the elastoplastic matrix increases. However, this increase is insignificant and the ratios nnz/N^2 in Table 3.3 remains unchanged.

Table 3.3: 3D finite element meshes of the strip footing

	Mesh size*		
	12×3×12	24×6×24	32×8×32
Number of elements	432	3,456	8,192
Number of nodes	2,431	16,525	37,521
Number of unknowns (N)	5,700	43,584	102,080
Number of Gauss points (N_{ip})	11,664	93,312	2,211,184
Number of nonzero (*nnz*)			
Soil profile 1			
Elastic system	715,515	6,475,800	15,738,341
Elastoplastic system at 295kPa	717,405	6,493,019	15,828,212
Soil profile 2			
Elastic system	715,528	6,477,348	15,744,688
Elastoplastic system at 27kPa	717,714	8,547,672	15,835,064
Soil profile 3			
Elastic system	715,701	6,475,903	15,744,149
Elastoplastic system at 40kPa	716,902	6,484,670	15,792,981
nnz/N^2 (%)			
Soil profile 1			
Elastic system	2.20	0.34	0.15
Elastoplastic system at 295kPa	2.21	0.34	0.15
Soil profile 2			
Elastic system	2.20	0.34	0.15
Elastoplastic system at 27kPa	2.21	0.34	0.15
Soil profile 3			
Elastic system	2.20	0.34	0.15
Elastoplastic system at 40kPa	2.21	0.34	0.15

** Mesh size x×y×z means x element in x direction, y element in y direction and z element in z direction*

Table 3.4, Table 3.5, and Table 3.6 report the number of matvec and total iteration time required by Bi-CGSTAB and IDR(s) with s consequently taken the value of 1, 4, 6, 10, and 20 when combined with different preconditioners. The nonsymmetric linear system solved is at the last load step for soil profile 1, 2 and 3. The linear systems at the last load step are chosen because this system requires the most matvec and time to solve. This point will be elaborate later in the discussion of Figure 3.3 to Figure 3.11 and Section 3.7.

Table 3.4: Comparison of Bi-CGSTAB and IDR(s) with different preconditioners. Soil profile 1 is used. Matvec and time in second are reported at the last load step, 280 kPa.

12×3×12	Bi-CGSTAB		IDR(1)		IDR(4)		IDR(6)		IDR(10)		IDR(20)	
No preconditioner	2801	6.2 (0.0)	2948	7.8 (0.0)	$\boxed{3149}$	8.6 (0.0)	2360	6.0 (0.0)	1762	5.1 (0.0)	1277	3.8 (0.0)
Jacobi	888	2.0 (0.0)	953	2.4 (0.0)	810	2.6 (0.0)	769	2.4 (0.0)	673	1.8 (0.0)	622	2.2 (0.0)
SSOR-LR	191	1.3 (0.0)	193	1.4 (0.0)	174	1.4 (0.0)	164	1.2 (0.0)	159	1.1 (0.0)	158	1.3 (0.0)
SSOR-L	197	2.3 (0.0)	208	2.5 (0.0)	185	2.1 (0.0)	167	1.9 (0.0)	163	1.9 (0.0)	162	1.9 (0.0)
ILU0	92	0.6 (0.1)	96	0.6 (0.1)	73	**0.5** (0.1)	70	**0.5** (0.1)	70	**0.5** (0.1)	**67**	**0.5** (0.1)
24×6×24												
No preconditioner	7973	83.3 (0.0)	5573	70.1 (0.0)	34,164	382.7 (0.0)	14,4478	166.1 (0.0)	12,106	145.0 (0.0)	5705	77.8 (0.0)
Jacobi	2794	32.6 (0.1)	2604	30.0 (0.1)	$\boxed{3012}$	36.3 (0.1)	2555	31.5 (0.1)	2160	27.7 (0.1)	1670	23.5 (0.1)
SSOR-LR	527	20.5 (0.1)	522	20.4 (0.1)	492	19.4 (0.1)	443	17.7 (0.1)	409	16.6 (0.1)	397	16.7 (0.1)
SSOR-L	527	30.9 (0.1)	519	30.5 (0.1)	485	28.7 (0.1)	477	28.4 (0.1)	427	25.6 (0.1)	407	25.0 (0.1)
ILU0	398	12.3 (0.5)	401	12.5 (0.5)	222	7.3 (0.5)	207	**6.7** (0.5)	198	**6.7** (0.5)	**192**	6.8 (0.5)
32×8×32												
No preconditioner	12,956	314.5	14,389	338.6 (0.0)	41,753	1057.5	41,320	1,058,9 (0.0)	39,463	1,116.4 (0.0)	20,355	668.8 (0.0)
Jacobi	2979	145.1 (0.1)	2951	146.4 (0.1)	$\boxed{3593}$	183.9 (0.1)	$\boxed{3712}$	211.1 (0.1)	$\boxed{3158}$	183.2 (0.1)	2292	151.2 (0.1)
SSOR-LR	528	69.8 (0.1)	531	77.1 (0.1)	$\boxed{562}$	67.0 (0.1)	491	62.8 (0.1)	460	57.6 (0.1)	422	59.1 (0.1)
SSOR-L	528	83.2 (0.1)	575	84.5 (0.1)	523	90.6 (0.1)	471	80.1 (0.1)	425	76.4 (0.1)	427	74.0 (0.1)
ILU0	275	42.5 (2.5)	273	42.4 (2.5)	205	**32.7** (2.5)	202	**32.7** (2.5)	196	**32.7** (2.5)	**192**	34.2 (2.5)

Table 3.5: Comparison of Bi-CGSTAB and IDR(s) with different preconditioners. Soil profile 2 is used. Matvec and time in second are reported at the last load step, 26 kPa.

12×3×12	Bi-CGSTAB		IDR(1)		IDR(4)		IDR(6)		IDR(10)		IDR(20)	
No preconditioner	4422	11.7 (0.0)	3967	12.3 (0.0)	$\boxed{4800}$	15.6 (0.0)	3933	12.7 (0.0)	2662	10.0 (0.0)	1951	7.0 (0.0)
Jacobi	1141	2.6 (0.0)	1161	2.9 (0.0)	$\boxed{1602}$	5.1 (0.0)	1005	3.4 (0.0)	450	2.7 (0.0)	771	2.7 (0.0)
SSOR-LR	265	2.3 (0.0)	249	2.3 (0.0)	$\boxed{266}$	2.3 (0.0)	255	2.1 (0.1)	256	2.2 (0.0)	209	1.8 (0.0)
SSOR-L	259	1.4 (0.0)	291	1.6 (0.0)	$\boxed{306}$	1.9 (0.0)	274	1.5 (0.0)	235	1.3 (0.0)	229	1.4 (0.0)
ILU0	111	0.7 (0.1)	111	0.7 (0.1)	85	0.6 (0.1)	81	**0.6** (0.1)	77	**0.6** (0.1)	**77**	**0.6** (0.1)
24×6×24												
No preconditioner	7011	65.0 (0.0)	6085	56.9 (0.0)	89,362	871.9 (0.0)	22,760	227.4 (0.0)	34,100	357.8 (0.0)	7608	89.1 (0.0)
Jacobi	3043	36.7 (0.1)	3087	36.7 (0.1)	$\boxed{\text{Fail}}$	–	$\boxed{3942}$	51.3 (0.1)	$\boxed{3590}$	48.6 (0.1)	2783	41.1 (0.1)
SSOR-LR	541	33.5 (0.1)	500	31.0 (0.1)	$\boxed{616}$	37.4 (0.1)	$\boxed{551}$	33.3 (0.1)	484	30.0 (0.1)	451	28.6 (0.1)
SSOR-L	553	33.2 (0.1)	573	34.6 (0.1)	$\boxed{645}$	39.1 (0.1)	$\boxed{650}$	39.1 (0.1)	$\boxed{645}$	39.9 (0.1)	523	33.0 (0.1)
ILU0	265	7.1 (0.5)	259	7.0 (0.5)	193	5.5 (0.5)	177	**5.1** (0.5)	171	**5.1** (0.5)	**165**	**5.1** (0.5)
32×8×32												
No preconditioner	17,110	400.2 (0.0)	17,255	429.1 (0.0)	304,511	8969.3 (0.0)	297,384	8020.6 (0.0)	155,668	4353.6 (0.0)	106,538	3350.0 (0.0)
Jacobi	4,406	103.0 (0.1)	4,611	118.8 (0.1)	19,989	533.4 (0.1)	9,712	265.7 (0.1)	7,049	200.9 (0.1)	3,558	112.1 (0.1)
SSOR-LR	398	36.7 (0.1)	406	37.8 (0.1)	394	38.1 (0.1)	$\boxed{475}$	46.4 (0.1)	381	37.9 (0.1)	336	35.2 (0.1)
SSOR-L	412	57.1 (0.1)	421	58.4 (0.1)	$\boxed{428}$	60.1 (0.1)	329	45.1 (0.1)	294	34.2 (0.1)	290	34.0 (0.1)
ILU0	454	29.5 (1.3)	466	30.6 (1.3)	265	17.6 (1.3)	250	**16.8** (1.3)	235	16.9 (1.3)	**231**	**16.8** (1.3)

Table 3.6: Comparison of Bi-CGSTAB and IDR(s) with different preconditioners. Soil profile 3 is used. Matvec and time in second are reported at the last load step, 40 kPa.

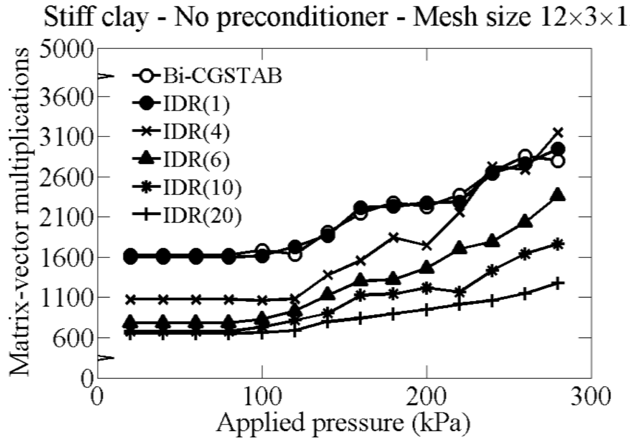
12×3×12	Bi-CGSTAB		IDR(1)		IDR(4)		IDR(6)		IDR(10)		IDR(20)	
No preconditioner	2131	4.1 (0.0)	2243	5.3 (0.0)	$\boxed{2430}$	5.7 (0.0)	2008	4.4 (0.0)	1330	3.1 (0.0)	980	2.6 (0.0)
Jacobi	846	1.7 (0.0)	861	1.8 (0.0)	674	1.9 (0.0)	638	1.7 (0.0)	577	1.4 (0.0)	548	1.7 (0.0)
SSOR-LR	189	2.0 (0.0)	191	2.0 (0.0)	184	2.1 (0.0)	168	1.8 (0.0)	149	1.6 (0.0)	145	1.7 (0.0)
SSOR-L	178	1.2 (0.0)	187	1.3 (0.0)	186	1.3 (0.0)	160	1.1 (0.0)	153	1.1 (0.0)	140	1.2 (0.0)
ILU0	83	0.5 (0.1)	83	0.5 (0.1)	60	**0.4** (0.1)	58	**0.4** (0.1)	57	**0.4** (0.1)	**56**	**0.4** (0.1)
24×6×24												
No preconditioner	10,300	146.0 (0.0)	10,307	147.1 (0.0)	103,019	1539.1 (0.0)	25,409	385.1 (0.0)	13,768	221.3 (0.0)	10,201	184.2 (0.0)
Jacobi	2965	91.1 (0.1)	3021	95.0 (0.1)	$\boxed{3252}$	106.0 (0.1)	2568	86.3 (0.1)	2068	72.6 (0.1)	1454	57.3 (0.1)
SSOR-LR	512	32.4 (0.1)	517	33.1 (0.1)	$\boxed{584}$	37.9 (0.1)	623	40.6 (0.1)	441	29.4 (0.1)	369	25.7 (0.1)
SSOR-L	555	71.7 (0.1)	537	69.8 (0.1)	536	70.3 (0.1)	468	61.7 (0.1)	416	55.7 (0.1)	339	47.1 (0.1)
ILU0	325	16.3 (0.9)	335	16.9 (0.9)	138	7.6 (0.9)	134	**7.3** (0.9)	128	**7.3** (0.9)	**124**	7.5 (0.9)
32×8×32												
No preconditioner	20,210	450.2 (0.0)	20,320	469.1 (0.0)	235,611	7869.3 (0.0)	10,738	2520.6 (0.0)	16,666	3353.6 (0.0)	10,664	2350.0 (0.0)
Jacobi	3764	182.1 (0.1)	3746	187.3 (0.1)	$\boxed{4747}$	250.1 (0.1)	$\boxed{4582}$	244.8 (0.1)	$\boxed{3921}$	221.7 (0.1)	2801	195.1 (0.1)
SSOR-LR	672	103.3 (0.1)	679	105.3 (0.1)	$\boxed{760}$	119.6 (0.10	$\boxed{759}$	120.6 (0.1)	633	102.7 (0.1)	541	92.8 (0.1)
SSOR-L	742	173.8 (0.1)	719	169.0 (0.1)	$\boxed{733}$	174.2 (0.1)	684	163.7 (0.1)	595	144.4 (0.1)	566	142.5 (0.1)
ILU0	3565	461.1 (2.5)	3429	436.7 (2.5)	398	54.1 (2.5)	349	50.4 (2.5)	306	44.7 (2.5)	**289**	**44.1** (2.5)

The general trend in these tables is: the matvec count reduces from left to right and from top to bottom for each problem size. Hence the matvec count is minimum at the bottom right corner and maximum at the top left corner. The number of matvec required by Bi-CGSTAB and IDR(1) is very close in all the cases with less than 10 percent differences. These differences in matvec count between Bi-CGSTAB and IDR(1) is due to the round-off error. Figure 3.3 to Figure 3.11 also show that Bi-CGSTAB and IDR(1) do behave almost identically in all the cases as expected from the theory of IDR method.

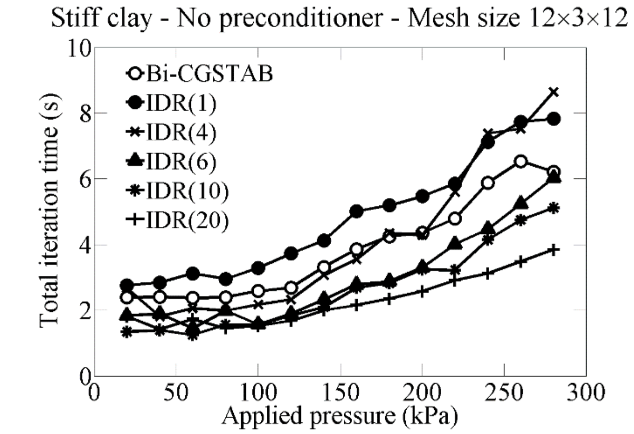
The matvec count and the total iteration time reduce when s increases, which shows that IDR is more efficient than Bi-CGSTAB. There are cases, the boxed numbers in Table 3.4, Table 3.5, and Table 3.6, that IDR($s > 1$) requires more matvec than IDR(1) and Bi-CGSTAB. This could be due to the choice of shadow matrix P as random matrix. When $s = 1$, P is set to the initial residual vector and this choice has shown to be a good choice. However, for $s > 1$, as discussed in Section 2.1.2, currently there is no similar recommendation for P hence random matrix is recommended. Random matrix P is not a bad choice either because it works well when ILU0 preconditioner is used and ILU0 performs better than other preconditioner tested. Sonneveld and Gijzen[162] (2008) recommended the use of s = 4 for short recurrence of IDR method. However for the linear systems tested here, IDR(4) is not the optimal in both matvec count and total iteration time. As discuss in Section 2.1.2, when s increases, the matvec count reduces but the storage of matrix $P^{N\times s}$ increases and the time spent to solve the linear system $s \times s$ increases. This is reflected in the tables, IDR(20) has the least matvec count but longer total iteration time especially for denser meshes. IDR(6) and IDR(10) are competitive in total iteration time. IDR(10) requires less matvec but about the same time as IDR(6). In this study, IDR(6) is chosen and is used in further discussion because smaller s is preferred when taking into account of the generation and storing the random matrix $P^{N\times s}$.

As discussed in the previous paragraph, the matvec count reduces from top to bottom for each problem size. This reflects the efficiency of the preconditioner. When no preconditioner is used, all the Krylov solvers do not converge within 5000 matvec for the medium ($24\times6\times24$ mesh) and large ($32\times8\times32$ mesh) problem sizes. Jacobi is the cheapest preconditioner because it is trivial to form but is also the least efficient preconditioner. The total iterationt time is dominated by the iteration time due to large amount of matvec required. SSOR-L is just as efficient as SSOR-LR and Bi-CGSTAB with SSOR-L does not show outstanding performance as shown by Mroueh, and Shahrour[115] (1999). Moreover, with the same matvec, total iteration time from SSOR-L is often more than from SSOR-LR because the matvec step requires more operations. ILU0 is the most expensive preconditioner here: the time to form this preconditioner is 25 times more than time to form Jacobi preconditioner. Nevertheless, the reduction in matvec count pays off and the total iteration time is the least among all the preconditioners tested. These numerical results agree with review in Section 2.2. Discussions in later parts will adopt ILU0 as the default preconditioner for the nonsymmetric linear system due to non-associated MC model.

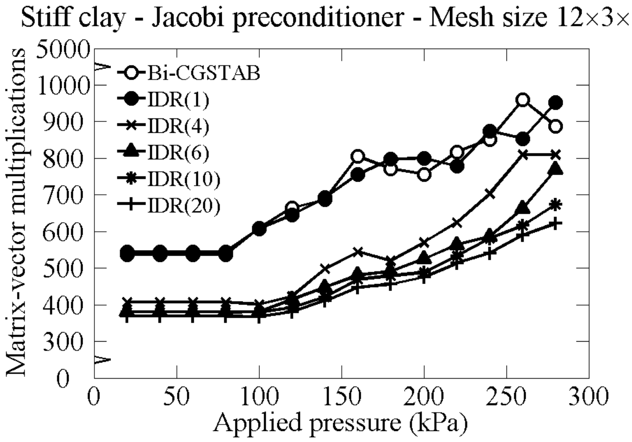
Figure 3.3 to Figure 3.11 plot the number of matvec and total iteration time required by Bi-CGSTAB and IDR(*s*) with *s* consequently taking the value of 1, 4, 6, 10, 20 when the applied pressure increase to the maximum value in Table 3.2. The matvec counts increase when the applied pressure increases, makes the linear system at the last load step the hardest to solve as mentioned earlier in this section. These figures agree with the conclusion from previous tables: IDR(1) is equivalent to Bi-CGSTAB; IDR(20) is the most efficient in term of matvec count, even converges when other methods fail but is not the most efficient in term of total iteration time; IDR(6) is the most optimal in total iteration time and memory storage; ILU0 is the most efficient preconditioner among those tested.


Stiff clay - No preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

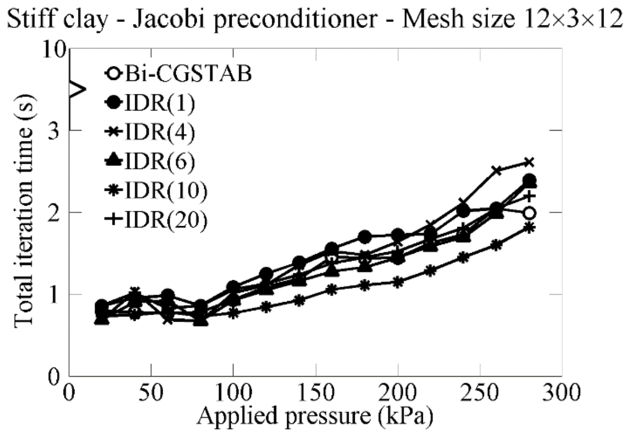
(a)


Stiff clay - No preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

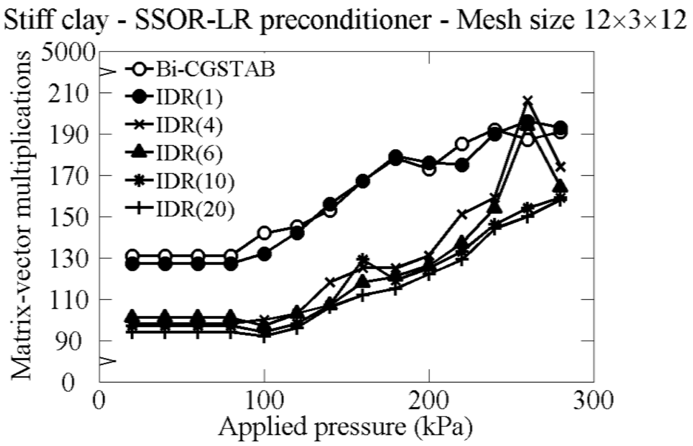
(b)


Stiff clay - Jacobi preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

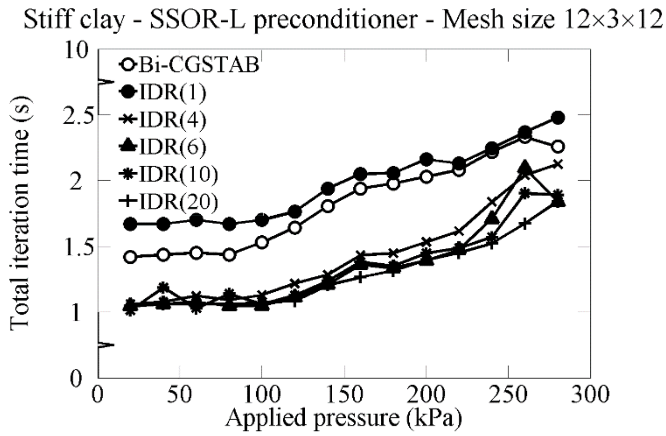
(c)


Stiff clay - Jacobi preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(d)


Stiff clay - SSOR-LR preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(e)


Stiff clay - SSOR-L preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(f)

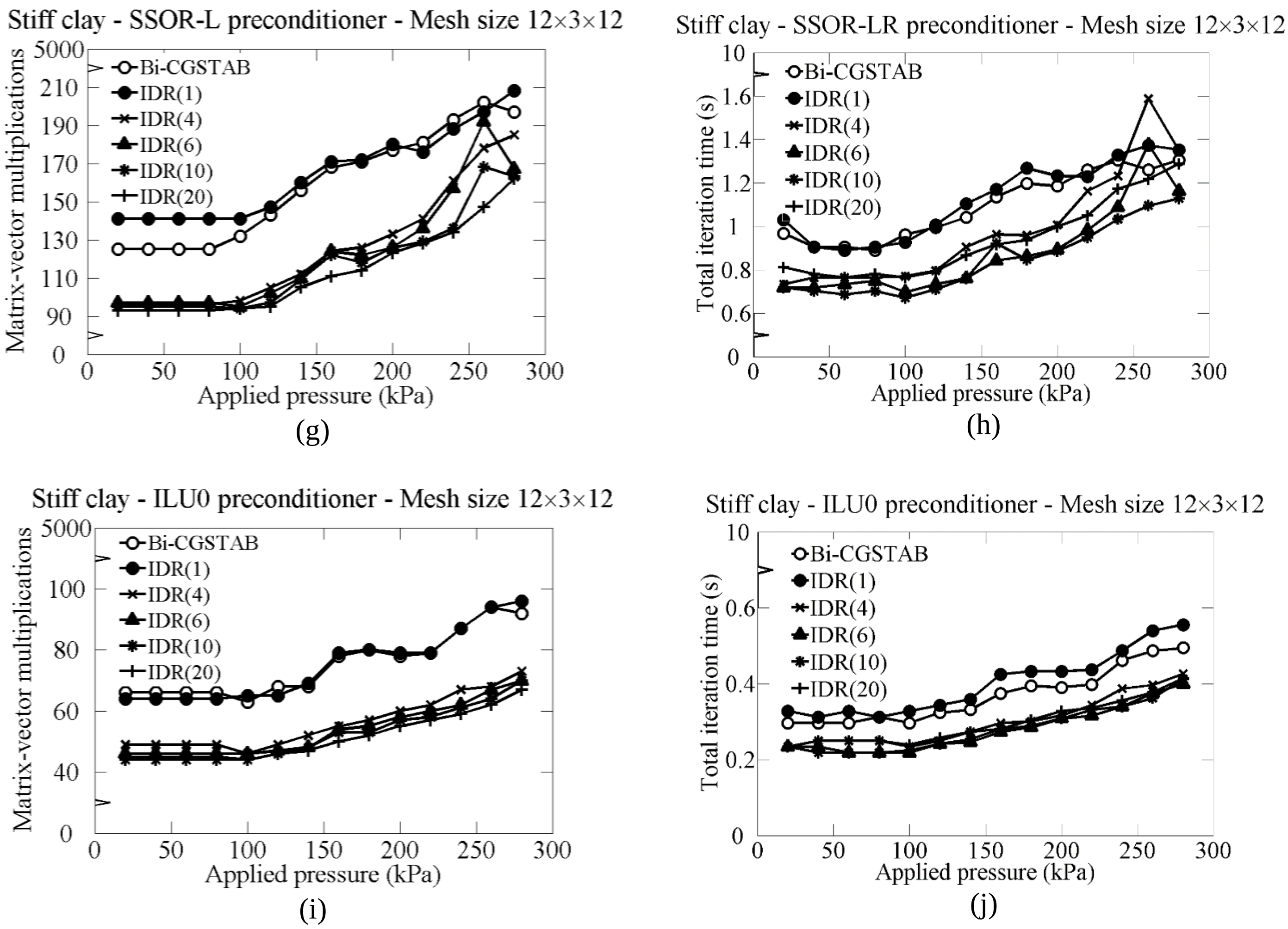
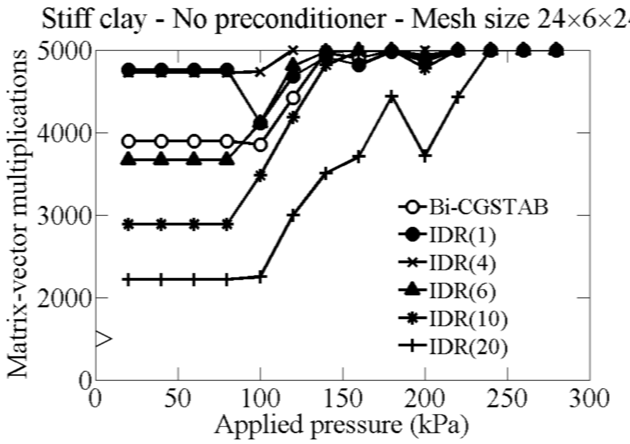
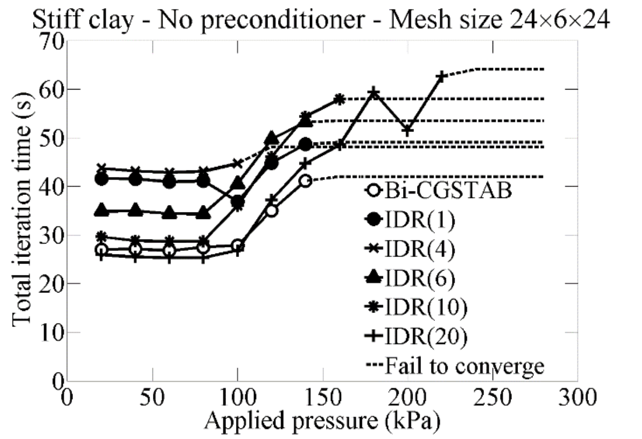


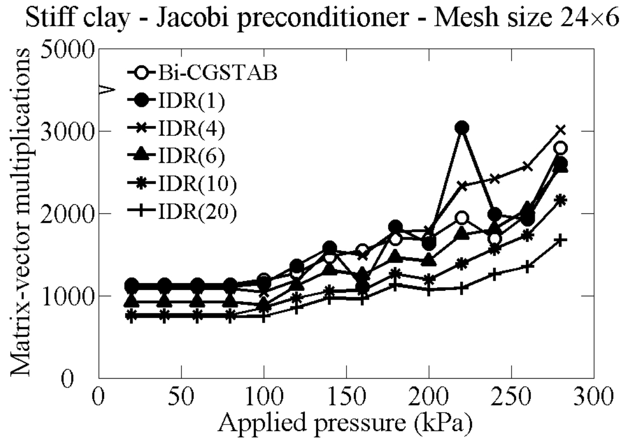
Figure 3.3: Comparison of Bi-CGSTAB and IDR(s) with s = 1, 4, 6, 10, and 20. Mesh size 12×3×12. Soil profile 1 is used.


Stiff clay - No preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

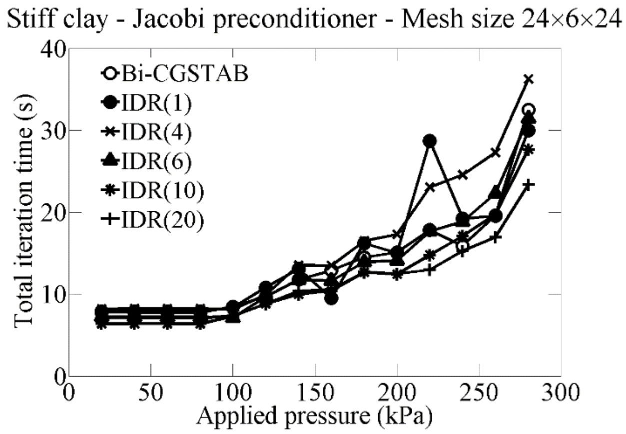
(a)


Stiff clay - No preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
Fail to converge

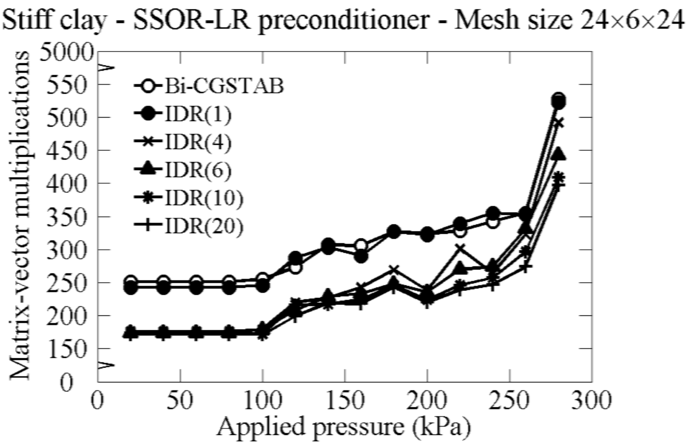
(b)


Stiff clay - Jacobi preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

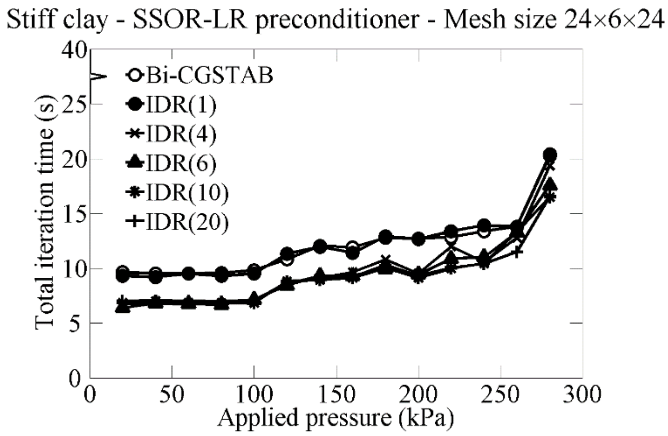
(c)


Stiff clay - Jacobi preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(d)


Stiff clay - SSOR-LR preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(e)


Stiff clay - SSOR-LR preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(f)

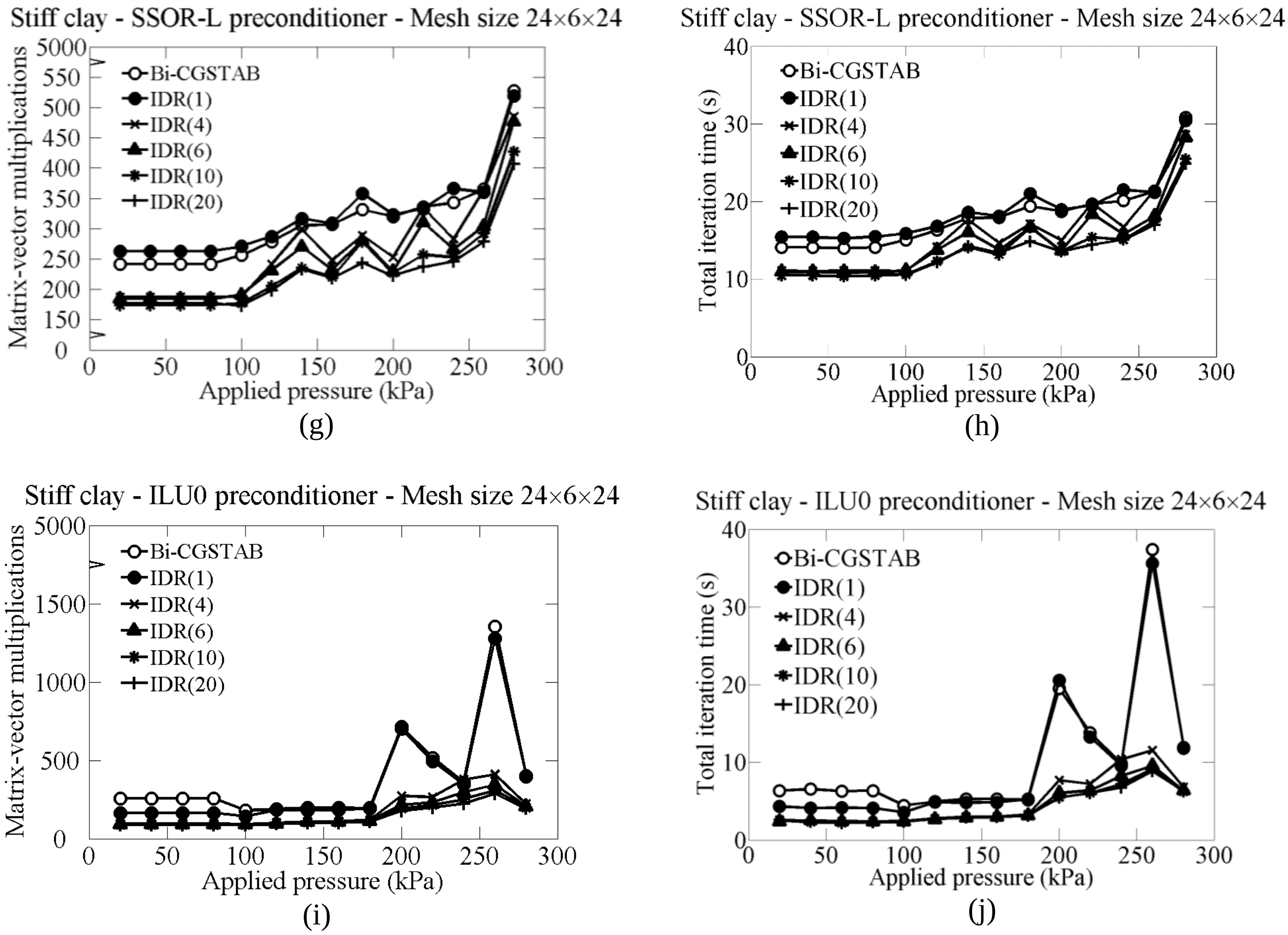
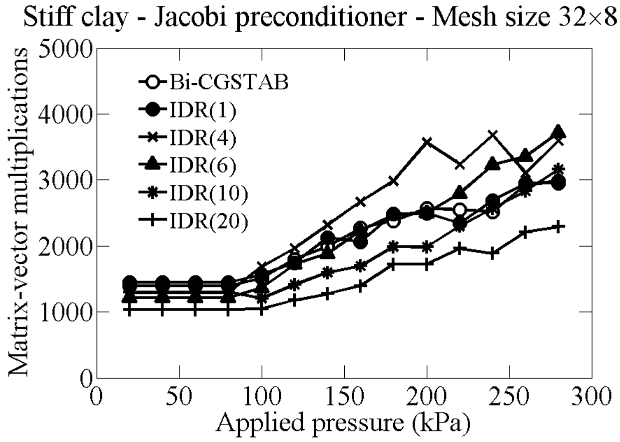
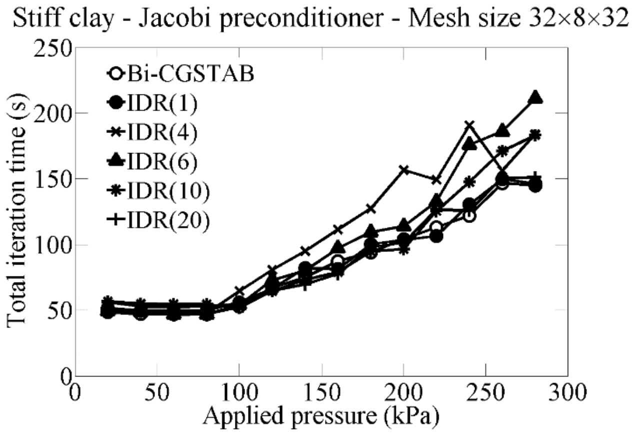


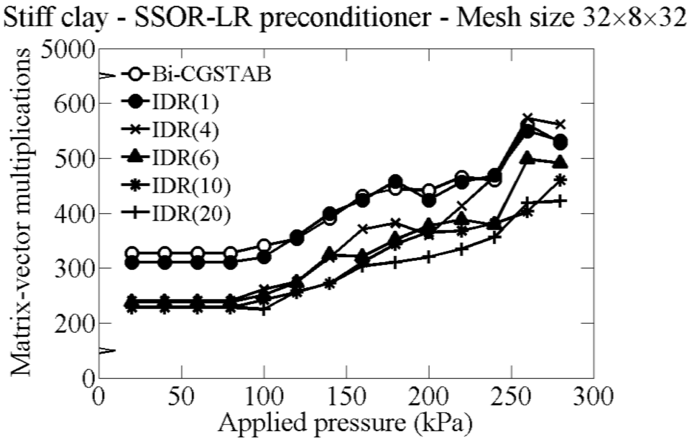
Figure 3.4: Comparison of Bi-CGSTAB and IDR(*s*) with s = 1, 4, 6, 10, and 20. Mesh size 24×6×24. Soil profile 1 is used.


Stiff clay - Jacobi preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

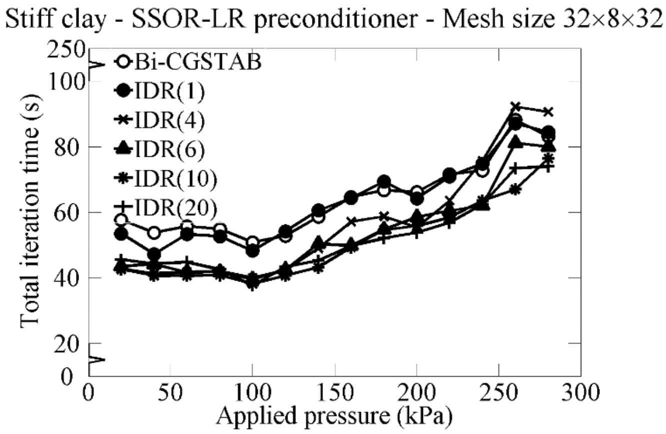
(a)


Stiff clay - Jacobi preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

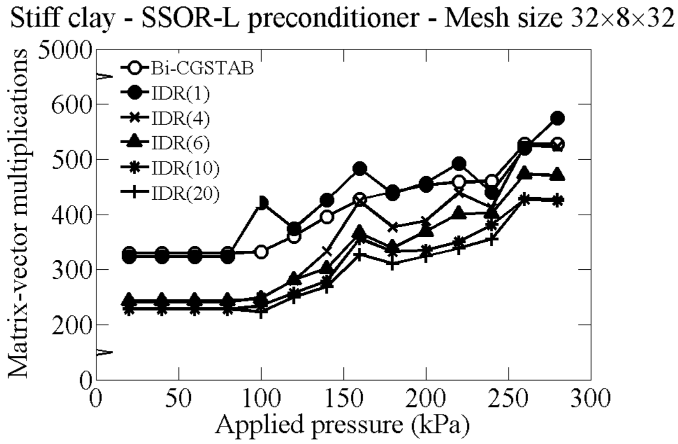
(b)


Stiff clay - SSOR-LR preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

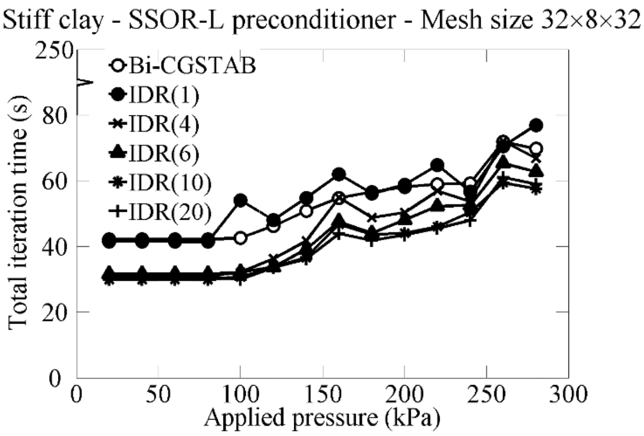
(c)


Stiff clay - SSOR-LR preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

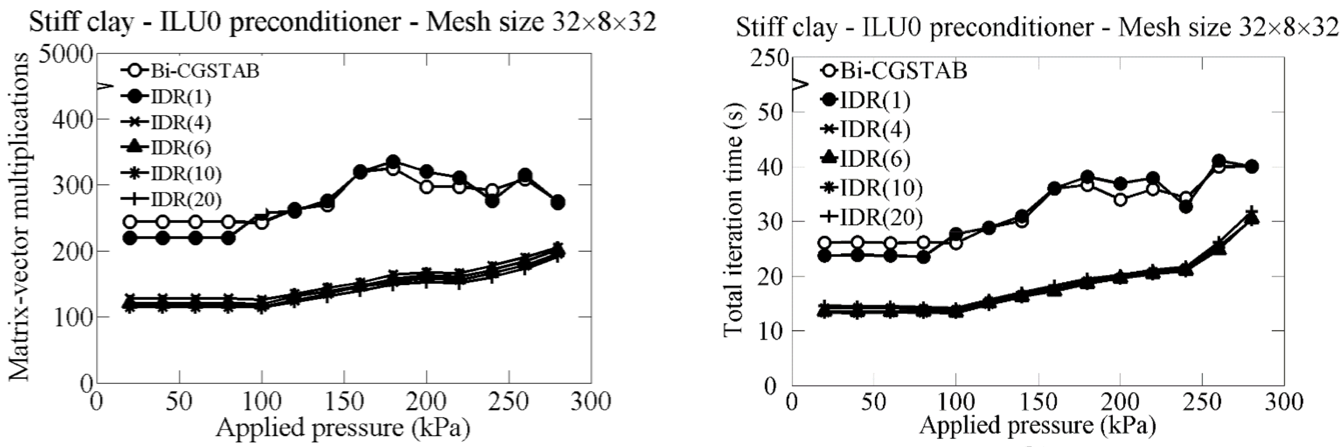
(d)


Stiff clay - SSOR-L preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(e)

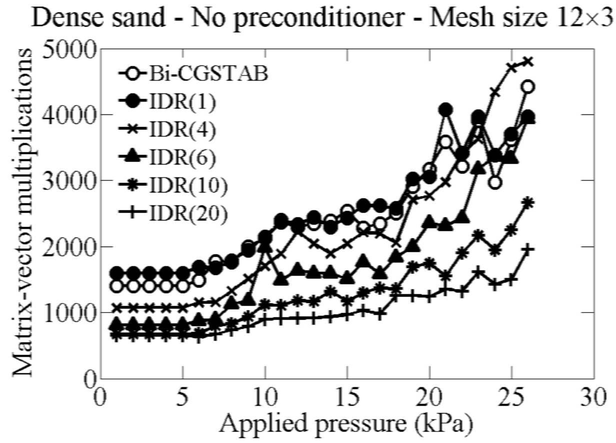

Stiff clay - SSOR-L preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(f)

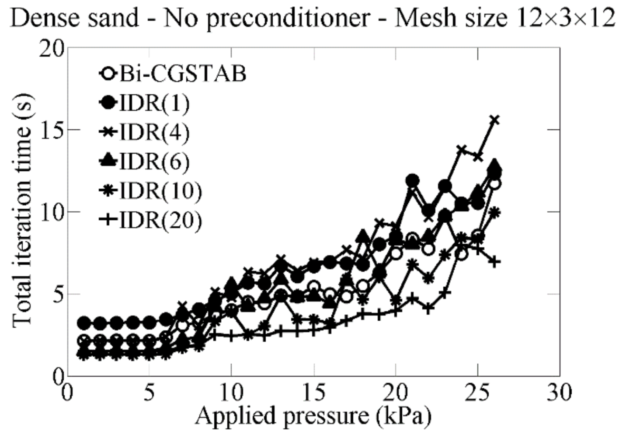


(g) (h)

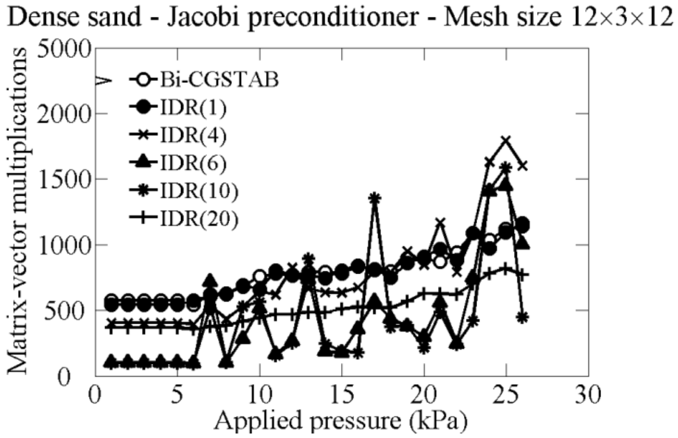
Figure 3.5: Comparison of Bi-CGSTAB and IDR(s) with s = 1, 4, 6, 10, and 20 Mesh size $32 \times 8 \times 32$. Soil profile 1 is used. All the methods do not converge when there is no preconditioner hence this case is not plotted here.


Dense sand - No preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

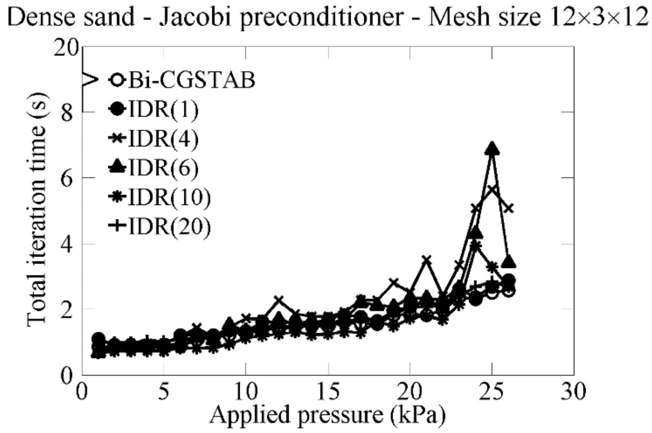
(a)


Dense sand - No preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

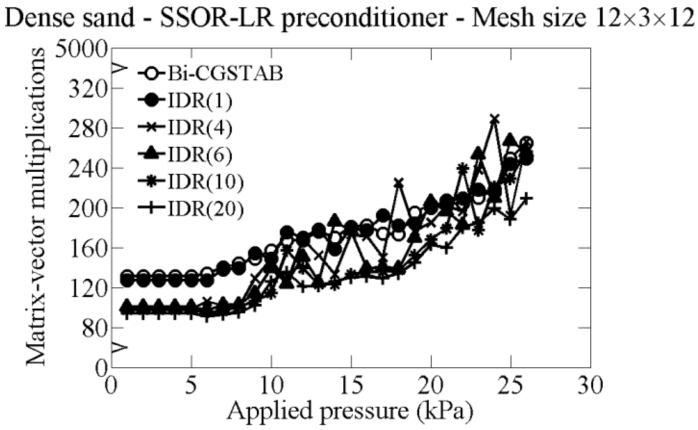
(b)


Dense sand - Jacobi preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

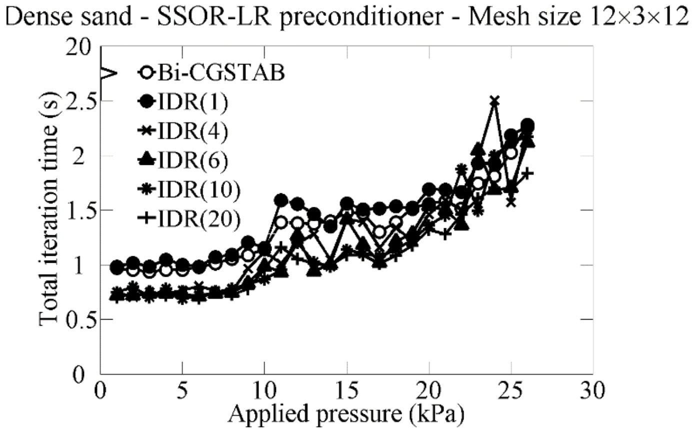
(c)


Dense sand - Jacobi preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(d)


Dense sand - SSOR-LR preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(e)


Dense sand - SSOR-LR preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(f)

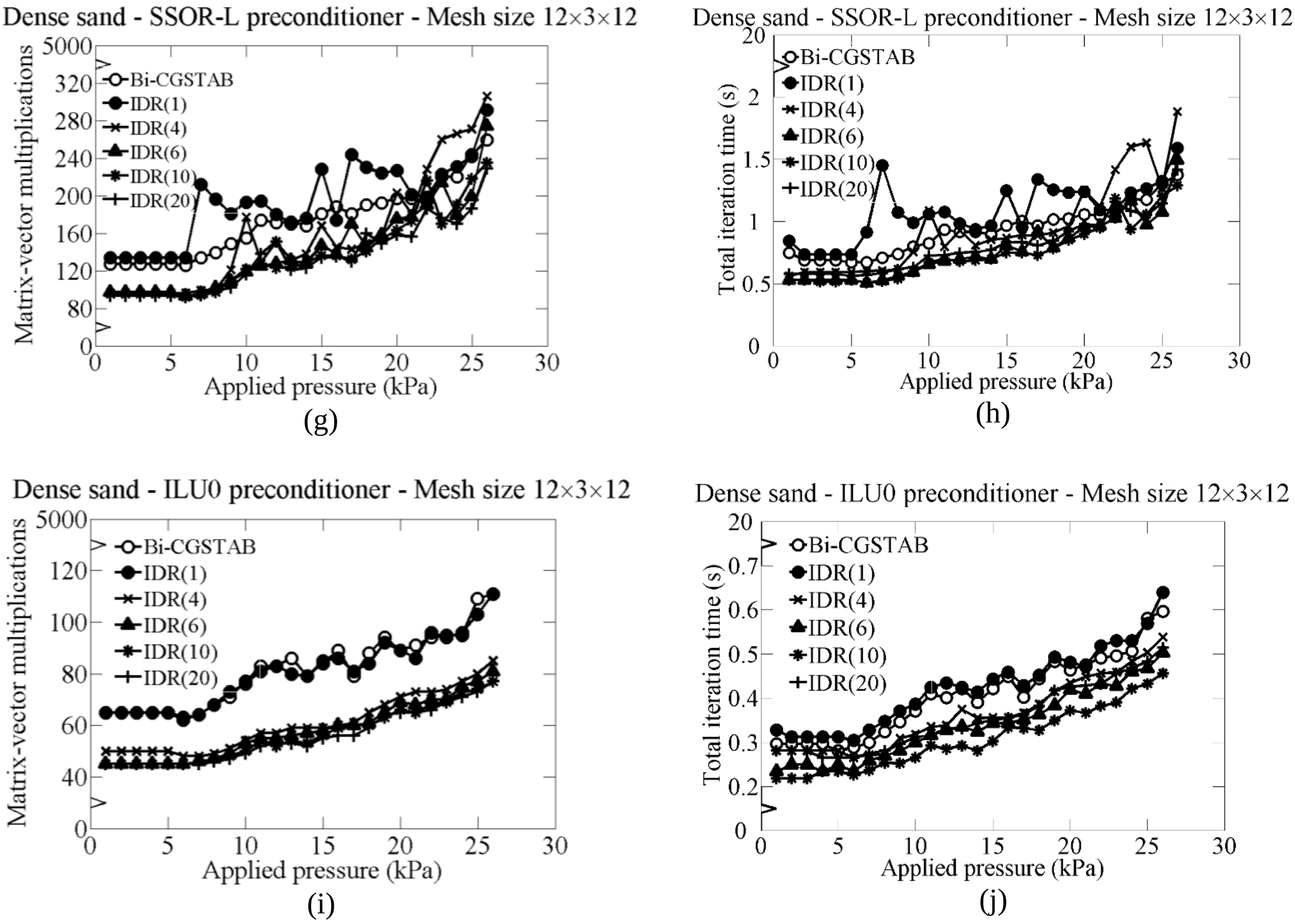
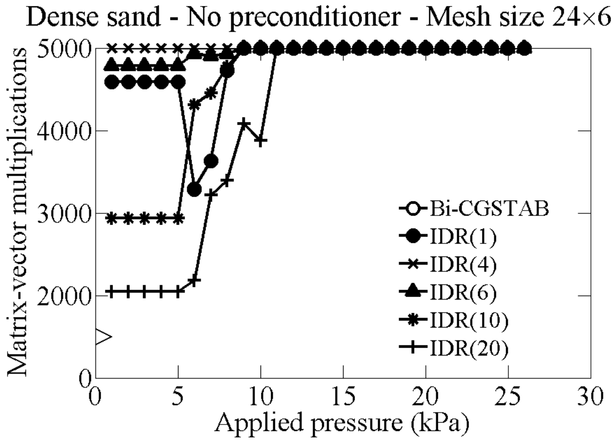
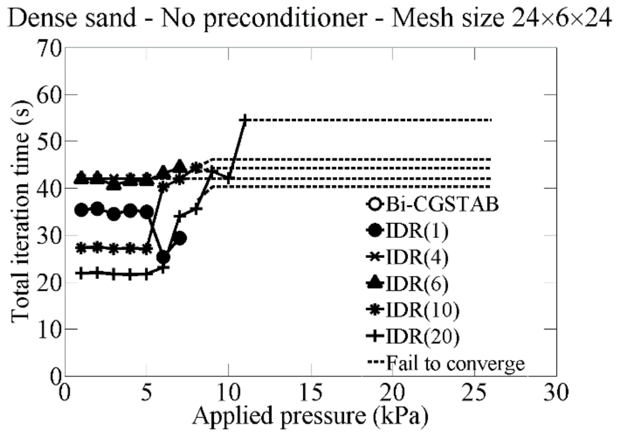
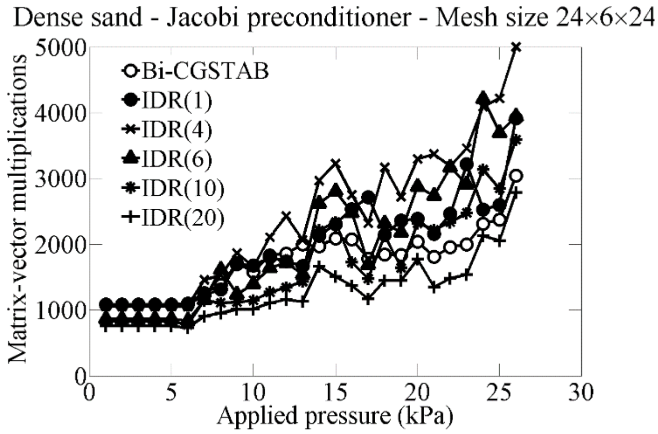
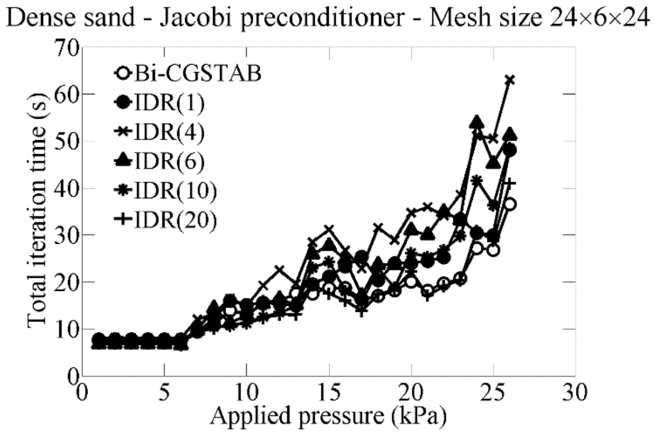


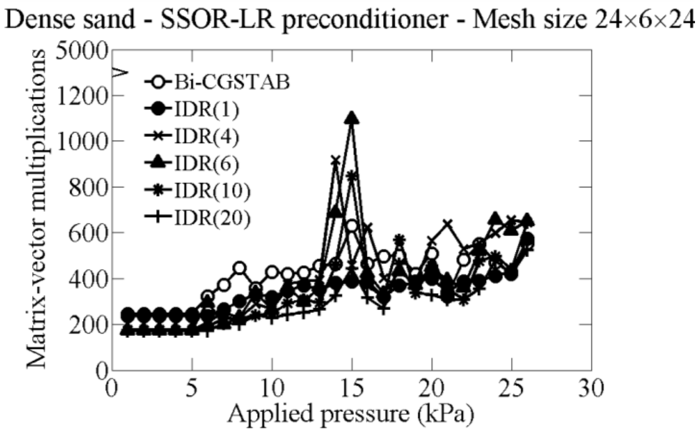
Figure 3.6: Comparison of Bi-CGSTAB and IDR(*s*) with s = 1, 4, 6, 10, and 20. Mesh size 12×3×12. Soil profile 2 is used.

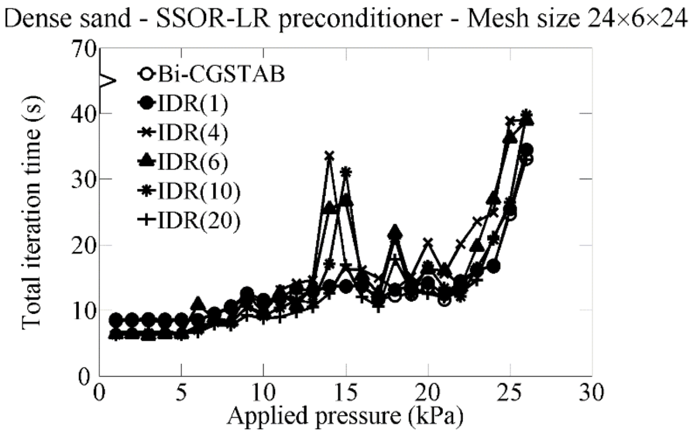

Dense sand - No preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
(a)

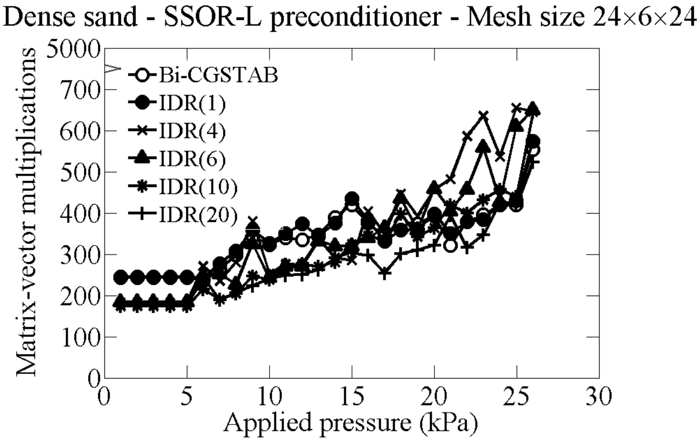

Dense sand - No preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
Fail to converge
(b)


Dense sand - Jacobi preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
(c)

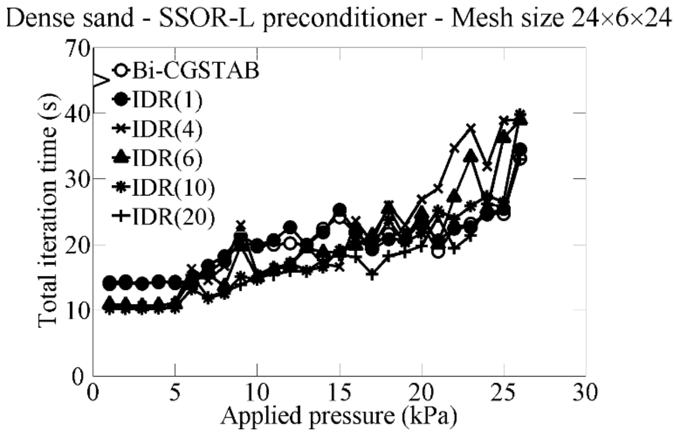

Dense sand - Jacobi preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
(d)


Dense sand - SSOR-LR preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
(e)

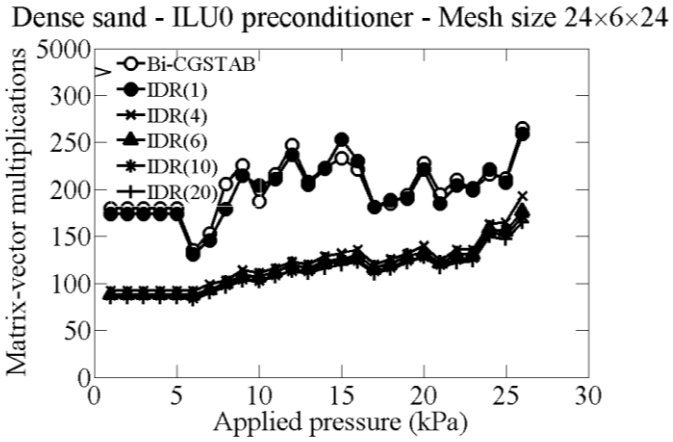

Dense sand - SSOR-LR preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
(f)



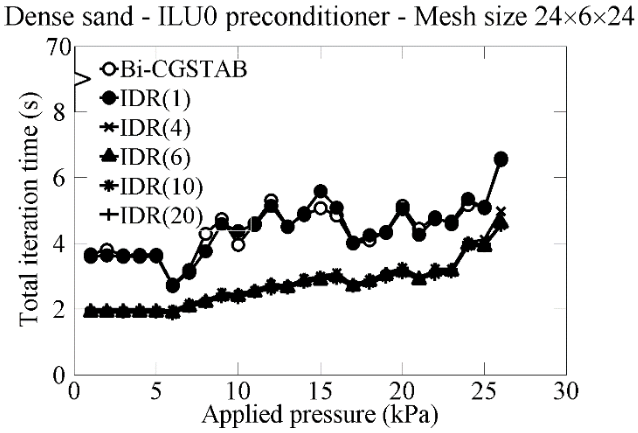
(g)



(h)

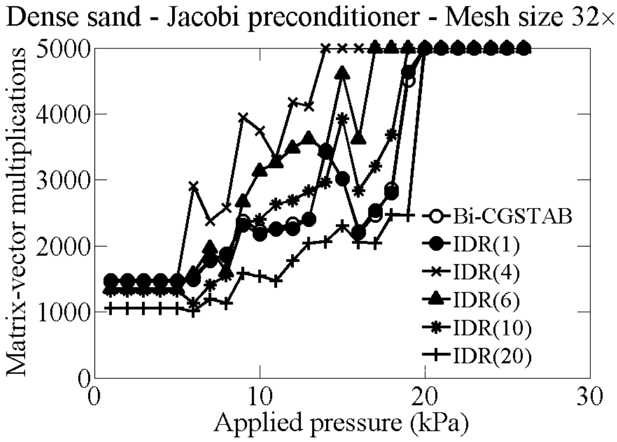


(i)

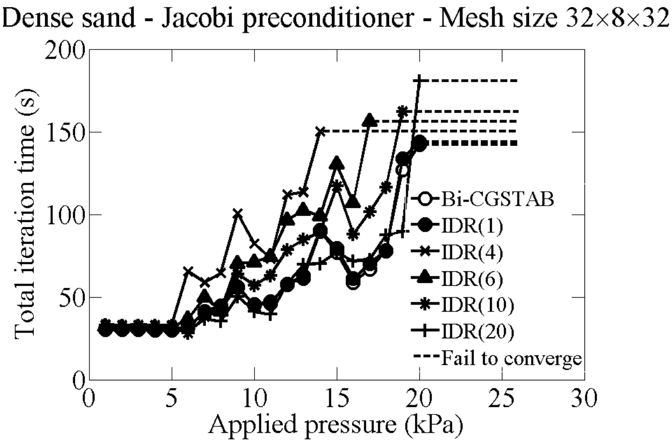


(j)

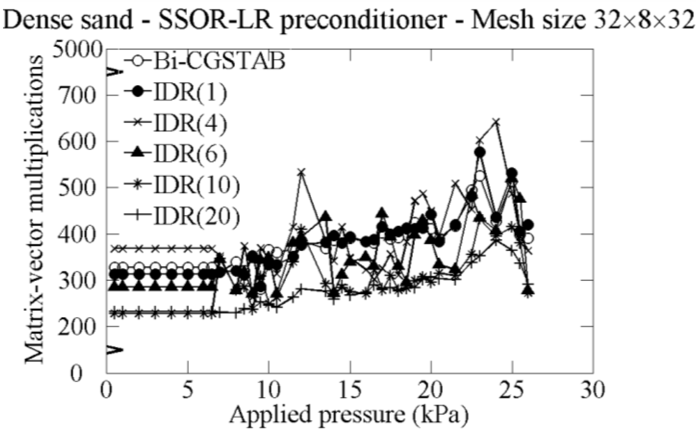
Figure 3.7: Comparison of Bi-CGSTAB and IDR(s) with s = 1, 4, 6, 10, and 20. Mesh size 24×6×24. Soil profile 2 is used.


Dense sand - Jacobi preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

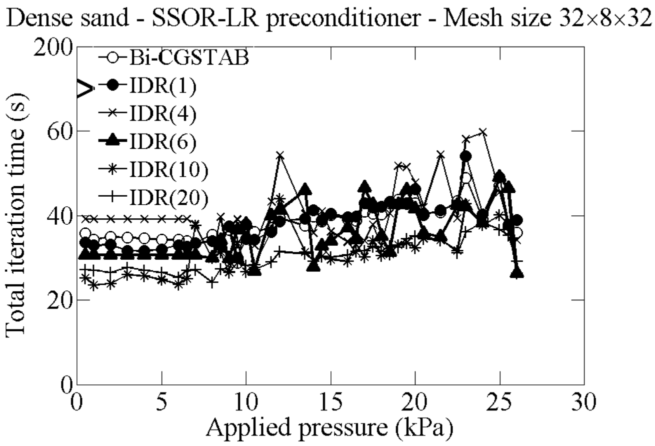
(a)


Dense sand - Jacobi preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
Fail to converge

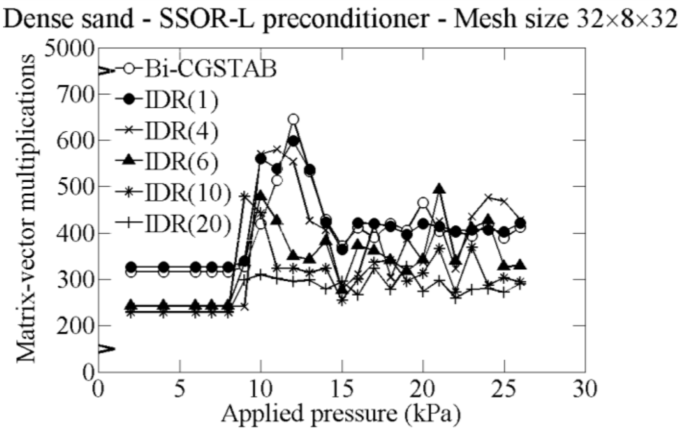
(b)


Dense sand - SSOR-LR preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

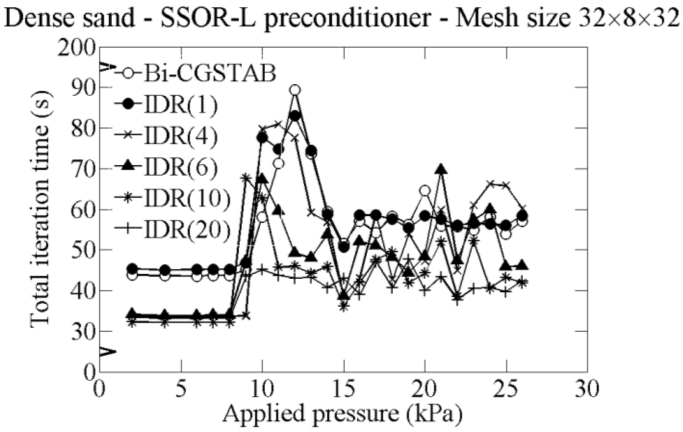
(c)


Dense sand - SSOR-LR preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

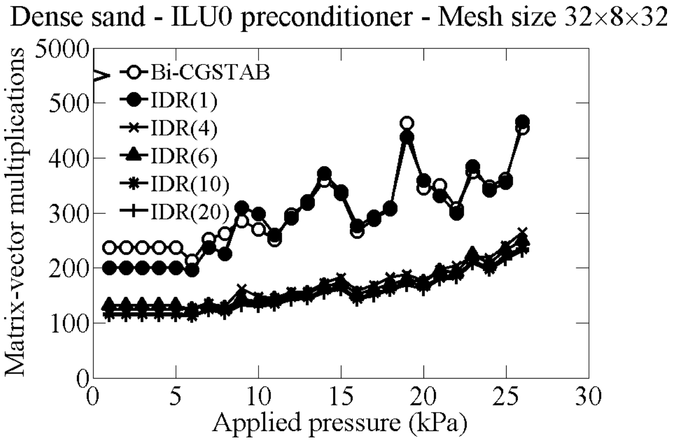
(d)


Dense sand - SSOR-L preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

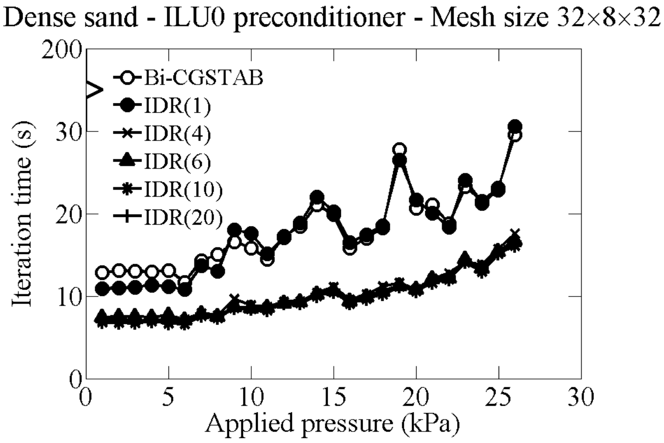
(e)


Dense sand - SSOR-L preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(f)

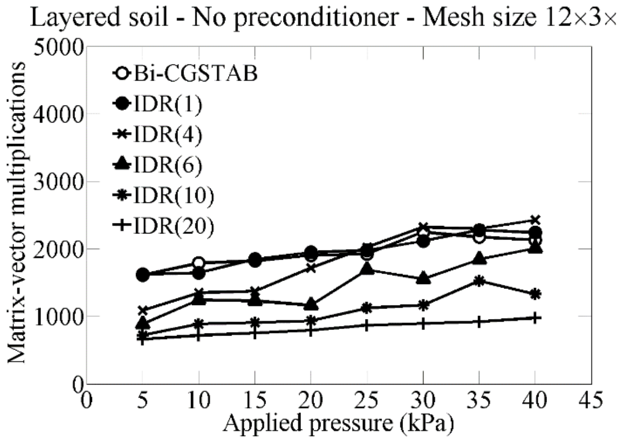


(g)

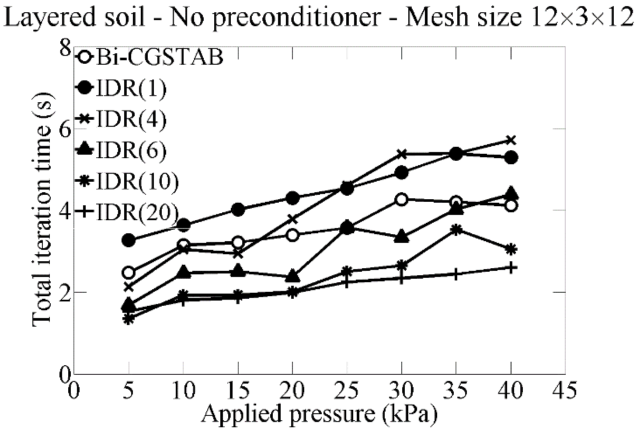


(h)

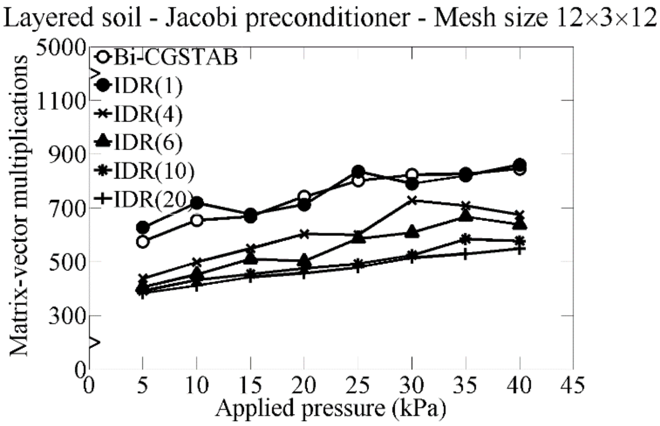
Figure 3.8: Comparison of Bi-CGSTAB and IDR(s) with s = 1, 4, 6, 10, and 20. Mesh size 32×8×32. Soil profile 2 is used. All the methods do not converge when there is no preconditioner hence this case is not plotted here.


Layered soil - No preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

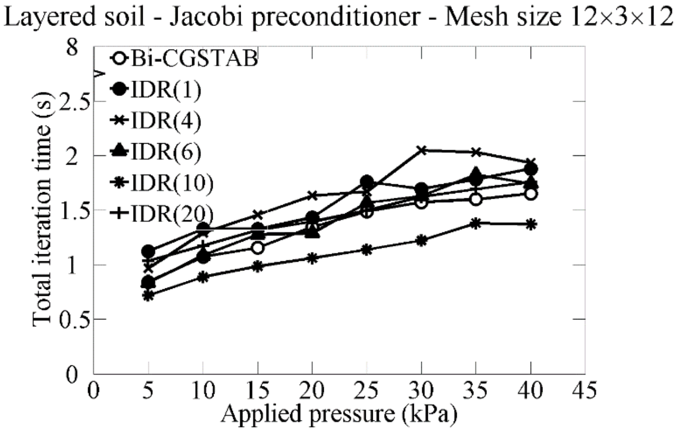
(a)


Layered soil - No preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

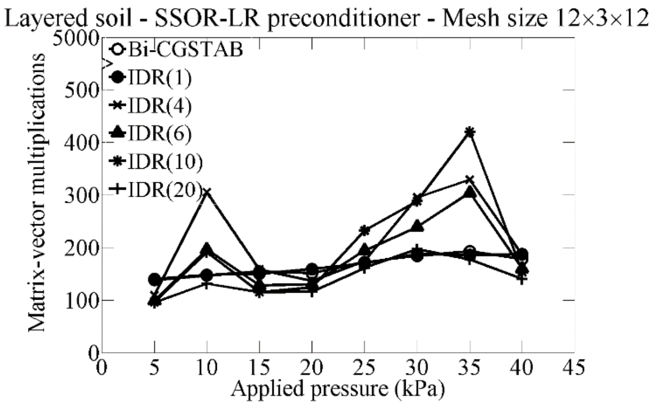
(b)


Layered soil - Jacobi preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

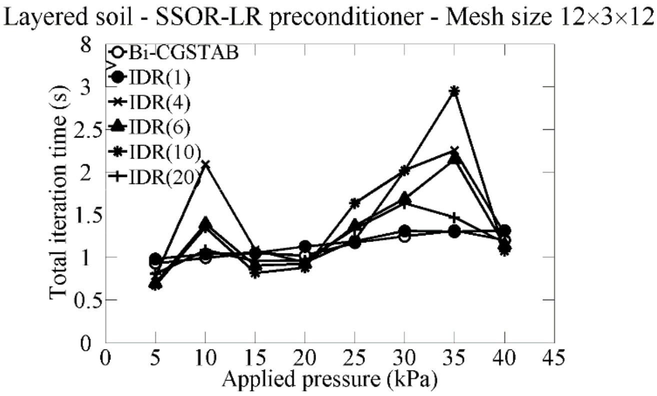
(c)


Layered soil - Jacobi preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

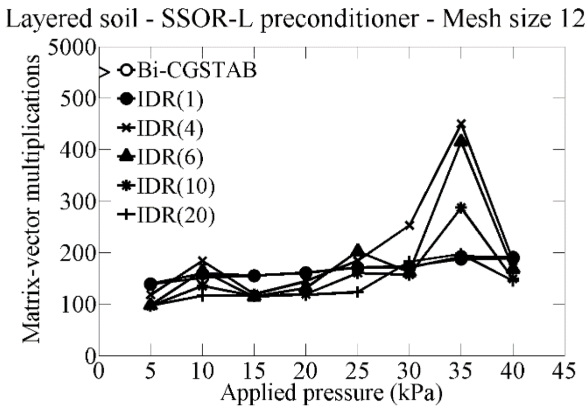
(d)


Layered soil - SSOR-LR preconditioner - Mesh size 12×3×12
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

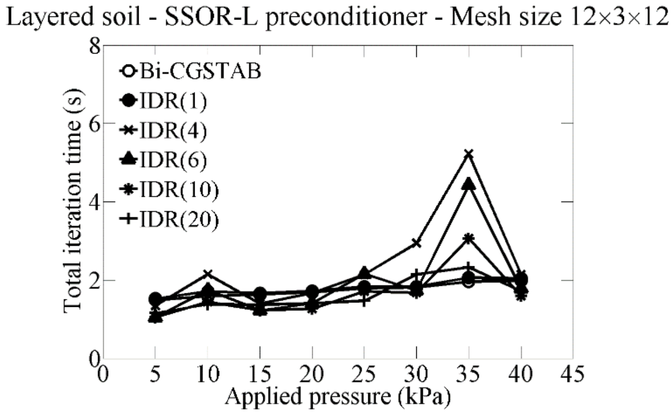
(e)


Layered soil - SSOR-LR preconditioner - Mesh size 12×3×12
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

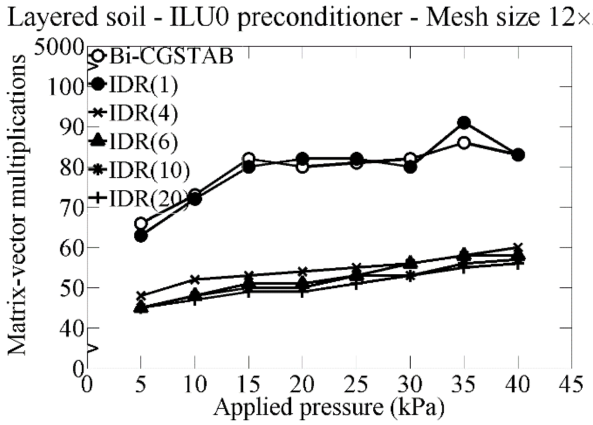
(f)



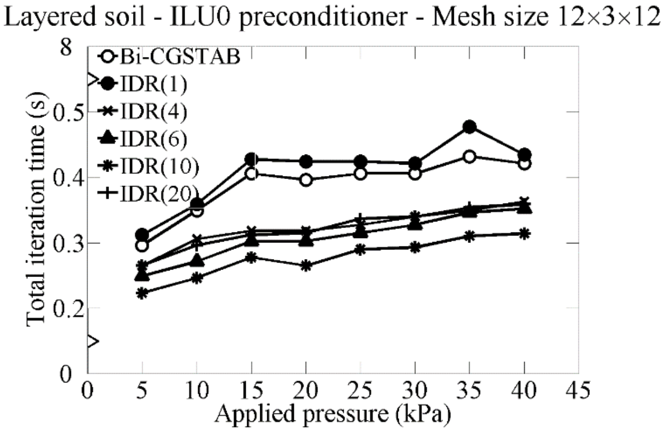
(g)



(h)

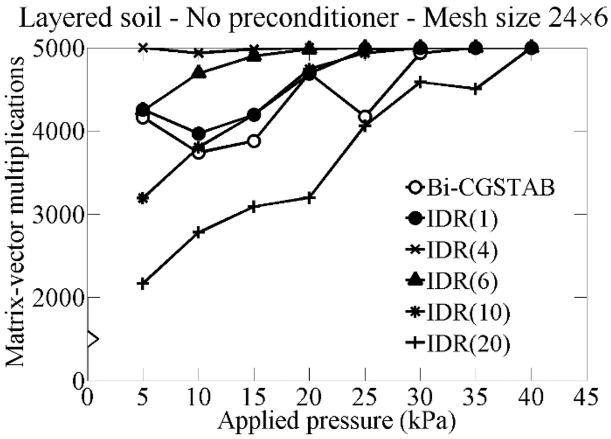


(i)

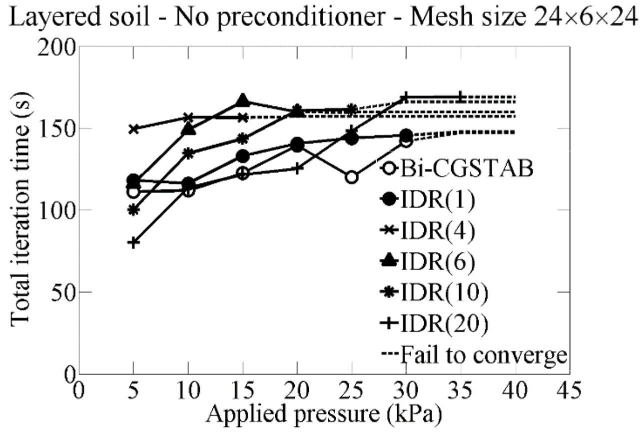


(j)

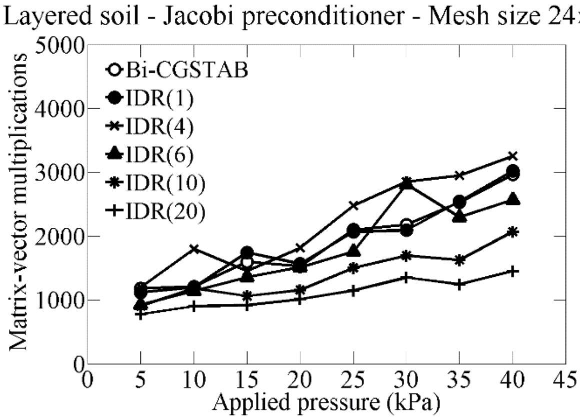
Figure 3.9: Comparison of Bi-CGSTAB and IDR(*s*) with s = 1, 4, 6, 10, and 20. Mesh size 16×3×16. Soil profile 3 is used.


Layered soil - No preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

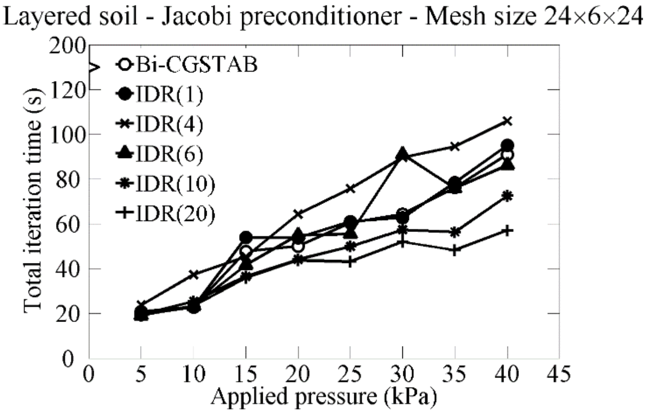
(a)


Layered soil - No preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)
Fail to converge

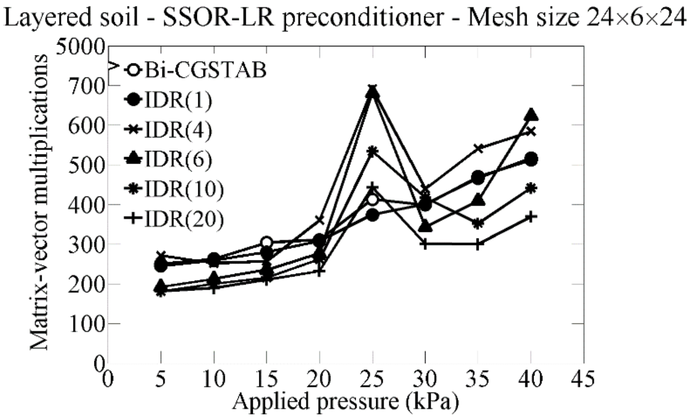
(b)


Layered soil - Jacobi preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

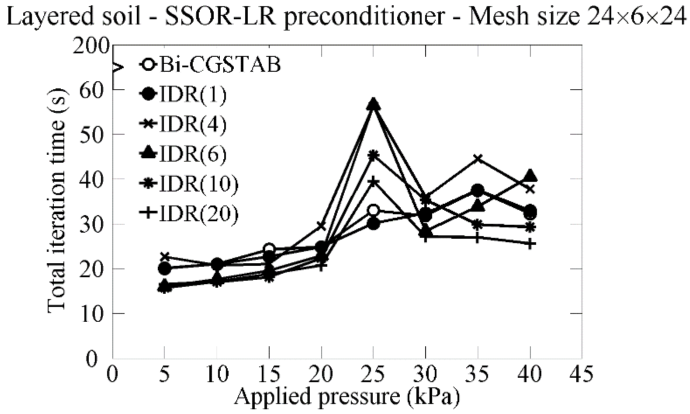
(c)


Layered soil - Jacobi preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

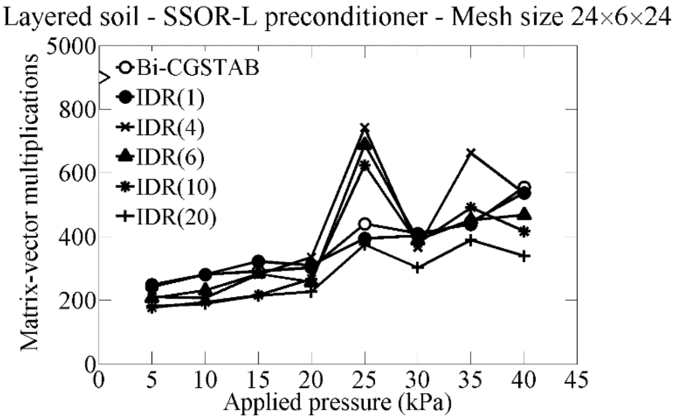
(d)


Layered soil - SSOR-LR preconditioner - Mesh size 24×6×24
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

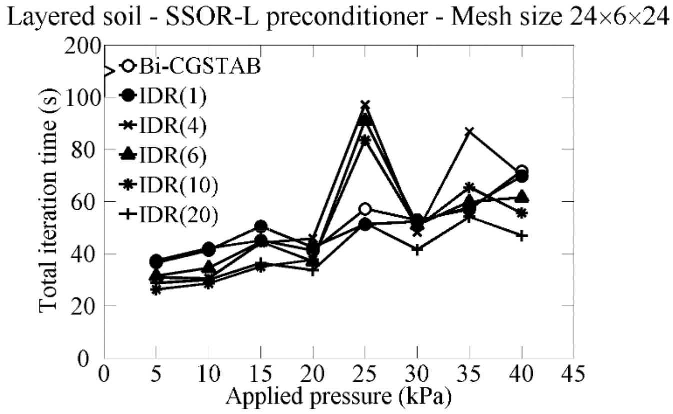
(e)


Layered soil - SSOR-LR preconditioner - Mesh size 24×6×24
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

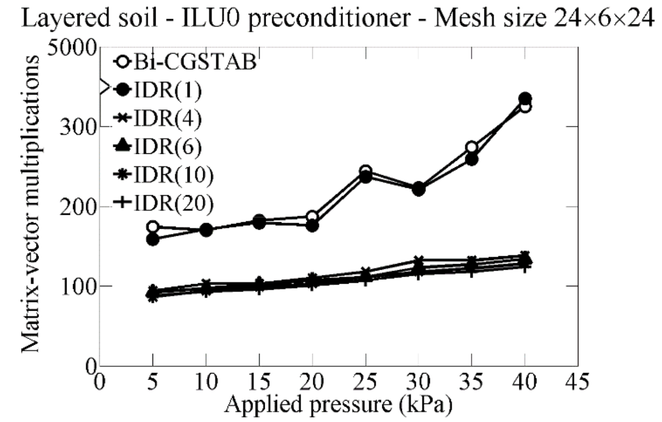
(f)



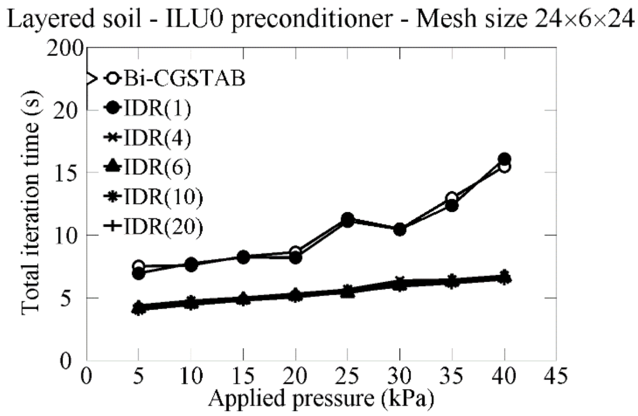
(g)



(h)

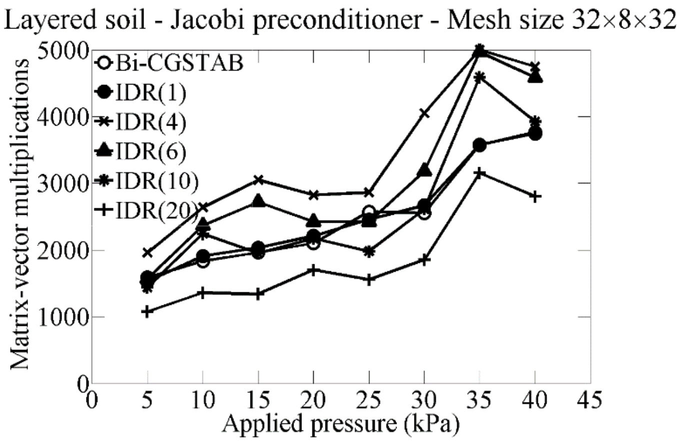


(i)

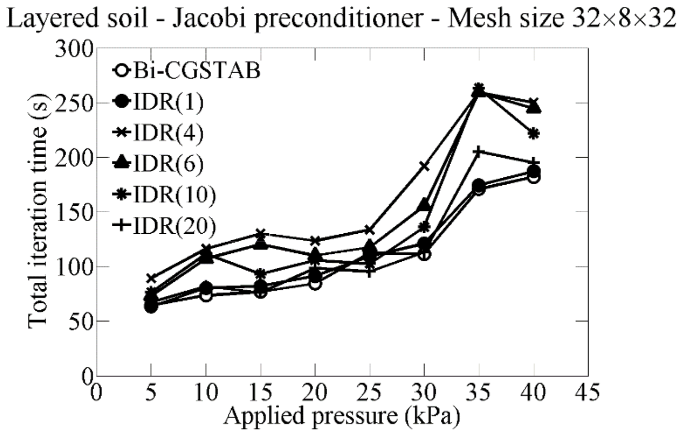


(j)

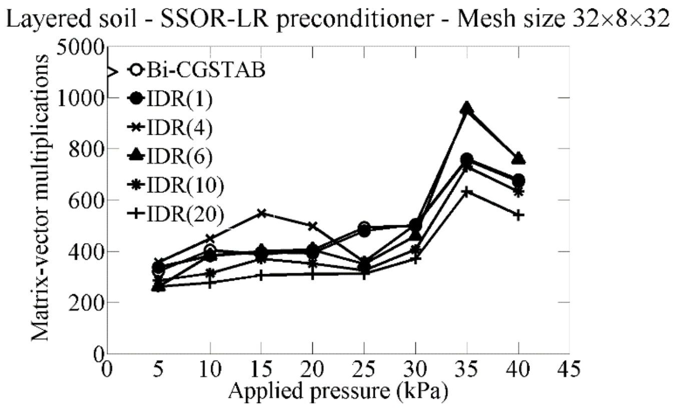
Figure 3.10: Comparison of Bi-CGSTAB and IDR(*s*) with s = 1, 4, 6, 10, and 20. Mesh size 24×6×24. Soil profile 3 is used.


Layered soil - Jacobi preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

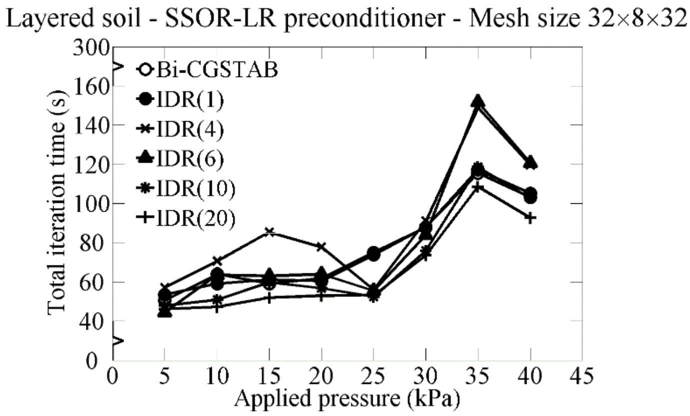
(a)


Layered soil - Jacobi preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

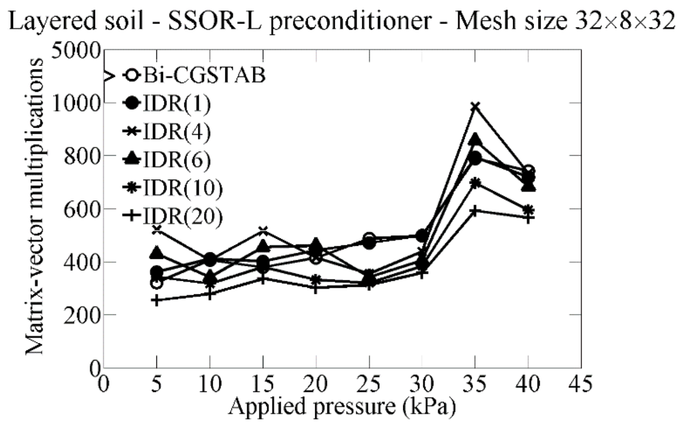
(b)


Layered soil - SSOR-LR preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

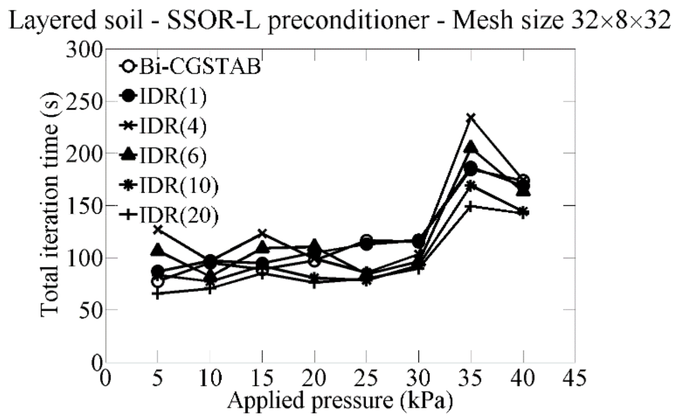
(c)


Layered soil - SSOR-LR preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

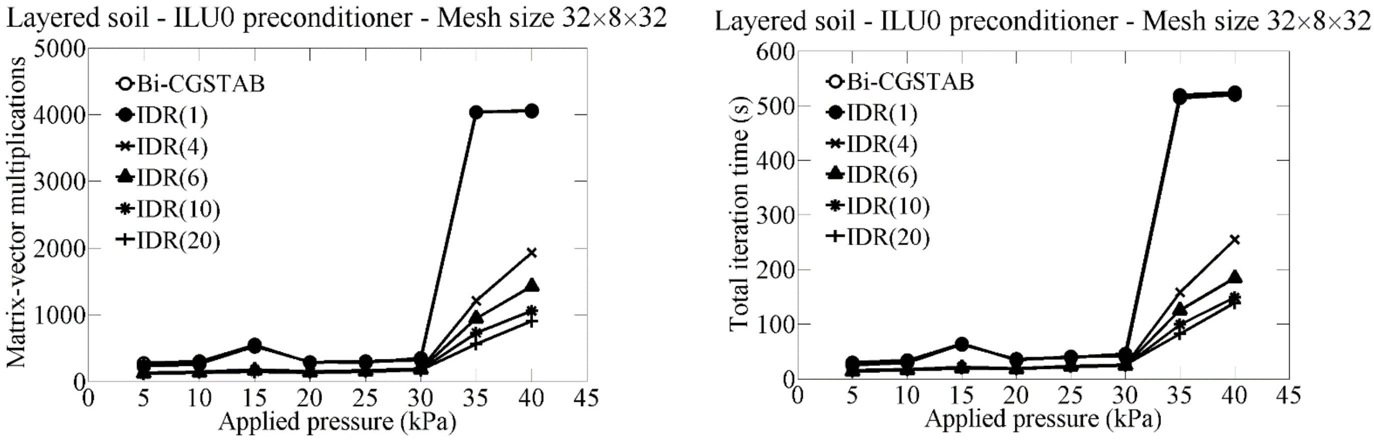
(d)


Layered soil - SSOR-L preconditioner - Mesh size 32×8×32
Matrix-vector multiplications
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(e)


Layered soil - SSOR-L preconditioner - Mesh size 32×8×32
Total iteration time (s)
Applied pressure (kPa)
Bi-CGSTAB
IDR(1)
IDR(4)
IDR(6)
IDR(10)
IDR(20)

(f)



(g) (h)

Figure 3.11: Comparison of Bi-CGSTAB and IDR(s) with s = 1, 4, 6, 10, and 20. Mesh size 32×8×32. Soil profile 3 is used. All the methods do not converge when there is no preconditioner hence this case is not plotted here.

3.5 Comparison of ILU0 and ILU(ρ, τ)

Section 3.4 has shown that ILU0 is a better preconditioner than Jacobi, SSOR-LR and SSOR-L hence implies that ILU0 is a better approximation of the nonsymmetric stiffness matrix K_{ep}. Section 2.2.3 has mentioned that ILUT can be more efficient and a competitor to ILU0 in the current problem. This section shows the comparison of ILU0 and ILUT(ρ, τ) for their practical application.

Figure 3.12, Figure 3.13 and Figure 3.14 show the comparison of ILU0 and ILUT(ρ, τ) with different values of ρ and τ. Due to the dropping scheme, not all ILUT are more efficient than ILU0. When ρ is less than 50, ILUT requires more matvec than ILU0. When ρ is equal to 50 or 100, ILUT requires less matvec than ILU0 but the differences are marginal. With the proper choice of ρ and τ, ILUT performs better than ILU0 but this choice of ρ and τ is not known in advance but through a trial and error process as this study has done. Hence, even from the matvec count aspect, ILUT is not a better preconditioner than ILU0.

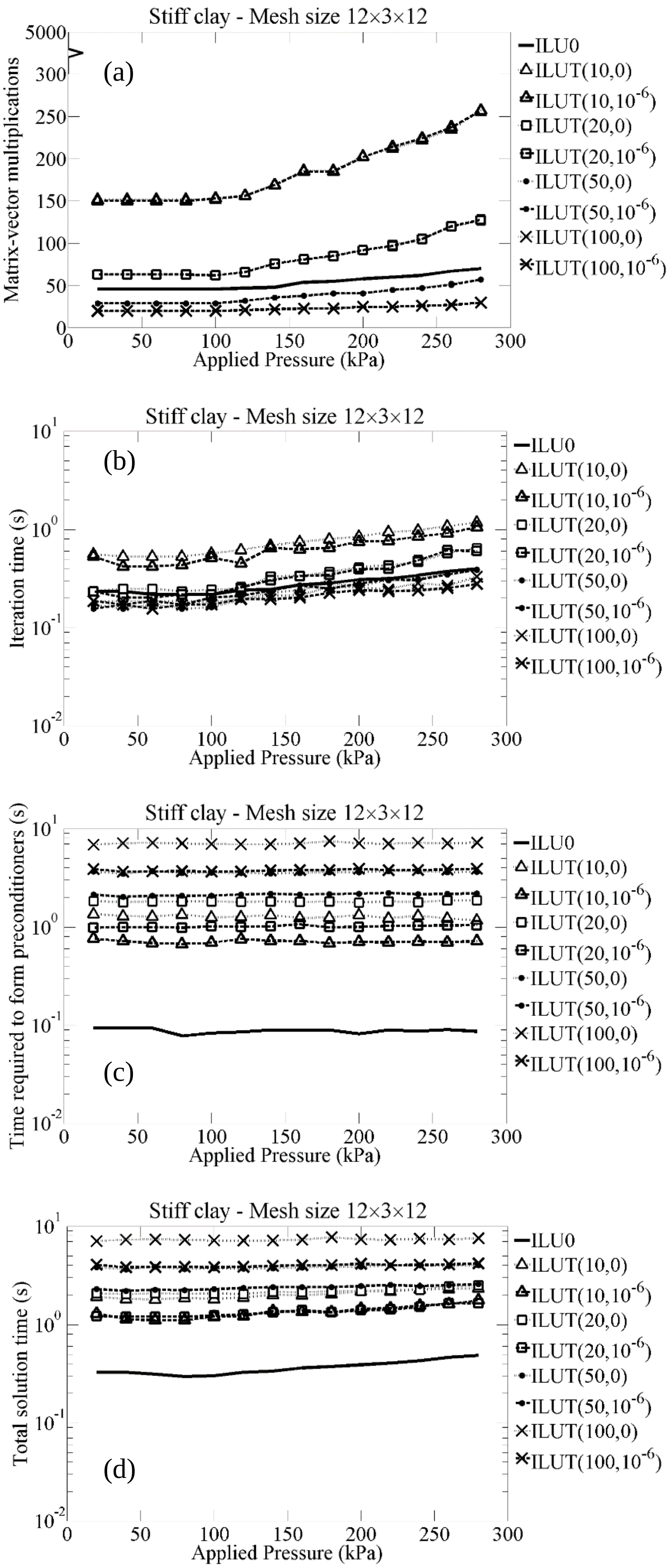


Figure 3.12: Comparison of ILU0 and ILUT(ρ, τ). Soil profile 1 is used with problem size of 12×3×12.

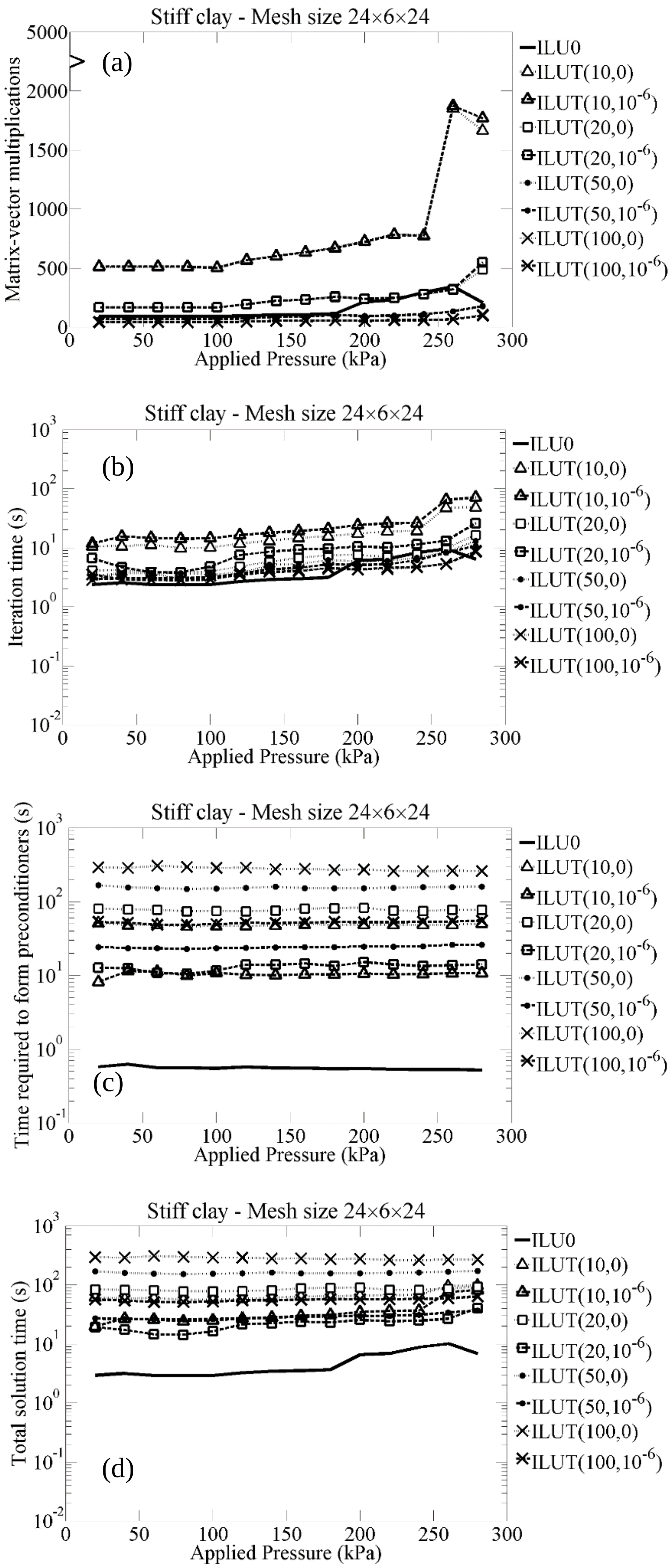


Figure 3.13: Comparison of ILU0 and ILUT(ρ, τ). Soil profile 1 is used with problem size of 24×6×24

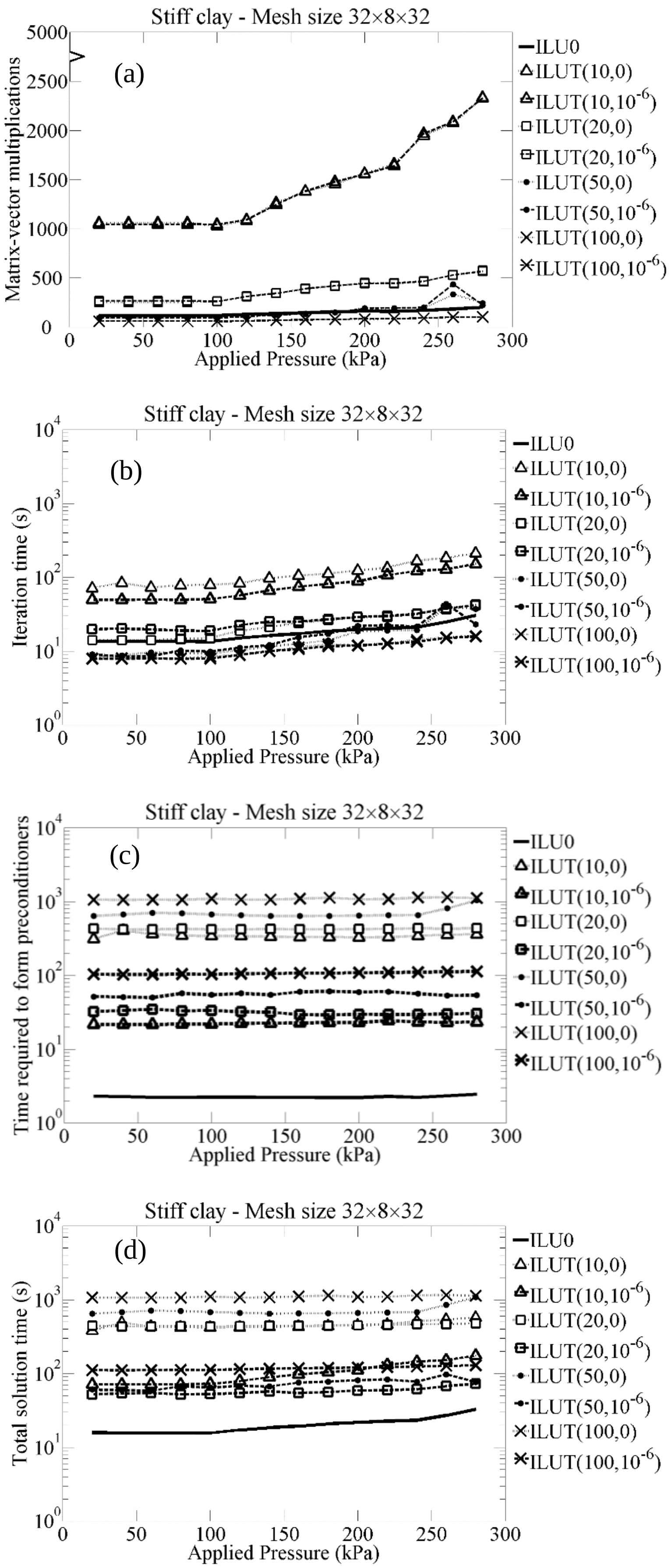


Figure 3.14: Comparison of ILU0 and ILUT(ρ, τ). Soil profile 1 is used with problem size of 32×8×32

Moreover, in the practical aspect – total iteration time, in all the tested cases, ILUT is not more efficient than ILU0. The time to form ILUT is at least 10 times more than it is to form ILU0 but the iteration time does not reduce accordingly to pay off for this amount of time because one matvec with ILUT also takes more time than with ILU0. Hence the total iteration time of IDR(6) with ILUT is always at least 10 times more than with ILU0. Although ILUT is recommended by many researchers, this preconditioner is not a good choice for this study. Hence, ILU0 is still chosen as the default preconditioner of K_{ep} in later parts.

3.6 Effect of convergence criteria and iteration tolerance

3.6.1 Effect of the variation of iteration tolerance, *i_tol*

This section discusses the effect of the tolerance of IDR(6), *i_tol*, on the accuracy of the FE analysis of the strip footing. Figure 3.15 and Figure 3.16 plot the vertical displacement of the center of the strip footing resting on soil profile 1 and 2 respectively when the applied load increases up to the predicted failure loads in Table 3.2.

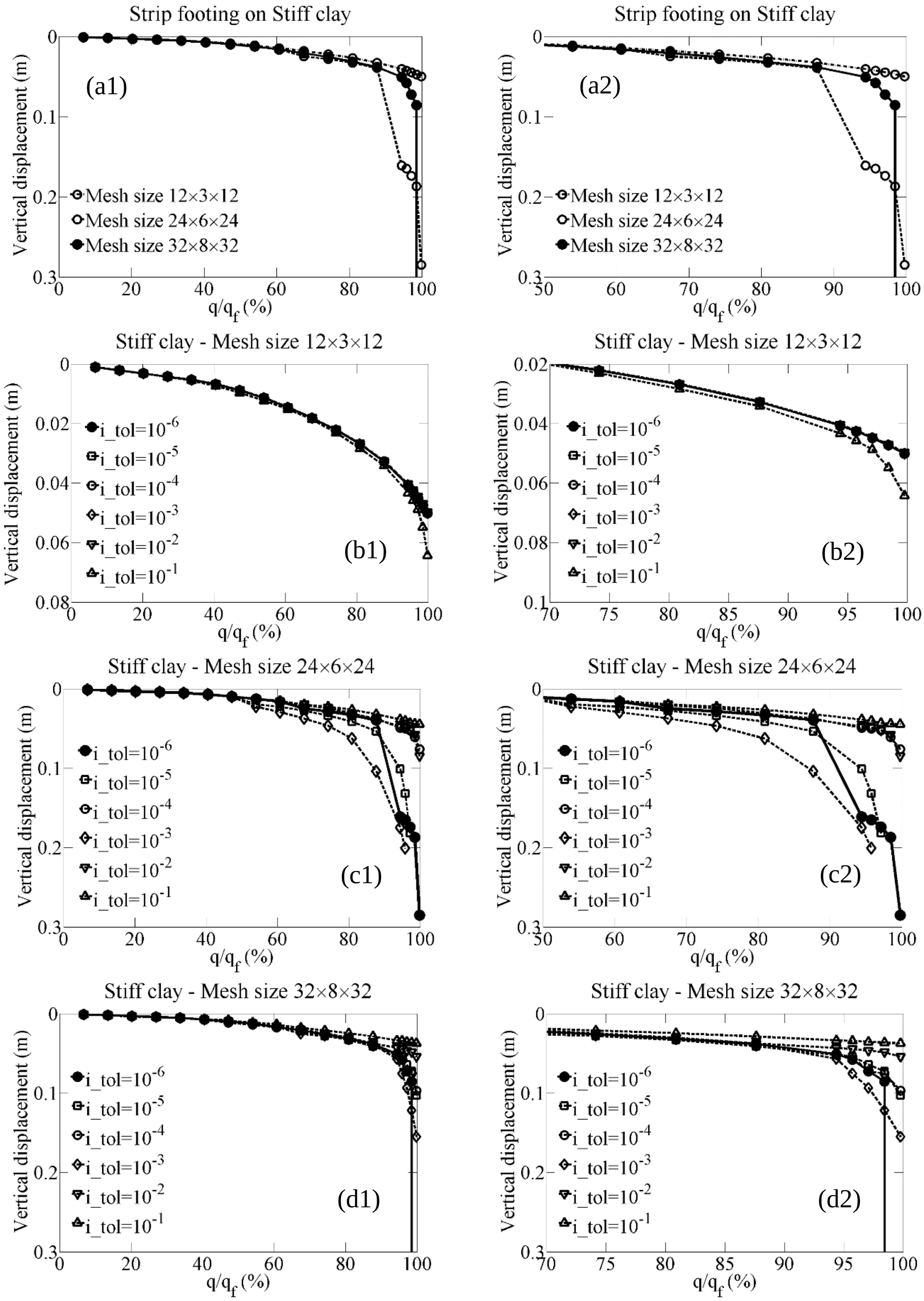


Figure 3.15: Comparison of different *i_tol*. Soil profile 1 is used.

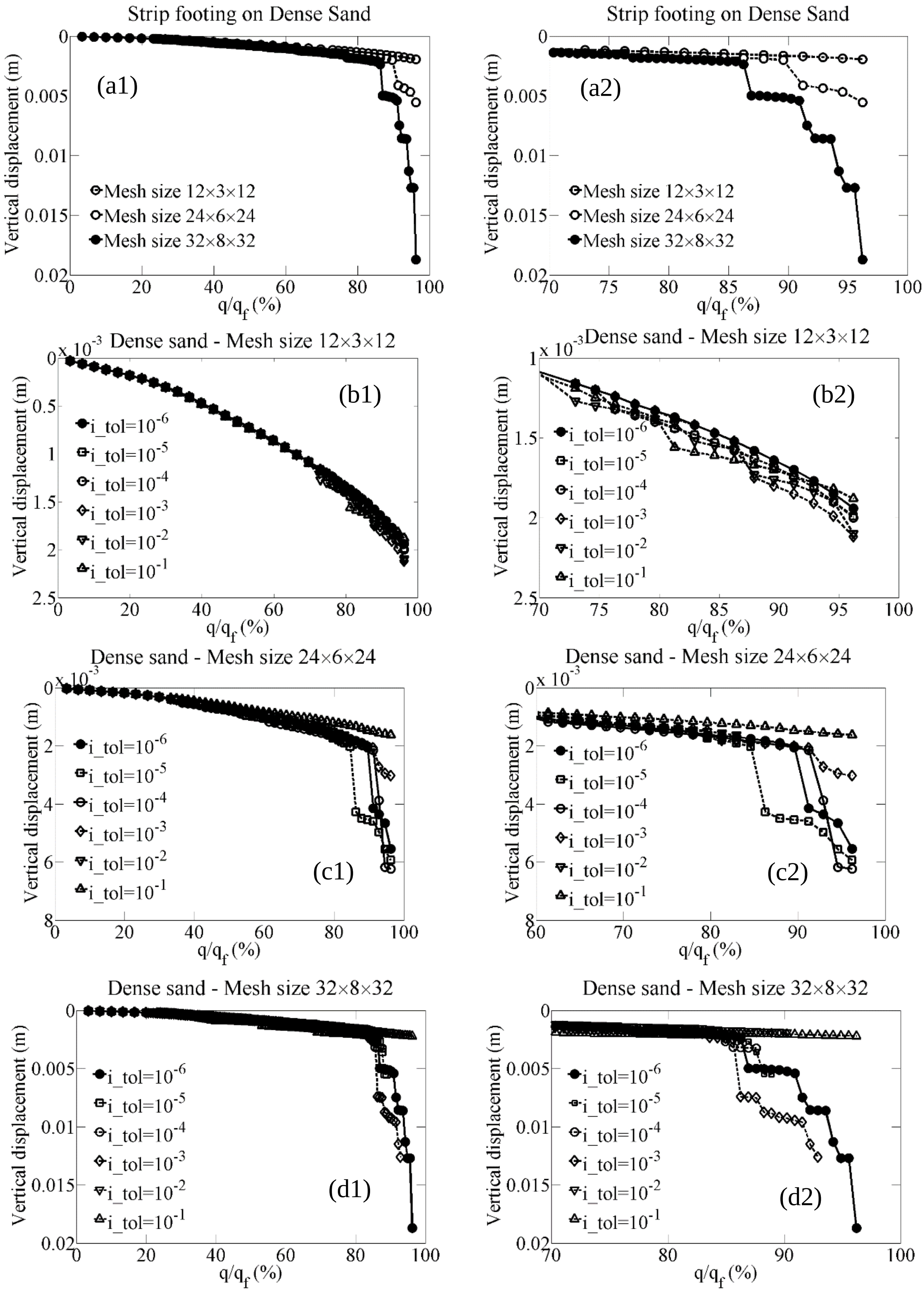


Figure 3.16: Comparison of different *i_tol*. Soil profile 2 is used.

The nonlinear FE analysis of the strip footing resting on the layer of homogeneous soil includes three numerical approximations with reducing level of influence on the accuracy of the analysis: FE approximation of the partial differential equations, NR iteration to solve the nonlinear system resulted from FE approximation, and Krylov iteration – IDR(6) with ILU0 preconditioner – to solve the linear system resulted from each NR iteration. The influence of FE approximation is shown in Figure 3.16(a1) and Figure 3.15(a1) which show the convergence of FE analysis when the mesh is denser. The tolerance of NR is 10^{-6}. *i_tol* is 10^{-6} which is the strictest tolerance tested. When the systems are far from failure (q/q_f < 50%), the vertical displacement resulted from the coarse mesh (12×3×12) is as good as from the densest mesh (32×8×32). However when the applied load is closed to the predicted failure load, denser meshes (24×6×24 and 32×8×32) show the failure phenomenon: there is large increase in vertical displacement and the load-displacement curve becomes steeper. This shows that the influence of FE approximation is the strongest among the three. The tolerance of NR iteration certainly affects the accuracy of the simulation. This thesis discusses the application of Krylov iteration methods and preconditioner hence the tolerance of NR is kept constant in all the analyses. The effect of NR tolerance is ignored because with the tolerance of 10^{-6} and the stopping criteria shown in Eq.(3.5), the FE analysis produces acceptable results when compared with the predicted theoretical failure as shown in Figure 3.16(a1) and Figure 3.15(a1).

It is worth to note that the load-displacement curves in Figure 3.16(a1) and Figure 3.15(a1) are not smooth because there are abrupt changes in vertical displacement, which may not be ideal. This may lie on the return mapping method (refer to Appendix A.2). The numerical model is not stable when the systems are close to failure hence more Gauss points are forced to lie on the apex of MC envelope (as mentioned in Section 3.3), which is spurious. The load step was reduced as a remedy but much smaller load step implies that much more simulation time will be required to produce a perfectly smooth load-displacement curve. Moreover, the current non-ideal load-displacement curve is not physically wrong hence this thesis continues using its current loading scheme for further discussion on the main objective: preconditioners and tolerance of IDR(*s*) method.

The rest of figures in Figure 3.16 and Figure 3.15 show the effects of *i_tol* on the vertical displacement of the strip footing. When q/q_f is less than 60%, the system is far from failure, the vertical displacement can be predicted well with *i_tol* as large as 10^{-1}. When the applied load is close to the predicted failure load, *i_tol* affects greatly on the failure prediction of FE analysis when the failure phenomenon is shown through the analysis. The coarse mesh (12×3×12) fails to predict the failure hence *i_tol* does not change the trend except making the vertical displacement fluctuate in a small range, while it affects the denser meshes (24×6×24 and 32×8×32). Hence for coarse mesh, *i_tol* = 10^{-6} can be consider 'stringent' because it does not offer better solution than *i_tol* = 10^{-1}.

For denser meshes, it should be highlighted first that the failure phenomenon is only shown with *i_tol* = 10^{-6}. When *i_tol* is increased to 10^{-5} and 10^{-4}, there is large increase in the vertical displacement but the failure phenomenon is not clearly shown, except in Figure 3.16(c2). When *i_tol* is as large as 10^{-1}, no sign of failure is shown at all and the load-displacement curve of the densest mesh (32×8×32) is similar to that of the coarse mesh (12×3×12) and the investment on the denser mesh with the purpose of getting better prediction does not pay off. Hence *i_tol* = 10^{-6} is not too 'stringent' for dense mesh and will be continued using throughout this thesis to predict as accurate physical phenomena as possible. In practice, if one finds *i_tol* = 10^{-6} to be too stringent, *i_tol* = 10^{-5} and 10^{-4} can be recommended with the condition that denser mesh should be used, and the value of *i_tol* less than 10^{-4} is not recommended. But it is worth to highlight that in complicated and realistic geotechnical problem, failure loads are not priorly known so too relaxed tolerance can lead to wrong impression about the physical behavior of the problems.

3.6.2 More discussion on the interaction of *i_tol, NR_tol* and load increment

This section only discusses the interaction of *i_tol, NR_tol* and load increment so that accurate results can be produced. Their interaction so that the optimal

NR iteration, Krylov iterations and total solution time can be achieved has not yet researched in-depth and is left for further study.

Table 3.7 to Table 3.9 present the number of NR iterations and average Krylov matvecs at each load increment. The reason that large *i_tol* cannot provide reasonable results is the unbalanced force due to the material non-linearity cannot be fully captured. This is shown by the reduction of the number of yielded Gauss points in the mesh caused by the applied load when *i_tol* increases. When there are few yielded Gauss points in the mesh, the mesh is considered “far from failure” although the load is close to the theoretical failure load.

Table 3.7: 12×3×12 mesh – Summary of NR iteration, average Krylov iteration and yielded Gauss point. Soil profile 1 is used.

Number	Load	i_tol = 10^{-6}		i_tol = 10^{-5}		i_tol = 10^{-4}		i_tol = 10^{-3}		i_tol = 10^{-2}		i_tol = 10^{-1}	
1	20	1 / 46	0	1 / 42	0	1 / 41	0	1 / 36	0	1 / 27	0	1 / 21	0
2	40	1 / 46	0	1 / 42	0	1 / 41	0	1 / 36	0	1 / 27	0	1 / 21	0
3	60	1 / 46	0	1 / 42	0	1 / 41	0	1 / 36	0	1 / 27	0	1 / 21	0
4	80	1 / 46	0	1 / 42	0	1 / 41	0	1 / 36	0	1 / 27	0	1 / 21	0
5	100	3 / 46	384	4 / 41	389	4 / 36	389	4 / 29	389	4 / 18	383	4 / 13	782
6	120	4 / 47	1175	5 / 43	1183	5 / 38	1183	5 / 30	1183	5 / 24	1165	5 / 15	1441
7	140	4 / 48	1708	4 / 46	1708	4 / 41	1708	4 / 36	1708	4 / 26	1694	4 / 17	1895
8	160	4 / 54	2117	4 / 47	2115	4 / 44	2117	4 / 38	2117	5 / 33	2106	4 / 19	2246
9	180	4 / 55	2507	4 / 51	2430	4 / 46	2507	5 / 41	2509	4 / 30	2493	4 / 18	2556
10	200	4 / 58	2835	4 / 52	2835	4 / 46	2835	4 / 38	2835	4 / 26	2797	5 / 14	2896
11	220	4 / 60	3301	4 / 54	3301	4 / 49	3301	5 / 40	3308	6 / 32	3303	5 / 17	3562
12	240	5 / 62	3897	5 / 57	3897	5 / 51	3897	5 / 45	3897	6 / 35	3886	6 / 18	4201
13	260	5 / 67	4543	5 / 60	4543	5 / 53	4543	5 / 45	4543	9 / 35	4517	6 / 21	4751
14	280	7 / 70	5172	8 / 61	5176	8 / 57	5176	11 / 44	5192	10 / 32	5174	7 / 20	5435

Table 3.8: 24×6×24 mesh – Summary of NR iteration, average Krylov iteration and yielded Gauss point. Soil profile 1 is used.

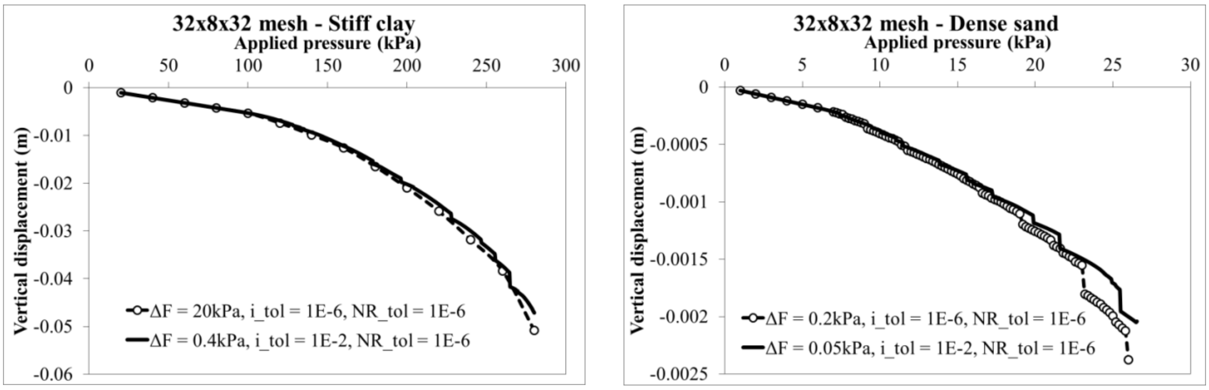
Number	Load	i_tol = 10^{-6}		i_tol = 10^{-5}		i_tol = 10^{-4}		i_tol = 10^{-3}		i_tol = 10^{-2}		i_tol = 10^{-1}	
1	20	1 / 92	0	1 / 85	0	1 / 75	0	1 / 66	0	1 / 54	0	1 / 33	0
2	40	1 / 92	0	1 / 85	0	1 / 75	0	1 / 66	0	1 / 54	0	1 / 33	0
3	60	1 / 92	0	1 / 85	0	1 / 75	0	1 / 66	0	1 / 54	0	1 / 33	0
4	80	1 / 92	0	1 / 85	0	1 / 75	0	1 / 66	0	1 / 54	0	1 / 33	0
5	100	4 / 92	3132	4 / 89	3132	4 / 74	3132	4 / 52	3114	4 / 35	3313	5 / 20	5487
6	120	7 / 97	9591	7 / 97	9591	6 / 81	9552	7 / 65	9589	7 / 50	9559	6 / 30	10333
7	140	14 / 106	13647	14 / 101	13647	13 / 86	13618	12 / 70	13972	10 / 50	13567	14 / 24	13610
8	160	18 /106	17047	20 / 110	19345	9 / 87	16725	20 / 71	20175	12 / 45	16996	9 / 34	17190
9	180	17 / 114	19747	10 / 112	22270	5 / 91	18831	20 / 81	23044	10 / 49	18923	7 / 34	18722
10	200	17 / 216	25317	10 / 114	25028	17 / 97	23108	18 / 82	25503	7 / 62	21390	8 / 30	22622
11	220	10 / 231	27284	12 / 108	28240	7 /94	25928	16 / 87	30012	7 / 64	24860	8 / 33	24621
12	240	7 / 296	31285	12 / 139	33630	10 / 101	31618	10 / 89	38142	16 / 50	30998	11 / 27	28521
13	260	9 / 344	37011	16 / 155	41256	12 / 95	35739	12 / 81	43382	17 / 67	35469	10 / 39	34062
14	280	25 / 207	59799	20 / 170	48707	13 / 109	43397	12 / 101	48450	10 / 65	41900	12 / 39	38490

Table 3.9: $32\times8\times32$ mesh – Summary of NR iteration, average Krylov iteration and yielded Gauss point. Soil profile 1 is used.

Number	Load	i_tol = 10^{-6}		i_tol = 10^{-5}		i_tol = 10^{-4}		i_tol = 10^{-3}		i_tol = 10^{-2}		i_tol = 10^{-1}	
1	20	1 / 120	0	1 / 113	0	1 / 100	0	1 / 84	0	1 / 70	0	1 / 50	0
2	40	1 / 120	0	1 / 113	0	1 / 100	0	1 / 84	0	1 / 70	0	1 / 50	0
3	60	1 / 120	0	1 / 113	0	1 / 100	0	1 / 84	0	1 / 70	0	1 / 50	0
4	80	1 / 120	0	1 / 113	0	1 / 100	0	1 / 84	0	1 / 70	0	1 / 50	0
5	100	4 / 119	7290	4 / 107	7290	4 / 90	7302	4 / 65	7242	4 / 45	7059	1 / 50	0
6	120	14 / 131	25002	11 / 122	24286	11 / 113	32161	11 / 84	24373	14 / 53	24928	8 / 33	17454
7	140	13 / 139	31926	10 / 128	30794	10 / 117	40900	10 / 91	30825	10 / 68	30353	17 / 32	28969
8	160	9 / 147	38515	9 / 142	37525	10 / 136	43503	8 / 104	37538	10 / 76	36788	14 / 25	36821
9	180	12 / 157	44282	11 / 136	45709	12 / 133	47791	14 / 104	44250	10 / 59	43316	20 / 15	42899
10	200	10 / 163	52127	11 / 141	52972	12 / 135	53976	14 / 85	65402	13 / 71	50268	11 / 36	50716
11	220	13 / 161	60740	18 / 135	67502	19 / 127	64913	9 / 117	69722	10 / 90	58523	8 / 41	55168
12	240	13 / 171	71646	12 / 149	73139	13 / 140	74581	14 / 123	76283	14 / 83	70055	12 / 17	64473
13	260	19 / 184	84364	20 / 152	86241	19 / 149	88913	21 / 129	92414	21 / 49	86427	11 / 38	68535
14	280	29 / 196	106109	15 / 180	104734	20 / 166	103900	22 / 169	109382	16 / 84	92460	11 / 26	75823

The results and discussion in Section 3.6.1 are based on the condition that *NR_tol* and load increment are constant when *i_tol* varies. When the unbalanced force is not fully captured, the analysis is closed to the explicit increment method which requires sufficiently small load increment to obtain accurate results (Abbo[2], 2007; Krenk[96], 2009 pp7-14). Hence, the cause of unreasonable results in this case is not because *i_tol* is larger than *NR_tol* (so less NR iterations are performed) but because the load increment is too large. The same unreasonable results would occur when *NR_tol* is too large and *i_tol* is small. Figure 3.17 below shows that when *i_tol* = 10^{-2}, *NR_tol* = 10^{-6} and the load increment is very small, the results are as good as when *i_tol* = 10^{-6}, *NR_tol* = 10^{-6} and the load increment is large.

Besides, the FE algorithm used in this thesis belongs to the class of Newton-Krylov method, also known as inexact Newton method. Blaheta[26] (1997), Axelsson and others[14] (1997) showed theoretically that the inexact Newton method used in elastoplastic problems converges to the right solution when the load increment is sufficiently small relative to the state of the structure at the computing point and the convergence rate depends on the choice of *i_tol*, e.g. constant value or series of reducing values.



(a) (b)

Figure 3.17: Interaction of *i_tol*, *NR_tol* and load increment

3.7 Eigenvalue distribution of nonsymmetric linear systems

This section shows qualitatively the theoretical reason for the increase of matvec when applied pressure increase as observed in Section 3.4. Section 2.4.1 has discussed the effects of spectral properties on the convergence of

Krylov iterative methods. Qualitatively, the iterative solvers require less matvec to converge if the ellipse (under suitable normalization) in the complex plane circumscribing all the eigenvalues is small. Figure 3.18 shows the eigenvalue distributions of the elastic stiffness matrix K_e and the un-preconditioned and preconditioned global stiffness matrix K_{ep}.

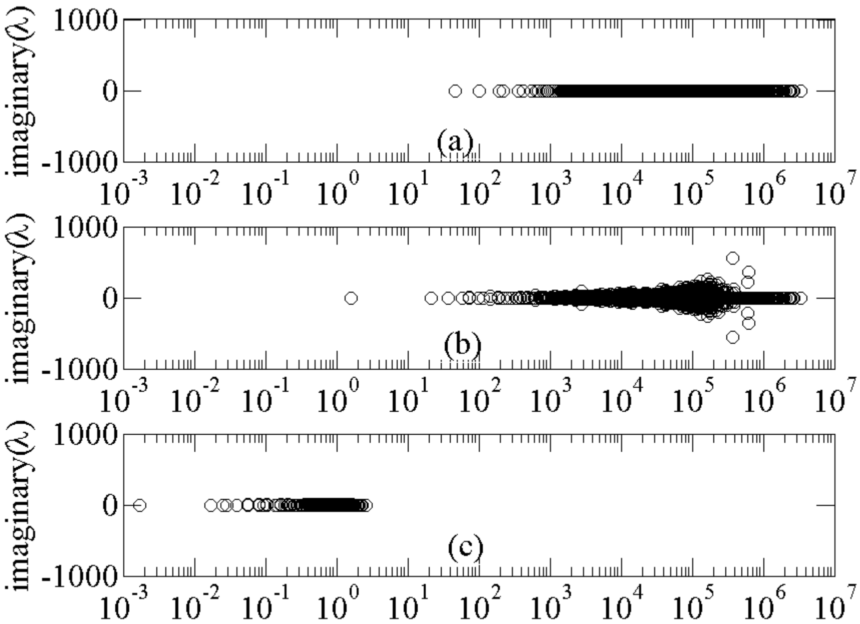
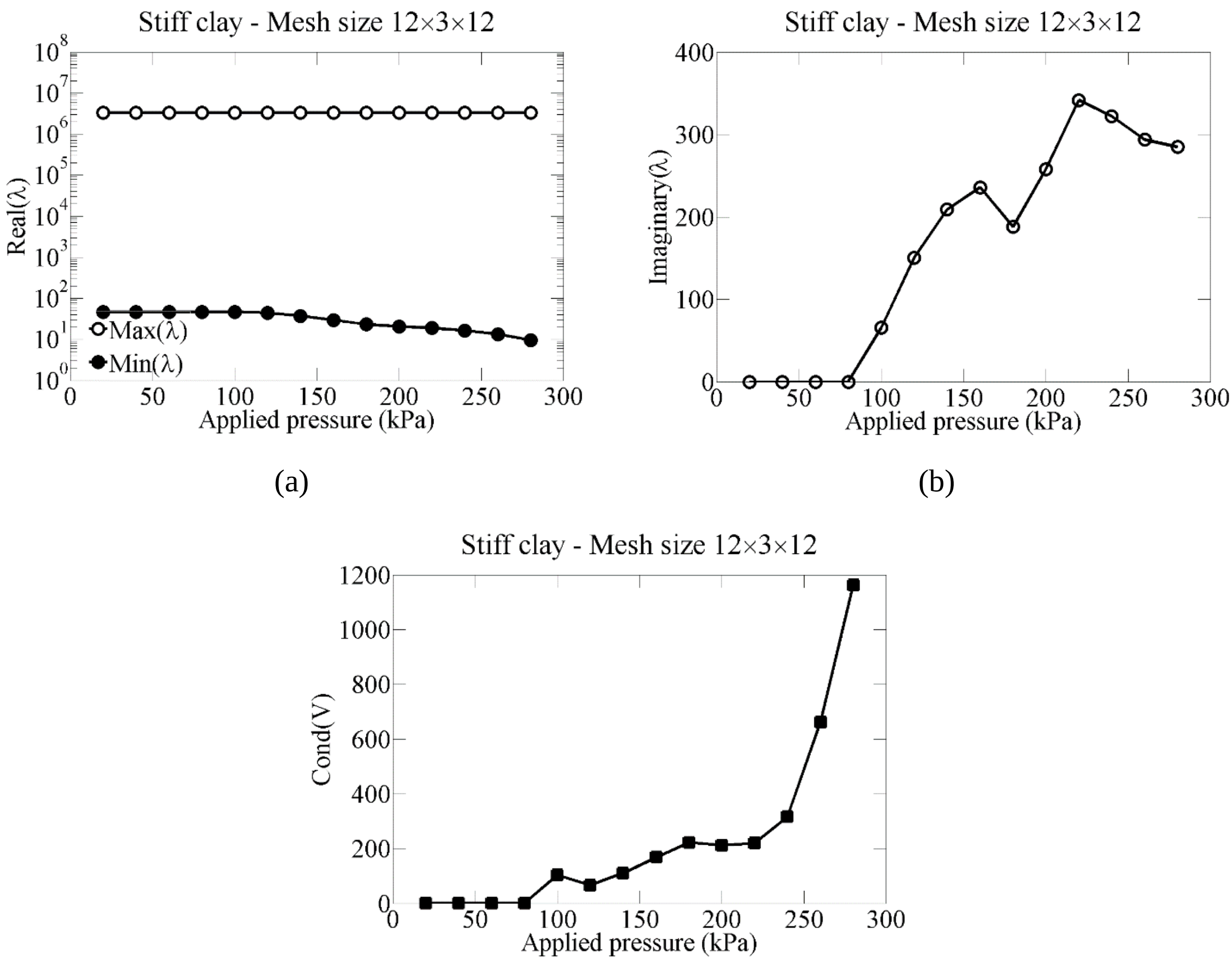


Figure 3.18: Eigenspectra of matrix (a) K_e; (b) Unpreconditioned K_{ep}; (c) K_{ep} preconditioned with ILU0. Problem size 12×3×12 with soil profile 1

When there are no yielded Gauss points in the mesh, the global stiffness matrix is K_e, which is symmetric and all the eigenvalues are positive and real numbers as shown in Figure 3.18a. When there are yielded Gauss points, K_{ep} applies and the global stiffness matrix becomes nonsymmetric hence the some of the eigenvalues are complex number (Figure 3.18b). Figure 3.19a and Figure 3.19b show that when the applied load increases, the maximum and minimum real parts of eigenvalues are almost unchanged while the maximum imaginary parts of these complex eigenvalues increase, which enlarge the circumscribing ellipse. Section 2.4.1 also mentioned that the condition number of the matrix X in Eq.(2.30) is involved in the convergence of Krylov iterative methods. Figure 3.19c shows that this condition number increases when the applied load increases. All these observations imply that the nonsymmetric linear system is harder to solve when the applied load increases and explains the reason of the increase in matvec of IDR(s) in Section 3.4. ILU(0) is the most efficient preconditioner shown in Section 3.4. This efficiency is reflected in the clustering of the eigenspectrum. Figure 3.18c shows the eigenvalues distributions of K_{ep} preconditioned by ILU(0). The eigenvalues distribute over a much smaller range than those from the un-preconditioned K_{ep}, which explains the efficiency gained when solving the preconditioned linear system iteratively.



(a) (b)

(c)

Figure 3.19: Characteristics of eigenspectrum: (a) Maximum and minimum eigenvalue; (b) Maximum imaginary part of eigenvalues; (c) Condition number of matrix X (Eq.(2.30)). Problem size 12×3×12 with soil profile 1 is used.

3.8 Summary

This chapter performs numerical experiments on the strip footing problem resting on three soil profiles, of which properties have typical values of soil as well as have wide range of input values. The key findings from the numerical results are:

1. IDR(1) is theoretically equivalent to Bi-CGSTAB, which has been validated numerically. Bi-CGSTAB or IDR(1) with $P = r_0$ is a good choice and can sometimes perform better than IDR(s > 1) with P as a random matrix. However, with good preconditioner like ILU0, a random matrix P is not a bad choice because IDR(s > 1) performs better than Bi-CGSTAB.

2. When s is larger, IDR requires less matvec as expected but more time is spent on solving the $s \times s$ linear system and more memory is used to store $P^{N \times s}$. IDR(6) and IDR(10) show competitive performance in iteration time. IDR(6) is chosen for further implementation because this will limit the memory required to store $P^{N \times s}$ especially when N grows large in practical problems.

3. ILU0 is the most efficient preconditioner for the K_{ep} matrix among Jacobi, SSOR-LR, SSOR-L and ILUT(ρ, τ). ILUT(ρ, τ) can be competitive in term of matvec but less competitive in term of solution time. Moreover, the fill-in number is not known priorly and also problem dependent.

4. Eigenvalue distribution shows that when there are more yielded Gauss points in the mesh, K_{ep} is harder to solve because there are more complex eigenvalue making the ellipse circumscribing them bigger and the eigenspectrum characteristic more complicated.

CHAPTER 4 PRECONDITIONERS FOR 1-BY-1 BLOCK MATRICES: DRAINED/UNDRAINED ANALYSIS

4.1 Introduction

This chapter aims to discuss two issues on preconditioning the 1-by-1 block matrix: Section 4.2 discusses techniques to precondition effectively a sequence of linear systems which occurs in the nonlinear FE analysis; and Section 4.3 discusses effects of the penalty method, which will be elaborated in the section, on IDR(*s*) and preconditioners. Chapter 3 has concluded that IDR(6) preconditioned with ILU0 requires the least time to solve the nonsymmetric linear system due to the non-associated MC model, hence this chapter continues to use IDR(6) and ILU0. Geotechnical problems considered in the numerical experiments are: flexible strip footing (Figure 3.1a) and square footing (Figure 4.1a) resting on homogenous soil layer, and vertical smooth wall (Figure 4.1b) subjected to horizontal prescribed displacements. Theoretical results are available for these problems and are used as the reference for numerical predictions.

4.2 Efficient preconditioning for a sequence of linear systems in drained analysis

This section considers a flexible strip footing (Figure 3.1a) and a flexible square footing (Figure 4.1a) resting on the homogenous soil profile 1 and 2 which have been described in Section 3.2. The boundary conditions of the square footing are similar to those of the strip footing described in Section 3.2. The characteristics of the 3D meshes of the square footing are presented in Table 4.1. The properties of the soil following the non-associated MC model are given in Table 3.1.

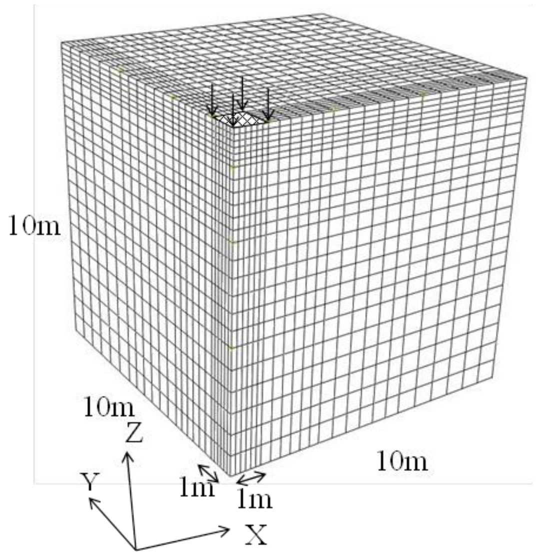


Figure 4.1: 3D finite element mesh of the square footing

Table 4.1: 3D FE meshes of the square footing resting on soil profile 1 and 2

	Mesh size	
	16×16×16	24×24×24
Number of elements	4,096	13,824
Number of nodes	18,785	60,625
Number of unknowns (N)	50,656	169,296
Number of Gauss points (N_{ip})	110,592	373,248
Number of nonzero (*nnz*)		
Soil profile 1		
Elastic system	7,809,113	27,355,315
Elastoplastic system at 380kPa	7,831,757	27,373,538
Soil profile 2		
Elastic system	7,814,366	27,360,464
Elastoplastic system at 47kPa	7,847,086	27,391,751
nnz/N^2 (%)		
Soil profile 1		
Elastic system	0.3	0.095
Elastoplastic system at 380kPa	0.31	0.096
Soil profile 2		
Elastic system	0.30	0.095
Elastoplastic system at 47kPa	0.31	0.096

The prediction of failure load of the strip footing has been presented in Section 3.2. When the foundation is a rectangle, the shape factors are multiplied to the Terzaghi's formula in Eq.(3.1) to get Eq.(4.1). The shape factor ξ_{cs} for the rectangular footing is given in Eq.(4.2).

$$q_f = \xi_{\gamma s}\left(\frac{1}{2}\gamma B N_\gamma\right) + \xi_{cs}\left(cN_c\right) + \xi_{qs}\left(\gamma D N_q\right) \tag{4.1}$$

$$\xi_{cs} = 1 + \left(B/L\right)\left(N_q / N_c\right) \tag{4.2}$$

The predicted failure load of the square footing resting on the soil profile 1 and 2 and the maximum applied load for each soil profile are tabulated in Table 4.2. Similar to the point noted in Section 3.2, the predicted failure load in Eq.(4.1) does not take into account the non-associated flow rule, which is considered in this whole thesis, hence the actual failure load may be 10 or 20 percent different from the predicted values.

Table 4.2: Ultimate bearing capacity of the strip footing and square footing on the homogenous soil layer and the maximum applied pressure used in numerical experiments

	N_q	N_c	ξ_{cs} with B/L = 1	$q_{f\text{-}square} = \xi_{cs} c' N_c$ (kPa)	$q_{max\text{-}square}$ (kPa)
Soil profile 1	6.4	14.83	1.43	424.68	380 (= 90%$q_{f\text{-}square}$)
Soil profile 2	18.40	30.14	1.61	48.54	47 (= 97%$q_{f\text{-}square}$)

4.2.1 By forming the global stiffness matrix implicitly

From the elastoplastic stress-strain matrix in Eq.(1.3), the global stiffness matrix in drained analysis in Eq.(2.1) can be written as summation of two matrices K_e and Δ as in Eq.(4.3).

$$K_{ep} = \sum_{\text{element}} \left(\int_V B^T D_e B dV\right) + \sum_{\text{element}} \left[\int_V B^T \left(-\frac{D_e\left(\frac{\partial g}{\partial \sigma}\right)\left(\frac{\partial f}{\partial \sigma}\right)^T D_e}{\left(\frac{\partial f}{\partial \sigma}\right)^T D_e \left(\frac{\partial g}{\partial \sigma}\right)}\right) B dV\right] = K_e + \Delta \tag{4.3}$$

The integral in Eq.(4.3) to assemble Δ is evaluated with Gauss quadrature formula given below

$$\Delta = \sum_{\text{element}} \sum_{i=1}^{n_y} B_i^T \left(-\frac{D_e \left(\frac{\partial g}{\partial \sigma} \right) \left(\frac{\partial f}{\partial \sigma} \right)^T D_e}{\left(\frac{\partial f}{\partial \sigma} \right)^T D_e \left(\frac{\partial g}{\partial \sigma} \right)} \right)_i B_i \|J_i\| W_i \tag{4.4}$$

in which n_y is the number of yielded Gauss points in each element, $\|J_i\|$ is determinant of the Jacobian matrix and W_i is the Gauss quadrature weighting coefficient at each Gauss point.

From Eq. (4.3) and Eq.(4.4), the upper bound on the rank of Δ is evaluated in Eq.(4.5) and Eq.(4.6),

$$\operatorname{rank}(\Delta) \le \sum_{\text{element}} \sum_{i=1}^{n_y} \operatorname{rank}\left[B_i^T \left(-\frac{D_e \left(\frac{\partial g}{\partial \sigma} \right) \left(\frac{\partial f}{\partial \sigma} \right)^T D_e}{\left(\frac{\partial f}{\partial \sigma} \right)^T D_e \left(\frac{\partial g}{\partial \sigma} \right)} \right)_i B_i \|J_i\| W_i \right] \le \sum_{\text{element}} n_y = N_y \tag{4.5}$$

$$\operatorname{rank}(\Delta) \le \min(N, N_y) \tag{4.6}$$

in which N_y is the total number of yielded Gauss points in the mesh and N is the dimension of both K_e and $K_{ep.}$

When geotechnical systems are far from failure, N_y is much smaller than N. Hence, the rank of Δ is much smaller than N. On the contrary, when geotechnical systems approach failure, N_y is larger than N. However, numerical results show that the rank of Δ is still much smaller than N. In this sense, Δ can be considered as a perturbation of K_e. In Section 4.2.2, ILU0-K_{ep} denotes the ILU0 derived from the K_{ep} matrix while ILU0-K_e denotes the ILU0 derived from the K_e matrix.

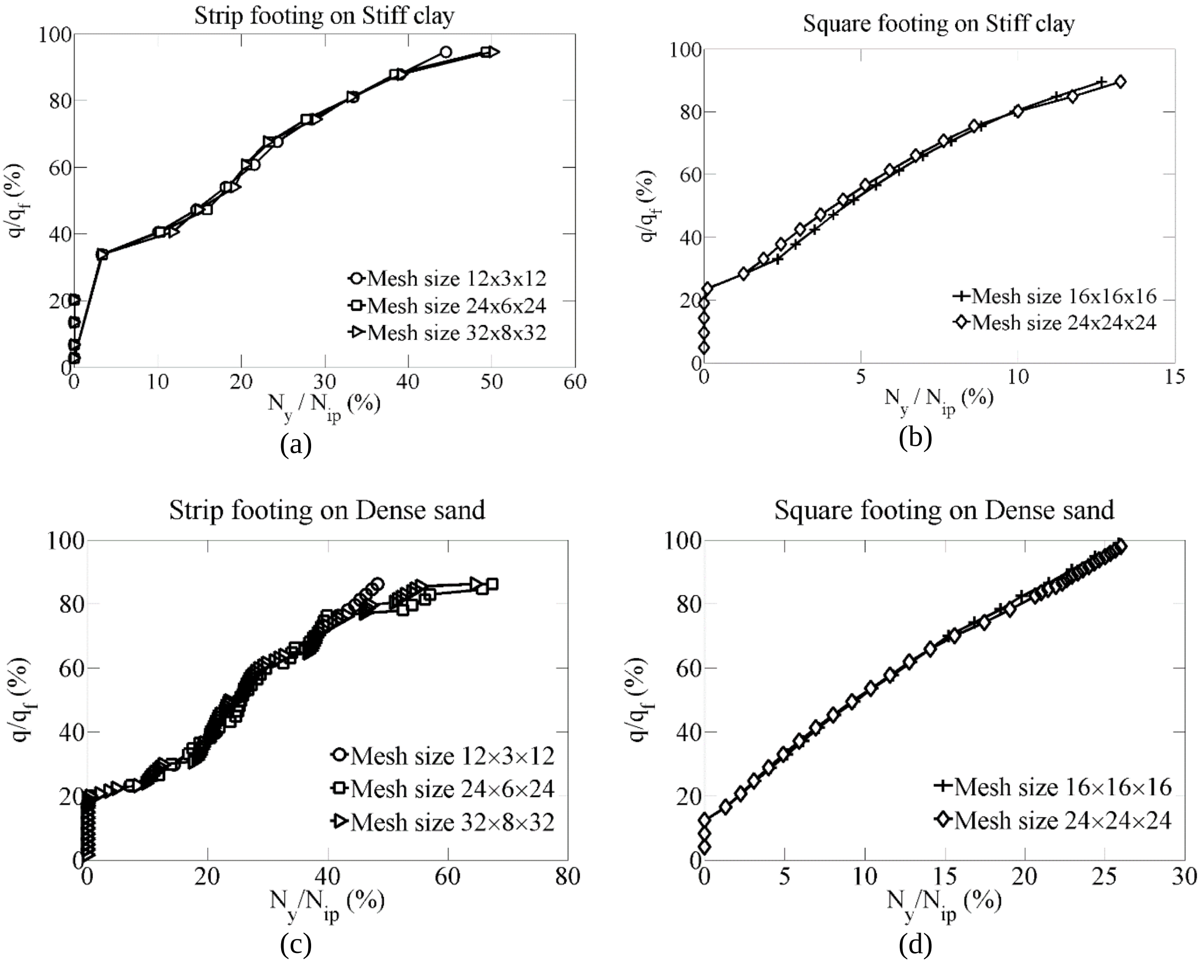


Figure 4.2: Ratio of applied pressure q over the bearing capacity q_f versus percentage of yielded Gauss points in the 3D mesh of: (a)(c) Strip footing, (b)(d)Square footing.

When full NR is used, K_{ep} has to be formed at each NR iteration. This formation can be done explicitly as in Eq.(2.1) or implicitly as in Eq.(4.3). In Eq. (4.3), only the Δ matrix has to be formed at each NR iteration. Figure 4.3 shows the comparison of time to form K_{ep} and time to form Δ. The figures are plotted against the percentage of yielded Gauss points in the mesh (N_y/N_{ip}).

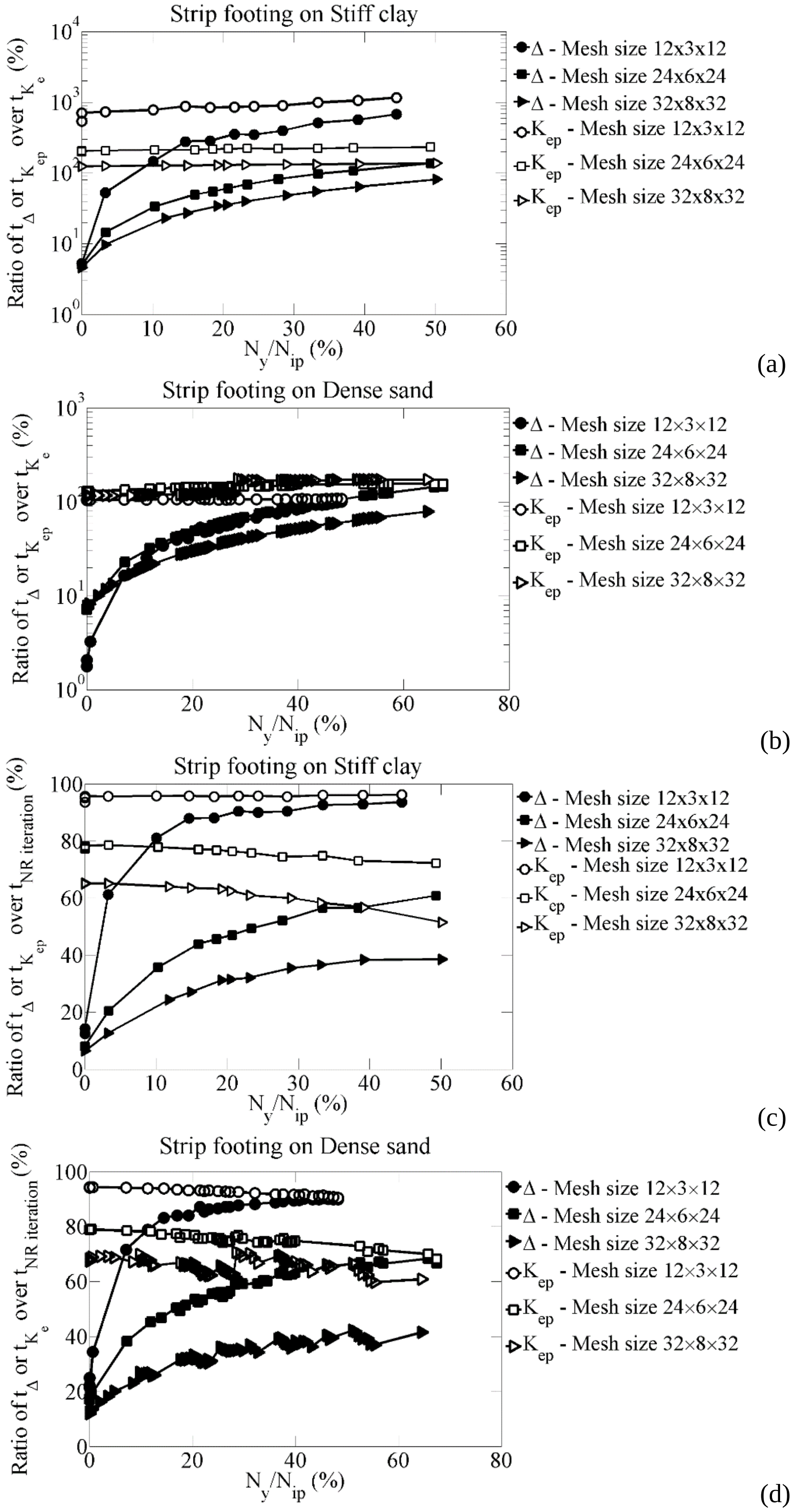


Figure 4.3: (a) (b) Ratio of time to form Δ and K_{ep} over time to form K_e; (c) (d)Ratio of time to form Δ and K_{ep} over total time consumed in each NR iteration when IDR(6) with ILU0 is used to solve the linear systems.

The time to form K_{ep} is always several times more than the time to form K_e because of the stress returning procedure in nonlinear FE analysis (Appendix A.2). In Figure 4.3a and Figure 4.3b, the time to form K_{ep} does not vary significantly regardless of the increases of yielded Gauss points. Hence, in each NR iteration, the same length of time has to be spent to form K_{ep}. On the contrary, the time to form Δ only increases when the number of yielded Gauss points increases. This time is smaller than the time to form K_{ep} even at the final load stage, where 50 (in soil profile 1) to 65 (in soil profile 2) percent of the Gauss points have yielded. In practice, the FE analysis is performed at the working load stage, which is typically less than 50 percent of the bearing capacity. At this stage, 15 percent of the Gauss points have yielded when soil profile 1 is considered (Figure 4.2a) and the time to form Δ is 30 percent of the time to form K_{ep} for the largest problem size tested; while 25 percent of the Gauss points have yielded when soil profile 2 is considered (Figure 4.2c) and the time to form Δ is 50 percent of the time to form K_{ep} for the largest problem size tested.

Section 3.4 has shown that IDR(6) with ILU0-K_{ep} preconditioner requires the least time to solve the non-symmetric linear system in each NR iteration. Figure 4.3c and Figure 4.3d show that when this preconditioner is used, the time to form K_{ep} takes more than 75 percent of total time consumed in each NR iteration for small and medium problem sizes, meaning forming K_{ep} is a much more critical procedure than solving the large-scale linear system and minimizing the linear system solving time is not tackling the more time consuming part of the solution process. Although this ratio reduces to 50 to 65 percent for the largest problem size tested, this figure is still considered significantly large.

Similar to Figure 4.3(a)(b), Figure 4.3(c)(d) show that the ratio of the time to form Δ over the total time consumed in each NR iteration increases with the increase of yielded Gauss points. For small problem size, this ratio is as large as it is for K_{ep} while it is greatly reduced when the problem size increases. For the largest problem size tested, this ratio is up to 40 percent. At the working load, this ratio is 30 percent when soil profile 1 is used and 36 percent when soil profile 2 is used while it is 60 percent for forming K_{ep}. Hence, computing just Δ accrues significant time saving compared with the ratio of 60 percent for the baseline case of forming K_{ep} afresh.

4.2.2 By freezing the preconditioner

4.2.2.1 Use preconditioner from the elastic global stiffness matrix K_e

Since Δ is a low-rank matrix, a preconditioner derived from the dominant component K_e may be as effective as a similar one derived from K_{ep}. Figure 4.4(a)(c) and Figure 4.5(a)(c) show the comparison of ILU0-K_e preconditioner derived from K_e and ILU0-K_{ep} derived from K_{ep} when the strip footing problem is considered. For all the problem sizes of the strip footing, IDR(6) with ILU0-K_e requires more matvec to converge than with ILU0-K_{ep}. This agrees with results of Augarde and others[11] (2007) which discusses element-by-element (EBE) preconditioners. ILU0-K_e is only formed once at the beginning of the solution process, there are differences in the total iteration time to solve the linear system. However, the time saved from forming the preconditioner is dominated by the increasing time to perform more matvec when there are more yielded Gauss points. Moreover, the efficiency of ILU0-K_e reduces when the problem size increases because for large–scale problems, the time to form the preconditioner becomes less significant compared with the time to solve the linear system. However, for all the problem sizes, ILU0-K_e is more time effective than ILU0-K_{ep} when the percentage of yielded Gauss points is less than 15 percent.

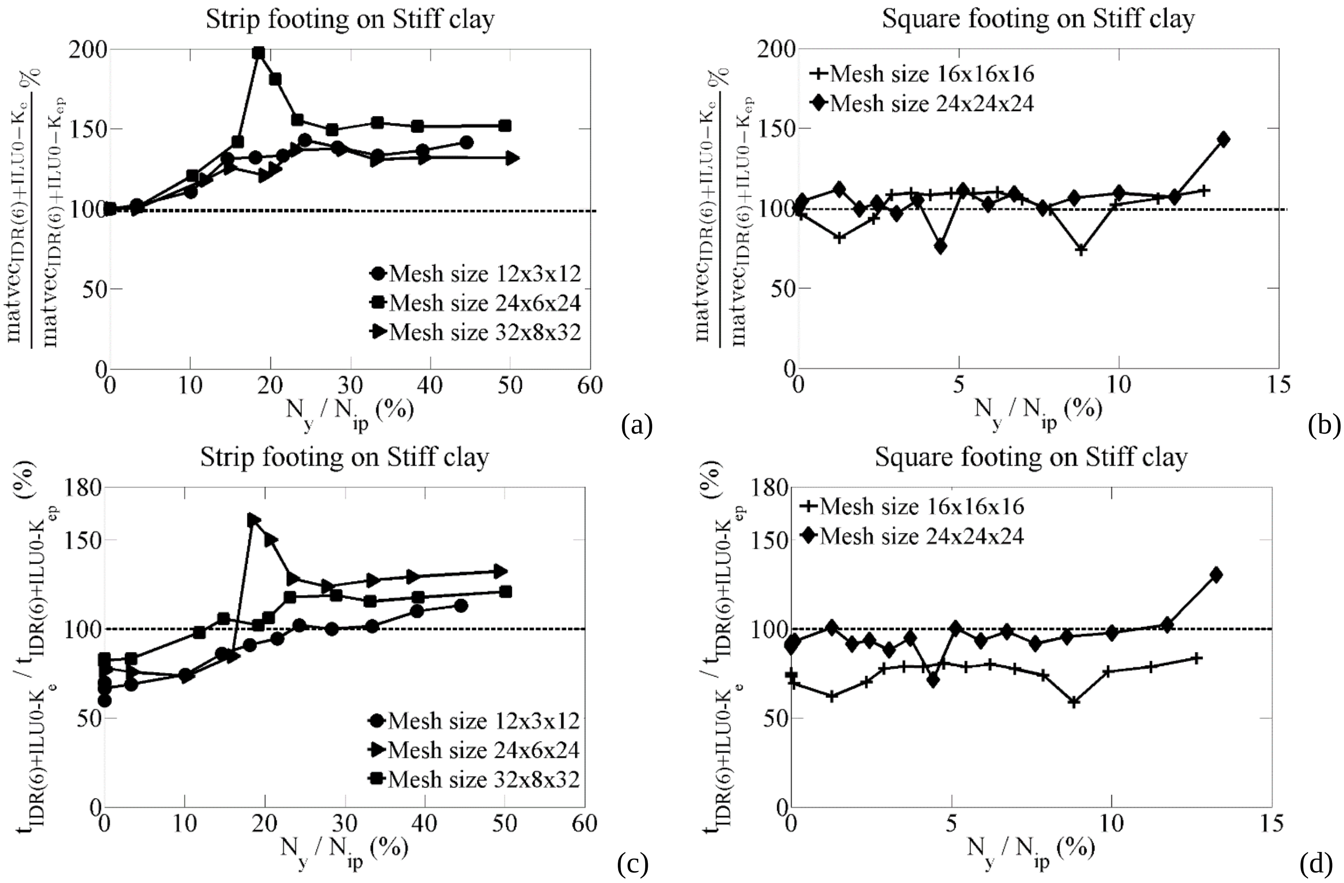


Figure 4.4: Comparison of efficiency of ILU0-K_e and ILU0-K_{ep} . Soil profile 1 is used.

For the current strip footing example, 50 to 60 percent of the Gauss points have yielded at 90 percent of the bearing capacity. A more practical geotechnical problem – square footing – is tested and shows that only 14 to 25 percent of the Gauss points yield at 90 percent of the bearing capacity (Figure 4.2b and Figure 4.2d). Figure 4.4(b)(d) and Figure 4.5(b)(d) compare the matvec and total iteration time of IDR(6) when ILU0-K_e and ILU0-K_{ep} are used. Similar to the case of strip footing, the number of matvec required by ILU0-K_e is more than that required by ILU0-K_{ep}. However, the total iteration time required by ILU0-K_e is less than by ILU0-K_{ep} when the percentage of yielded Gauss point is less than 15 percent. This agrees with the conclusion made for the strip footing problem. Geotechnical problems tend to fail locally so the percentage of yielded Gauss points is not significantly large. When the percentage is less than 15, IDR(6) with ILU0-K_e is effective in reducing the iteration time.

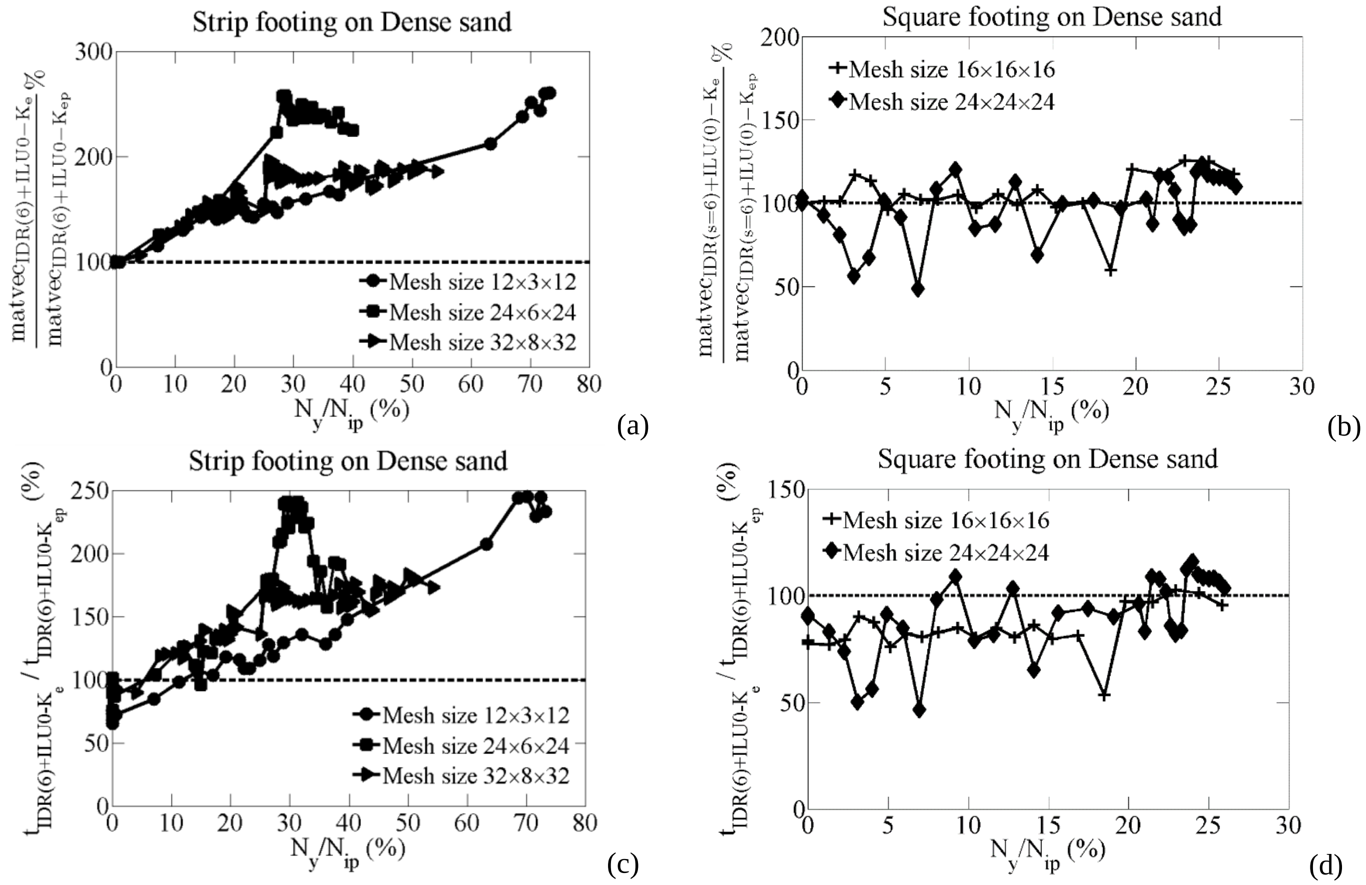


Figure 4.5: Comparison of efficiency of ILU0-K_e and ILU0-K_{ep}. Soil profile 2 is used.

As noted in Section 1.1.3, before the development of IDR(s) method, Bi-CGSTAB is considered the most efficient method and its combination with ILU0-K_{ep} preconditioner yields the shortest iteration time compared with other preconditioners (Section 3.4). Thus, by default, to attain practical length of simulation time, one would use Bi-CGSTAB with ILU0-K_{ep} as well as a newly assembled global stiffness matrix K_{ep} in each NR iteration. This section proposes the use of IDR(6) with ILU-K_e (when the percentage of yield points is less than 15) and assembling Δ at each NR iteration to form K_{ep}. Figure 4.6 plots the ratio of total iteration time used in each NR iteration by the latter method over total iteration time used by the former method. The latter method only requires at most 40 percent of time required by the former. Hence, by using the proposed method, the total simulation time can be reduced by 60 percent. Besides, when the 24×24×24 mesh is used for the soil profile 1, Bi-CGSTAB with ILU0-K_{ep} fail to converge within 5000 matvec at every loading stage. This highlights the superior efficiency of IDR(6) over Bi-CGSTAB in solving the nonsymmetric linear system from non-associated MC model.

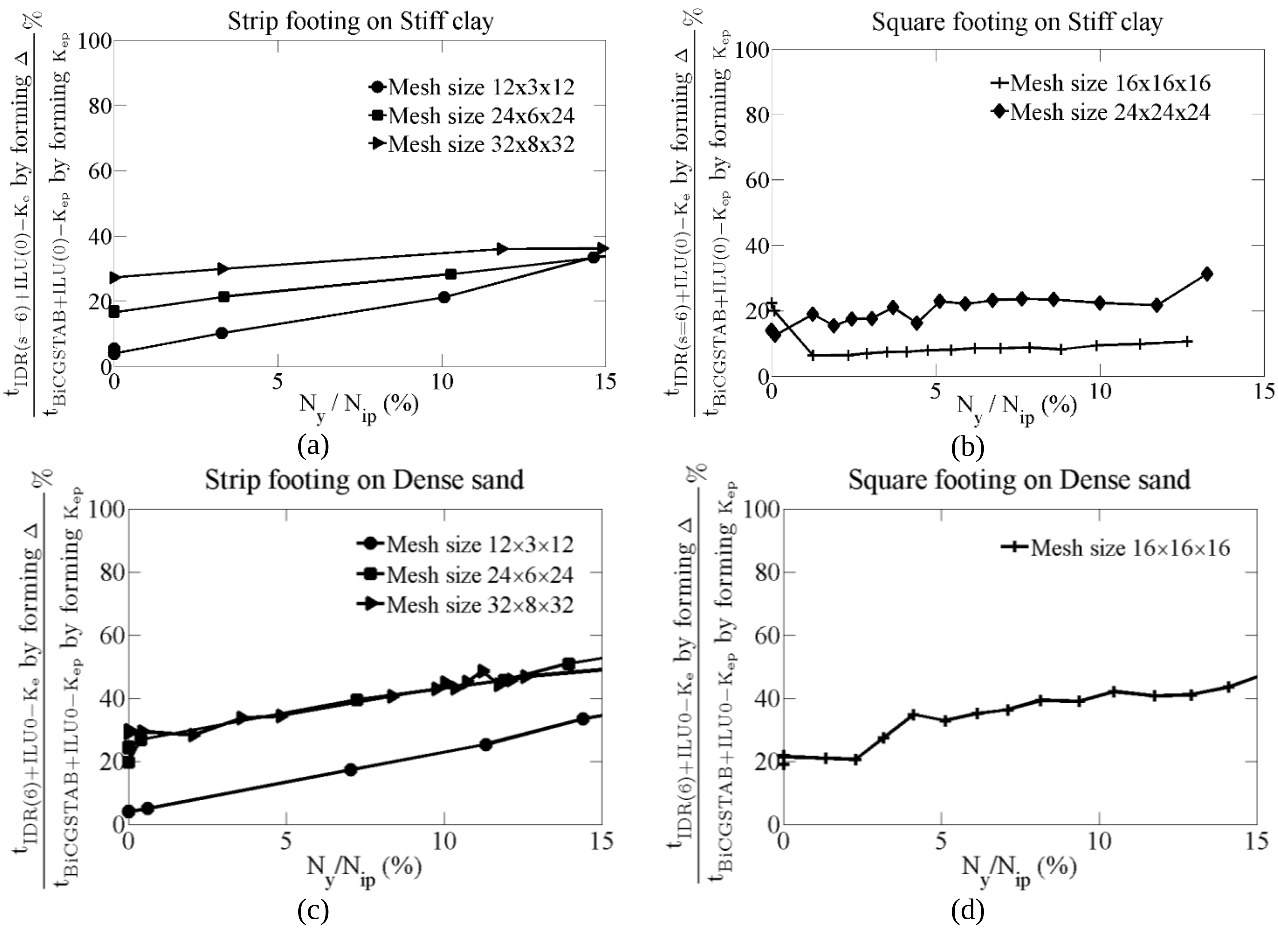


Figure 4.6: Ratio of total time consumed in each NR iteration by method (1): using IDR(6) with ILU0-K_e and forming Δ over method (2): using Bi-CGSTAB with ILU0-K_e and forming K_{ep}.(a)(c) Strip footing. (b)(d) Square footing

4.2.2.2 Update preconditioner after the new load increment is applied

Section 4.2.2.1 has recommended the use of ILU0-K_e to reduce the total iteration time of IDR(6) when the percentage of yielded Gauss points (N_y/N_{ip}) is less than 15 percent. ILU0-K_e loses its time efficiency when N_y/N_{ip} is more than 15 because of the increase of matvec count. From Figure 4.4a and Figure 4.5a , the matvec count required by ILU0-K_e can be up to three times of the matvec count required by ILU0-K_{ep}, which makes the total iteration time consumed by the former to be up to 2.5 times of that consumed by the latter (Figure 4.4c and Figure 4.5c). The increase in matvec count required by ILU0-K_e is because when the number of yielded Gauss points increases, the Δ matrix contributes more in forming K_{ep} hence ILU0-K_e becomes less accurate in approximating K_{ep}. This section proposes different schemes to update ILU0 preconditioner during the nonlinear FE analysis to hopefully reduce the matvec count of IDR(6) and hence reduce the total iteration time when the number of yielded Gauss points increases.

The increase of matvec count lies in the increase of yielded Gauss points therefore this section first considers to update the ILU0 preconditioner at every α% [1] increment of N_y/N_{ip}. Besides, the increase of yielded Gauss points, N_y, is caused by the increase of applied load. Figure 4.7 shows that in each load step, N_y often increase significantly right after the load increment is applied and reaches a stable value when the NR starts converging. Similarly, the matvec required by IDR(6) with ILU0-K_{ep} also increases after the load increment is applied and reduces when the NR iteration converges. Therefore updating the ILU0 preconditioner once every load step right after the load increment is applied may tackle the most difficult case to solve K_{ep} among all the NR iterations within that load step. Table 4.3 summarizes all the preconditioner updating schemes discussed in this section. Different notations of ILU0 preconditioner are also proposed in Table 4.3 to distinguish these updating schemes.

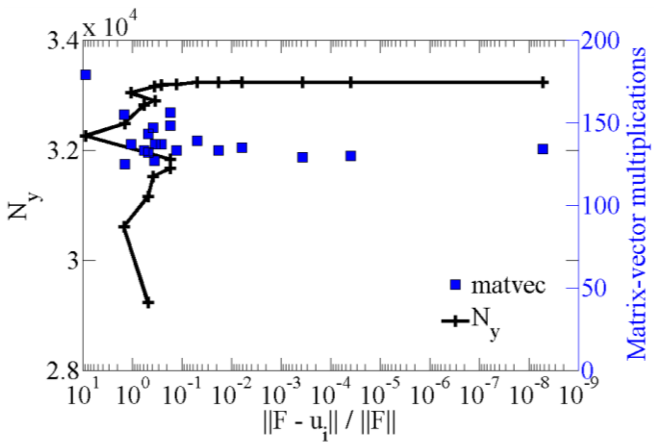


Figure 4.7: Typical trend of variation of N_y and matvec required by IDR(6) with ILU0-K_{ep} within each load step

[1] It should be noted that this α (%) has no relation with the α in the block preconditioners in Section 2.3.3.

Table 4.3: Different schemes to update ILU0 preconditioner during the simulation

Number	Description	Preconditioner notation
1	The default scheme which updates preconditioner at every NR iteration.	ILU0-K_{ep}
2	The scheme discussed in Section 4.2.2.1 which forms the preconditioner from K_e once at the beginning of the simulation.	ILU0-K_e
3	Update the preconditioner at every α% increment of N_y/N_{ip}.	ILU0, α = 5% ILU0, α = 25%
4	Update the preconditioner once in each load step right after the load increment is applied.	ILU0-NR

Figure 4.8 to Figure 4.11 show the comparison of these proposed preconditioner updating schemes. The matvec reported is the average value over all the NR iterations in each load step. The time reported for scheme 2, 3, and 4 is the cumulative solution time during the simulation normalized by the corresponding cumulative solution time of the default scheme 1. Cumulative solution time is a better parameter to analyze than the average time in each load step, which has been used in Chapter 3 and Section 4.2.2, because the preconditioner is updated at different points in time during the simulation.

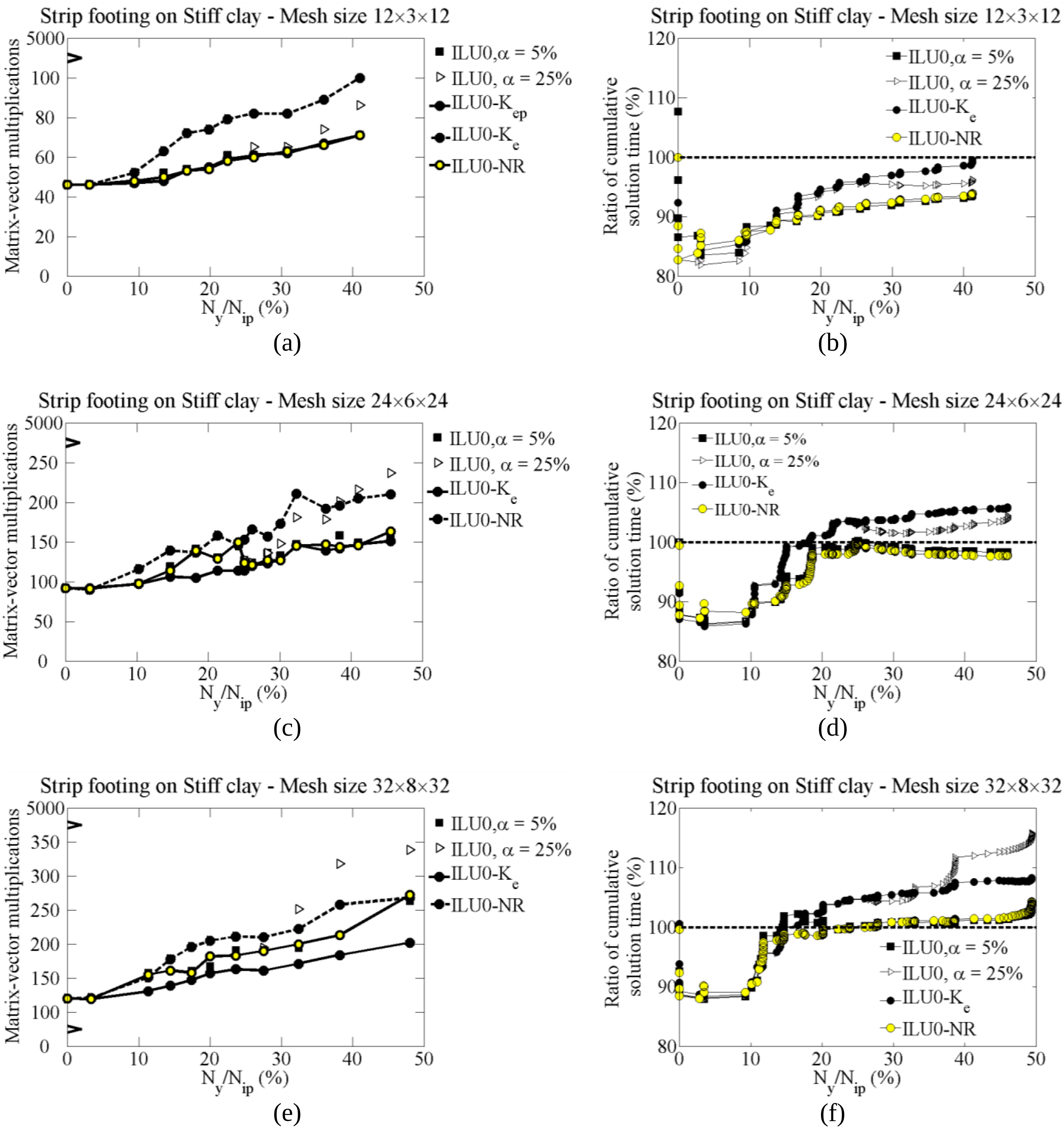


Figure 4.8: Comparison of different schemes of updating ILU0 preconditioner. Strip footing resting on Soil profile 1 is considered.

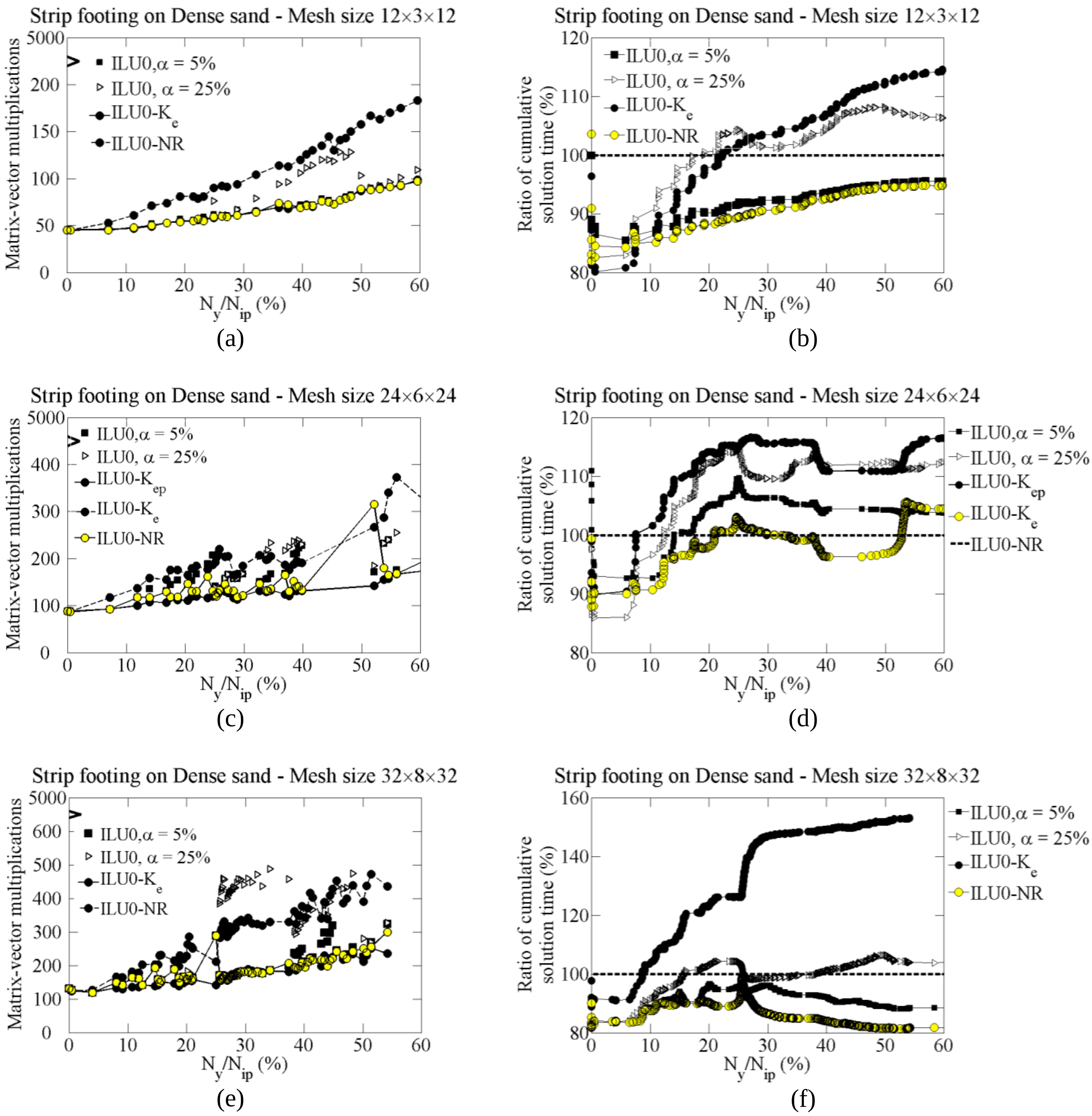


Figure 4.9: Comparison of different schemes of updating ILU0 preconditioner. Strip footing resting on Soil profile 2 is considered.

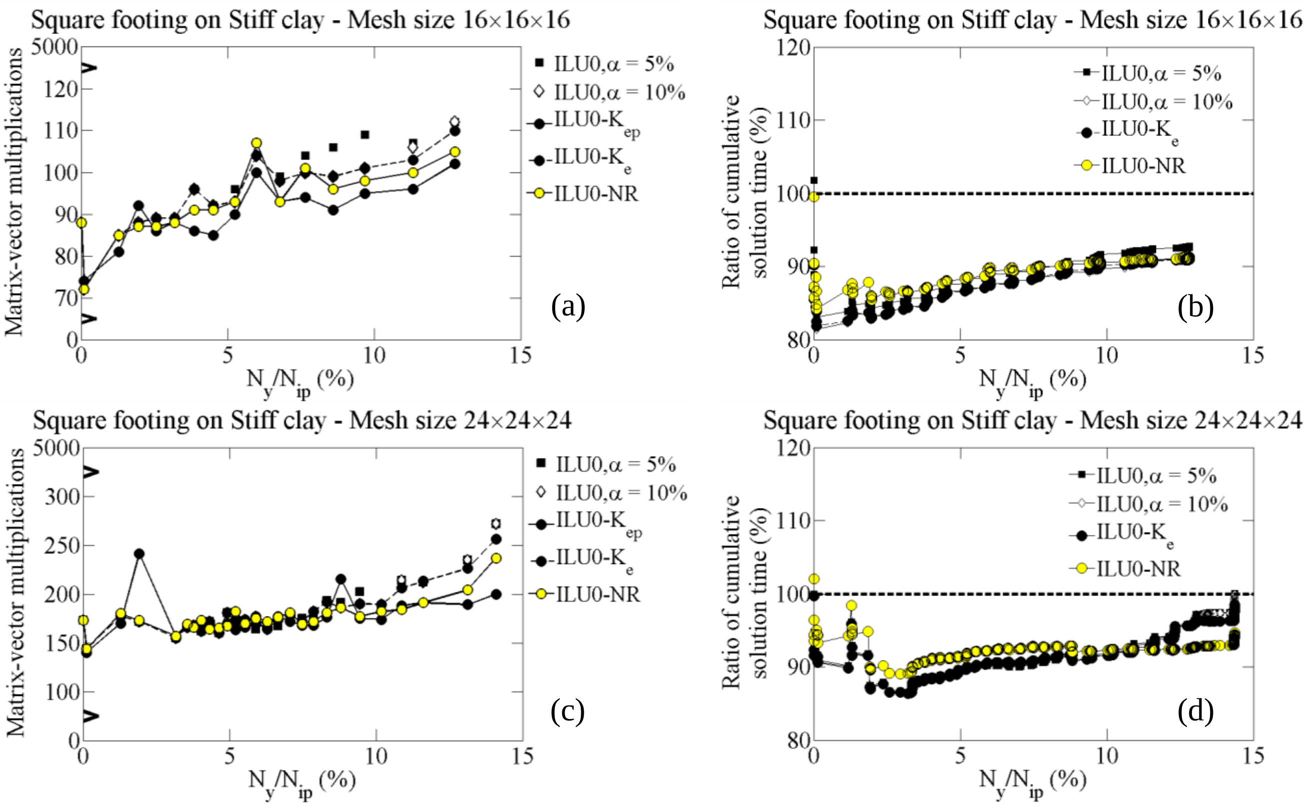


Figure 4.10: Comparison of different schemes of updating ILU0 preconditioner. Square footing resting on Soil profile 1 is considered.

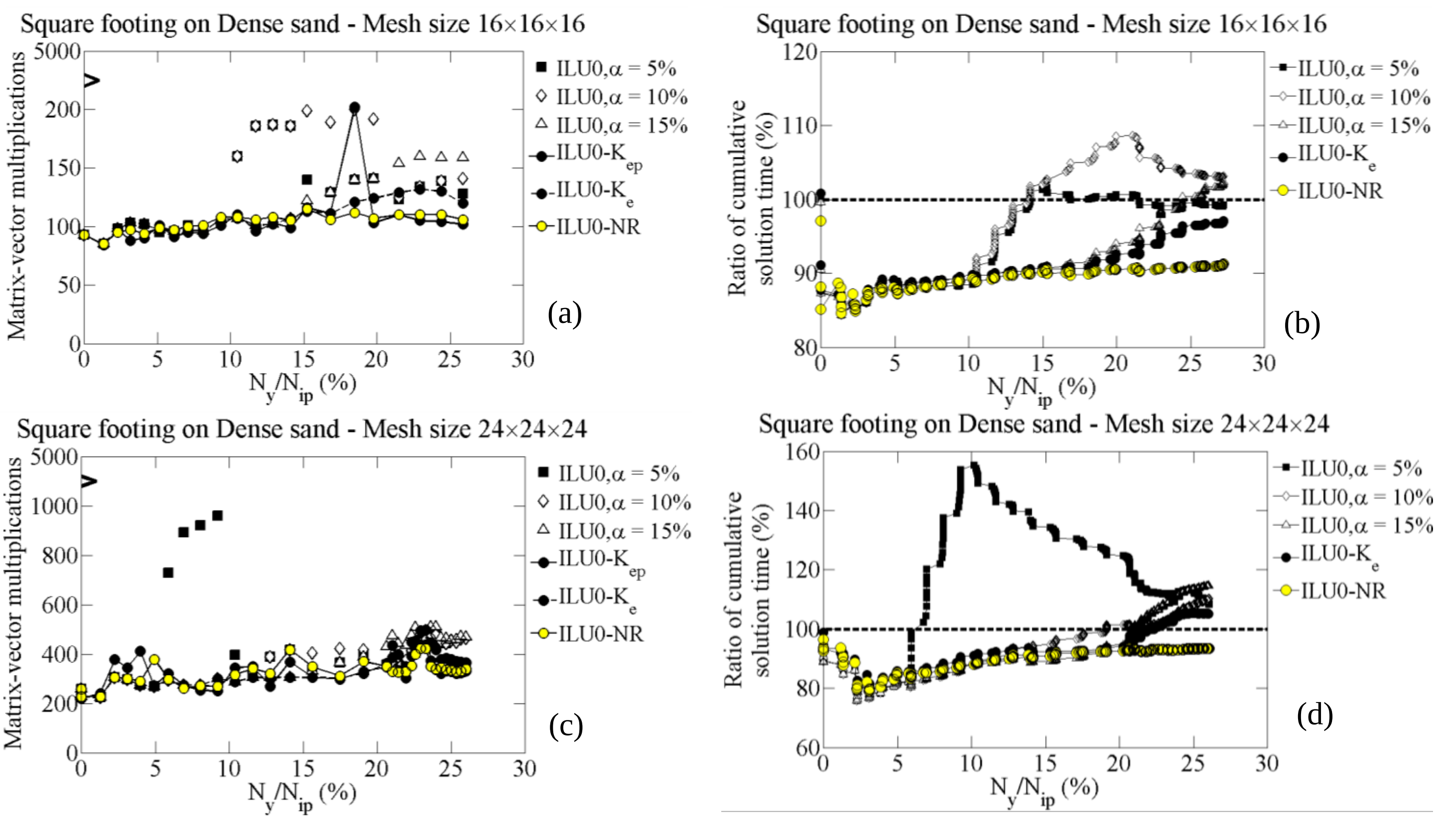


Figure 4.11: Comparison of different schemes of updating ILU0 preconditioner. Square footing resting on Soil profile 2 is considered.

As stated in previous paragraph, the objective of these preconditioner updating schemes is to reduce the matvec count and the total iteration time of IDR(6). Scheme 1 and 2 are expected to produce the lower bound and upper bound of matvec count respectively because at any time during the simulation, ILU0-K_{ep} in scheme 1 approximates K_{ep} the closest and ILU0-K_e in scheme 2 approximates K_{ep} the crudest. This explanation is partly valid in Figure 4.8a and Figure 4.9a when the smallest problem size is used. However it does not apply for larger problem sizes in the rest of the figures from Figure 4.8 to Figure 4.11. In particular, scheme 2 does not provide the upper bound of matvec count. The matvec of scheme 3 are often larger than of scheme 2 although the preconditioner is updated more frequently. This can be because the ILU0 formed at that NR step is unstable and IDR(6) requires more matvec to converge. Section 4.3.2 will show that there are cases that unstable ILU0 can make IDR(*s*) fail to converge within the maximum number of matvec. In the current case, the unstable ILU0 only causes IDR(6) to take significantly more matvec. In scheme 3, when the ILU0 is updated at every α% increment of N_y/N_{ip}, the updated ILU0 could be unstable at that NR iteration but is kept unchanged until the next α% increment of N_y/N_{ip}, hence IDR(6) preconditioned by this unstable ILU0 requires large amount of matvec to converge. This is reflected in Figure 4.9e and Figure 4.11(a)(c). In constrast with scheme 3, the ILU0-NR in scheme 4 does not show unstable behavior during the simulation although it is also updated after certain amount of NR iterations. Moreover, the average matvec count of IDR(6) using ILU0-NR in scheme 4 is as small as the lower bound set by ILU0-K_{ep} in scheme 1.

Observation on matvec count is strongly relatived to the observation of cumulative solution time. Because the time to form ILU0 preconditioner is minimal compared to the iteration time, when there is the rise in the matvec count, there is the rise in the cumulative solution time. The plots on the cumulative solution time of scheme 2 in Figure 4.8(d)(f) and Figure 4.9(b)(d)(f), agree with conclusion from Section 4.2.2.1: this scheme requires less cumulative solution time than the default scheme 1 when N_y/N_{ip} is less than 15%. Scheme 3 can be better than scheme 2 when the matvec required by scheme 3 is less than it is required by scheme 2. This is the case when α = 5%.

ILU0-α = 5% gives the best performance of all the choices of α. The cumulative solution time of this case is even less than the cumulative time of scheme 1 in several cases (Figure 4.8(b)(d) and Figure 4.9(b)(f)). With other choices of α, in Figure 4.8f, Figure 4.9(d)(f), and Figure 4.11(b)(f), the cumulative solution time grows large at some point of the simulation because of the rise in matvec count due to the unstable ILU0 preconditioner as explained in the previous paragraph.

In all the figures from Figure 4.8 to Figure 4.11, the cumulative solution time of scheme 4 is always the smallest even when N_y/N_{ip} grows up to 60 percent. Exception is seen in Figure 4.8f and Figure 4.9d but the cumulative time of scheme 4 in these case is asymptotic to that of scheme 1 so scheme 4 is still considered the most time efficient. Scheme 3 with α = 5% is competitive with scheme 3 for other cases in the strip footing problem (Figure 4.8(b)(d) and Figure 4.9b) but not in the square footing problem. For square footing, although scheme 2 has been shown to be more time efficient than scheme 1 in Section 4.2.2.1, Figure 4.10 and Figure 4.11 show that scheme 4 can achieve even smaller cumulative solution time. From all of the above observations, this section recommends the preconditioner updating scheme 4, which updates the preconditioner after the load increment is applied, to solve the sequence of nonsymmetric linear system from non-associated MC model. This scheme can help to reduce up to 20 percent of total simulation time compared with using ILU0-K_{ep} (Figure 4.9f).

Finally, to make the numerical experiment complete, the cumulative solution time of scheme 1, 2 and 3 are compared with the cumulative solution time by using Bi-CGSTAB with ILU0-K_{ep}, which is the default approach before the development of IDR(s) method. This comparison is shown in Figure 4.12. Section 4.2.2.1 has noted that when soil profile 2 is considered, Bi-CGSTAB does not converge within 5000 matvec when the 24×24×24 mesh is used hence the results are not plotted in Figure 4.12d. The updating scheme 4 still gives the smallest cumulative solution time athough the differences with scheme 1 and 2 are marginal. The most striking difference can be seen in Figure 4.12c. For both the strip footing and square footing, scheme 4 can save at least 40 percent of total simulation time compared with using Bi-CGSTAB.

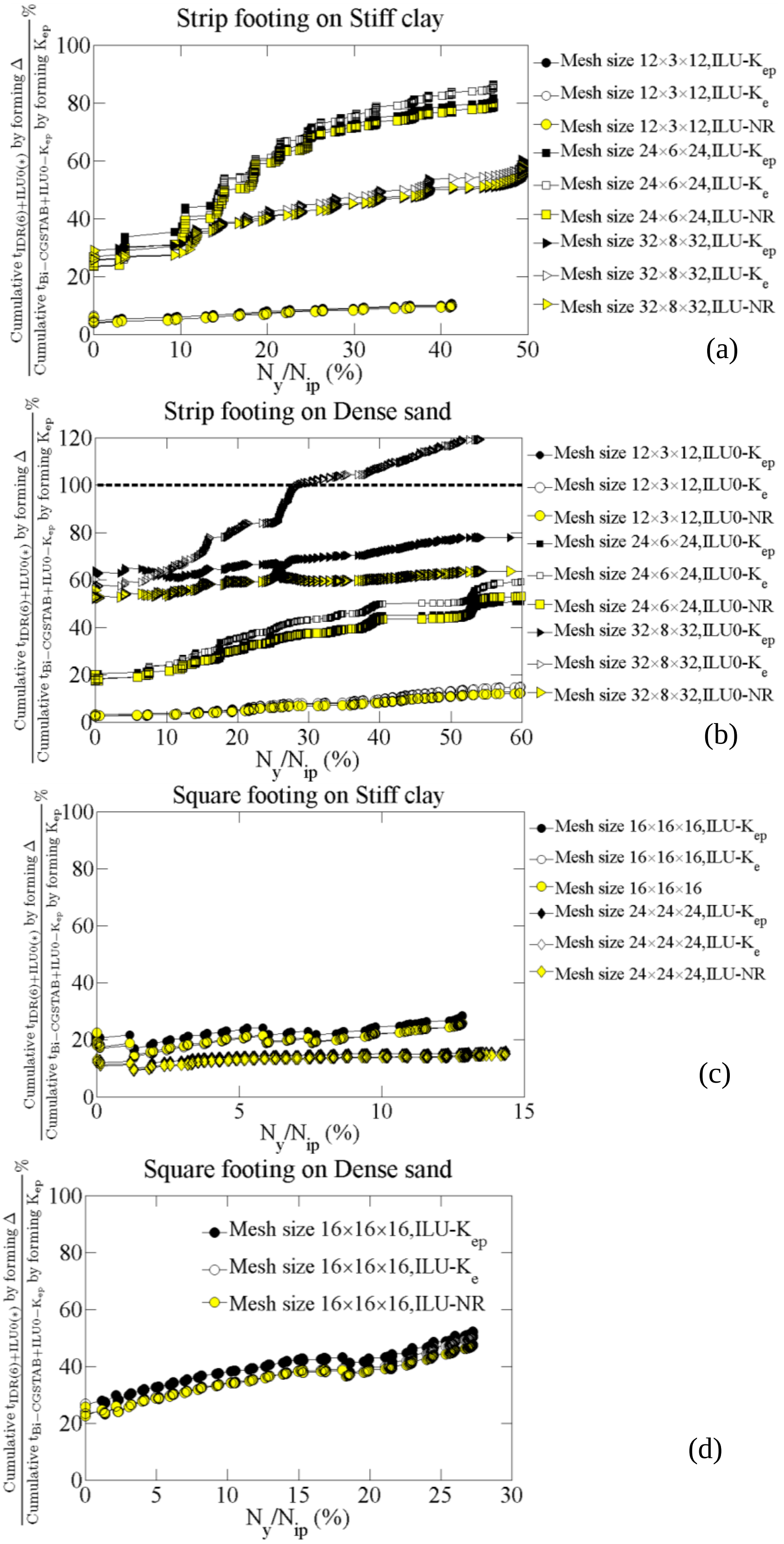


Figure 4.12: Comparison of cumulative solution time of IDR(6) versus Bi-CGSTAB

4.3 Effect of penalty method for prescribed degrees of freedom and undrained analysis on IDR(*s*) and ILU0 preconditioner

This section considers two popular cases of imposed constraints in geotechnical problems: 1) prescribed degrees of freedom (d.o.f) in passive pressure analysis, and 2) prescribed volumetric strain in undrained analysis using effective stress approach. Penalty method is a way to impose constraints. Zienkiewicz and others[187] (2005) have discussed the use of penalty functions and penalty method in FE analysis to impose constraints on the solutions. Penalty method involves a positive 'penalty number', w_p, of which the larger the value the better the constraints are achieved.

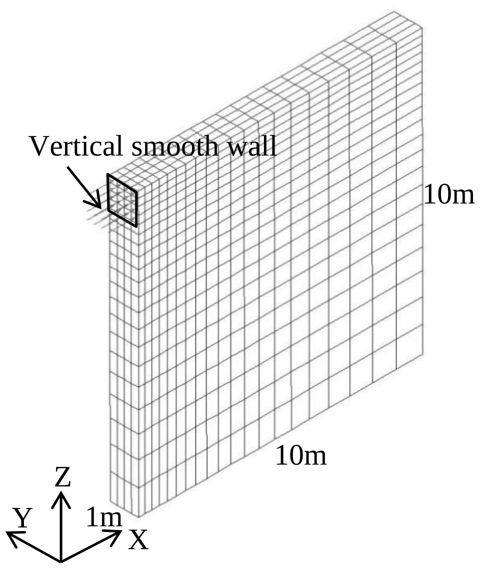


Figure 4.13: 3D FE mesh for the passive pressure analysis

The first case involves a vertical smooth wall (Figure 4.13) subjected to horizontal prescribed displacements to push the wall toward the soil behind it. The theoretical passive resistance of the soil is given in Table 4.5.

Table 4.4: Properties of Mohr-Coulomb soil

	Effective Young's modulus E' (kPa)	Poisson's ratio v'	Effective cohesion c' (kPa)	Effective friction angle ϕ' (^{o})	Dilation angle ψ (^{o})	Self-weight γ (kN/m^3)	"At rest" earth pressure coefficient K_o
Stiff clay	60000	0.3	20	20	0	20	1
Dense sand	105000	0.3	1	30	5	20	1

Table 4.5: Total passive resistance on the 1m height smooth vertical wall

	K_p	H (m)	P_p (kN)
Soil profile 1	2.04	1	77.52
Soil profile 2	3	1	33.46

For each row of K_{ep} corresponding with the prescribed d.o.fs, the penalty method is applied by adding the penalty number, w_p, the diagonal entries and replacing the right-hand-side entries with the product of w_p and the prescribed value. This formation of K_{ep} is denoted as "unscaled K_{ep}" and is demonstrated in Eq.(4.7) when u_i is the prescribed degree of freedom. Very minimal amount of d.o.fs is constrained in this case hence the value of w_p does not affect the convergence of IDR(6) as shown in Figure 4.15 and Figure 4.14. Chen and Phoon[40] (2009) recommended scaling the rows and columns containing the penalty number so that the values of the corresponding diagonal entries are close to 1. By this way, the global stiffness matrix is better conditioned. Eq.(4.8) demonstrates this scaling process when ui is the prescribed degree of freedom. The K_{ep} modified by this scaling process is denoted as "scaled K_{ep}". This recommendation is useful and essential for the problems tested in this section.

$$\begin{bmatrix} k_{11} & \dots & k_{1i} & \dots & k_{1n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i1} & \dots & k_{ii}+w_p & \dots & k_{in} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{n1} & \dots & k_{ni}\big/\sqrt{w_p} & \dots & k_{nn} \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_i = c_i \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ c_i w_p \\ \vdots \\ 0 \end{bmatrix} \tag{4.7}$$

$$\begin{bmatrix} k_{11} & \dots & k_{1i}\big/\sqrt{w_p} & \dots & k_{1n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{i1}\big/\sqrt{w_p} & \dots & (k_{ii}+w_p)/w_p \approx 1 & \dots & k_{in}\big/\sqrt{w_p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ k_{n1} & \dots & k_{ni}\big/\sqrt{w_p} & \dots & k_{nn} \end{bmatrix} \begin{bmatrix} u_1 \\ \vdots \\ u_i = c_i\sqrt{w_p} \\ \vdots \\ u_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ c_i\sqrt{w_p} \\ \vdots \\ 0 \end{bmatrix} \tag{4.8}$$

In Figure 4.14 and Figure 4.15, without scaling, although IDR(6) with ILU0-K_{ep} preconditioner do not require more matvec to converge, the iterative solver often fails when w_p becomes too large. IDR(s) fails due to $\mu_{k,k}$ in Figure 2.2 becomes zero and the set of basic vectors of the new subspace G_j cannot be formed. Varying the shadow matrix P and increasing s do not help with the convergence. This failure may be due to the round-off error because w_p is much larger than the matrix entries.

The second case considers the same flexible strip footing resting on the homogenous soil layer. In this case, the footing is under undrained loading, in which the volumetric strain is zero – or there is no change in volume of the soil mass considered. Penalty number affects the global stiffness matrix significantly in this case. In the global stiffness matrix given in Eq.(2.2), the penalty number is the term K_w/n, which involves in each element stress-strain matrix. The rest of this section focuses on the behavior of ILU0 in solving this undrained analysis.

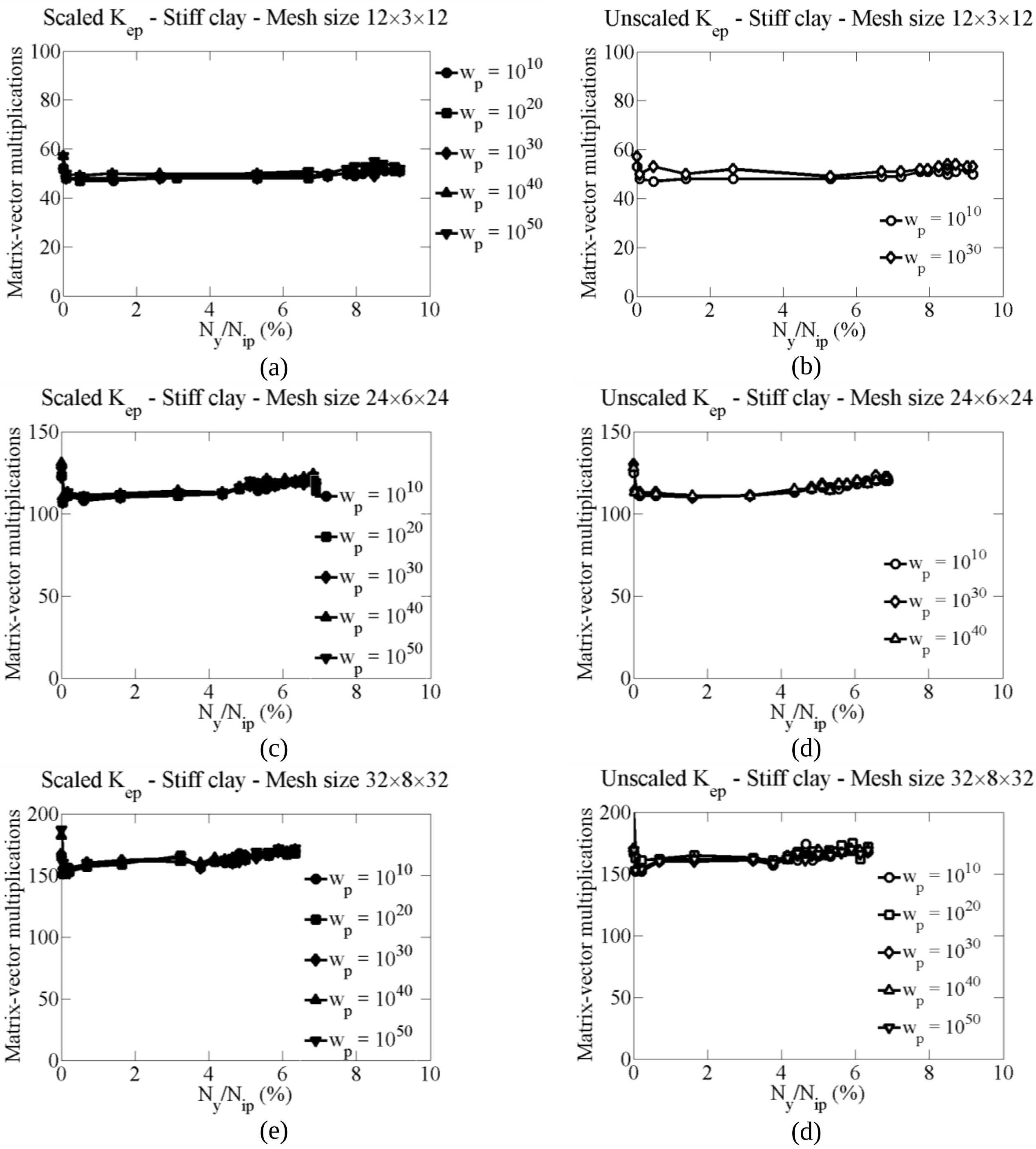


Figure 4.14: Matrix-vector multiplications of IDR(6) with ILU0-K_{ep} when solving the retaining wall subjected to prescribed horizontal displacements. Soil profile 1 is used.

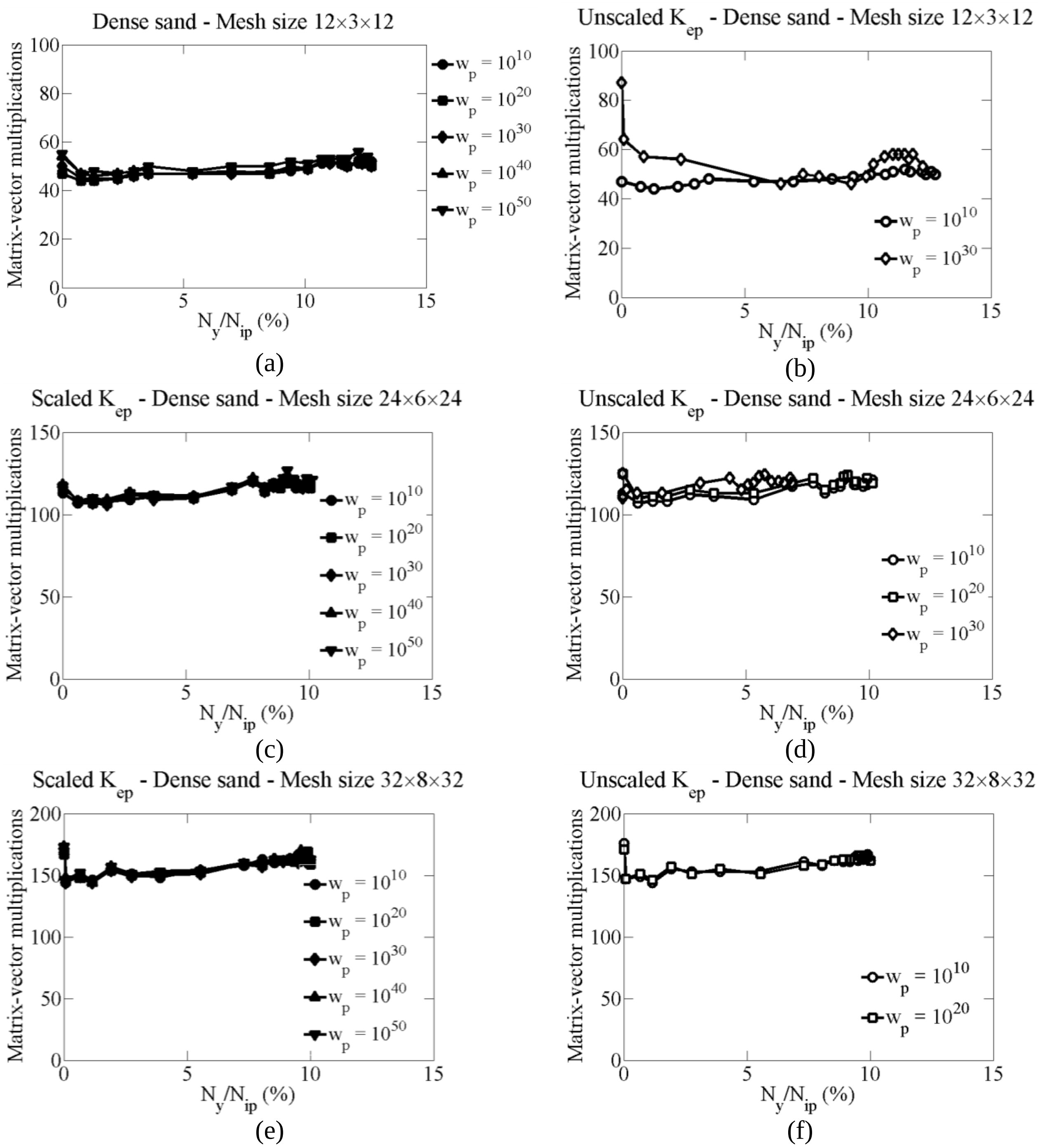


Figure 4.15: Matrix-vector multiplications of IDR(6) with ILU0-K_{ep} when solving the retaining wall subjected to prescribed horizontal displacements. Soil profile 2 is used.

4.3.1 Undrained analysis of the strip footing using effective stress method

The failure load of the strip footing in undrained analysis can be calculated with Eq.(3.1) with equivalent undrained parameters derived from the drained parameters by Eq.(4.9), (4.10), and (4.11). The total friction angle is zero.

$$E_u = \frac{3E'}{2(1+\nu')} \tag{4.9}$$

$$c_u = \frac{2c'\sqrt{N_\phi}}{1+N_\phi} \tag{4.10}$$

$$N_\phi = \frac{1+\sin\phi'}{1-\sin\phi'} \tag{4.11}$$

One shortcoming of the MC model is that the geotechnical systems do not fail when the dilation angle ψ is different from zero. Theoretical results of triaxial test show that the volumetric strain keeps increasing when the applied load increases. One rectification recommended is to manually set the dilation angle to zero to control the volumetric strain. Hence in this section, the dilation angle is set to zero for both dense sand and stiff clay, which still satisfies the non-associated flow rule.

Table 4.6: Total stress parameters of Mohr-Coulomb yield criterion

	Young's modulus, E_u (MPa)	Poisson's ratio, v_u	Cohesion, c_u (kPa)	Friction angle, ϕ_u (degree)	Dilation angle, ψ (degree)
Stiff clay	69.23	0.5	18.79	0	0
Dense sand	121.15	0.5	0.8660	0	0

Table 4.7: Ultimate bearing capacity of strip footing on homogenous soil layers

	N_c	$q_{f\text{-}strip}$ = c_uN_c (kPa)	q_{max} (kPa)
Soil profile 1	5.14	96.58	90 (= 93% $q_{f\text{-}strip}$)
Soil profile 2	5.14	4.45	4.4(= 99% $q_{f\text{-}strip}$)

The following section will show that IDR(6) with ILU0-K_{ep} does not converge when solving Eq. (2.2). Hence IDR(6) was preconditioned with SSOR-LR to carry on the analysis and obtain K_{ep}, of which ILU0 factorization is investigated. IDR(6) with SSOR-LR did converge but SSOR-LR is not an efficient remedy because the iterative solver always requires more than 5000 matvec to converge and often shows stagnant behaviour when the footing is close to failure.

4.3.2 Problem with ILU0 factorization

Figure 4.16 shows the relative residual norm when solving the undrained problem with IDR(6) and Bi-CGSTAB preconditioned with ILU0-K_{ep}. Both of the methods do not converge within the prescribed matvec but Bi-CGSTAB behaviour is worse because the relative residual norm keeps increasing and finally break down while it is stagnant in IDR(6). Increasing *s* in IDR(*s*) does not help the convergence hence the problems may lie in ILU0-K_{ep} preconditioner.

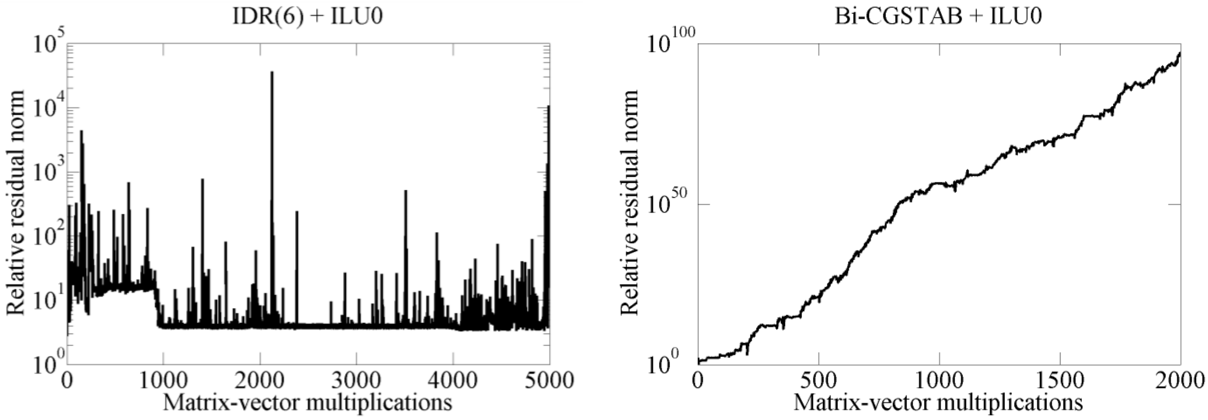


Figure 4.16: Typical relative residual norm of an unstable ILU0 preconditioner: (a) IDR(6) method; (b) Bi-CGSTAB method

Table 4.8 and Table 4.9 presents the statistics of ILU0-K_{ep} according to the recommendation of Chow and Saad[43] (1997) with the variation of penalty number, K_w/n. These statistics include *condest*, 1/pivot, and $\max(\overline{\mathrm{L}} + \overline{\mathrm{U}})$, of which meanings and computing procedure have been presented in Table 2.1 and Figure 2.6.

When $K_w/n = 0$, the problem is returned to drained analysis and all the statistics are small, which proves the stability and efficiency of ILU0-K_{ep} in drained analysis as shown in Chapter 3 and Section 4.2. When K_w/n increases, *condest* and $\max(\overline{\mathrm{L}} + \overline{\mathrm{U}})$ grow extremely large: on the order of 10^{15} following the guideline from Chow and Saad[43] (1997) while 1/*pivot* is still as small as when K_w/n = 0. The extremely large values are highlighted in Table 4.8 and Table 4.9. Although Chow and Saad[43] (1997) reported that they did not observed any system that has large $\max(\overline{\mathrm{L}}+\overline{\mathrm{U}})$ and small 1/*pivot*, the small 1/pivot and large $\max(\overline{\mathrm{L}}+\overline{\mathrm{U}})$ are found in this undrained system. 1/*pivot* is small or pivot value is reasonable large can be because the large penalty number K_w/n is added to the system. Chow and Saad recommended that when ILU preconditioner has small 1/*pivot* and large *condest*, the iterative solver fails to converge due to unstable triangular solves. Stabilizing ILU0 by adding the threshold is irrelevant because the pivot values are large enough. ILUT(50, 10^{-6}) was used to precondition IDR(6) and Bi-CGSTAB and the same trend of relative residual norm with Figure 4.16 was found. Benzi and others[20] (1999) also noticed that when the problem does not lie in the accuracy of ILU0 factorization, allowing more fill-ins does not help with the convergence. They recommended reordering the original matrix to improve the ILU factorization and RCM is recommended in general. The statistics of ILU0 of the RCM reordering K_{ep} are presented in Table 4.8 and Table 4.9. Condest of this ILU0 are 10^2 – 10^{30} times larger than of ILU0-K_{ep} from the original K_{ep} and IDR(6) does not converge when preconditioned with this ILU0. Hence RCM ordering is not useful in this case.

Table 4.8: ILU statistics and possible reasons of failure for soil profile 1 – Stiff clay

	condest	1/*pivot*	$\max(\overline{\mathrm{L}} + \overline{\mathrm{U}})$
Mesh size 16×3×16			
$K_w/n = 0K' = 0$			
Elastic	2.27×10^{-3}	1.86×10^{-4}	8.68×10^{5}
Elastic (90kPa)	2.27×10^{-3}	1.86×10^{-4}	8.68×10^{5}
$K_w/n = 50K'$			
Elastic	$\mathbf{5.83\times10^{25}}$	3.11×10^{-3}	2.82×10^{11}
Elastoplastic (90kPa)	$\mathbf{3.66\times10^{23}}$	1.56×10^{-2}	1.35×10^{12}
$K_w/n = 50K'$– RCM (K_{ep})			
Elastic	$\mathbf{5.36\times10^{22}}$	1.25×10^{-2}	2.00×10^{12}
Elastoplastic (90kPa)	$\mathbf{4.76\times10^{24}}$	4.38×10^{-2}	4.36×10^{12}
$K_w/n = 500K'$			
Elastic	$\mathbf{7.32\times10^{22}}$	3.67×10^{-4}	6.19×10^{13}
Elastoplastic (90kPa)	$\mathbf{1.66\times10^{23}}$	3.05×10^{-3}	1.75×10^{13}
$K_w/n = 500K'$– RCM (K_{ep})			
Elastic	$\mathbf{2.24\times10^{23}}$	2.35×10^{-3}	1.44×10^{13}
Elastoplastic (90kPa)	$\mathbf{4.07\times10^{24}}$	8.02×10^{-4}	5.60×10^{13}
Mesh size 24×6×24			
$K_w/n = 0K' = 0$			
Elastic	6.46×10^{-3}	3.72×10^{-4}	4.35×10^{5}
Elastic (90kPa)	6.46×10^{-3}	3.72×10^{-4}	4.35×10^{5}
$K_w/n = 50K'$			
Elastic	$\mathbf{2.47\times10^{63}}$	1.27×10^{-2}	$\mathbf{1.12\times10^{15}}$
Elastoplastic (90kPa)	$\mathbf{3.96\times10^{63}}$	8.04×10^{-2}	$\mathbf{3.06\times10^{17}}$
$K_w/n = 50K'$– RCM (K_{ep})			
Elastic	$\mathbf{2.02\times10^{78}}$	6.82×10^{-2}	3.90×10^{14}
Elastoplastic (90kPa)	$\mathbf{2.30\times10^{80}}$	1.12×10^{0}	$\mathbf{1.46\times10^{17}}$
$K_w/n = 500K'$			
Elastic	$\mathbf{6.61\times10^{61}}$	3.28×10^{-2}	4.67×10^{13}
Elastoplastic (90kPa)	$\mathbf{2.94\times10^{64}}$	1.17×10^{-2}	$\mathbf{2.67\times10^{16}}$
$K_w/n = 500K'$– RCM (K_{ep})			
Elastic	$\mathbf{3.62\times10^{75}}$	2.72×10^{-2}	$\mathbf{7.75\times10^{17}}$
Elastoplastic (90kPa)	$\mathbf{2.28\times10^{78}}$	8.20×10^{-3}	$\mathbf{7.75\times10^{17}}$
Mesh size 32×8×32			
$K_w/n = 0K' = 0$			
Elastic	9.35×10^{-3}	4.96×10^{-4}	3.26×10^{5}
Elastic (90kPa)	9.35×10^{-3}	4.96×10^{-4}	3.26×10^{5}
$K_w/n = 50K'$			
Elastic	$\mathbf{3.94\times10^{95}}$	8.32×10^{-2}	1.62×10^{14}
Elastoplastic (90kPa)	$\mathbf{9.08\times10^{95}}$	1.05×10^{0}	$\mathbf{4.10\times10^{16}}$
$K_w/n = 50K'$– RCM (K_{ep})			
Elastic	$\mathbf{8.78\times10^{127}}$	5.91×10^{0}	$\mathbf{6.89\times10^{19}}$
Elastoplastic (90kPa)	$\mathbf{2.10\times10^{128}}$	7.00×10^{1}	$\mathbf{8.53\times10^{26}}$
$K_w/n = 500K'$			
Elastic	$\mathbf{5.85\times10^{95}}$	1.82×10^{-2}	$\mathbf{7.18\times10^{15}}$
Elastoplastic (90kPa)	$\mathbf{4.32\times10^{95}}$	7.41×10^{-2}	$\mathbf{1.49\times10^{16}}$
$K_w/n = 500K'$– RCM (K_{ep})			
Elastic	$\mathbf{4.32\times10^{124}}$	2.07×10^{-2}	$\mathbf{3.85\times10^{21}}$
Elastoplastic (90kPa)	$\mathbf{2.97\times10^{129}}$	4.19×10^{-2}	$\mathbf{3.06\times10^{21}}$

Table 4.9: ILU statistics and possible reasons of failure for soil profile 2 – Dense sand

	condest	1/*pivot*	max($\overline{\mathrm{L}}+\overline{\mathrm{U}}$)
Mesh size 16×3×16			
K_w/n = 0K' = 0			
Elastic	1.35×10^{-3}	1.06×10^{-4}	1.52×10^{6}
Elastic (4.4kPa)	1.35×10^{-3}	1.06×10^{-4}	1.52×10^{6}
K_w/n = 50K'			
Elastic	$\mathbf{1.11\times10^{24}}$	2.07×10^{-3}	1.37×10^{12}
Elastoplastic (4.4kPa)	$\mathbf{6.32\times10^{21}}$	5.83×10^{-3}	5.07×10^{11}
K_w/n = 50K' – RCM (K_{ep})			
Elastic	$\mathbf{4.07\times10^{21}}$	3.11×10^{-3}	8.03×10^{11}
Elastoplastic (4.4kPa)	$\mathbf{6.08\times10^{21}}$	1.62×10^{-3}	3.35×10^{12}
K_w/n = 500K'			
Elastic	$\mathbf{2.26\times10^{21}}$	8.72×10^{-4}	2.71×10^{14}
Elastoplastic (4.4kPa)	$\mathbf{6.58\times10^{21}}$	2.26×10^{-2}	$\mathbf{1.47\times10^{17}}$
K_w/n = 500K' – RCM (K_{ep})			
Elastic	$\mathbf{5.37\times10^{21}}$	2.00×10^{-4}	$\mathbf{2.88\times10^{15}}$
Elastoplastic (4.4kPa)	$\mathbf{1.68\times10^{22}}$	5.69×10^{-3}	$\mathbf{1.94\times10^{16}}$
Mesh size 24×6×24			
K_w/n = 0K' = 0			
Elastic	3.56×10^{-3}	2.13×10^{-4}	7.61×10^{5}
Elastic (4.4kPa)	3.56×10^{-3}	2.13×10^{-4}	7.61×10^{5}
K_w/n = 50K'			
Elastic	$\mathbf{5.97\times10^{60}}$	3.10×10^{-2}	1.05×10^{13}
Elastoplastic (4.4kPa)	$\mathbf{7.41\times10^{59}}$	1.04×10^{-1}	1.13×10^{14}
K_w/n = 50K'– RCM (K_{ep})			
Elastic	$\mathbf{9.19\times10^{77}}$	9.26×10^{-2}	8.86×10^{14}
Elastoplastic (4.4kPa)	$\mathbf{2.25\times10^{76}}$	2.35×10^{-2}	5.29×10^{14}
K_w/n = 500K'			
Elastic	$\mathbf{4.92\times10^{61}}$	8.59×10^{-3}	$\mathbf{2.04\times10^{16}}$
Elastoplastic	$\mathbf{2.48\times10^{59}}$	1.12×10^{-3}	3.68×10^{14}
K_w/n = 500K'– RCM (K_{ep})			
Elastic	$\mathbf{4.44\times10^{75}}$	5.70×10^{-3}	$\mathbf{9.10\times10^{16}}$
Elastoplastic	$\mathbf{5.23\times10^{79}}$	2.88×10^{-3}	$\mathbf{9.10\times10^{16}}$
Mesh size 32×8×32			
K_w/n = 0K' = 0			
Elastic	5.25×10^{-3}	2.84×10^{-4}	5.71×10^{5}
Elastic (4.4kPa)	5.25×10^{-3}	2.84×10^{-4}	5.71×10^{5}
K_w/n = 50K'			
Elastic	$\mathbf{6.05\times10^{93}}$	3.07×10^{-2}	$\mathbf{1.33\times10^{15}}$
Elastoplastic (4.4kPa)	$\mathbf{6.48\times10^{91}}$	1.67×10^{-1}	$\mathbf{1.19\times10^{15}}$
K_w/n = 50K'– RCM (K_{ep})			
Elastic	$\mathbf{2.94\times10^{125}}$	5.69×10^{-2}	$\mathbf{2.66\times10^{22}}$
Elastoplastic (4.4kPa)	$\mathbf{4.03\times10^{125}}$	2.38×10^{-2}	$\mathbf{1.30\times10^{19}}$
K_w/n = 500K'			
Elastic	$\mathbf{6.52\times10^{98}}$	4.94×10^{-3}	$\mathbf{7.09\times10^{22}}$
Elastoplastic (4.4kPa)	$\mathbf{1.96\times10^{92}}$	1.09×10^{-1}	$\mathbf{3.83\times10^{17}}$
K_w/n = 500K'– RCM (K_{ep})			
Elastic	$\mathbf{2.52\times10^{130}}$	1.46×10^{-2}	$\mathbf{2.61\times10^{21}}$
Elastoplastic	$\mathbf{2.00\times10^{129}}$	1.92×10^{-2}	$\mathbf{2.61\times10^{21}}$

It is worth to note that undrained problem can be solved by total stress approach, meaning to use the total stress parameters in Table 4.6 and solve the 1-by-1 block linear system as usual. The shortcoming of this approach is the pore pressure is unavailable in the solution. In this case, both the friction angle and dilation angle have to be set to zero, implying associated flow rule and symmetric global stiffness matrix K_{ep}. The Poisson's ratio is set to 0.499. Although this K_{ep} is symmetric and possesses better eigenspectrum (i.e. no complex eigenvalues), IDR(6) with ILU0-K_{ep} does not converge and the relative residual norm shows the stagnant behavior as in Figure 4.16. Hence ILU0 is not suitable to precondition the linear system of undrained analysis in Eq.(2.2).

4.3.3 Recommendation for remedy

The last paragraph in Section 4.3.1 has mentioned that the undrained analysis in this chapter was performed with SSOR-LR preconditioner. Hence one easy and cheap remedy is to use Jacobi or SSOR-L/SSOR-LR preconditioner instead of ILU0. However Section 4.3.1 also shown that IDR(6) with these preconditioners always requires more than 5000 matvec to converge and does not converge when the system is close to failure. Hence unless Eq.(2.2) is demanded to obtain the undrained behavior of the geotechnical problems, this remedy is not recommended.

A better remedy is using Biot's consolidation analysis to simulate undrained behavior of soil by tuning either the permeability [*k*] or the time step Δt to small values. Phoon and others[131] compared this approach with the analysis using Eq. (2.2) when the soil is linear elastic and concluded that this approach is more advantageous. Chapter 5 will show that IDR(*s*) and ILU0 preconditioner can solve Biot's consolidation equations, Eq.(2.8), efficiently when the soil follow the non-associated MC model, and undrained behavior can be obtained by tuning [*k*] and Δt without difficulty.

4.4 Summary

This chapter performs numerical experiments to observe the behavior of IDR(6) preconditioned with ILU0 preconditioner when solving sequence of linear systems and when penalty method is applied. The key recommendations and observations are summarized below:

1. The techniques to save the total simulation time in dealing with sequence of nonsymmetric linear systems are recommended as:
    a. Forming the elastoplastic global stiffness matrix $K_{ep} = K_e + \Delta$ implicitly by forming the elastic global stiffness matrix K_e once and update the Δ matrix at every NR iteration. Forming Δ matrix only takes up to 40 percent of the time consumed in each NR iteration while forming the complete K_{ep} will take at least 60 percent.
    b. Using ILU0-K_e to save the time to form preconditioner when the percentage of yielded Gauss points N_y/N_{ip} is less than 15 percent.
    c. Using the preconditioner updating scheme 4 in Table 4.3 to update ILU0-NR preconditioner after the load increment is applied. ILU0-NR is more time efficient than ILU0-K_e and is still effective when N_y/N_{ip} is more than 15. When this updating scheme is combined with technique (a), the total simulation time can be reduced by 60 percent (Figure 4.12).
2. Penalty method involves adding a large penalty number to the global stiffness matrix to impose some constraints on the system. The passive pressure problem demonstrates the prescribed displacements as constraints and penalty number has minimal effects on IDR(6) and ILU0-K_{ep} because the number of constrained d.o.fs is minimal compared with the total number of d.o.f. On the contrary, the penalty number greatly affects the undrained analysis using Eq.(2.2) because this case has the constraint such that the volumetric strain is zero over the whole domain. ILU0-K_{ep} has been shown to be unstable and both IDR(6) and Bi-CGSTAB fail to converge when ILU0-K_{ep} is used. Increasing the fill-ins by using ILUT(50, 10^{-6}) and reordering K_{ep} with RCM method do not mitigate the situation. Jacobi and SSOR-L/SSOR-LR are recommended when Eq.(2.2) is demanded and the linear system is small. A more practical remedy for large-scale problems is using

Biot's consolidation analysis in Eq.(2.8) to simulate the undrained behavior by tuning the permeability and time step. This recommendation is discussed in detail in Chapter 5.

Part of this chapter has been accepted for publication in:

Tran, H.H.T., Toh, K.C. and Phoon, K.K. (2013), Preconditioned IDR(s) iterative solver for non-symmetric linear system associated with FEM analysis of shallow foundation. Int. J. Numer. Anal. Meth. Geomech.. doi: 10.1002/nag.2171

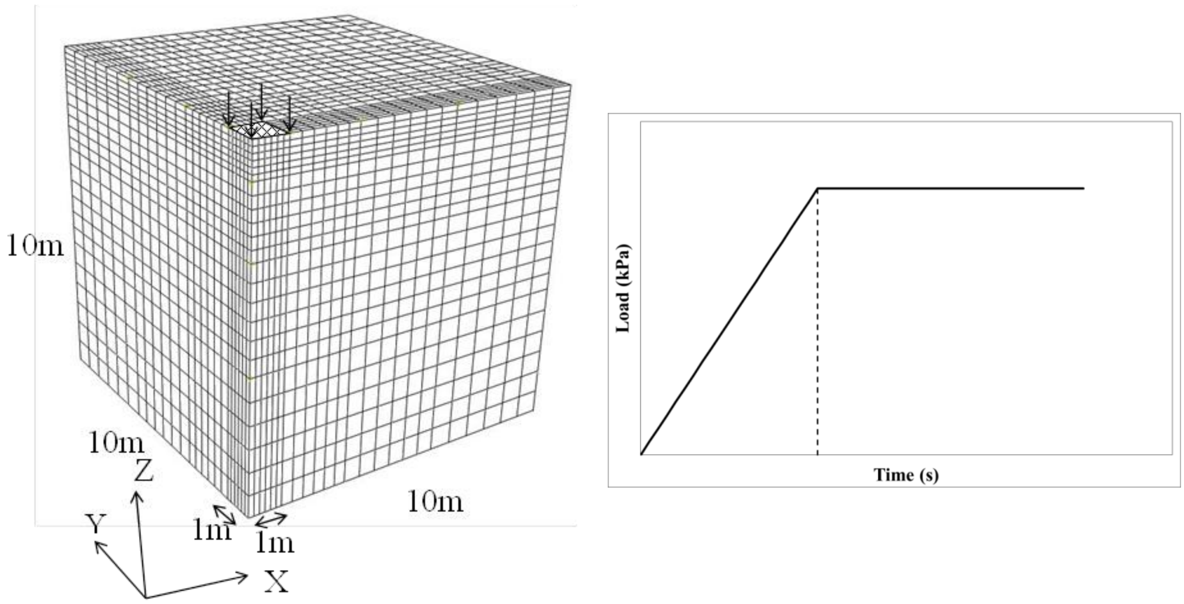
CHAPTER 5 PRECONDITIONERS FOR 2-BY-2 BLOCK MATRICES: CONSOLIDATION ANALYSIS

5.1 Introduction

This chapter discusses the application of preconditioners in Section 2.3 to the Biot's consolidation analysis when the soil follows the non-associated MC model. These preconditioners include diagonal block preconditioner M_d, constrained block preconditioner M_c, MSSOR and ILU0. From Section 2.3, block preconditioners mainly approximates the elastoplastic global stiffness matrix by varying $\hat{K}$ and $\hat{S}$, which are approximations of K_{ep} and Schur complement S respectively. This chapter will first perform numerical experiments to compare the efficiency of diagonal block preconditioner M_d and constrained block preconditioner M_c with different $\hat{K}$ and $\hat{S}$. Then the most efficient block preconditioner will be compared with MSSOR and ILU0, which treat the 2-by-2 block global stiffness matrix as a 1-by-1 block matrix. Secondly, this chapter will adopt the preconditioner updating scheme proposed in Section 4.2.2.2 to investigate its efficiency in saving the simulation time to solve the Biot's consolidation problem. Thirdly, the eigenvalue distribution of the global stiffness matrix is presented to explain the convergence of IDR(s) and the efficiency of the block preconditioner. Finally, this chapter will show that undrained analysis can be simulated using Biot's consolidation equation and the block preconditioner can be used to speed up the simulation time hence resolve the difficulty observed in Section 4.3.2.

5.2 Problem description

All the numerical experiments in this chapter are performed on the flexible square footing problem. Figure 5.1a shows the 3D FE mesh of the flexible square footing resting on a layer of homogenous soil and subjected to uniform pressure, q. Table 5.1 summaries the characteristics of the three FE meshes ($16\times16\times16$, $20\times20\times20$, $24\times24\times24$) used in this section. The water table is set at the ground surface and is in hydrostatic condition at the initial stage; the soil is assumed to be fully saturated. The base is fixed in all directions and impermeable. The side faces are fixed in transversed directions and free in in-plane directions for both displacement and water flux. The top surface is free in all direction and free-draining with pore pressure assumed to be zero. Two cases of soil profile are considered: soil profile 1 is a homogeneous stiff clay layer; soil profile 2 is a homogeneous dense sand layer. The soil follows a non-associated MC model and the effective parameters are presented in Table 5.2.



(a) (b)

Figure 5.1: (a) 3D mesh of the square footing; (b) Ramp loading

Table 5.1: 3D finite element meshes of the square footing

	Mesh size*			
	8×8×8	16×16×16	20×20×20	24×24×24
Number of elements	512	4,096	8,000	13,824
Number of nodes	2673	18,785	35,721	60,625
Number of unknowns (N)	7160	55,280	107,180	184,296
Displacement DOFs, nd	6512	50,626	98,360	169,296
Pore pressure DOFs, np	648	4,624	8,820	15,000
np/N (%)	9.05	8.36	8.23	8.14
Number of Gauss points (N_{ip})	13,824	110,592	216,000	373,248
Number of nonzero (nnz)				
Soil profile 1				
Elastic system		9,751,246	19,428,239	34,122,447
Elastoplastic system at 130kPa		9,765,468	19,443,665	34,131,365
Soil profile 2				
Elastic system		9,757,406	19,434,819	34,121,660
Elastoplastic system at 20kPa		9,781,416	19,467,142	34,143,209
nnz/N^2 (%)				
Soil profile 1				
Elastic system		0.32	0.17	0.1
Elastoplastic system at 130kPa		0.32	0.17	0.1
Soil profile 2				
Elastic system		0.32	0.17	0.1
Elastoplastic system at 20kPa		0.32	0.17	0.1

* *Mesh size x×y×z means x element in x direction, y element in y direction and z element in z direction*

Table 5.2: Effective parameters of the soil following a non-associated MC model

	Young's modulus E' (MPa)	Poisson's ratio v	Cohesion c' (kPa)	Friction angle Φ' (degree)	Dilation angle ψ (degree)	Saturated unit weight γ_{sat} (kN/m^3)	Permeability $k_x/\gamma_w = k_x/\gamma_w$ with γ_w = 10kN/m^3, Unit of k_x,k_y = m/s
Stiff clay	60	0.3	20	20	0	10	10^{-10}
Dense sand	105	0.3	1	30	5	10	10^{-6}

A uniformly distributed load is applied to the square footing. The applied load is a function of time and follows the ramp loading diagram in Figure 5.1b, in which q_{max} is the maximum applied load and t_0 is the maximum loading time. When the consolidation time t is less than t_0, the load increment Δf in Eq.(2.8) is added to the system every time step, Δt. When the consolidation time t exceeds t_0, the maximum assigned load has been reached and the load increment Δf in Eq.(2.8) is set to zero. The system is considered in the consolidation stage. The loading informations for soil profile 1 and 2 are summarized in Table 5.3. In this table, Δt is the time step in Eq.(2.11), *incs* denotes the number of load increments, and *nstep* denotes the number of time steps. *nstep* should be always greater or equal to *incs*.

Table 5.3: Loading information

	t_0 (s)	q_{max}	Number of load increment, *incs*	Time step Δt	Number of time step, *nstep*
Soil profile 1	1	130	20	0.05	20
Soil profile 2	1	20	20	0.05	20

Similar to the 1-by-1 block matrix in Section 4.2.1, the elastoplastic global stiffness matrix of Biot's consolidation analysis can be written as the summation of the elastic global stiffness matrix and a low-rank matrix as shown in Eq.(5.1). The changes of the global stiffness matrix in each NR iteration purely come from the changes of K_{ep}. This chapter will apply the recommendation in Section 4.2.1 to form the global elastoplastic stiffness matrix implicitly by forming the elastic global stiffness matrix once and update the Δ matrix at every NR iteration.

$$A = \begin{bmatrix} K_{ep} & B \\ B^T & -C \end{bmatrix} = \begin{bmatrix} K_e & B \\ B^T & -C \end{bmatrix} + \begin{bmatrix} \Delta & 0 \\ 0 & 0 \end{bmatrix} \tag{5.1}$$

In previous chapters, the degree of freedoms (d.o.fs) in the elastoplastic global stiffness matrix K_{ep} have always been ordered in natural order which results directly from FE formulation. For Biot's consolidation, when the d.o.fs are in natural order, the global stiffness matrix has the same pattern in 1-by-1 block matrix in Figure 2.4a. This chapter aims to exploit the block structure in Eq. (2.8) so the d.o.fs are re-ordered to obtain the pattern of 2-by-2 block matrix in Figure 2.4b. Natural ordering will also be used in the later part of this chapter when MSSOR and ILU0 preconditioner are compared with block preconditioners. This numerical experiment will show that the natural ordering is not as convenient as the block ordering when preconditioners are taken into account.

This chapter continues to use IDR(s) with s = 6, which has been shown in Chapter 3 to be the optimal value of s. The stopping criteria of IDR(6) and NR iteration are similar to previous chapters and are presented again in Eq.(5.2) and (5.3) respectively,

$$\frac{\left\|r^{(i)}\right\|_2}{\left\|r_0\right\|_2} \le i\_tol = 10^{-6} \text{ or } i \ge 5000 \tag{5.2}$$

$$\frac{\left\|F - K_{ep}u\right\|_2}{\left\|F\right\|_2} \le 10^{-6} \tag{5.3}$$

with $r^{(i)} = K_{ep}u^{(i)} - F$ and $r_0 = F$. IDR(6) is considered “Fail” when the number of matvec exceeding 5000. The values of matvec and total iteration time reported are average values over all the NR iterations in each load step. Total iteration time includes the time spent to form preconditioner and the time spent by iterative solver.

5.3 Comparison of preconditioners and effect of node ordering

5.3.1 Preconditioners derived from the 2-by-2 block ordering

The approximations of $\hat{K}$ and $\hat{S}$ in Eq.(5.4) and (5.5) are used in diagonal block preconditioner, M_d, and block constrained preconditioner M_c. The approximations are numbered from the crudest to the finest. The finer the approximation is, the more time it takes to form that approximation. The soil stiffness matrix K_{ep} in Eq.(2.8) is the same as the K_{ep} in drained analysis in Eq.(2.1). Hence the approximations of K_{ep} are taken as preconditioner for 1-by-1 block matrix discussed in Chapter 3 and Chapter 4. Section 3.4 and 3.5 have shown numerically that ILU0-K_{ep} is the most time efficient to precondition K_{ep} hence this section uses $\hat{K}_3 = \text{ILU0}(K_{ep})$ as the finest approximation of K_{ep}. The notation ‘ILU0(A)’ in this case denotes the incomplete LU factorization with zero fill-in of the matrix A inside the brackets. The notation ‘SSOR(A)’ denotes the SSOR preconditioner in Eq.(2.4) extracted from the matrix A with ω =1. The GJ preconditioner in Section 2.3.3.1 is termed here as $M_d(\hat{K}_1, \hat{S}_1)$.

$$\hat{K}_1 = \text{diag}(K_{ep})\,;\, \hat{K}_2 = \text{SSOR}\,(K_{ep})\,;\; \hat{K}_3 = \text{ILU0}(K_{ep}) \tag{5.4}$$

$$\begin{aligned} \hat{S}_1 &= \text{diag}(B^T \text{diag}(K_{ep})^{-1} B + C) \\ \hat{S}_2 &= \text{ILU0}\left(B^T \text{diag}(K_{ep})^{-1} B + C\right) \\ \hat{S}_3 &= \text{ILU0}\left(B^T \text{ILU0}(K_{ep})^{-1} B + C\right) \end{aligned} \tag{5.5}$$

Section 2.3.3.1 has presented the theorem from Phoon *et al.*[131] (2002) about the effect of α on the eigenvalue distribution of the 2-by-2 block matrix preconditioned by a diagonal block preconditioner. This section first will use α = –4 to compare M_d and M_c when $\hat{K}$ and $\hat{S}$ vary.

Table 5.4 and Table 5.5 present the matvec and total iteration time to solve the Biot's consolidation equation using M_d and M_c preconditioner at the final load step, $q_{\max}$. The time presented in bracket is the overhead time to extract the nessecary block matrices (such as block B in Eq.(2.8) for M_c preconditioner) and to form the preconditioner. Section 2.3.3 has reviewed that M_c is a better approximation of Eq.(2.8) than M_d hence IDR(6) preconditioned with M_c is expected to converge faster than when preconditioned with M_d. However more time may be required to form M_c as well as to perform the preconditioning step.

In Table 5.4 and Table 5.5, for each problem size, the matvec reduces from left to right and from top to bottom when the approximations $\hat{K}$ and $\hat{S}$ change from the crudest to the finest. Among all the combination of $\hat{K}$ and $\hat{S}$, $M_d(\hat{K}_1, \hat{S}_1)$ is the crudest approximation of A hence requires the most matvec while $M_c(\hat{K}_3, \hat{S}_3)$ is the finest approximation of A hence requires the least matvec. Botchev and Golub[30] (2006) recommended the use of $\hat{K}_2$ in M_c preconditioner when K_{ep} in the 2-by-2 block matrix A is nonsymmetric. However the numerical results show that even with the finest approximation of S, $M_c(\hat{K}_2, \hat{S}_3)$ does not achieve less matvec than $M_c(\hat{K}_3, \hat{S}_3)$ or less total iteration time than $M_d(\hat{K}_3, \hat{S}_1)$. With the same $\hat{S}$, the matvec count reduces greatly when $\hat{K}$ changes from $\hat{K}_1$ to $\hat{K}_3$. While with the same $\hat{K}$, the matvec count only reduces minimally when $\hat{S}$ changes from $\hat{S}_1$ to $\hat{S}_3$. This shows that a good approximation of K_{ep} is more crucial to the efficiency of a preconditioner than a good approximation of *S*. The reason may be because K_{ep} is a major block in the 2-by-2 block matrix *A*. Table 5.1 shows that the size of K_{ep} submatrix is more than 90 percent of the matrix *A*.

Table 5.4: Comparison of diagonal block preconditioner M_d and constrained block preconditioner M_c. Time presented in brackets is overhead time including time required to form preconditioners and extracting required block matrices. Soil profile 1 is used. Results are reported at the last load step.

16×16×16		$\hat{K}_1$		$\hat{K}_2$		$\hat{K}_3$	
$\hat{S}_1$	M_d	$\boxed{1291}$	32.1 (0.1)	274	17.5 (1.3)	$\boxed{201}$	**12.8** (1.9)
	M_c	$\boxed{1370}$	39.5 (0.1)	216	23.2 (1.3)	$\boxed{255}$	25.2 (2.0)
$\hat{S}_2$	M_d	1273	$\boxed{30.8}$ (6.5)	303	$\boxed{27.3}$ (9.4)	$\boxed{173}$	21.3 (10.9)
	M_c	712	$\boxed{29.1}$ (8.5)	210	$\boxed{19.2}$ (5.7)	$\boxed{177}$	27.5 (10.6)
$\hat{S}_3$	M_d	1173	235.9 (177.3)	264	156.3 (139.9)	168	243.5 (227.1)
	M_c	Fail	–	199	159.7 (142.1)	**105**	280.5 (272.5)
20×20×20		$\hat{K}_1$		$\hat{K}_2$		$\hat{K}_3$	
$\hat{S}_1$	M_d	1896	63.4 (0.1)	346	43.2 (2.8)	$\boxed{262}$	**22.7** (3.7)
	M_c	1835	106.9 (0.2)	275	38.2 (2.2)	$\boxed{336}$	67.8 (4.6)
$\hat{S}_2$	M_d	1606	83.0 (23.1)	428	$\boxed{83.1}$ (33.0)	$\boxed{223}$	59.0 (34.1)
	M_c	909	88.5 (32.9)	272	$\boxed{68.9}$ (25.4)	$\boxed{313}$	94.1 (36.4)
$\hat{S}_3$	M_d	1542	1379.3 (1083.3)	344	773.7 (716.8)	250	904.3 (857.1)
	M_c	Fail	–	234	1196.8 (1117.4)	**129**	1170.8 (1127.5)
24×24×24		$\hat{K}_1$		$\hat{K}_2$		$\hat{K}_3$	
$\hat{S}_1$	M_d	$\boxed{2708}$	221.6 (0.2)	493	113.9 (4.5)	338	**79.8** (7.4)
	M_c	$\boxed{2769}$	229.0 (0.2)	376	130.3 (4.6)	315	106.7 (7.6)
$\hat{S}_2$	M_d	2275	229.3 (72.6)	554	$\boxed{231.7}$ (94.6)	$\boxed{322}$	123.6 (76.2)
	M_c	1367	241.1 (95.8)	388	$\boxed{222.2}$ (105.4)	$\boxed{410}$	226.1 (94.5)
$\hat{S}_3$	M_d	2095	3934.4 (3198.1)	438	3670.2 (2569.6)	311	3649.9 (2561.7)
	M_c	Fail	–	348	3208.7 (3034.9)	**197**	3164.3 (3074.7)

Table 5.5: Comparison of diagonal block preconditioner M_d and constrained block preconditioner M_c. Time presented in brackets is overhead time including time required to form preconditioners and extracting required block matrices. Soil profile 2 is used. Results are reported at the last load step.

$16\times16\times16$		$\hat{K}_1$		$\hat{K}_2$		$\hat{K}_3$	
$\hat{S}_1$	M_d	1155	26.3 (0.0)	276	13.3 (1.1)	$\boxed{174}$	**10.4** (2.0)
	M_c	1145	30.5 (0.0)	218	20.8 (1.2)	$\boxed{179}$	17.8 (2.0)
$\hat{S}_2$	M_d	1113	$\boxed{35.3}$ (8.6)	309	20.7 (7.0)	145	17.0 (9.6)
	M_c	630	$\boxed{17.8}$ (5.7)	215	28.6 (8.9)	114	21.4 (10.7)
$\hat{S}_3$	M_d	1051	296.7 (228.8)	252	190.6 (171.8)	125	190.0 (180.6)
	M_c	Fail	–	200	243.7 (217.9)	**73**	229.6 (220.7)
$20\times20\times20$		$\hat{K}_1$		$\hat{K}_2$		$\hat{K}_3$	
$\hat{S}_1$	M_d	$\boxed{1771}$	73.0 (0.1)	371	32.2 (2.2)	282	**24.6** (3.7)
	M_c	$\boxed{1879}$	99.6 (0.2)	282	40.9 (2.2)	208	30.2 (3.7)
$\hat{S}_2$	M_d	1597	57.0 (16.8)	437	$\boxed{59.7}$ (23.4)	267	46.9 (26.2)
	M_c	885	79.3 (30.9)	294	$\boxed{58.2}$ (20.5)	209	51.8 (25.9)
$\hat{S}_3$	M_d	1493	1121.0 (892.2)	341	705.16 (760.5)	250	867.4 (821.7)
	M_c	Fail	–	260	881.5 (817.7)	**108**	860.3 (835.4)
$24\times24\times24$		$\hat{K}_1$		$\hat{K}_2$		$\hat{K}_3$	
$\hat{S}_1$	M_d	$\boxed{2377}$	193.3 (0.1)	490	73.7 (4.0)	274	**53.4** (6.8)
	M_c	$\boxed{2445}$	241.2 (0.3)	341	116.4 (4.5)	262	80.5 (6.8)
$\hat{S}_2$	M_d	2068	$\boxed{285.7}$ (99.7)	559	$\boxed{229.7}$ (106.9)	299	109.1 (70.9)
	M_c	1154	$\boxed{195.0}$ (88.1)	535	$\boxed{200.8}$ (69.6)	290	177.9 (94.2)
$\hat{S}_3$	M_d	1810	4654.3 (3829.9)	454	3874.2 (3645.4)	272	4516.8 (3318.4)
	M_c	Fail	–	355	5017.0 (4742.6)	**130**	3727.7 (3650.2)

With the same combination of $\hat{K}$ and $\hat{S}$, the matvec required by M_c is smaller than that required by M_d, which agrees with the prediction. There are cases – numbers in box – that M_c requires more matvec than M_d. The differences in matvec count, which are marginal, can be due to round-off error. Although M_c requires less matvec than M_d, the total iteration time required by M_c is often more than that required by M_d. This is expected because the forming process and the preconditioning step of M_c take more time than that of M_d. There are cases that the total iteration time required by M_c is less than that required by M_d. These cases are marked as boxed numbers in the total iteration time columns in Table 5.4 and Table 5.5. This happens because there is significant reduction of matvec when M_c is used in compared with when M_d is used, which leads to the reduction in total iteration time. This reduction of matvec often occurs with $M_c(\hat{K}_2, \hat{S}_2)$, which belongs to the class of preconditioner recommended by Botchev and Golub[30] (2006). Although in general the use of

$\hat{K}_2$ does not offer the least matvec count or the least total iteration time, the use of $M_c(\hat{K}_2, \hat{S}_2)$ is more efficient than other combinations of $\hat{K}$ and $\hat{S}$ because $M_c(\hat{K}_2, \hat{S}_2)$ succeeds in reducing the matvec count and the total iteration time when compared with $M_d(\hat{K}_2, \hat{S}_2)$.

While the iteration time is controlled by the preconditioner, which M_d has been shown to be more time efficient than M_c, the time to form the preconditioner depends greatly on $\hat{K}$ and $\hat{S}$. With the same $\hat{S}$, the time to form M_d or M_c increases when $\hat{K}$ varies from $\hat{K}_1$ to $\hat{K}_3$ as expected but the increase is minimal. On the contrary, with the same $\hat{K}$, the time to form M_d or M_c increases drastically when $\hat{S}$ varies from $\hat{S}_1$ to $\hat{S}_3$. $\hat{S}_3$ is the closet to S and the time to form preconditioners involving $\hat{S}_3$ requires more than 400 times the time to form those involving $\hat{S}_1$. Previous paragraph in this section has discussed that a better $\hat{K}$ is more crucial than a better $\hat{S}$ because the reduction in matvec due to a better $\hat{S}$ is very minimal. Hence a fine approximation of S reduces the iteration time minimally but requires an extensive time to form. Among all of the combination of $\hat{K}$ and $\hat{S}$, $M_d(\hat{K}_3, \hat{S}_1)$ requires the least total iteration time hence with a good approximation of K_{ep} like $\hat{K}_3$, a simple approximation of S like $\hat{S}_1$ is sufficient.

Besides, Table 5.4 and Table 5.5 show that $M_c(\hat{K}_1, \hat{S}_3)$ fails to converge for all the tested cases. The reason for this problem is still unknown. Table 5.6 shows that this problem can be mitigated by increasing the value of s. However, there are two issues with this mitigation. First, although IDR(*s*) converges with large value of *s*, the matvec and the total iteration time are large as well. The matvec reduces minimally with a large increase of *s* therefore it is actually not beneficial to increase *s*. Second, this minimum value of *s* to obtain convergence grows when the size of the linear system increases and cannot be predicted in general case. Nevertheless, the problem associated with $M_c(\hat{K}_1, \hat{S}_3)$ is not relevant to this study because previous paragraph has discussed that $\hat{S}_3$ is not practical to use hence $M_c(\hat{K}_1, \hat{S}_3)$ is not used in further discussion.

Table 5.6: Matrix-vector multiplications required by IDR(*s*) preconditioned with $M_c(\hat{K}_1, \hat{S}_3)$. Soil profile 2 is used. The applied pressure is 3kPa when yielded Gauss points first appear and the linear system becomes nonsymmetric.

	Mesh size		
	16×16×16	20×20×20	24×24×24
s = 100	Fail	Fail	Fail
s = 200	2506	Fail	Fail
s = 400	2205	Fail	Fail
s = 500	–	2876	Fail
s = 600	–	–	4635

Figure 5.2 to Figure 5.7 plot the comparison of M_d and M_c at every time step. These figures agree with conclusions from Table 5.4 and Table 5.5. Whenever $\hat{S}_3$ is involved, the total iteration time always increases significantly. Among all the cases tested and at every time step, $M_d(\hat{K}_3, \hat{S}_1)$ requires the least total iteration time. Hence $M_d(\hat{K}_3, \hat{S}_1)$ is the most efficient block preconditioner for the nonsymmetric 2-by-2 block matrix from Biot's consolidation analysis.

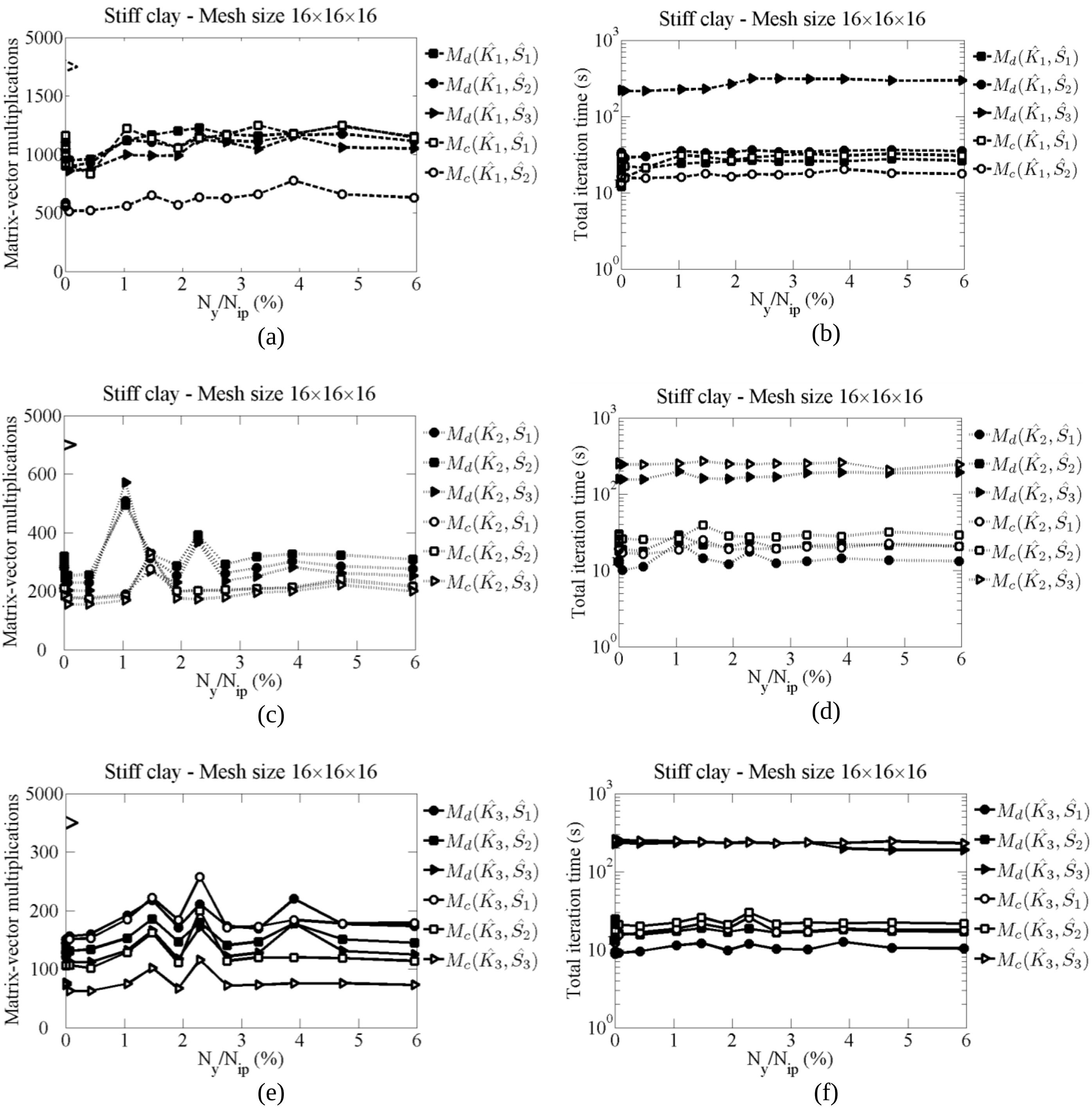


Figure 5.2: Comparison of M_d and M_c with variation of approximations of K_{ep} and S. Mesh size of 16×16×16 and soil profile 1 is used.

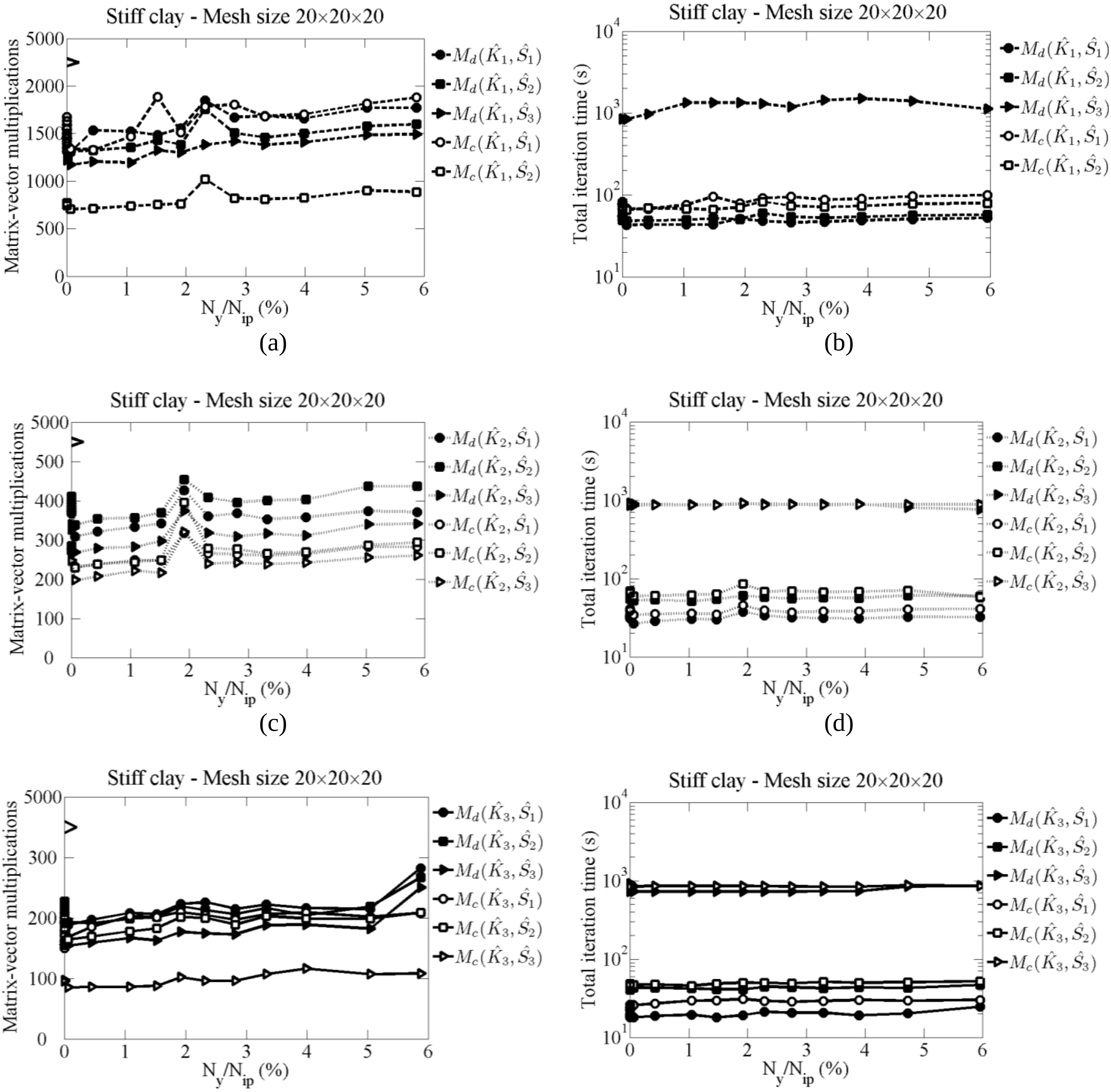


Figure 5.3: Comparison of M_d and M_c with variation of approximations of K_{ep} and S. Mesh size of 20×20×20 and soil profile 1 is used.

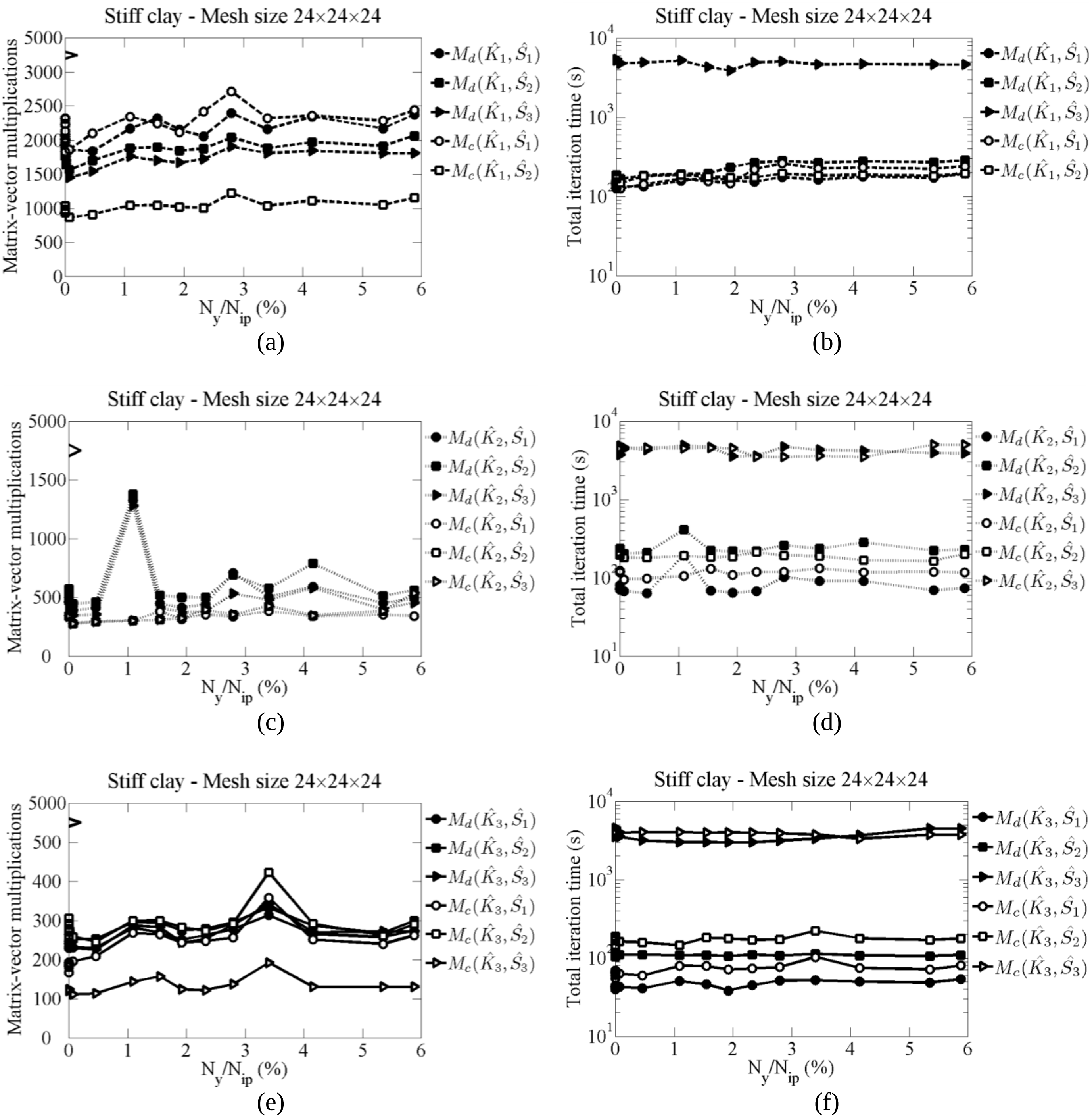


Figure 5.4: Comparison of M_d and M_c with variation of approximations of K_{ep} and S. Mesh size of 24×24×24 and soil profile 1 is used.

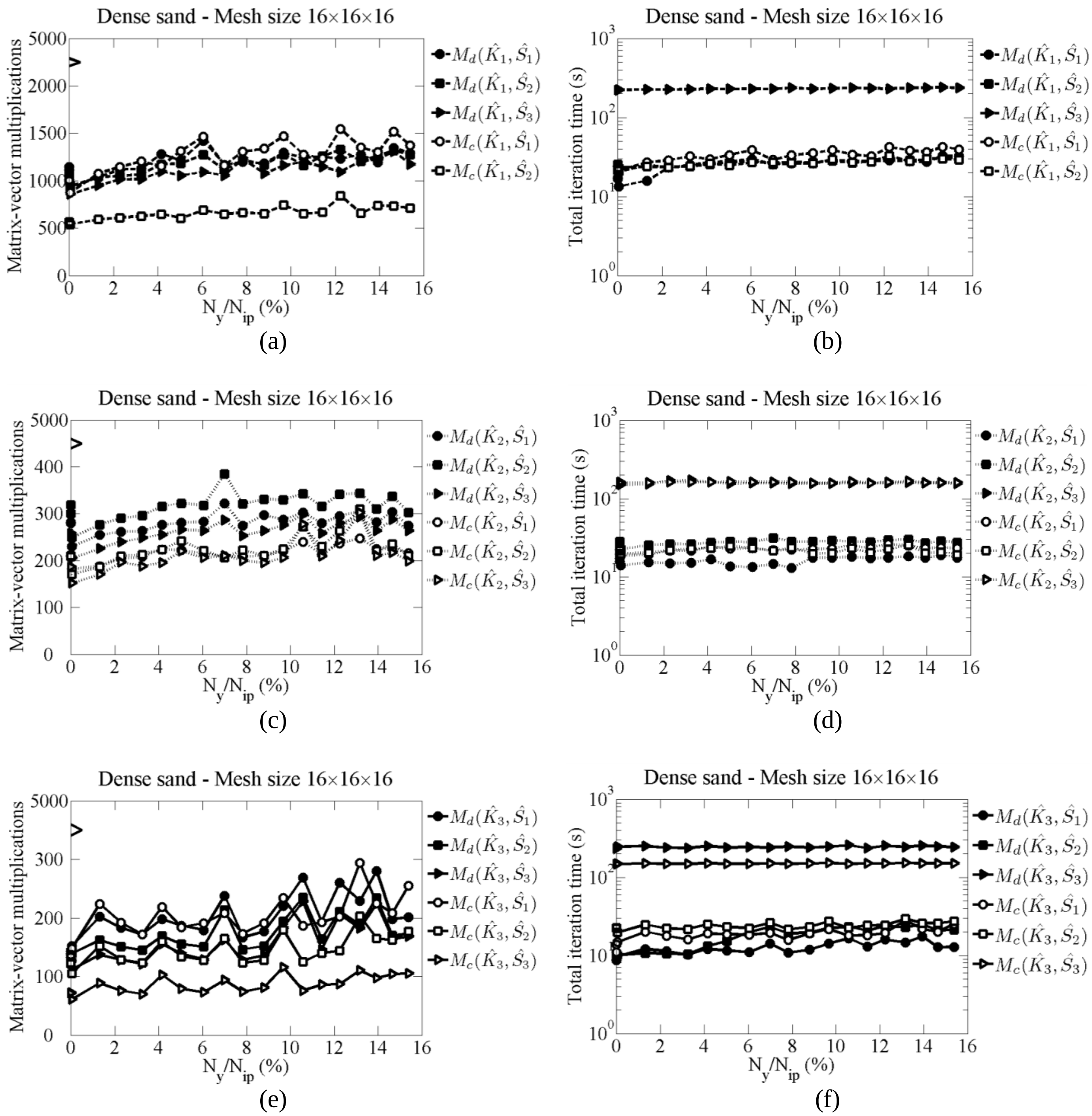


Figure 5.5: Comparison of M_d and M_c with variation of approximations of K_{ep} and S. Mesh size of 16×16×16 and soil profile 2 is used.

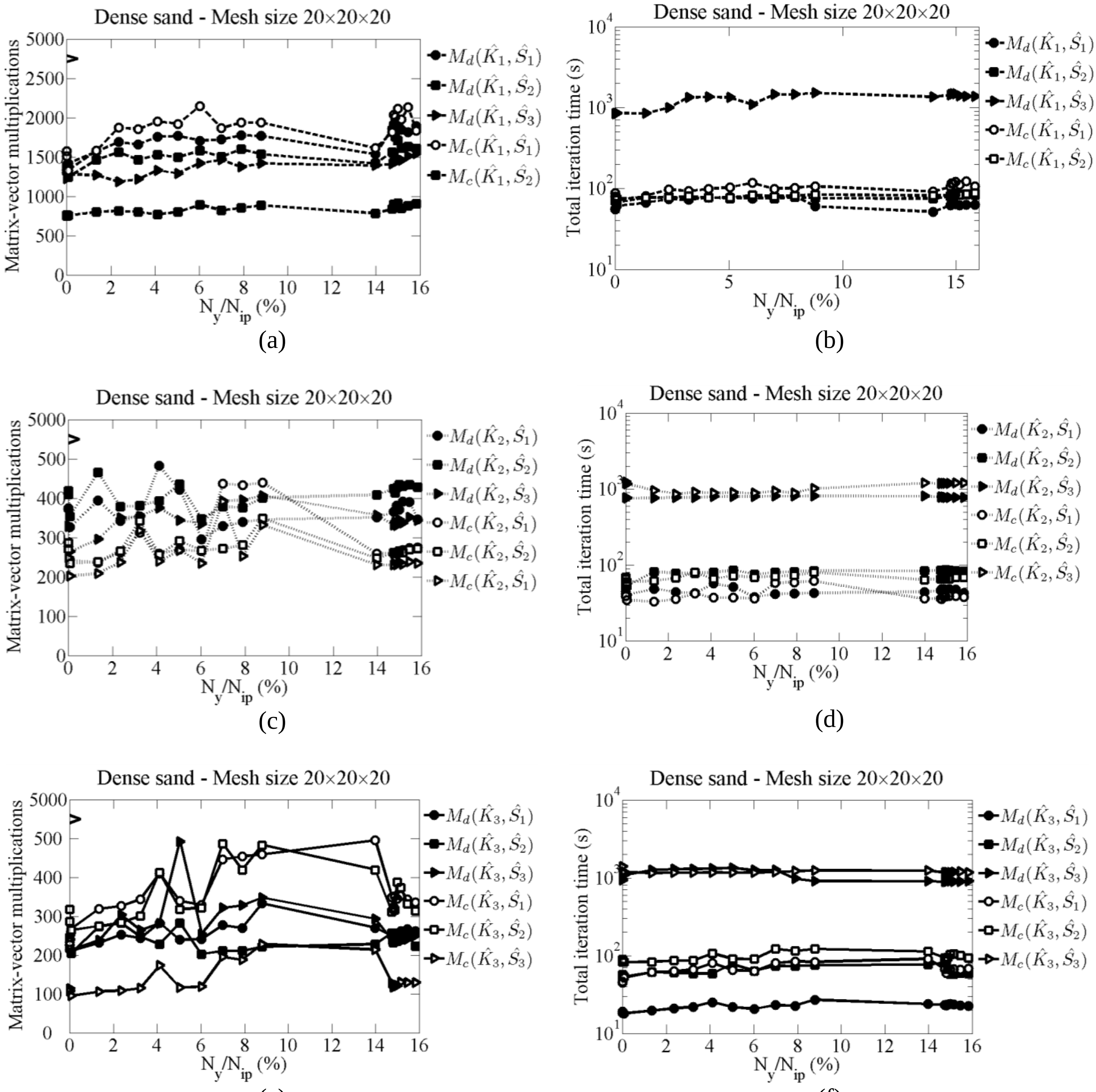


Figure 5.6: Comparison of M_d and M_c with variation of approximations of K_{ep} and S. Mesh size of 20×20×20 and soil profile 2 is used.

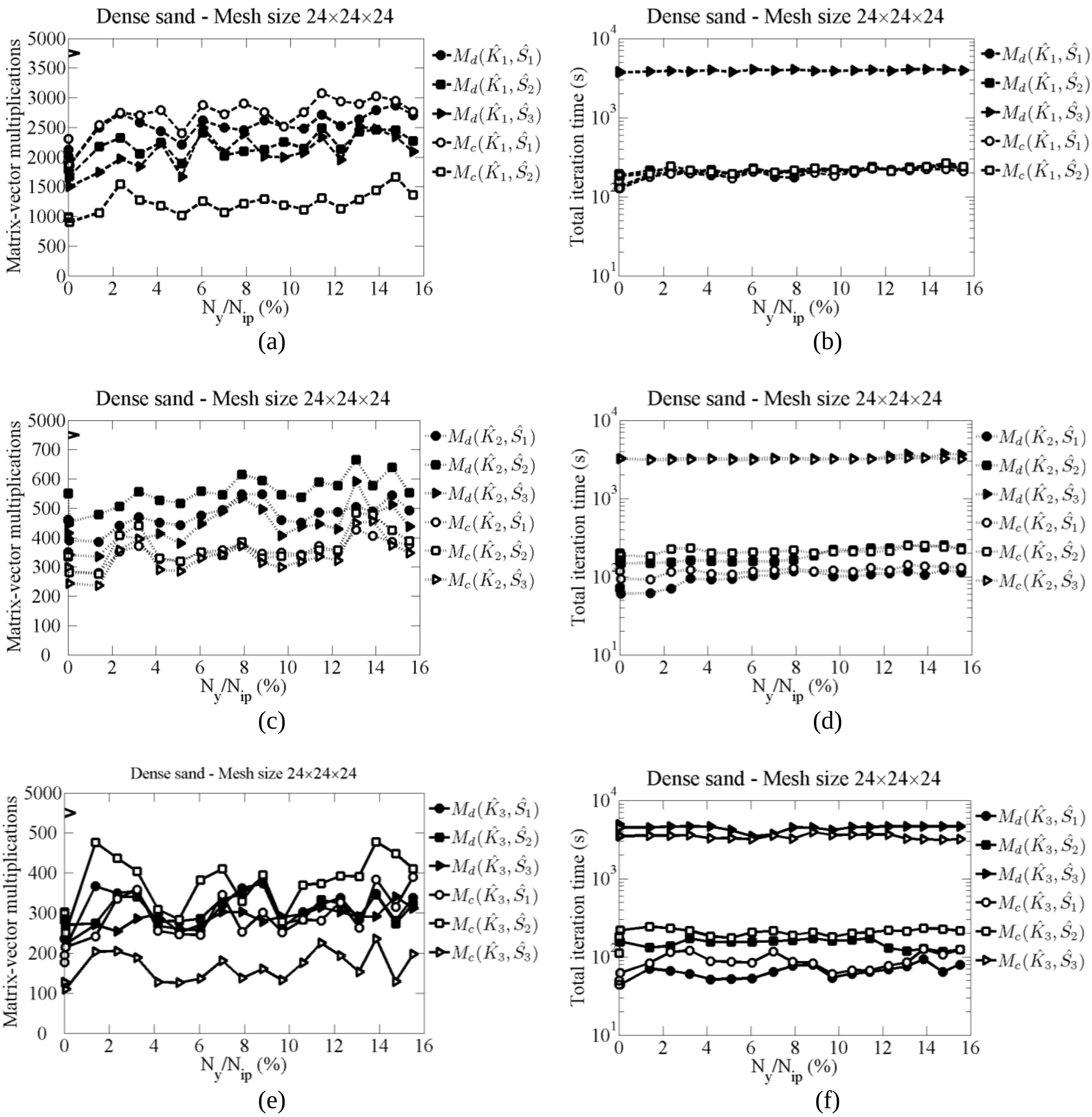


Figure 5.7: Comparison of M_d and M_c with variation of approximations of K_{ep} and S. Mesh size of 24×24×24 and soil profile 2 is used.

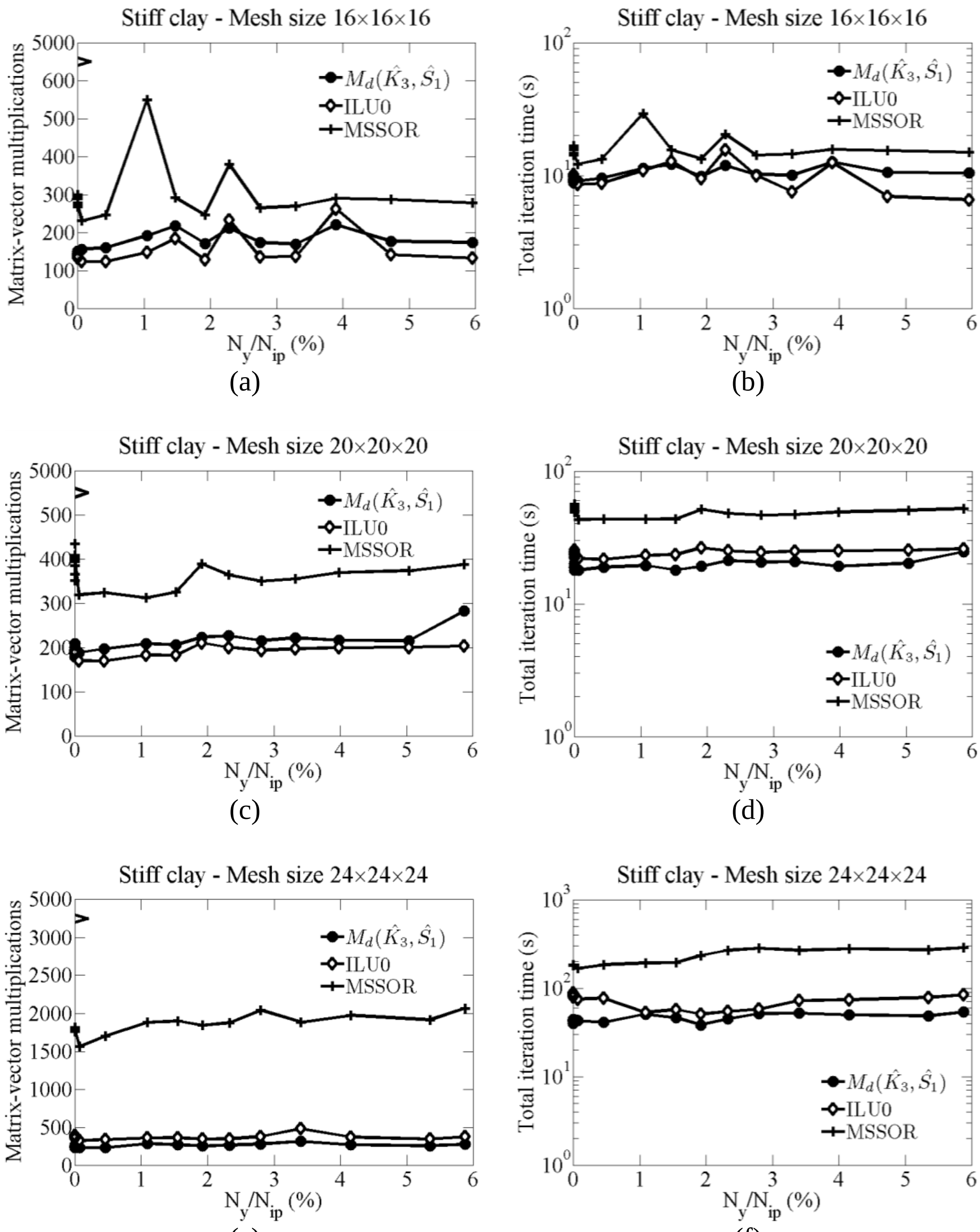


Figure 5.8: Comparison of $M_d(\hat{K}_3, \hat{S}_1)$ versus MSSOR and ILU0. Soil profile 1 is used.

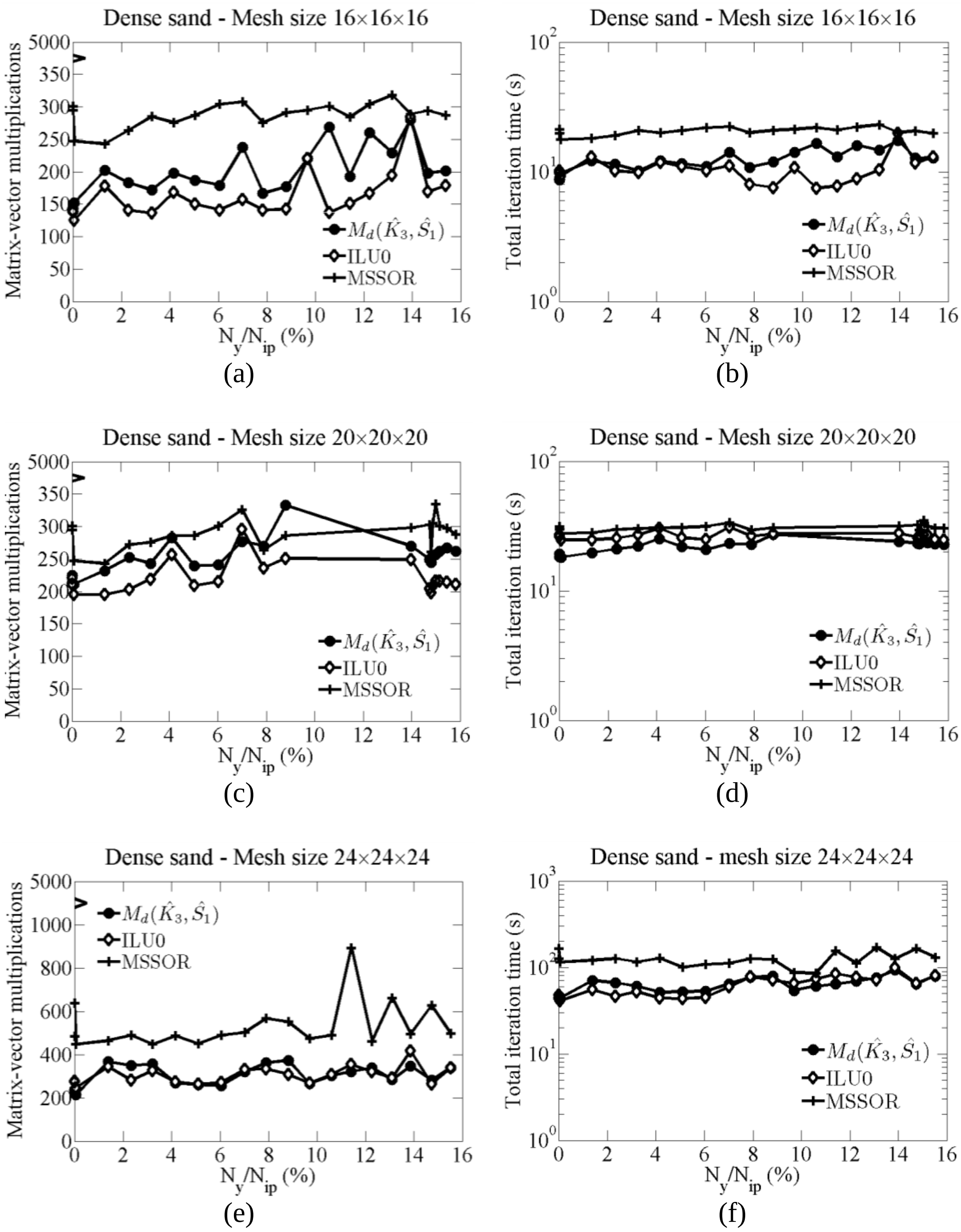


Figure 5.9: Comparison of $M_d(\hat{K}_3, \hat{S}_1)$ versus MSSOR and ILU0. Soil profile 2 is used.

$M_d(\hat{K}_3, \hat{S}_1)$ is compared with MSSOR and ILU0 in Figure 5.8 and Figure 5.9. MSSOR and ILU0 in these cases are formed when the global stiffness matrix A is in the 2-by-2 block form. MSSOR is a cheap preconditioner but always require more matvec and total iteration time than ILU0 and $M_d(\hat{K}_3, \hat{S}_1)$. ILU0 shows competitive performance with $M_d(\hat{K}_3, \hat{S}_1)$ especially when the soil profile 1 is used. However in general, $M_d(\hat{K}_3, \hat{S}_1)$ requires either less or equal total iteration time than ILU0 hence this section concludes that $M_d(\hat{K}_3, \hat{S}_1)$ is the most efficient preconditioner for the nonsymmetric 2-by-2 block matrix from Biot's consolidation analysis. We note that $\hat{K}_3$ is ILU0 of K_{ep}.

5.3.2 Preconditioners derived from the natural ordering

Section 5.3.1 has used the block ordering form of the global stiffness matrix to exploit the block preconditioners. However block ordering requires overhead time to order the unknowns. Natural ordering is the order resulting directly FE analysis and does not require any overhead time to form. ILU0 and MSSOR treat the matrix as 1-by-1 block matrix hence they are still applicable when the global stiffness matrix is in natural ordering. Figure 5.10 plots the comparison of ILU0 and MSSOR when the global stiffness matrix is in the block ordering and natural ordering.

Figure 5.10 shows that ILU0 is more effective than MSSOR when the soil profile 2 is used. However ILU0 has difficulty to converge when soil profile 1, with low permeability, is used. Figure 5.11 shows the typical relative residual of IDR(6) preconditioned with ILU0 when the soil profile 1 is used. This can be because ILU0 is unstable when the permeability is low. This problem was also observed by Chauhary (2010) when the soil follows the linear elastic model. Although MSSOR is more robust than ILU0, its performance in natural ordering is not as stable as in block ordering. This is because when the matrix is in natural ordering, the stress returning process tends to return the Gauss points to the apex and forces the stress-strain matrix become zero, which makes the global stiffness matrix more ill-conditioned and MSSOR requires more matvec to converge. Hence it is more convenient to arrange the stiffness matrix in block ordering and exploit the block preconditioner.

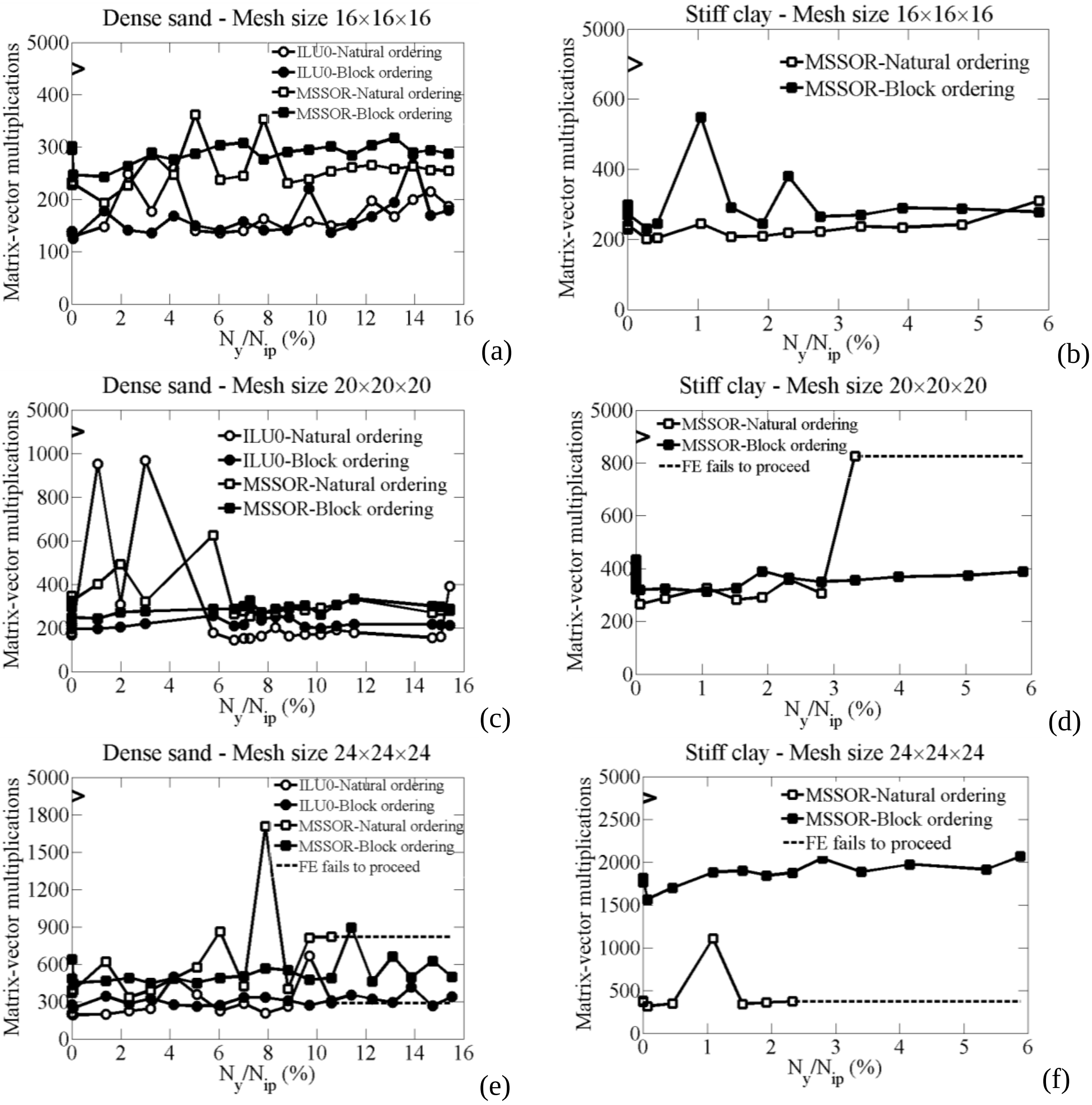


Figure 5.10: The effect of node ordering in the global stiffness matrix on ILU0 and MSSOR preconditioner

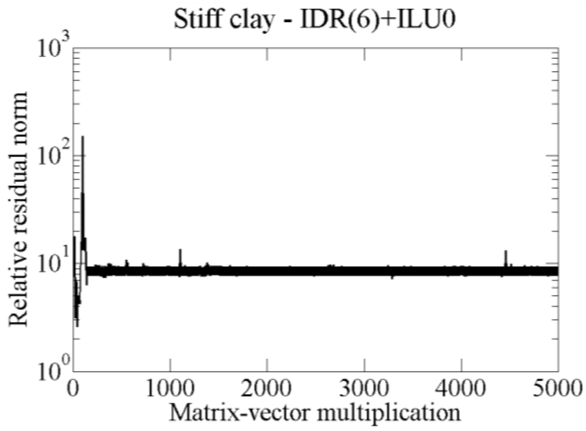


Figure 5.11: Typical relative residual norm of an unstable ILU0 preconditioner when the global stiffness matrix is in natural ordering and soil profile 1 is used.

5.3.3 Eigenvalue distribution

The eigenvalue distribution of the elastic global stiffness matrix and the elastoplastic global stiffness matrix are plotted in Figure 5.12 (a) and (b) respectively. The global stiffness matrix has the size of 7160 in which block K_{ep} has the size of 6512 and block C has the size of 648. The elastic global stiffness matrix is indefinite symmetric hence all the eigenvalues are real numbers, in which 6512 are positive and 648 are negative. The global stiffness matrix is nonsymmetric hence there are complex eigenvalues, which makes the ellipse circumscribes this eigenspectrum bigger than that of the elastic stiffness matrix. Figure 5.13a shows that the maximum and minimum real part of the eigenvalues do not change when the number of yielded Gauss points increases. Hence the increase in the imaginary part alone makes the eigenvalue distribution larger. Figure 5.13b shows that the condition number of matrix X increases when the number of yielded Gauss points increases. This implies the global stiffness matrix becomes more nonsymmetric. This observation agrees with those seen in the drained analysis.

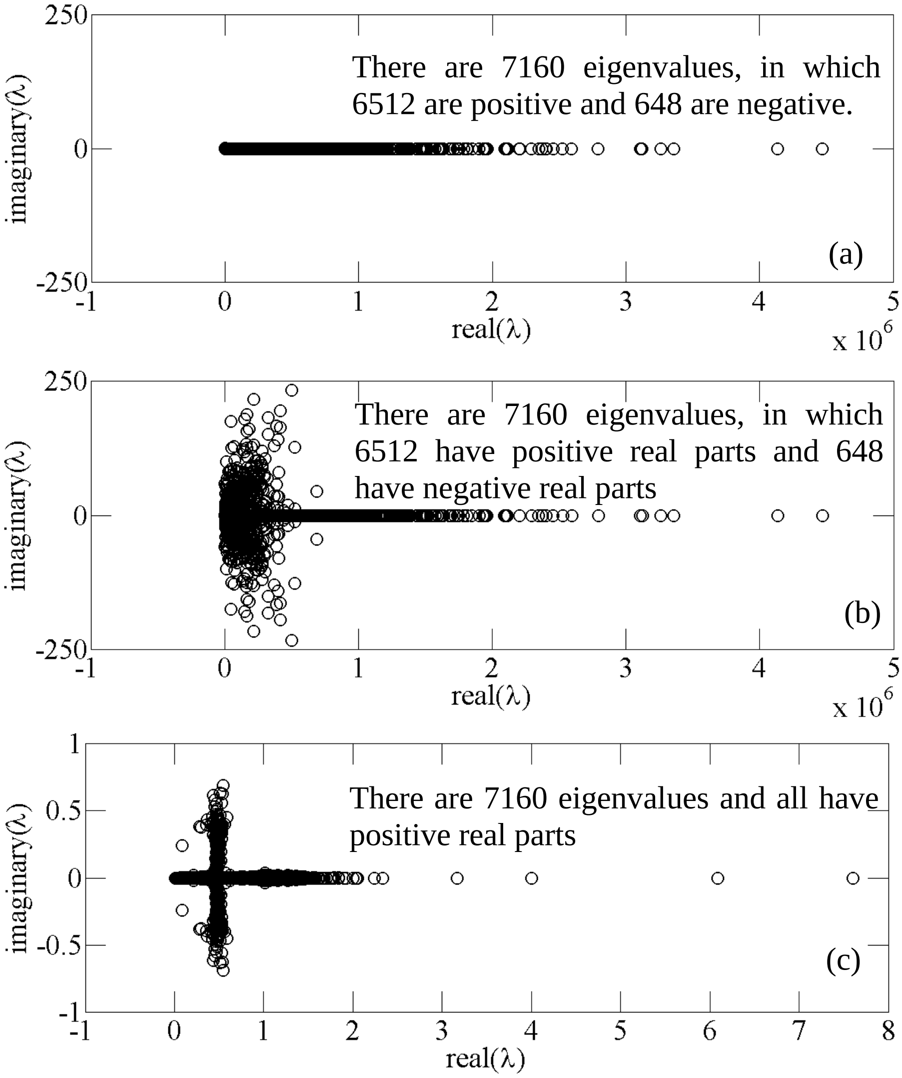
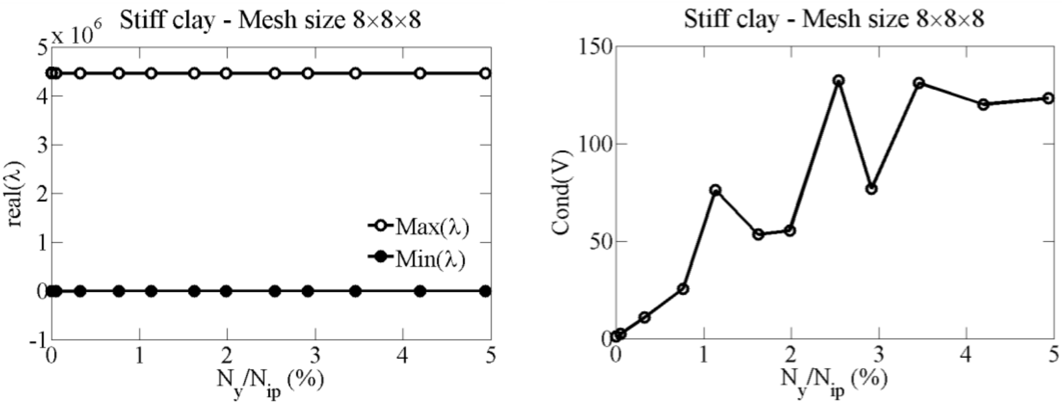


Figure 5.12: Eigenspectrum of: (a) the elastic global stiffness matrix; (b) the elastoplastic global stiffness matrix the final load step of 130kPa; (c) the elastoplastic global stiffness matrix preconditioned with $M_d(\hat{K}_3, \hat{S}_1)$. Soil profile 1 is used with the 8×8×8 FE mesh.



(a) (b)

Figure 5.13: Characteristics of eigenspectrum: (a) Maximum and minimum eigenvalue; (c) Condition number of matrix X (Eq.(2.30)). Soil profile 1 is used with the 8×8×8 FE mesh.

Figure 5.12c shows the eigenvalue distribution of the elastoplastic global stiffness matrix when it is preconditioned with $M_d(\hat{K}_3, \hat{S}_1)$ with $\alpha = -4$. The theorem in Section 2.3.3.1 states that when $\alpha = -4$, the eigenvalues will cluster around two point ½ and 1 with the radius of $O(\|S^{-1}C\|)$, which is reflected in Figure 5.12c. The soil profile 2 with $k_x/\gamma_w = 10^{-6}$ hence the radius $O(\|S^{-1}C\|)$ is small. The range of the eigenvalues has been reduced significantly when compared with the distribution in Figure 5.12b, which reflects the efficiency of the preconditioner $M_d(\hat{K}_3, \hat{S}_1)$.

5.4 Undrained analysis with 2-by-2 block matrix

Section 4.3.2 has shown that the ILU0 preconditioner is unstable to solve the undrained analysis in the conventional way. Biot's consolidation is the general equation of soil-water interation hence undrained analysis can be simulated by tuning the product $k_w\Delta t$ to a very small value like 10^{-14}. k_w is kept constant for soil profile 1 and 2 while Δt is varied. The theoretical undrained failure loads of the square footing are presented in Table 5.7. The numerical simulations load the square footing up to 90 percent of the theoretical failure. The dilation angles of both of the soil profiles are manually set to 0 as explained in Section 4.3.1.

Table 5.7: Ultimate bearing capacity of square footing q_f (kPa)

	Drained	Undrained
Soil 1 – Dense sand	48.54	7.17
Soil 2 – Stiff clay	424.68	138.31

When $k_w\Delta t$ is small, the value of $O(\|S^{-1}C\|)$ in the theorem in Section 2.3.3.1 is small and the eigenvalues will be clustered closer hence $\alpha = -4$ is efficient in making the eigenvalues cluster at two points. This section will use $\alpha = -4$ for the preconditioner $M_d(\hat{K}_3, \hat{S}_1)$. Figure 5.14 plots the matvec required by IDR(6) preconditioned with $M_d(\hat{K}_3, \hat{S}_1)$. The matvec varies in small ranges when $k_w\Delta t$ changes. Figure 5.15 shows the the excess pore pressure right below the square footing when $k_w\Delta t$ and the mesh change. The excess pore pressure is close to the applied load, which reflects the undrained behaviour of the system. However, the accuracy of the study does not only lie in the product of $k_w\Delta t$ but also in the mesh convergence. When the system is close to failure, there are fluctuations in excess pore pressure in the coarse mesh (16×16×16). Figure 5.14 and Figure 5.15 show that undrained analysis can be simulated with Biot's consolidation equations and $M_d(\hat{K}_3, \hat{S}_1)$ is an efficient preconditioner for this nonsymmetric linear system.

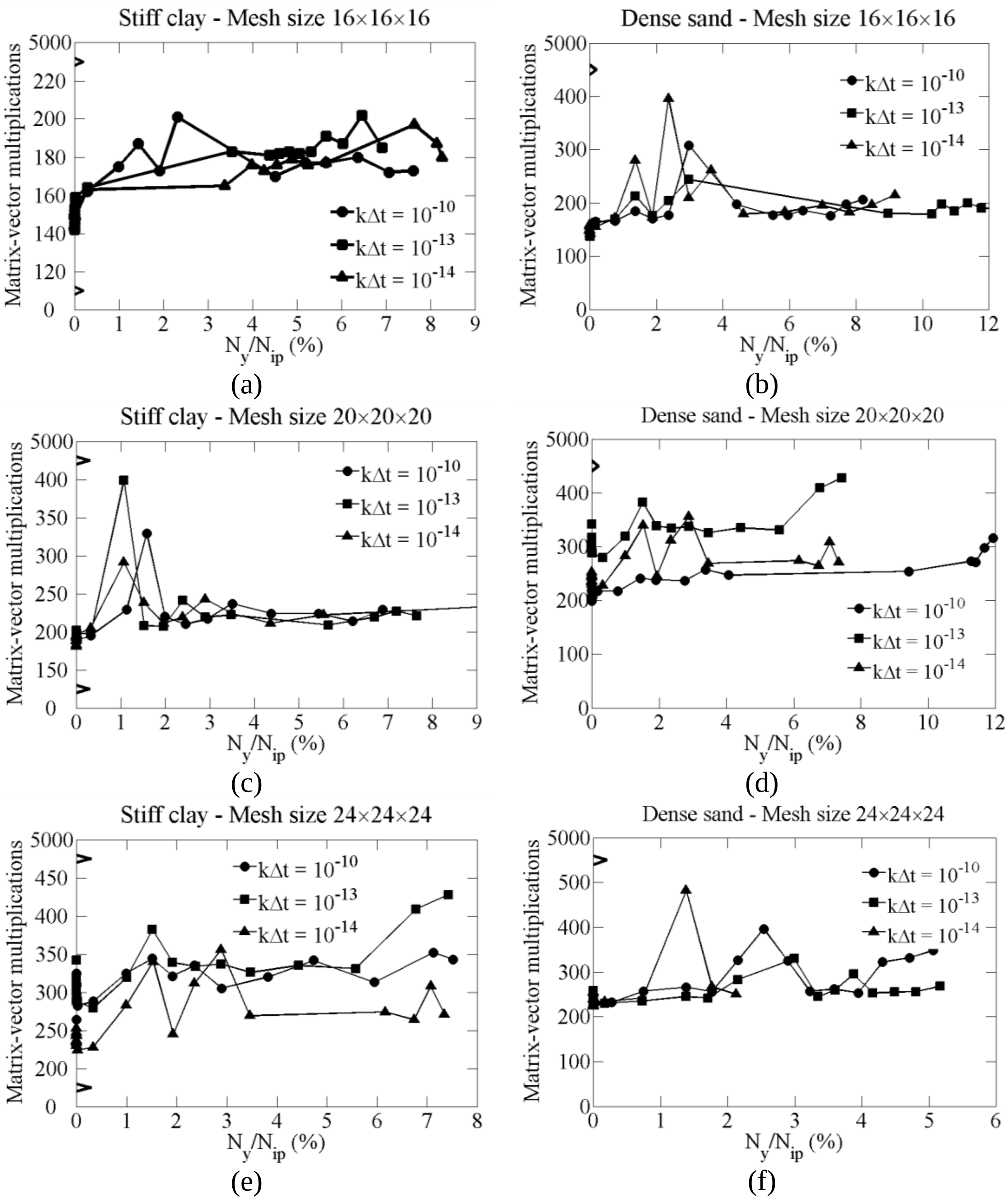


Figure 5.14: Effect of $k\Delta t$ on the convergence of IDR(6) + $M_d(\hat{K}_3, \hat{S}_1)$

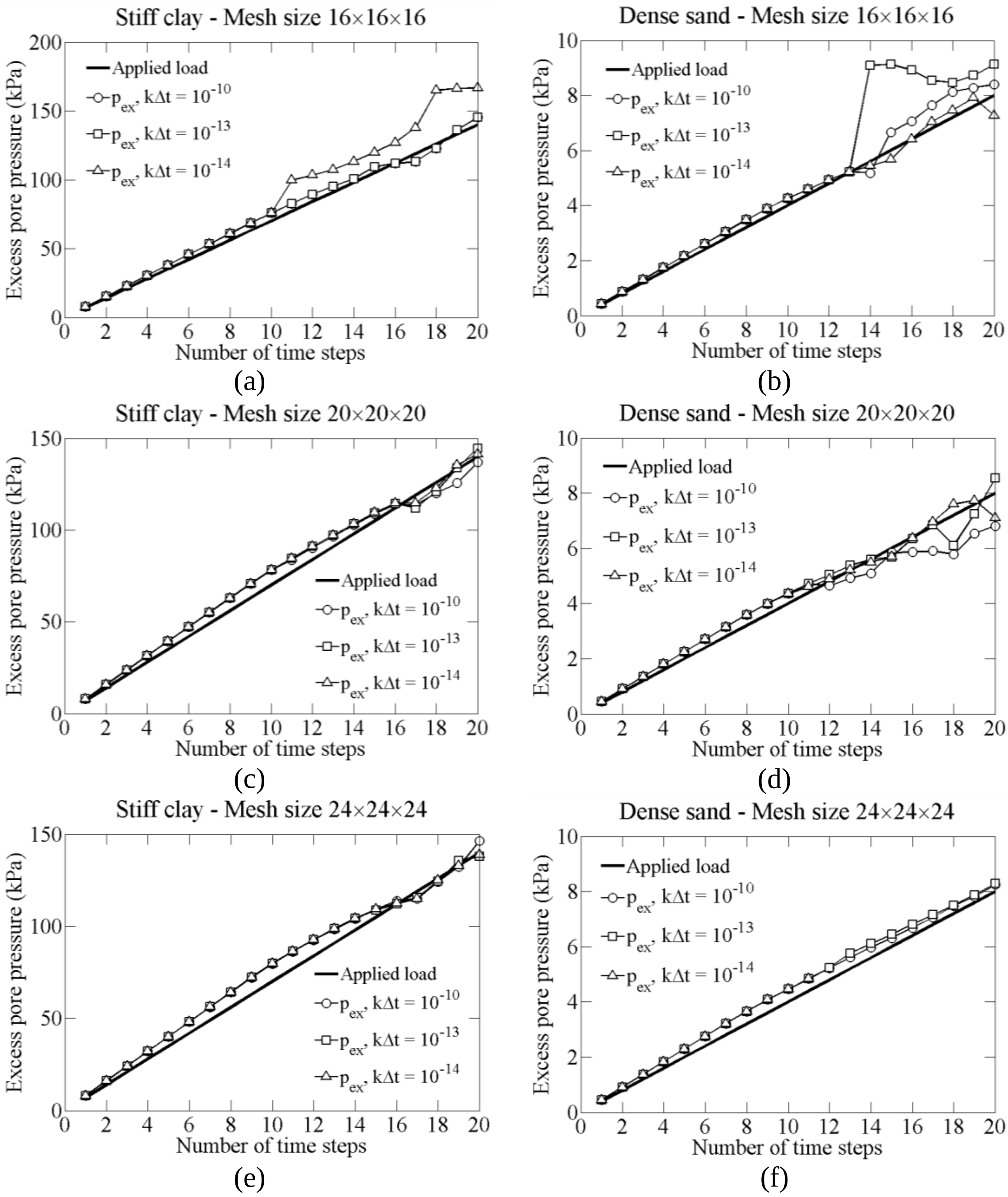


Figure 5.15: Excess pore pressure at the point right below the square footing

5.5 Applying the preconditioner updating schemes in Section 4.2.2

Section 4.2.2 has presented several preconditioner updating schemes to reduce the total simulation time when solving the 1-by-1 nonsymmetric block matrix. Section 4.2.2 recommended either using the preconditioner which is extracted from the elastic global stiffness matrix or updating the preconditioner at the beginning of the new load step. This section applies these two schemes on the 2-by-2 block matrix to observe their efficiency. Table 5.8 summarizes the preconditioner updating schemes and the notation of block preconditioners used in this section.

Table 5.8: Different schemes to update ILU0 preconditioner during the simulation

Number	Description	Preconditioner notation
1	The default scheme which updates preconditioner at every NR iteration.	$M_d(\hat{K},\hat{S})$-EP $M_c(\hat{K},\hat{S})$-EP
2	The scheme discussed in Section 4.2.2.1 which forms the preconditioner from K_e once at the beginning of the simulation.	$M_d(\hat{K},\hat{S})$-E $M_c(\hat{K},\hat{S})$-E
3	Update the preconditioner once in each load step right after the load increment is applied.	$M_d(\hat{K},\hat{S})$-NR $M_c(\hat{K},\hat{S})$-NR

Section 5.3.1 has shown numerically that $M_d(\hat{K}_3,\hat{S}_1)$ is the most time efficient in preconditioning the 2-by-2 block matrix resulted from Biot's consolidation analysis. This conclusion is drawn when the first updating scheme, which updates $M_d(\hat{K}_3,\hat{S}_1)$–EP at every NR iteration. When the second or third updating sheme is applied, the same preconditioner is used for several NR iterations hence it may be more advantage to use a better preconditioner at the beginning. Hence M_c is used to compare with $M_d(\hat{K}_3,\hat{S}_1)$–EP. This section continues using $\hat{K}_3$ as the approximation of K_{ep} because $\hat{K}_3$ has been shown many times in this thesis to be an efficient preconditioner for K_{ep}. $\hat{S}_3$ is not practical to use because it takes very long time to form hence $\hat{S}_1$ and $\hat{S}_2$ are considered in this section.

Figure 5.16 and Figure 5.17 show the comparison of the three updating schemes in Table 5.8. Section 5.3.1 has commented that M_c approximates the global stiffness matrix better hence it always requires less matvec than M_d does but the reduction in matvec is minimal while the time to form M_c and the preconditioning step are more than those required by M_d. Figure 5.16 and Figure 5.17 show the same trend: although the same M_c is used for the whole simulation or the whole load step, the preconditioning step is still more time comsuming hence the cumulative time using M_c is always larger than $M_d(\hat{K}_3, \hat{S}_1)$–EP. Hence M_c is not recommended in general to precondition the 2-by-2 block matrix coming from Biot's consolidation equations. Among all the case tested, $M_d(\hat{K}_3, \hat{S}_1)$–E is the most time efficient. The cumulative of simulation time can be reduced at least 20 percent when $M_d(\hat{K}_3, \hat{S}_1)$–E is used.

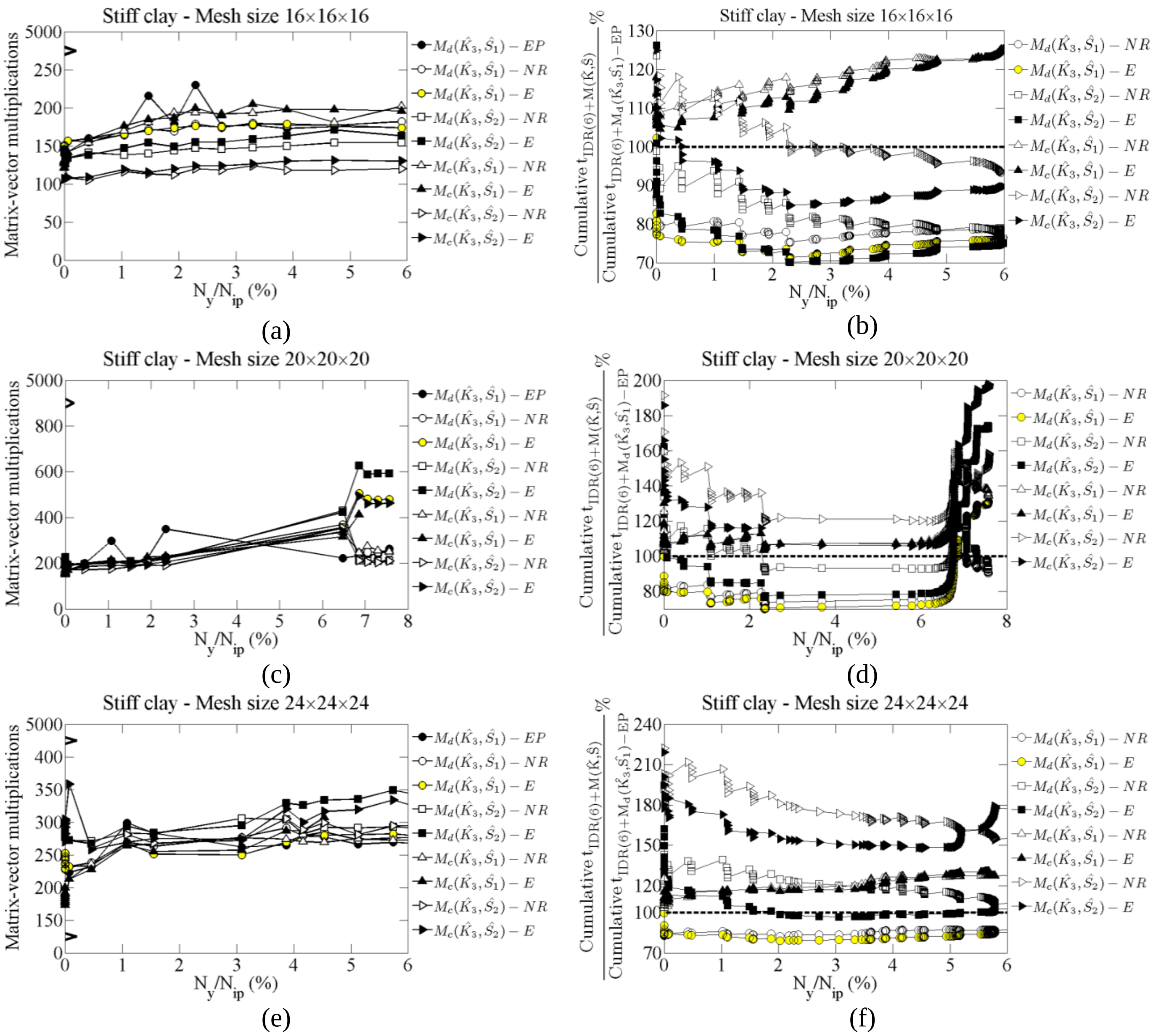


Figure 5.16: Comparison of different schemes of updating block preconditioners. Square footing resting on Soil profile 1 is considered.

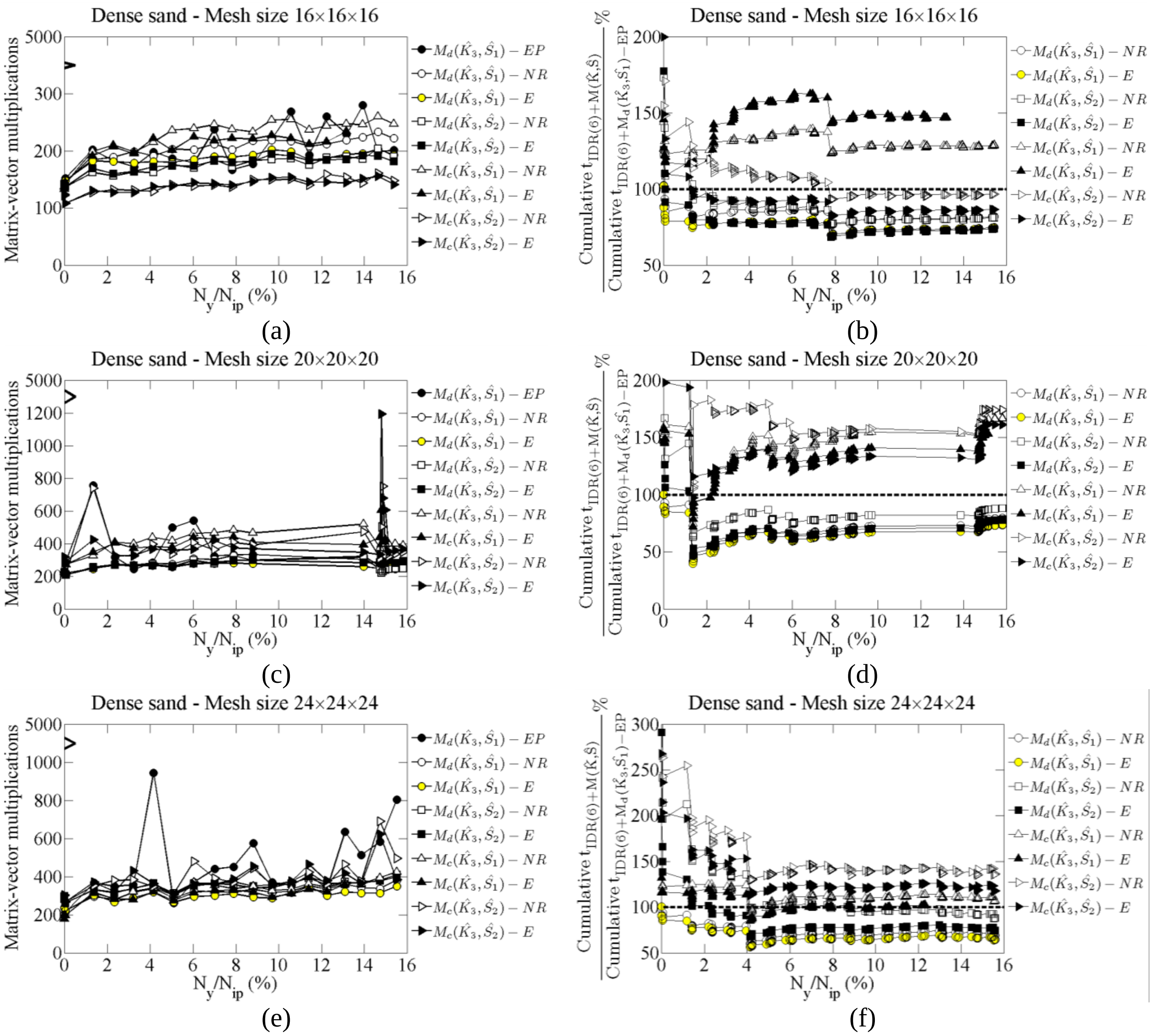


Figure 5.17: Comparison of different schemes of updating block preconditioners. Square footing resting on Soil profile 2 is considered.

5.6 Summary

This chapter applies IDR(6) and block preconditioners to solve the 2-by-2 block nonsymmetric linear system from Biot's consolidation equations. The numerical experiments on the flexible square footing result the following observations and recommendation:

1. The most optimal preconditioner for 2-by-2 block matrix is $M_d(\hat{K}_3, \hat{S}_1)$. To efficiently precondition the 2-by-2 block matrix, a better $\hat{K}$ is more crucial than a better $\hat{S}$ because the reduction in matvec due to a better $\hat{S}$

is very minimal. Besides, the constrained block preconditioner M_c is not recommended for the block matrix from Biot's consolidation equations. M_c does help IDR(6) to converge with less matvec but the reduction of matvec is not proportional with the reduction of total iteration time. The preconditioner updating scheme, $M_d(\hat{K}_3, \hat{S}_1)$–E, is recommended to save at least 20 percent of the total simulation time.

2. The narural ordering is not recommended for the Biot's consolidation analysis. This is first because block preconditioners cannot be exploited in this form. Second, the nonlinear FE analysis cannot be proceeded with large applied load because the stress-returning procedure tends to return the Gauss points to the apex of the MC evelope, which does not occur very often in block orering. Third, ILU0 and MSSOR is less time efficient than $M_d(\hat{K}_3, \hat{S}_1)$ and ILU0 is unstable in natural ordering when the permeability is small.
3. Undrained analysis can be simulated with Biot's consolidation equations, and $M_d(\hat{K}_3, \hat{S}_1)$ is efficient in preconditioning this nonsymmetric linear system. Hence the problem on unstable preconditioner occuring in Section 4.3.2 has been solved.

CHAPTER 6 APPLICATION OF PRECONDITIONERS ON PRACTICAL GEOTECHNICAL PROBLEMS

6.1 Introduction

Chapter 3, Chapter 4, and Chapter 5 have discussed the application of IDR(*s*) on solving the nonsymmetric linear system arising from the FE analysis with the non-associated MC model. Preconditioners have been recommended for the 1-by-1 block matrix coming from drained analysis and 2-by-2 block matrix coming from Biot's consolidation analaysis. Strip footing, square footing and retaining wall problems have been considered to demonstrate the efficiency of the proposed preconditioners. This chapter aims to validate the use of these preconditioners for other common geotechnical problems: 1) laterally loaded pile and 2) tunnelling. The geotechnical software package GeoFEA is used as pre-processor and post-processor of the nonlinear FE analysis.

6.2 GeoFEA implementation

GeoFEA is a commercial FE software which has implemented several popular iterative solvers such as SQMR to solve symmetric linear systems and Bi-CGSTAB to solve nonsymmetric linear systems, together several preconditioners from Chen[39] (2005) and Chauhary[37] (2011) which are optimal for symmetric linear systems. An advantage of GeoFEA is that this software allows the use of user-defined solvers and preconditioners. This chapter takes advantage of this feature to implement IDR(*s*) and the proposed preconditioners to solve the large-scale practical problems. The steps to implement user-defined solver in GeoFEA are summarized as follows:

1. Create the USOLV.DLL file with the FORTRAN code containing the user-defined solver and preconditioner. A sample FORTRAN code is provided with the software package and can be located at 'C:\Program Files\GeoFEA\USOLV.F90'.
2. Place the new USOLV.DLL file in the directory 'C:\Program Files\GeoFEA'

3. Create the finite element model with all assignments and boundary conditions as is for other inbuilt solvers.
4. In the 'SOLVE' window, check the box beside 'Generate input files only' and click on 'OK' button. This will generate three input files (geosoil.gad, geosoil.gpd, and geosoil.cnn) at 'C:\Program Files\GeoFEA'.
5. Open the 'geosoil.gad' file using any text editor (such as Notepad/Wordpad) and change the very first integer to 99. This is the only change needed by the user to use the user interface solver.
6. Go back to the 'SOLVE' window and check the box beside 'Use existing input files (geosoil.gpd, geosoil.gad)'. Click on 'OK' to solve the problem using the user-define solver.

6.3 Drained analysis

6.3.1 Problem descriptions

This section models two drained problems from literature: laterally loaded pile following Brown and Shie[31] (1990) and tunnelling following Mroueh and Shahrour[117] (2008). These two problems are geotechnical problems that cannot be simplified into 2D analysis and always require 3D modelling.

Table 6.1 summarizes the properties of the soil and structural materials used in the two problems. The soil is considered to follow the non-associated MC model. The structural materials (pile and lining) are considered to follow the linear elastic mode. The laterally loaded pile has the diameter of 0.28m and the length of 4.8m. Figure 6.2 shows the dimension and the boundary conditions of the tunnelling problem. The outer diameter of the tunnel, D_{tunnel}, is 7.5m and the lining thickenss is 0.5m.

Table 6.1: Geomaterials used in the laterally loaded pile and the tunnelling excavation problems

	Young's modulus, E' (MPa)	Poisson's ratio, v'	Cohesion, c' (kPa)	Friction angle, ϕ' (degree)	Dilation angle, ψ (degree)	Unit weight γ (kN/m^3)
Laterally loaded pile problem: D_{pile} = 0.28m; L_{pile} = 4.6m						
Soil	87+51z	0.3	13.8	23	0	18.9
Pile	4.8×10^7	0.3	-	-	-	18.9
Tunnelling problem: D_{tunnel} = 4.5m; thickness = 0.5m						
Silty sand	30	0.3	0.005	27	5	20
Lining	3.5×10^7	0.25	-	-	-	25

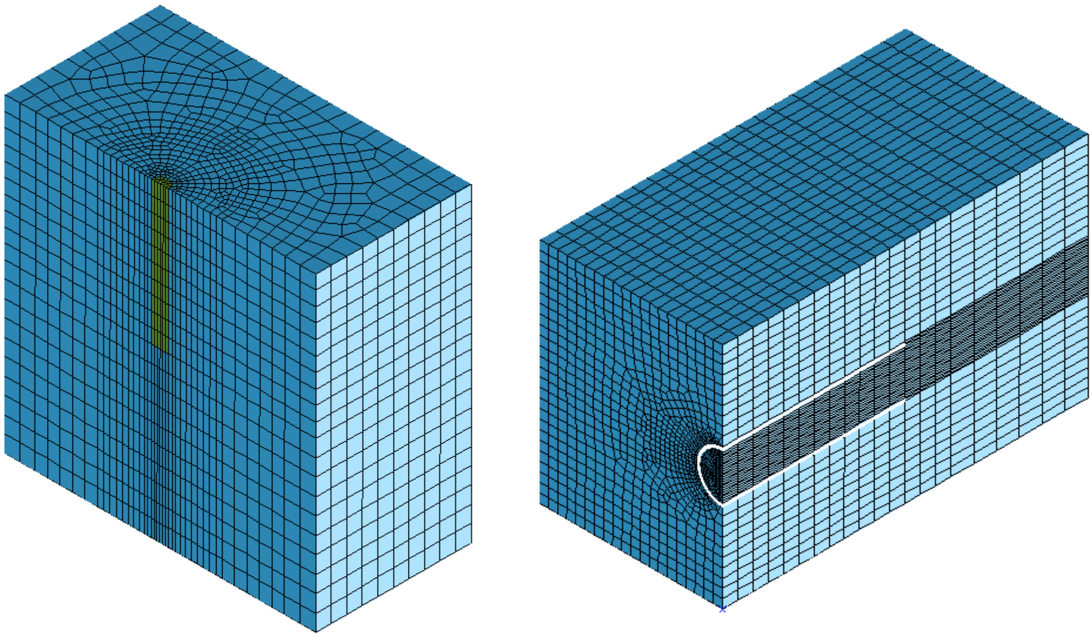
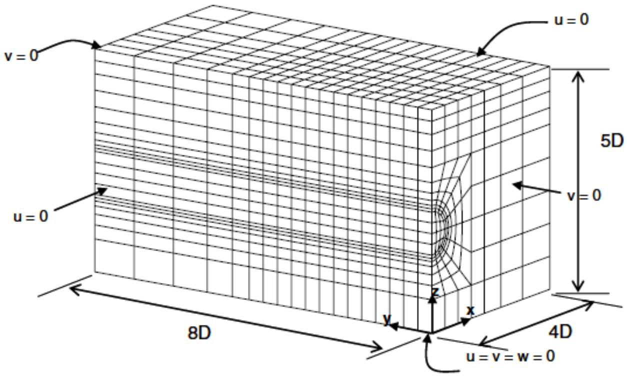
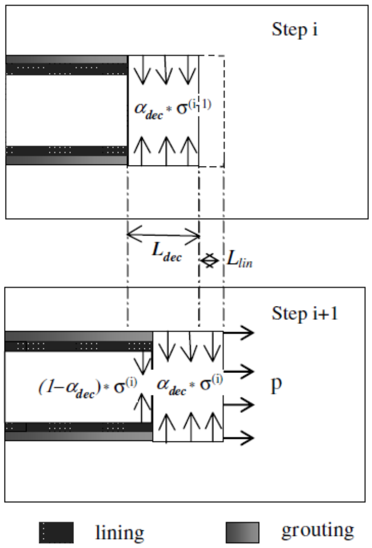

(a) (b)

Figure 6.1: 3D FE mesh of: (a) Laterally loaded pile; (b) Tunnelling excavation

The 3D meshes of the laterally loaded pile and the tunnelling problem are shown in Figure 6.1(a) and (b) respectively. The mesh of the laterally loaded piles includes 10,740 hexahedral elements and results a linear system of 141,276 unknowns. This number of unknowns does not change during the simulation. The mesh of the tunnelling excavation includes 101,101 hexahedral elements and results a linear system of 303,303 unknowns. This number of unknowns varies during the excavation and lining installing process. The excavation process follows the TBM method discussed in Mroueh & Shahrour[117] (2008) and is shown in Figure 6.2. The face pressure and the wall pressure are considered uniformed and equal to the initial axial stress at the tunnel axis.



(a)



(b)

Figure 6.2: (a) Dimension and boundary condition of the tunnelling problem; (b) Method used for the tunnel construction using TBM (Mroueh & Shahrour[117], 2008).

Drained analysis is considered hence IDR(6) with ILU0 preconditioner are used as user-defined solver and preconditioner for GeoFEA. The dynamic libray USOLV.DLL forms the global stiffness matrix K_{ep} explicitly following Eq.(2.1) therefore the technique to form K_{ep} implicitly proposed in Section 4.2 cannot be demonstrated in this section.

6.3.2 Implementation of preconditioner updating schemes

Figure 6.3 presents the implemetation of preconditioner updating schemes proposed in Section 4.2.2.2 (with the notations explained in Table 4.3) on the laterally loaded pile and tunnelling excavation problem. For the laterally loaded pile, the number of unknowns does not change hence ILU0-K_e is still applicable. For tunnelling excavation, the number of unknowns reduces during excavation process and increases during lining process. ILU0-Stage is denoted in this section as the scheme similar to ILU0-K_e. ILU0-Stage scheme updates ILU0 preconditioner at the beginning of a stage within which the number of unknowns does not change. The results in Figure 6.3 agree with the conclusion in Section 4.2 . Figure 6.3(b) and (d) show that ILU0-K_e and ILU0-Stage are only useful at the beginning of the simulation when the number of yielded

Gauss points is not significant. When there are more yielded Gauss points, ILU0-NR is recommended. This scheme can save up to 10 percent of the total simulation time in compared with ILU0-K_{ep} scheme.

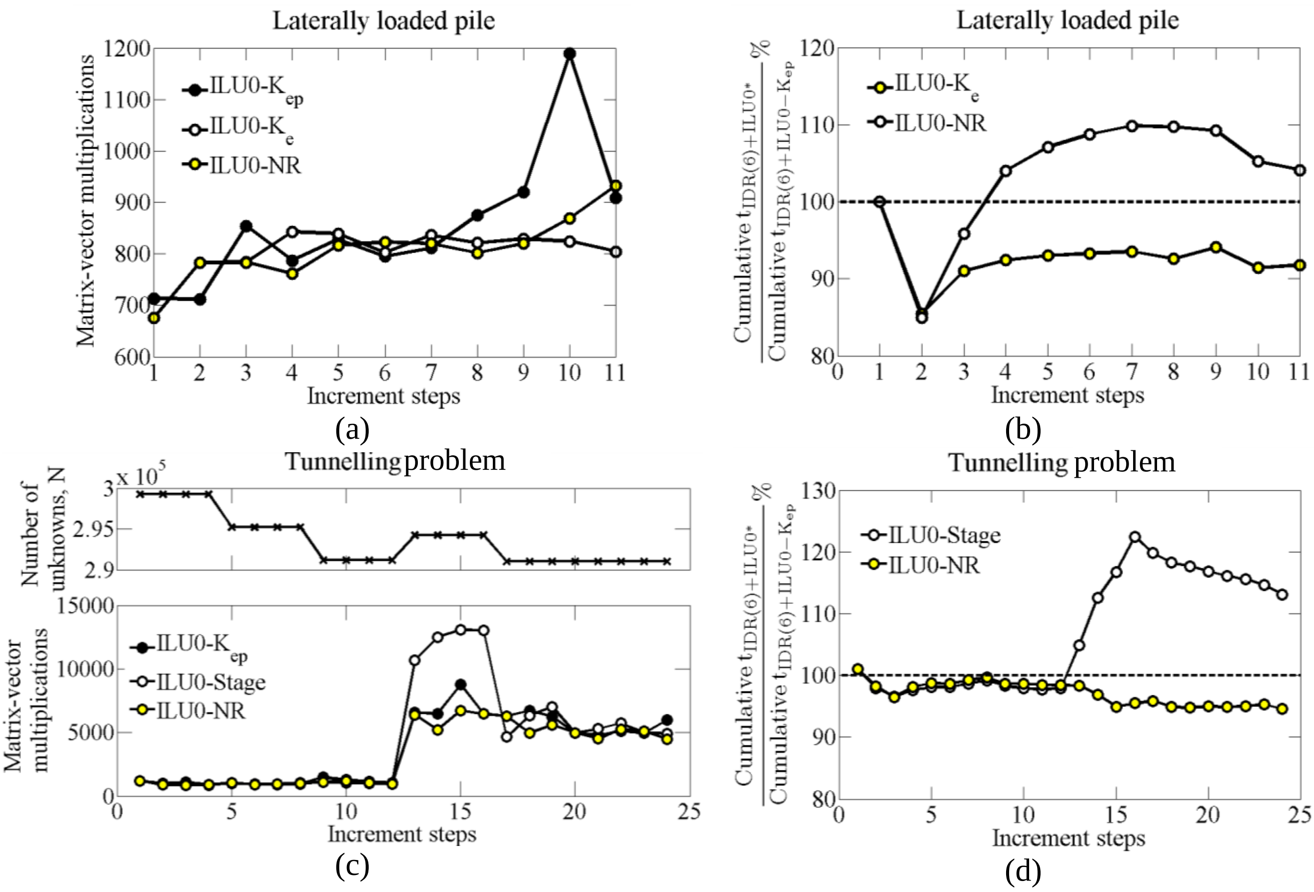


Figure 6.3: Comparison of preconditioner updating scheme in drained analysis of: (a)(b) Laterally loaded pile; (c)(d) Tunnelling problem.

IDR(6) method with ILU0 preconditioner is compared with the built-in Bi-CGSTAB method with Jacobi preconditioner. The latter failed to converge in the laterally loaded pile problem and the analysis could not be carried on. Figure 6.4 shows the comparison of the two methods in solving the tunnelling excavation problem. IDR(6) with ILU0 is much faster than Bi-CGSTAB and this is expected because IDR(6) has been shown to be faster than Bi-CGSTAB and ILU0 is a much better preconditioner than Jacobi. Regardless of preconditioner updating schemes, IDR(6) with ILU0 can save at least 65 percent of the total simulation time when compared with the built-in Bi-CGSTAB with Jacobi.

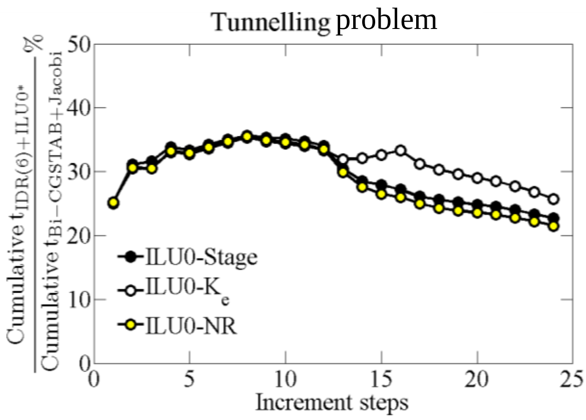
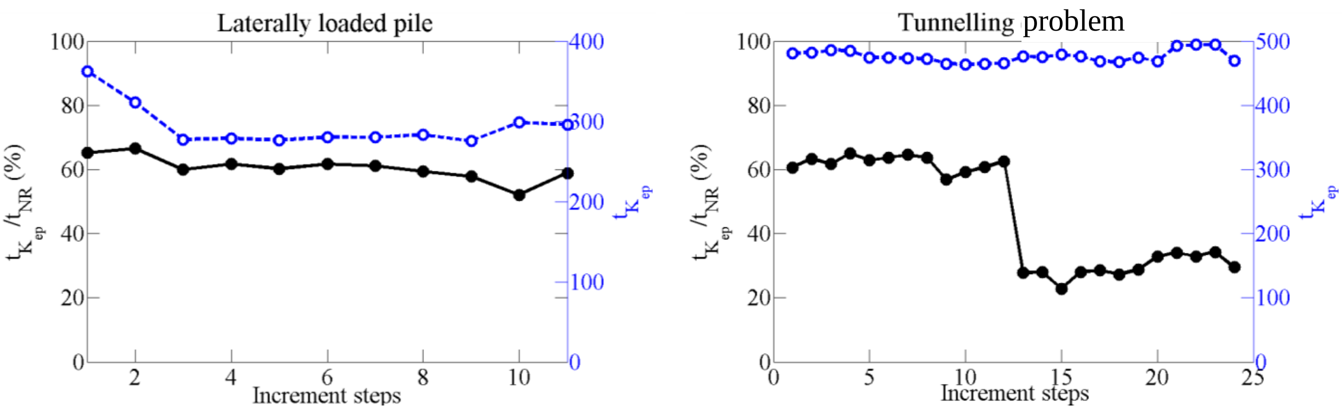


Figure 6.4: Comparison of cumulative solution time of IDR(6) versus Bi-CGSTAB

Besides, the disadvantage of forming K_{ep} matrix explicitly at every load step or NR step is highlighted in Figure 6.5. The average time in each NR step is reported. The time to form K_{ep} consumes 60 percent of the total time spent in each NR iteration. There is a reduction of this percentage in tunnelling excavation problem because the IDR requires more matvec to converge but the absolute time to form K_{ep} does not change. This means a same amount of time has to be spared in each NR iteration to form K_{ep}, which is not necessary if the implicit way to form K_{ep} proposed in Section 4.2.1 is used.



(a) (b)

Figure 6.5: Ratio of the time to form K_{ep} over total time consumed in each NR iteration when IDR(6) with ILU0-K_{ep} is used to solve the linear systems in: (a) Laterally loaded pile; (b) Tunnelling excavation

6.4 Summary

GeoFEA is a convenient software that can be used as pre-processor and post-processor to simulate complex geotechnical problems with user-defined solvers and preconditioners. The numerical results on the two practical examples show that IDR(6) with ILU0-NR is efficient in solving nonsymmetric linear system coming from large-scale geotechnical problems. The implicit way to form K_{ep} proposed in Section 4.2.1 is effective in reducing the total simulation time.

CHAPTER 7 CONCLUSION AND RECOMMENDATION

7.1 Summary and conclusions

The nonsymmetric linear system in this thesis comes from the FE discretization of the drained, undrained and Biot's consolidation of the soil following the non-associated MC model. This linear system is large-scale and highly sparse. This thesis presented the application of IDR(s) method to solve this linear system iteratively with preconditioners as accelerator.

The following findings and recommendations emerge from the numerical studies conducted in this thesis:

1. IDR(1) is theoretically equivalent to Bi-CGSTAB and IDR(s >1) is more efficient than Bi-CGSTAB in both matvec count and total iteration time. The choice s = 6 has been shown numerically to be the most optimal for IDR(s) applied to nonsymmetric linear systems arising from the non-associated MC model.

2. ILU0 is the most efficient preconditioner for K_{ep} matrix among Jacobi, SSOR-LR, SSOR-L and ILUT(ρ, τ). ILUT(ρ, τ) can be competitive in term of matvec but less competitive in term of solution time. Moreover, the fill-in number is not known priorly and also problem dependent.
3. $M_d(\hat{K}_3, \hat{S}_1)$ is the most efficient preconditioner for the 2-by-2 block stiffness matrix from Biot's consolidation analysis. The constrained block preconditioner M_c is not recommended for the block matrix from Biot's consolidation equations. M_c does help IDR(6) to converge with less matvec but the reduction of matvec is not proportional with the reduction of total iteration time.
4. Two techniques to save the total simulation time in dealing with a sequence of nonsymmetric linear systems are recommended for both 1-by-1 and 2-by-2 block matrix as follows:

a. Forming the elastoplastic global stiffness matrix implicitly by forming the elastic global stiffness matrix once and update the low-rank matrix at every NR iteration.

b. Employing the preconditioner updating scheme. Two schemes are recommended: 1) using preconditioners from the elastic stiffness matrix for the whole simulation; 2) updating preconditioners at the beginning of each load steps.

c. When these two techniques are used concurrently, the total simulation time of 1-by-1 block matrix can be reduced by 60 percent compared with the default procedure.

5. IDR(6) with ILU0-K_{ep} has been applied in the laterally loaded pile and tunnelling excavation problem and shows more time efficient than the built-in Bi-CGSTAB with Jacobi preconditioner. At least 65 percent to the default total simulation time can be saved by using IDR(6) with ILU0-*NR*.

7.2 Limitations and recommendations

This thesis has presented a wide-range and somewhat in-depth study on preconditioners for the nonsymmetric linear system arising from the non-associated MC model. However this thesis by no mean has taken into account and tackled all the problems. The followings list out the limitations of this thesis and recommendations for future works

1. There are other preconditioners developed for 1-by-1 block matrix. Approximate inverse preconditioner is one of the available options. This class of preconditioner is a promising advancement instead of using traditional preconditioner like Jacobi, SSOR and ILU. Similarly, for 2-by-2 block matrix, there are several others discussions on nonsymmetric saddle points problem that this thesis did not explore.
2. Consistent tangent stress-strain matrix has been shown to be able to preserve the quadratic convergence of NR iteration and is expected to be better than the continuum stress-strain matrix in this thesis. Hence it is promising to repeat the numerical experiments on this matrix and find the optimal preconditioners.

3. This thesis used the full NR iteration to solve the nonlinear FE equation. No acceleration technique for NR is applied in this thesis. If these techniques are applied, the total simulation time should be saved more.
4. As discussed in the Introduction, MC model is used because it is popular and simple. However, MC model has drawbacks in modeling soil behavior (such as zero dilation angle has to be indicated for undrained analysis to control the volumetric strain). Cam-clay model is a better choice to tackle the nonlinear behaviour of soil. This model follows an associated flow rule but its consistent tangent stress-strain matrix is nonsymmetric. Hence, it is interesting to study the use of IDR(*s*) on these models.

REFERENCE

1. ABAQUS/CAE user's manual: version 6.9. (2010).
2. Abbo, A. J. (1997). *Finite Element Algorithms for Elastoplasticity and Consolidation.* Doctor of Philosophy, University of Newcastle.
3. Abbo, A. J., & Sloan, S. W. (1995). A Smooth Hyperbolic Approximation To The Mohr-Coulomb Yield Criterion. *Computers and Structures, 54*(3), 427-441.
4. Abbo, A. J., Sloan, S. W. (2000). SNAC (User manual) (Version 2.0). Department of Civil, Surveying and Environmental Engineering. Callaghan: University of Newcastle.
5. Almeida e Sousa, J., Negro, A., Matos Fernandes, M., & Cardoso, A. (2011). Three-dimensional nonlinear analyses of a metro tunnel in Sao Paulo Porous clay, Brazil. *Journal of Geotechnical and Geoenvironmental Engineering, 137*(4), 376-384.
6. Almeida, V. S., & Paiva, J. B. d. (2004). A mixed BEM-FEM formulation for layered soil-superstructure interaction. *Engineering Analysis with Boundary Elements, 28*, 1111-1121.
7. Araujo, F. C. d., Mansur, W. J., Dors, C., & Martins, C. J. (2004). New developments on BE/BE multi-zone algorithms based on Krylov solvers - Application to 3D frequency-dependent problems. *Journal of the Brazilian Society of Mechanical Sciences and Engineering, 26*(2), 231-248.
8. Araujo, F. C., Silva, K. I., & Telles, J. C. F. (2006). Generic domain decomposition and iterative solvers for 3D BEM problems. *International Journal For Numerical Methods In Engineering, 68*, 448-472.
9. Arioli, M., Loghin, D., & J.Wathen, A. (2005). Stopping criteria for iterations in finite element methods. *Numerische Mathematik, 99*, 381–410.
10. Augarde, C. E., & Burd, H. J. (1995). A three-dimensional finite element model of tunnelling. In G. N. Pande & S. Pietruszczak (Eds.), *Numerical Models in Geomechanics - NUMOG V* (pp. 457-462). Rotterdam: Balkema.
11. Augarde, C. E., Ramage, A., & Staudacher, J. (2007). Element-based preconditioners for elasto-plastic problems in geotechnical engineering. *International journal for numerical methods in engineering, 71*(7), 757-779.
12. Axelsson, O., & Blaheta, R. (2010). Preconditioning of matrices partitioned in 2x2 block form: Eigenvalue estimates ans Schwarz DD for mixed FEm. *Numerical Linear Algebra with Applications, 17*, 787-810.

13. Axelsson, O., & Neytcheva, M. (2012). A general approach to analyse preconditioners for two-by-two block matrices. *Numerical linear algebra with applications, 19*.

14. Axelsson, O., Blaheta, R., & Kohut, R. (1997). Inexact Newton solvers in plasticity: Theory and Experiments. *Numerical Linear Algebra with Applications*, *4*(3), 133-152.

15. Bank, R. E., & Douglas, C. C. (1985). An efficient implementation for SSOR and incomplete factorization preconditionings. *Applied Numerical Mathematics*, *1*, 489-492.

16. Barrett, R., Berry, M., F.Chan, T., Demmel, J., M.Donato, J., Dongarra, J., et al. (1994). *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*: SIAM.

17. Bathe, K. J., & Cimento, A. P. (1980). Some practical procedures for the solution of nonlinear finite element equations. *Computer methods in applied mechanics and engineering, 22*, 59-85.

18. Benzi, M. (2002). Preconditioning techniques for large linear systems: A survey. *Journal of Computational Physics, 182*, 418-477.

19. Benzi, M., & Golub, G. H. (2004). A preconditioner for generalized saddle point problems. *SIAM Journal on Matrix Analysis and Applications*, *26*(1), 20-41.

20. Benzi, M., Szyld, D. B., & Duin, A. V. (1999). Orderings for incomplete factorization preconditioning of nonsymmetric problems. *SIAM Journal on Scientific Computing*, *20*(5), 1652-1670.

21. Bergamaschi, L., Bru, R., Martinez, A., & Putti, M. (2012). Quasi-Newton acceleration of ILU preconditioners for nonlinear two-phase flow equations in porous media. *Advances in Engineering Software, 46*, 63-68.

22. Bergamaschi, L., Ferronato, M., & Gambolati, G. (2007). Novel preconditioners for the iterative solution to FE-discretized coupled consolidation equations. *Computer methods in applied mechanics and engineering, 196*, 2647-2656.

23. Bergamaschi, L., Ferronato, M., & Gambolati, G. (2008). Mixed Constraint Preconditioners for the iterative solution of FE coupled consolidation equations. *Journal of Computational Physics, 227*, 9885-9897.

24. Birken, P., Gassner, G., Haas, M., & Munz, C.-D. (2013). Preconditioning for model discontinuous Galerkin methods for unsteady 3D Navier-Stokes equations. *Journal of COmputational Physics, 240*, 20-35.

25. Bishop, A. W. (1966). The strength of soils as engineering materials. *Geotechnique, 16*, 91-128.

26. Blaheta, R. (1997). Convergence of Newton-type methods in incremental return mapping analysis of elasto-plastic problems. *Computational Methods in Applied Mechanics and Engineering, 147*, 167-185.

27. Bolton, M. D. (1986). The strength and dilatancy of sands. *Geotechnique, 36*(1), 65-78.

28. Bonet, J., & D.Wood, R. (2008). *Nonlinear continuum mechanics for finite element analysis*. New York: Cambridge University Press.

29. Borja, R. I. (1991). Composite Newton-PCG and Quasi-Newton iterations for nonlinear consolidation. *Computer methods in applied mechanics and engineering, 86*, 27-60.

30. Botchev, M. A., & Golub, G. H. (2006). A class of nonsymmetric preconditioners for saddle points problems. *SIAM Journal on Matrix Analysis and Applications, 27*(4), 1125-1149.

31. Brown, D. A., Shie, C-F. (1990). Three dimensional finite element model of laterally loaded piles. *Computers and Geotechnics, 10*, 59-79.

32. Bruaset, A. M. (1995). *A survey of preconditioned iterative methods*. New York: Longman Scientific & Technical.

33. Bruaset, A. M. (1997). Krylov subspace iterations for sparse linear systems. In M. Dahlen & A. Tveito (Eds.), *Numerical methods and software tools in industrial mathematics* (pp. 281-298). Birkhauser, Boston: Springer.

34. Bruyn, D. J. D., Ramaeckers, C. C., & Huergo, P. J. (1994). Excavation of galleries in deep clay formations: An analytical tool for the control of finite element results. *Computers and Geotechnics, 16*, 247-260.

35. Chaillat, S., Bonnet, M., & Semblat, J. F. (2009). A new fast multi-domain BEM to model seismic wave propagation and amplification in 3D geological structures. *Geophysical Journal International, 177*(2), 509-531.

36. Chan, S. H., Phoon, K. K., & Lee, F. H. (2001). A modified Jacobi preconditioner for solving ill-conditioned Biot's consolidation equations using symmetric quasi-minimal residual method. *International Journal for Numerical and Analytical Methods in Geomechanics, 25*(10), 1001-1025.

37. Chauhary, K. B. (2010). *Preconditioners for soil-structure interaction problems with significant material stiffness constrast.* Doctor of Philosophy, National University of Singapore, Singapore.

38. Chen, R. S., Ping, X. W., Wang, D. X., & Yung, E. K. N. (2004). SSOR preconditioned GMRES for the FEM analysis of waveguide discontinuities with anisotropic dielectric. *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, 17*, 105-118.

39. Chen, X. (2005). *Preconditioner for iterative solutions of large-scale linear systems arising from Biot's consolidation equations.* Doctor of Philosophy, National University of Singapore, Singapore.

40. Chen, X., & Phoon, K. K. (2009). Some numerical experiences on convergence criteria for iterative finite element solvers. *Computers and Geotechnics, 36*, 1272-1284.

41. Chen, X., & Phoon, K. K. (2012). Applications of Symmetric and Nonsymmetric MSSOR Preconditioners to Large-Scale Biot's Consolidation Problems with Nonassociated Plasticity. *Journal of Applied Mathematics, 2012*. doi: 10.1155/2012/352081

42. Chen, X., Toh, K. C., & Phoon, K. K. (2006). A modified SSOR preconditioner for sparse symmetric indefinite linear systems of equations. *International Journal for Numerical Methods In Engineering, 65*, 785-807.

43. Chow, E., & Saad, Y. (1997). Experimental study of ILU preconditioners for indefinite matrices. *Journal of Computational and Applied Mathematics, 86*(2), 387-414.

44. Clausen, J., & Krabbenhoft, K. (2008). Existence and Uniqueness of Solutions in Nonassociated Mohr-coulomb Elastoplasticity. In Schrefler, B. A., & Perego, U. (Eds.), *8th World Congress on Computational Mechanics WCCM8, 5th European Congress on Computational Methods in Applied Science and Engineering ECCOMAS 2008.* Spain.

45. Conte, E., Silvestri, F., & Troncone, A. (2010). Stability analysis of slopes in soils with strain-softening behaviour. *Computers and Geotechnics, 37*, 710-722.

46. Crisfield, M. A. (1987). Plasticity computations using the Mohr—Coulomb yield criterion. *Engineering Computations, 4*(4), 300 - 308.

47. Crisfield, M. A. (1998). *Non-linear finite element analysis of solids and structures: Essentials* (Vol. 1): John Wiley & Sons.

48. Dahl, O., & Wille, S. O. (1992). An ILU preconditioner with coupled node fill-in for iterative solution of the mixed finite element formulation of the 2D and 3D Navier-Stokes equations. *International Journal for Numerical Methods in Fluids, 15*, 525-544.

49. Diosady, L. T., & Darmofal, D. L. (2009). Preconditioning methods for discontinuous Galerkin solutions of the Navier-Stokes equations. *Journal of Computational Phyics, 228*, 3917-3935.

50. Driscoll, T. A., Toh, K.-C., & Trefethen, L. N. (1998). From potential theory to matrix iterations in six steps. *SIAM Review, 40*(3), 547-578.

51. Duncan, J. M. (1994). The role of advanced constitutive relations in practical applications *Proceddings of the Thirteenth international conference on soil mechanics and foundation engineering* (Vol. 5, pp. 31-48). Rotterdam: A. A. Balkema.

52. Dutto, L. C., & Habashi, W. G. (1999). Parallelization of the ILU(0) preconditioner for CFD problems on shared-memory computers. *International Journal for Numerical Methods in Fluids, 30*, 995-1008.

53. Eisenstat, S. C. (1981). Efficient implementation of a class of preconditioned conjugate gradient methods. *SIAM Journal on Scientific and Statistical Computing, 2*(1), 1-4.

54. Eisenstat, S. C., Elman, H. C., & Schultz, M. H. (1983). Variational iterative methods for nonsymmetric systems of linear equations. *SIAM Journal on Numerical Analysis*, *20*(2), 345-357.

55. Erickson, H. L., & Drescher, A. (2002). Bearing capacity of circular footings. *Journal of Geotechnical and Geoenvironmental Engineering, 128*(1), 38-43.

56. Faheem, H., Cai, F., & Ugai, K. (2004). Three-dimensional base stability of rectangular excavations in soft soils using FEM. *Computers and Geotechnics, 31*, 67-74.

57. Fata, S. N., & Gray, L. J. (2010). On the implementation of 3D Galerkin boundary integral equations. *Engineering Analysis with Boundary Elements, 34*(1), 60-65.

58. Ferronato, M. (2012). Preconditioning for Sparse Linear Systems at the Dawn of the 21st Century: History, Current Developments, and Future Perspectives. *ISRN Applied Mathematics, 2012*, 49. doi: 10.5402/2012/127647.

59. Ferronato, M., Bergamaschi, L., & Gambolati, G. (2010). Performance and robustness of block constraint preconditioners in finite element coupled consolidation problems. *International Journal for Numerical Methods In Engineering, 81*, 381-402.

60. Fischer, S., Frommer, A., Glassner, U., Lippert, T., Ritzenhofer, G., & Schilling, K. (1996). A paralllel SSOR preconditioner for lattice QCD. *Computer Physics Communications, 98*, 20-34.

61. Fletcher, R. (1976). Conjugate gradient methods for indefinite systems *Numerical Analysis* (pp. 73-89). Berlin, Heidelberg, New York: Springer-Verlag.

62. Freund, R. W., & Nachtigal, N. M. (1991). QMR: a quasi-minimal residual method for non-Hermitian linear systems. *Numerische Mathematik, 60*, 315-339.

63. Freund, R. W., Golub, G. H., & Nachtigal, N. M. (1992). Iterative solution of linear systems. *Acta Numerica, 1*(57-100).

64. Gambolati, G., Pini, G., & Ferronato, M. (2001). Numerical performance of projection methods in finite element consolidation models. *International Journal for Numerical and Analytical Methods in Geomechanics*, *25*, 1429-1447.

65. Gambolati, G., Pini, G., & Ferronato, M. (2002). Direct, partitioned and projected solution to finite element consolidtion models. *International Journal for Numerical and Analytical Methods in Geomechanics, 26*, 1371-1383.

66. Gambolati, G., Pini, G., & Ferronato, M. (2003). Scaling improves stability of preconditioned CG-like solvers for FE consolidation equations. *International Journal for Numerical and Analytical Methods in Geomechanics, 27*, 1043-1056.

67. GeoFEA (Version 8.0) (Software). (2006). *GeoSoft Pte. Ltd.*

68. GeoStudio (Version 8.0.7.6129) (Software). (2012). *GEO-SLOPE International*.

69. Ghoussoub, N., & Moradifam, A. (2010). A note on simultaneous preconditioning and symmetrization of non-symmetric linear systems. *Numerical Linear Algebra with Application, 18*, 343-349.

70. Gijzen, M. B. V., & Sonneveld, P. (2010). An elegant IDR(s) variant that efficiently exploits bi-orthogonality properties (D. o. A. M. Analysis, Trans.). Delft: Delft University of Technology.

71. Golub, G. H., & Vorst, H. A. v. d. (2001). Closer to the solutions: iterative linear solvers *The State of the Art in Numerical Analysis*: Clarendon Press.

72. Gose, S., Ugai, K., & Ochiai, H. (1997). The behavior of pile foundation using three-dimensional elasto-plastic FEM analysis. In S. Pietruszczak & G. N. Pande (Eds.), *Numerical models in geomechanics: proceedings of the Sixth International Symposium on Numerical Models in Geomechanics* (pp. 539-544). Rotterdam: A. A. Balkema.

73. Greenbaum, A. (1997). *Iterative methods for solving linear systems*. Philadelphia: SIAM.

74. Griffiths, D. V., & Lane, P. A. (1999). Slope stability analysis by finite elements. *Geotechnique, 49*(3), 387-403.

75. Gutknecht, M. H. (2007). A brief introduction to Krylov space methods for solving linear systems. In Y. Kaneda, H. Kawamura & M. Sasai (Eds.), *Frontiers of Computational Science - Proceedings of the International Symposium on Frontiers of Computational Science 2005* (pp. 53-62). Berlin Heidelberg: Springer-Verlag.

76. Hartmann, S., Tebbens, J. D., Quint, K. J., & Meister, A. (2009). Iterative solvers within sequences of large linear systems in non-linear structural mechanics. *Journal of Applied Mathematics and Mechanics, 89*(9), 711-728.

77. Hashash, Y. M. A., Song, H., & Osouli, A. (2011). Three-dimensional inverse analyses of a deep excavation in Chicago clays. *International Journal for Numerical and Analytical Methods in Geomechanics, 35*, 1059-1075.

78. Hata, Y., Ichii, K., & Nozu, A. (2012). Three-dimensional non-linear FEM analysis of a seismic induced crack at an airport runway. *Soil Dynamics and Earthquake Engineering, 42*, 105-118.

79. Henkel, D. J. (1959). The relationships between the strength, pore-water pressure, and volume-change characteristics of saturated clays. *Geotechnique*, 9(3), 119-135.

80. Hestenes, M. R., & Stiefel, E. (1952). Methods of Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards, 49*(6), 409-436.

81. Hettler, A., & Vardoulakis, I. (1984). Behaviour of dry sand tested in a large triaxial apparatus. *Geotechnique, 34*(2), 183-198.

82. Hicher, P.-Y. (1996). Elastic properties of soils. *Journal of Geotechnical Engineering, 122*(8), 641-648.

83. Houlsby, G. T. (1991). How the dilatancy of soils affects their behaviour (D. o. E. Science, Trans.). Oxford: University of Oxford.

84. Isaacson, E., & Keller, H. B. (1994). *Analysis of numerical methods*. New York: Dover Publications.

85. Jardine, R. J., Potts, D. M., Fourie, A. B., & Burland, J. B. (1986). Studies of the influence of non-linear stress-strain characteristics in soil-structure interaction. *Geotechnique, 36*(3), 377-396.

86. Jeremic, B., & Jie, G. (2008). Parallel finite element computations for soil-foundation-structure interaction problems (D. o. C. a. E. Engineering, Trans.): University of California.

87. Jiang, B.-n., Lin, T. L., & Povinelli, L. A. (1994). Large-scale computation of incompressible viscous flow by least-squares finite element method. *Computational Methods in Applied Mechanics and Engineering, 114*(3-4), 213-231.

88. Jing, Y.-F., Huang, T.-Z., Duan, Y., & Carpentieri, B. (2010). A comparative study of iterative solutions to linear systems arising in quantum mechanics. *Journal of Computational Phyics, 229*, 8511-8520.

89. Johnson, K., Karunasena, W., Guazzo, A., & Sivakugan, N. (2001). Load-deformation characteristics of axially loaded piles. In C. F. Lee, C. K. Lau, C. W. W. Ng, A. K. Kwong, P. L. R. Pang, J. H. Yin & Z. Q. Yue (Eds.), *Soft soil engineering: proceedings of the Third International Conference on Soft Soil Engineering* (pp. 339-344): A. A. Balkema Publishers.

90. Kacimi, A. E., & Laghrouche, O. (2011). Wavelet based ILU preconditioners for the numerical solution by PUFEM of high frequency elastic wave scattering. *Journal of Computational Physics, 230*(8), 3119-3134.

91. Kahyaoglu, M. R., Imancli, G., Ozturk, A. U., & Kayalar, A. S. (2009). Computational 3D finite element analyses of model passive piles. *Computational Materials Science, 46*, 193-202.

92. Kechroud, R., Soulaimani, A., Saad, Y., & Gowda, S. (2004). Preconditioning techniques for the solution of the Helmholtz equation by the finite element method. *Mathematics and Computers in Simulations, 65*, 303-321.

93. Kelesoglu, M. K., & Springman, S. M. (2011). Analytical and 3D numerical modelling of full-height bridge abutments constructed on pile foundations through soft soils. *Computers and Geotechnics, 38*(8), 934-948.

94. Kelley, C. T. (1995). *Iterative methods for linear and nonlinear equations*. Philadelphia: SIAM.

95. Knibbe, H., Oosterlee, C. W., & Vuik, C. (2011). GPU implementation of a Helmholtz Krylov solver preconditioned by a shifted Laplace multigrid method. *Journal of Computational and Applied Mathematics, 236*, 281-293.

96. Krenk, S. (2009) *Non-linear Modeling and Analysis of Solids and Structures* (pp. 7-14). Cambridge: Cambridge University Press.

97. Kumar, J. (2004). Stability factors for slopes with nonassociated flow rule using energy consideration. *Journal of Geotechnical and Geoenvironmental Engineering, 4*(4), 264-272.

98. Lan, C. W., & Liang, M. C. (1997). A three-dimensional finite-volume/Newton method for thermal-capillary problems. *International Journal For Numerical Methods In Engineering, 40*, 621-636.

99. Lee, F.-H., Hong, S.-H., Gu, Q., & Zhao, P. (2011). Application of large three-dimensional finite-element analyses to practical problems. *International Journal of Geomechanics, 11*(Special Issue: Material and Computer Modeling), 529-539.

100. Lee, K. H., & Rowe, R. K. (1990). Finite element modelling of the three-dimensional ground deformations due to tunnelling in soft cohesive soils: Part I - Method of analysis. *Computers and Geotechnics, 10*, 87-109.

101. Leung, Y. F., Soga, K., Lehane, B., & Klar, A. (2010). Role of linear elasticity in pile group analysis and load test intepretation. *Journal of Geotechnical and Geoenvironmental Engineering, 12*(136), 1686-1694.

102. Lewis, R. W., & Schrefler, B. A. (1998). *The finite element method in the static and dynamic deformation and consolidation of porous media*: John Wiley & Sons.

103. Lin, Y., & Grosh, K. (2003). Effectiveness of iterative solution strategies for the high frequency response of fluid-loaded structures. *Finite Elements in Analysis and Design, 39*, 951-964.

104.Liu, Q.-F., & Meyerhof, G. G. (1987). New method for non-linear analysis of laterally loaded flexible piles. *Computers and Geotechnics, 4*(3), 151-169.

105.Loukidis, D., & Salgado, R. (2009). Bearing capacity of strip and circular footings in sand using finite elements. *Computers and Geotechnics, 36*, 871-879.

106.Malas, T., & Gurel, L. (2007). Incomplete LU preconditioning with the multilevel fast multipole algorithm for electromagnetic scattering. *SIAM Journal on Scientific Computing, 29*(4), 1476-1494.

107.Manoharan, N., & Dasgupta, S. P. (1995). Bearing capacity of surface footings by finite elements. *Computers and Geotechnics, 54*(4), 563-586.

108.Manoharan, N., & Dasgupta, S. P. (1997). Collapse load computation for high friction soil. *Computers and Geotechnics, 62*(4), 681-684.

109.Manoharan, N., & Dasgupta, S. P. (1997). Collapse load computation for high-friction soil. *Computers and Structures, 62*(4), 681-684.

110.Manzari, M. T., & Nour, M. A. (2000). Significance of soil dilatancy in slope stability analysis. *Journal of Geotechnical and Geoenvironmental Engineering, 126*(1), 75-80.

111.Meurant, G. (1998). *Computer solution of large linear systems*. North Holland: Elsevier.

112.Migliazza, M., Chiorboli, M., & Giani, G. P. (2009). Comparison of analytical method, 3D finite element model with experimental subsidence measurements resulting from the extension of the Milan underground. *Computers and Geotechnics, 36*, 113-124.

113.Morgan, R. B. (2000). Implicitly restarted GMRES and Arnoldi methods for nonsymmetric systems of equations. *SIAM Journal on Matrix Analysis and Applications, 21*(4), 1112-1135.

114.Motta, E. (2013). Lateral delfection of horizontally loaded rigid piles in elastoplastic medium. *Journal of Geotechnical and Geoenvironmental Engineering, 139*(3), 501-506.

115.Mroueh, H., & Shahrour, I. (1999). Use of sparse iterative methods for the resolution of three-dimensional soil/structure interaction problems. *International Journal for Numerical and Analytical Methods in Geomechanics, 23*, 1961-1975.

116.Mroueh, H., & Shahrour, I. (2003). A full 3-D finite element analysis of tunneling-adjacent structures interactions. *Computers and Geotechnics, 30*, 245-253.

117.Mroueh, H., & Shahrour, I. (2008). A simplified 3D model for tunnel construction using tunnel boring machines. *Tunnelling and Underground Space Technology, 23*, 38-45.

118. Nachtigal, N. M., Reddy, S. C., & Trefethen, L. N. (1992). How fast are nonsymmetric matrix iterations? *SIAM Journal on Matrix Analysis and Applications, 13*(3), 778-795.

119. Nova, R. (2004). The role of non-normality in soil mechanics and some of its mathematical consequences. *Computers and Geotechnics, 31*, 185-191.

120. Oettl, G., Stark, R. F., & Hofstetter, G. (1998). A comparison of elastic-plastic soil models for 2D FE analyses of tunnelling. *Computers and Geotechnics, 23*, 19-38.

121. Osei-Kuffuor, D., & Saad, Y. (2010). Preconditioning Helmholtz linear systems. *Applied Numerical Mathematics, 60*, 420-431.

122. Ou, C. Y., & Kung, T. C. (2004). Analysis of ground surface settlement induced by excavation. In G. N. Pande & S. Pietruszczak (Eds.), *Numerical models in geomechanics: NUMOG IX: proceddings of the 9th Symposium on Numerical Models in Geomechanics*. Leiden: A. A. Balkema Publishers.

123. Owen, D. R. J., & Hinton, E. (1980). *Finite elements in plasticity: theory and practices*. Swansea: Pineridge Press.

124. Pakbaz, M., & Zolfagharian, M. (2005). Evaluation of earth fill parameters from Gavoshan dam with back analysis during construction *Proceedings of the 16th International Conference on Soil Mechanics and Geotechnical Engineering* (Vol. 3, pp. 1901-1904). Rotterdam: Millpress.

125. Parry, R. H. G. (1968). Field and laboratory behaviour of a lightly overconsolidated clay. *Geotechnique, 18*, 151-171.

126. Payer, H. J., & Mang, H. A. (1997). Iterative strategies for solving systems of linear, algebraic equations arising in 3D BE-FE analyses of tunnel drivings. *Numerical Linear Algebra with Applications, 4*(3), 239-268.

127. Peng, J. R., Rouainia, M., & Clarke, B. G. (2010). Finite element analysis of laterally loaded fin piles. *Computers and Structures, 88*, 1239-1247.

128. Persson, P. O., & Peraire, J. (2008). Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier-Stokes equations. *SIAM Journal on Scientific Computing, 30*(6), 2709-2733.

129. Phoon, K. K. (2008). Numerical recipes for reliability analysis - a primer. In K. K. Phoon (Ed.), *Reliability-based design in geotechnical engineering - Computations and applications* (pp. 1-75). London, New York: Taylor & Francis.

130. Phoon, K. K., Toh, K. C., & Chen, X. (2004). Block constrained versus generalized Jacobi preconditioners for iterative solution of large-scale Biot's FEM equations. *Computers and Structures, 82*, 2401-2411.

131. Phoon, K. K., Toh, K. C., Chan, S. H., & Lee, F. H. (2002). An efficient diagonal preconditioner for finite element solution of Biot's consolidation

equations. *International Journal For Numerical Methods In Engineering, 55,* 377-400.

132.Pillis, L. G. D. (1998). A comparison of iterative methods for solving nonsymmetric linear systems. *Acta Applicandae Mathematicae, 51,* 141-159.

133.PLAXIS 2D (Version 2012) (Software). (2012). *PLAXIS*.

134.PLAXIS 3D (Version 2012) (Software). (2012). *PLAXIS*.

135.Plumey, S., Muttoni, A., Vulliet, L., & Labiouse, V. (2010). Analytical and numerical analyses of the load-bearing capacity of retaining walls laterally supported at both ends. *International Journal for Numerical and Analytical Methods in Geomechanics*, *35*(9), 1019-1033.

136.Pontaza, J. P., & Reddy, J. N. (2004). Space-time coupled spectral/hp least-squares finite element formulation for the incompressible Navier-Stokes equations. *Journal of Computational Physics*, *197*, 418-459.

137.Potts, D. M., & Zdravkovic, L. (1999). *Finite element analysis in geotechnical engineering: Theory*. London: Thomas Telford.

138.Potts, D. M., & Zdravkovic, L. (2001). *Finite element analysis in geotechnical engineering: Application*. London: Thomas Telford.

139.Quarteroni, A., Sacco, R., & Saleri, F. (2007). *Numerical Mathematics* (2 ed.). New York: Springer.

140.Rehman, M. u., Vuik, C., & Segal, G. (2008). A comparison of preconditioners for incompressible Navier-Stokes solvers. *International Journal for Numerical Methods in Fluids*, *57*, 1731-1751.

141.Ribeiro, F. L. B., & Ferrerira, I. A. (2007). Parallel implementation of the finite element method using compressed data structures. *Computational Mechanics*, *41*(1), 31-48.

142.Roscoe, K. H., & Burland, J. B. (1968). On the generalised stress-strain behaviour of we clay. In J. Heyman & F. A. Leckie (Eds.), *Engineering plasticity: papers for a conference held in Cambridge* (pp. 535-609). Cambridge: Cambridge University Press.

143.Roscoe, K. H., Schofield, A. N., & Thuraurajahb, A. (1963). Yielding of clays in statees wetter than critical. *Geotechnique, 13*(3), 211-240.

144.Saad, Y. (2003). *Iterative Methods for Sparse Linear Systems*. Philadelphia: SIAM.

145.Saad, Y., & Schultz, M. H. (1986). GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, *7*(3), 856-869.

146. Saad, Y., & Vorst, H. A. v. d. (2000). Iterative solution of linear systems in the 20th century. *Journal of Computational and Applied Mathematics, 123*(1-2), 1-33.

147. SAGE-CRISP. (2000). Software, Version 4.02b. *Sage Consortium*.

148. Santandrea, S., & Sanchez, R. (2005). Analysis and improvements of the DPN acceleration technique for the method of characteristics in unstructured meshes. *Anals of Nuclear Energy, 32*, 163-193.

149. Schanz, T., & Vermeet. (1996). Angles of friction and dilatancy of sand. *Geotechnique, 46*(1), 145-151.

150. Schneider, S., & Marburg, S. (2003). Performance of iterative solvers for acoustic problems. Part II. Acceleration by ILU-type preconditioner. *Engineering Analysis with Boundary Elements, 27*, 751-757.

151. Schofield, A. N., & Wroth, P. (1968). *Critical state soil mechanics*: McGraw-Hill.

152. Schweiger, H. F. (2008). The role of advanced constitutive models in geotechnical engineering. *Geomechanics and Tunnelling, 1*(5), 336-344.

153. Shahrour, I. (1992). PECPLAS: un programme de calcul par elements finis pour la resolution des problemes de geotechnique. *Colloque International Geotechnique Informatique* (pp. 327-334). Paris: Presses de l'ENPC.

154. Sheu, T. W. H., Fang, C. C., & Tsai, S. F. (1999). Application of an element-by-element BiCGSTAB iterative solver to a monotonic finite element model. *Computers and Mathematics with Applications, 37*, 57-70.

155. Simoncini, V., & Szyld, D. B. (2007). Recent computational developments in Krylov subspace methods for linear systems. *Numerical Linear Algebra with Application, 14*, 1-59.

156. Sleijpen, G. L. G., Sonneveld, P., & Gijzen, M. B. v. (2010). Bi-CGSTAB as an induced dimension reduction method. *Applied Numerical Mathematics, 60*(11), 1100-1114.

157. Sloan, S. W., & Booker, J. R. (1986). Removal Of Singularities In Tresca And Mohr-Coulomb Yield Functions. *Communications In Applied Numerical Methods, 2*, 173-179.

158. Smith, I. M., & Griffiths, D. V. (2004). *Programming the Finite Element Method*. Chichester: John Wiley & Sons.

159. Smith, I. M., & Ho, D. K. H. (1992). Influence of construction technique on the performance of a braced excavation in marine clay. *International Journal for Numerical and Analytical Methods in Geomechanics, 16*, 845-862.

160. Sonneveld, P. (1989). CGS: a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing, 10*, 36-52.

161. Sonneveld, P. (2012). On the convergence behaviour of IDR(s) and related methods. *SIAM Journal on Scientific Computing, 34*(5), 2576-2598.

162. Sonneveld, P., & Gijzen, M. B. V. (2008). IDR(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. *SIAM Journal on Scientific Computing, 31*(2), 1035-1062.

163. Stute, B., Krupp, V., & Lieres, E. v. (2013). Performance of iterative equation solvers for mass transfer problems in three dimensional sphere packings in COMSOL. *Simulation Modelling Practice and Theory, 33*, 115-131.

164. Terzaghi, K. (1948). *Theoretical soil mechanics* (4 ed.). New York: John Wiley and Sons.

165. Toh, K. C., & Phoon, K. K. (2007). Comparison between iterative solution of symmetric and non-symmetric forms of Biot's FEM equations using the generalized Jacobi preconditioner. *International Journal for Numerical and Analytical Methods in Geomechanics, 32*(9), 1131-1146.

166. Toh, K.-C., Phoon, K. K., & Chan, S.-H. (2004). Block preconditioners for symmetric indefinite systems. *International Journal For Numerical Methods In Engineering, 60*, 1361-1381.

167. Umetani, N., MacLachlan, S. P., & Oosterlee, C. W. (2009). A multigrid-based shifted Laplacian preconditioner for a fourth-order Helmholtz discretization. *Numerical Linear Algebra with Applications, 16*, 603-626.

168. Vermeer, P. A., & De Borst, R. (1984). Non-associated plasticity for soils, concrete and rock. *Heron, 29*(3), 3-64.

169. Vermeer, P. A., & Langen, H. v. (1989). Soil collapse computations with finite elements. *Ingenieru-Archiv, 59*, 221-236.

170. Vorst, H. A. v. d. (1992). Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing, 13*(2), 631-644.

171. Vorst, H. A. v. d. (2002). Efficient and reliable iterative methods for linear systems. *Journal of Computational and Applied Mathematics, 149*, 251-265.

172. Vorst, H. A. v. d. (2003). *Iterative Krylov Methods for Large Linear Systems*. New York: Cambridge University Press.

173. Weiss, R. (1995). A theoretical overview of Krylov subspace methods. *Applied Numerical Mathematics, 19*, 207-233.

174. Wesseling, P., & Sonneveld, P. (1980). Numerical experiments with a multiple grid and a preconditioned Lanczos type method *Approximation Methods for Navier-Stokes Problems* (pp. 543-562). Berlin, Heidelberg, New York: Springer-Verlag.

175. White, J. A., & Borja, R. I. (2011). Block-preconditioned Newton–Krylov solvers for fully coupled flow and geomechanics. *Computational Geosciences, 15*(4), 647-659.

176. Whitman, R. V. (2000). Organizing and evaluating uncertainty in geotechnical engineering. *Journal of Geotechnical and Geoenvironmental Engineering, 126*(7), 583-593.

177. Whittle, A. J., Hashash, Y. M. A., & Whitman, R. V. (1993). Analysis of deep excavation in Boston. *Journal of Geotechnical Engineering, 119*(1), 69-90.

178. Wieners, C., Graf, T., Ammann, M., & Ehlers, W. (2005). Parallel Krylov methods and the application to 3-d simulations of a triphasic porous media model in soil mechanics. *Computational Mechanics, 36*(6), 409-420. doi: 10.1007/s00466-004-0654-1.

179. Wood, D. M. (1991). *Soil Behaviour and Critical State Soil Mechanics*: Cambridge University Press.

180. Wood, D. M. (2004). *Geotechnical Modelling* (2.2 ed.): Spon Press.

181. Wriggers, P. (2008). *Nonlinear finite element methods*. Berlin: Springer-Verlag.

182. Xiao, J., Ye, W., Cai, Y., & Zhang, J. (2012). Precorrected FFT accelerated BEM for large-scale transient elastodynamic analysis using frequency-domain approach. *International Journal for Numerical and Analytical Methods in Geomechanics, 90*, 116-134.

183. Yang, Z., & Jeremic, B. (2002). Numerical analysis of pile behaviour under lateral loads in layered elastic-plastic soils. *International Journal for Numerical and Analytical Methods in Geomechanics, 26*(14), 1385-1406.

184. Yin, J. H., Wang, Y. J., & Selvadurai, A. P. S. (2001). Influence of nonassociativity on the bearing capacity of a strip footing. *Journal of Geotechnical and Geoenvironmental Engineering, 127*(11), 985-989.

185. Yong, K. Y., Lee, F. H., Parnploy, U., & Lee, S. L. (1989). Elasto-plastic consolidation analysis for strutted excavation in clay. *Computers and Geotechnics, 8*, 311-328

186. Zdravkovic, L., Potts, D. M., & John, H. D. S. (2005). Modelling of a 3D excavation in finite element analysis. *Geotechnique, 55*(7), 497-513.

187. Zienkiewicz, O. C., Taylor, R. L., & Zhu, J. Z. (2005) *The finite element method: Its basis and fundamentals* (6 ed., pp. 54-102). Oxford; Burlington: Elsevier Butterworth-Heinemann.

188. Zvanut, P., Logar, J., & Majes, B. (2005). Back analyses of anchored bored-pile walls *Proceedings of the 16th International Conference on Soil Mechanics and Geotechnical Engineering* (Vol. 2, pp. 1001-1004). Rotterdam: Millpress.

APPENDIX A: NONLINEAR FINITE ELEMENT ANALYSIS

A.1 Pseudo-code for conventional and modified nonlinear FE analysis

<table>
<tr><th>Conventional nonlinear FE algorithm</th><th>Modified nonlinear FE algorithm</th></tr>
<tr><td>
Read input file
Set $u_0 = 0$, $F(u_0) = 0$, $D = D_e$
Form $K(u_0) = K_e$
for $i = 1,\ldots, n_{\text{load increment}}$
  for $j = 1$, $maxit_{\text{NR}}$

    Solve $K(u_{ij})\Delta u_{ij} = F_i - F(u_{i(j-1)})$
    $u_i = u_{i(j-1)} + \Delta u_{ij}$
    for all elements
      for all Gauss points
        Form elastic strain increment $\Delta\varepsilon_{ij} = B\Delta u_{ij}$
        Form elastic stress increment $\Delta\sigma_{ij} = D\Delta\varepsilon_{ij}$
        Total elastic trial stress $\sigma_i = \Delta\sigma_{ij} + \sigma_{i(j-1)}$
        if $f(\sigma_i) > 0$ then
          Return trial stress σ_i to the MC yield surface
          Set $D = D_{ep}$
        else
          Set $D = D_e$
        end if
      end for (all Gaussian points)
    end for (all elements)
    Form $K(u_{ij})$, $F(u_{ij})$
    if $\dfrac{\|F_i - F(u_{ij})\|}{\|F_i\|} \le NR\_tol$ then
      $u = u_i$
      STOP NR_iteration
    end if
  end for (NR iteration)
end for (load increment)
</td><td>
Read input file
Set $u_0 = 0$, $F(u_0) = 0$, $D = D_e$, $\Delta = 0$
Form $K(u_0) = K_e$
for $i = 1,\ldots, n_{\text{load increment}}$
  for $j = 1$, $maxit_{\text{NR}}$
    $K(u_{ij}) = K_e + \Delta(u_{ij})$
    Solve $K(u_{ij})\Delta u_{ij} = F_i - F(u_{i(j-1)})$
    $u_i = u_{i(j-1)} + \Delta u_{ij}$
    for all elements
      for all Gauss points
        Form elastic strain increment $\Delta\varepsilon_{ij} = B\Delta u_{ij}$
        Form elastic stress increment $\Delta\sigma_{ij} = D\Delta\varepsilon_{ij}$
        Total elastic trial stress $\sigma_i = \Delta\sigma_{ij} + \sigma_{i(j-1)}$
        if $f(\sigma_i) > 0$ then
          Return trial stress σ_i to the MC yield surface
          Set $M = D_{ep} - D_e$
        else
          Set $M = 0$
        end if
      end for (all Gaussian points)
    end for (all elements)
    Form $\Delta(u_{ij})$, $F(u_{ij})$
    if $\dfrac{\|F_i - F(u_{ij})\|}{\|F_i\|} \le NR\_tol$ then
      $u = u_i$
      STOP NR_iteration
    end if
  end for (NR iteration)
end for (load increment)
</td></tr>
</table>

A.2 Formulation of continuum tangent stiffness stress-strain matrix for Mohr-Coulomb model

A.2.1 Rounding of Mohr-Coulomb yield surface

With c and ϕ are cohesion and friction angle respectively, the Mohr Coulomb yield criterion can be expressed as Eq.(1). Figure 1a shows the Mohr-Coulomb yield surface at octahedral plane view.

$$f(\sigma) = \frac{1}{3} I_1 + \sqrt{J_2} A_f(\theta) - c\cos\phi \tag{1}$$

$$A_f(\theta) = \cos\theta - \frac{\sin\theta\cos\phi}{\sqrt{3}} \tag{2}$$

with stress component is $\{\sigma\}^T = [\sigma_x \quad \sigma_y \quad \sigma_z \quad \tau_{xy} \quad \tau_{yz} \quad \tau_{zx}]$, θ is Lode angle, and I_1, J_2 are stress invariants.

With dilation angle ψ, the plastic potential of Mohr-Coulomb yield function is given as

$$g(\sigma) = \frac{1}{3} I_1 + \sqrt{J_2} A_g(\theta) - c\cos\psi = 0 \tag{3}$$

$$A_g(\theta) = \cos\theta - \frac{\sin\theta\cos\psi}{\sqrt{3}} \tag{4}$$

The non-associated flow rule applies when dilation angle ψ in Eq.(3) is different from the friction angle ϕ in Eq. (1).

Figure 1b shows there are gradient discontinuities of MC yield surface at $\theta = \pm 30$ (the corners) and $J_2 = 0$ (the apex). Sloan and Booker[157] (1986) and Abbo and Sloan[3] (1993) propose rounding function to remove these singularities as in Eq. (5)

$$f(\sigma) = \frac{1}{3} I_1 + \sqrt{J_2 A_f^2(\theta) + \alpha^2 \sin^2\phi} - c\cos\phi = 0 \tag{5}$$

with α is an adjustable variable for rounding at the apex. The smaller α is, the closer the Eq. (5) is to Eq. (1). Abbo and Sloan[3] (1993) recommend $\alpha = 0.05c\cot\phi$. With θ_T is a specified transition angle, function $A_f(\theta)$ is adjusted as followings

$$A_f(\theta) = \begin{cases} A - B\sin 3\theta & |\theta| > \theta_T \\ \cos\theta - \dfrac{\sin\theta\cos\phi}{\sqrt{3}} & |\theta| \le \theta_T \end{cases} \tag{6}$$

$$A = \frac{1}{3}\cos\theta_T\left(3 + \tan\theta_T\,\tan 3\theta_T + \frac{1}{\sqrt{3}}sign(\theta)(\tan 3\theta_T - 3\tan\theta_T)\sin\phi\right) \tag{7}$$

$$B = \frac{1}{3\cos 3\theta_T}\left(sign(\theta)\sin\theta_T + \frac{1}{\sqrt{3}}\sin\phi\cos\theta_T\right) \tag{8}$$

$$sign(\theta) = \begin{cases} +1 & \text{for } \theta \ge 0^o \\ -1 & \text{for } \theta < 0^o \end{cases} \tag{9}$$

A.2.2 Return mapping method and continuum tangent stiffness stress-strain matrix for Mohr-Coulomb model

When NR method is employed, the stress vectors of Gauss points deducted from the tangential linear system may be overestimated and lie outside the MC yield surface. This also implies the points have yielded. In that case, the yield function value $f(\sigma)$ is greater than the stress relative error tolerance BE_tol (Abbo[2], 1997) and the Gauss points have to be dragged back the yield surface. Hence, return mapping procedure is the process returning the stress vectors to the yield surface.

The algorithm backward Euler method and forming of the tangent global stiffness matrix for MC model follows Crisfield[46] (1987). All the derivatives are taken at the trial stress. When one vector return is applicable (Figure 2a), the stress returned to the yield surface is calculated as in Eq.(10). The continuum tangent stiffness matrix is formed as in Eq.(1).

$$\sigma_{new} = \sigma_{trial} - \Delta\lambda D_e\frac{\partial f}{\partial\sigma} \tag{10}$$

with

$$\Delta\lambda = \frac{f(\sigma_{trial})}{\left(\frac{\partial f}{\partial \sigma}\right)^T D_e \frac{\partial g}{\partial \sigma}} \tag{11}$$

When two-vectored return is applicable (Figure 2b), the stress returned to the yield surface is calculated as in Eq.(12).

$$\sigma_{new} = \sigma_{trial} - \Delta\lambda_1 D_e \frac{\partial f_1}{\partial \sigma} - \Delta\lambda_2 D_e \frac{\partial f_2}{\partial \sigma} \tag{12}$$

with

$$\Delta\lambda_1 = \frac{bf_1(\sigma_{trial}) - d_1 f_2(\sigma_{trial})}{ab - d_1 d_2} \tag{13}$$

$$\Delta\lambda_2 = \frac{af_2(\sigma_{trial}) - d_2 f_1(\sigma_{trial})}{ab - d_1 d_2} \tag{14}$$

The continuum tangent stiffness is formed in Eq.(15)

$$D_{ep} = D_e - \frac{1}{ab - d_1 d_2}\begin{pmatrix} bD_e \frac{\partial g_1}{\partial \sigma}\frac{\partial f_1}{\partial \sigma}^T D_e - d_1 D_e \frac{\partial g_1}{\partial \sigma}\frac{\partial f_2}{\partial \sigma}^T D_e \\ -d_2 D_e \frac{\partial g_2}{\partial \sigma}\frac{\partial f_1}{\partial \sigma}^T D_e + aD_e \frac{\partial g_2}{\partial \sigma}\frac{\partial f_2}{\partial \sigma}^T D_e \end{pmatrix} \tag{15}$$

with

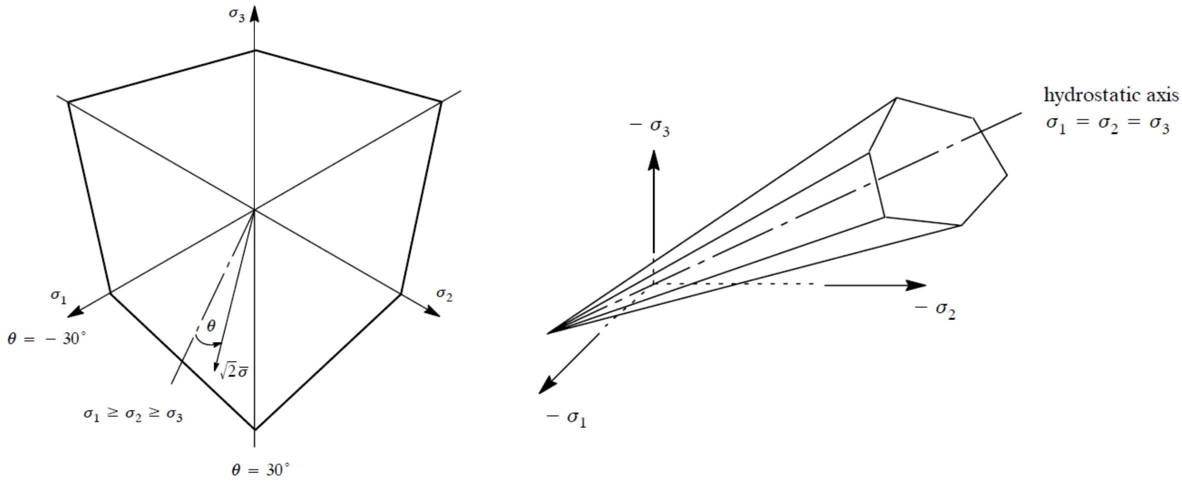
$$a = \frac{\partial f_1}{\partial \sigma}^T D_e \frac{\partial g_1}{\partial \sigma} \tag{16}$$

$$d_1 = \frac{\partial f_1}{\partial \sigma}^T D_e \frac{\partial g_2}{\partial \sigma} \tag{17}$$

$$b = \frac{\partial f_2}{\partial \sigma}^T D_e \frac{\partial g_2}{\partial \sigma} \tag{18}$$

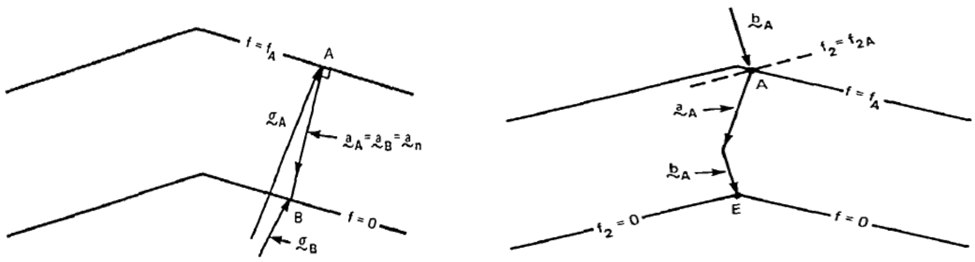
$$d_2 = \frac{\partial f_2}{\partial \sigma}^T D_e \frac{\partial g_1}{\partial \sigma} \tag{19}$$

There are cases that neither one-vectored return nor two-vectored return is applicable and Crisfield[46] (1987) recommends that the stress is returned to the apex of the yield surface. However, the apex is a fixed point and the tangent stiffness matrix at that point is [O] which is not realistic and makes the global stiffness matrix ill-conditioned. The author recommends that if such case occurs, the applied load should be reduced until either one or two-vectored return procedure is applicable.



(a) (b)

Figure 1: Mohr-Coulomb yield surface space in (Abbo[2], 1997): (a) Octahedral plane; (b) Principal stress space



(a) (b)

Figure 2: Backward Euler return mapping method (Crisfield[46],1987): (a) One-vector return; (b) Two-vectored return

APPENDIX B: SOURCE CODE IN FORTRAN 90

B.1 Subroutine for preconditioned IDR(*s*) to solve 1-by-1 block nonsymmetric linear system

```
SUBROUTINE idrs_r (pindx, n, jcsra, icsra,csra, jdel, idel, del,  &
  rhs, s, maxit,tol, matvec,relres,alu,jlu,ju,da)
!-------------------------------------------------------------------
! This subroutine uses preconditioned IDR method solve (A+delta)x=b
! nonsymmetric linear system with a right preconditioner.
! Preconditioners considered are: Jacobi, SSOR, ILU.
!
! Parameters:
!   On input:
!          pindx: preconditioner type
!                 = 0: no preconditioner
!                 = 1: Jacobi preconditioner, input requires DA
!                       vector, which is invert of diagonal entries
!                       of A
!                 = 2: ILU preconditioner, input requires alu, jlu,
!                       ju vectors, which stores incomplete LU
!                       factorization of A. Refer to ILU0 or ILUT
!                       subroutine for definition of each vector.
!                 = 3: SSOR preconditioner, input requires DA vector,
!                       which is invert of diagonal entries of A
!              n: dimension of coefficient matrix A
!jcsra,icsra,csra: CSR storage of coefficient matrix Ke, elastic
!                  stiffness
!  jdel,idel,del: CSR storage of coefficient matrix Delta, A=Kep=
!                 Ke+ Delta
!            rhs: at input, it is right hand vector b
!                 at output,it is returned approximate solution x
!              s: input for IDR method, dimension of shadow space P.
!                 Refer to IDR papers for more information
!          maxit: user-defined maximum iteration count
!            tol: user-defined stopping tolerance
!                 relative residual norm criterion for convergence
!   On output:
!            rhs: approximate solution x
!         matvec: matrix-vector multiplication count when IDR(s)
!                 converges
!         relres: relative residual when IDR(s) converges.
!-------------------------------------------------------------------
! Reference:
```

```fortran
!   Sonneveld, P., & Gijzen, M. B. V. (2008). IDR(s): A family of
!   simple and fast algorithms for solving large nonsymmetric
!   systems of linear equations. SIAM Journal on Scientific
!   Computing, 31(2), 1035-1062.
!------------------------------------------------------------------
IMPLICIT NONE
INTEGER,PARAMETER::iwp=SELECTED_REAL_KIND(15)
INTEGER,INTENT(IN):: pindx,n,jcsra(:),icsra(:),maxit,s
INTEGER,OPTIONAL,INTENT(IN)::jdel(:),idel(:),jlu(:),ju(:)
REAL(iwp),INTENT(IN):: csra(:),tol
REAL(iwp),OPTIONAL,INTENT(IN)::del(:),alu(:),da(:)
INTEGER,INTENT(OUT):: matvec
REAL(iwp),INTENT(OUT):: relres
REAL(iwp),INTENT(INOUT):: rhs(:)
INTEGER::i,j,d,k1,k2,k,sd=2,seed(2),iters,err
REAL(iwp),ALLOCATABLE::x(:),r(:),P(:,:),G(:,:),U(:,:),M(:,:),f(:),&
  v(:),temp(:,:),p1(:),p2(:),c(:),t(:),q(:),qi(:),c1(:),v1(:),tt(:)
REAL(iwp)::angle,normr,tolb,zero=0.0_iwp,omega,ns,nt,ts,rho,alpha,&
  beta,one=1.0_iwp,normb

ALLOCATE(x(n),r(n),P(n,s),G(n,s),U(n,s),M(s,s),f(s),v(n),t(n),    &
  v1(n),tt(n),p1(n),p2(n),c(s),temp(s,s))
!--------------------- Generate Random Matrix P --------------------
IF(s == 1)THEN ! set parameter similar to Bi-CGSTAB for comparison
  angle=zero
  P(:,1) = rhs
ELSE
  angle=0.7_iwp
  seed(1)=2147483562
  seed(2)=1
  CALL RANDOM_SEED (SIZE = sd)
  CALL RANDOM_SEED(PUT=seed(1:sd))
  CALL RANDOM_NUMBER(P)
END IF
!---------------------- Compute initial residual ------------------
x=zero
normb=SQRT(DOT_PRODUCT(rhs,rhs))
!--------------------------- Relative tolerance -------------------
tolb = tol*normb
r=rhs
normr=normb
!------------------------------------------------------------------
IF (normr <= tolb) THEN    ! Initial guess is a good enough solution
   rhs=x
   iters = 0
   matvec=0
   relres = normr/normb
   RETURN
```

```
END IF
!-----------------------------------------------------------------
G=zero
U=zero
M=zero
DO i=1,s
    M(i,i)=one
END DO
omega=one
iters = 0
matvec=0
iteration:DO WHILE ( normr > tolb .AND. iters < maxit )
  f=MATMUL(TRANSPOSE(P),r)
  DO k = 1,s
!------------------ Solve Mc=f using LU decomposition --------------
    IF(s/=1)THEN
      temp(1:s-k+1,1:s-k+1)=M(k:s,k:s)
      c(k:s)=f(k:s)
      CALL lubksb(temp(1:s-k+1,1:s-k+1),s-k+1,c(k:s))
    ELSE
      c(1)=f(1)/M(1,1)
    END IF
!-----------------------------------------------------------------
    v = r - MATMUL(G(:,k:s),c(k:s))
!-------------- Preconditioning: v=invert(preconditioner)*v --------
! If pindx = 0: no preconditioner so do nothing
    SELECT CASE (pindx)
      CASE(1) !Jacobi preconditioner
        IF(PRESENT(da))THEN
          v = da*v
        ELSE
          WRITE(*,'(A)')'Lack of DA vector'
          EXIT
        END IF
      CASE(2)
        IF(PRESENT(alu))THEN
          CALL lusol(n,v,v,alu,jlu,ju)
        ELSE
          WRITE(*,'(A)')'Lack of ILU matrix'
          EXIT
        END IF
      CASE(3)
        IF(PRESENT(da))THEN
!------------------ Solve v = (D+U)^-1*(D)*(D+L)^-1*v --------------
! v1 = (D+L)^-1*v --> Solve (D+L)*v1 = v
! v1 = D*v1
! v = (D+U)^-1*v1 --> Solve (D+U)*v = v1
!-----------------------------------------------------------------
```

```
          CALL lsolve(n,da,icsra,jcsra,csra,v,v1)
          v1 = v1/da
          CALL usolve(n,da,icsra,jcsra,csra,v1,v)
        ELSE
          WRITE(*,'(A)')'Lack of DA vector'
          EXIT
        END IF
    END SELECT
!-------------------------------------------------------------------
    U(:,k) = MATMUL(U(:,k:s),c(k:s)) + omega*v
!------- G(:,k) = MATMUL(A,U(:,k)) -- matrix-vector multiplication -
    CALL csrbx(icsra,jcsra,csra,U(:,k),G(:,k))
    IF (PRESENT(del))THEN
      CALL csrbx(idel,jdel,del,U(:,k),tt)
      G(:,k)=G(:,k)+tt
    END IF
    matvec=matvec+1      ! after matvec, then count
!-------------------------------------------------------------------
    DO i = 1,k-1
      p1=P(:,i)
      p2=G(:,k)
      alpha=DOT_PRODUCT(p1,p2)/M(i,i)
      G(:,k) = G(:,k) - alpha*G(:,i)
      U(:,k) = U(:,k) - alpha*U(:,i)
    END DO
    M(k:s,k) = MATMUL(TRANSPOSE(P(:,k:s)),G(:,k))
    IF (M(k,k)==zero)THEN
      WRITE (*,'(A)')"M(k,k) = 0.  IDR fails!"
      RETURN     !fail
    END IF
    beta = f(k)/M(k,k)
    r = r - beta*G(:,k)
    x = x + beta*U(:,k)
    normr=SQRT(DOT_PRODUCT(r,r))
    iters=iters+1         ! update x then count iters
    IF (normr < tolb.OR. iters == maxit)THEN
      rhs=x
      relres=normr/normb
      RETURN
    END IF
    IF (k < s)f(k+1:s)=f(k+1:s)-beta*M(k+1:s,k)
  END DO
  IF (normr <tolb .OR. iters == maxit )THEN
    rhs=x
    relres=normr/normb
    RETURN
  END IF
!-------------- Preconditioning: v=invert(preconditioner)*r --------
```

```
! If pindx = 0: no preconditioner so do nothing
  SELECT CASE (pindx)
    CASE(0)
      v = r
    CASE(1) !Jacobi preconditioner
      IF(PRESENT(da))THEN
        v = da*r
      ELSE
        WRITE(*,'(A)')'Lack of DA vector'
        EXIT
      END IF
    CASE(2)
      IF(PRESENT(alu))THEN
        CALL lusol(n,r,v,alu,jlu,ju)
      ELSE
        WRITE(*,'(A)')'Lack of ILU matrix'
        EXIT
      END IF
    CASE(3)
      IF(PRESENT(da))THEN
!----------------- Solve v = (D+U)^-1*(D)*(D+L)^-1*r----------------
! v1 = (D+L)^-1*r --> Solve (D+L)*v1 = r
! v1 = D*v1
! v = (D+U)^-1*v1 --> Solve (D+U)*v = v1
!-------------------------------------------------------------------
        CALL lsolve(n,da,icsra,jcsra,csra,r,v1)
        v1 = v1/da
        CALL usolve(n,da,icsra,jcsra,csra,v1,v)
      ELSE
        WRITE(*,'(A)')'Lack of DA vector'
        EXIT
    END IF
  END SELECT
!------------ t=MATMUL(A,v)-- matrix vector multiplication ---------
  CALL csrbx(icsra,jcsra,csra,v,t)
  IF (PRESENT(del))THEN
    CALL csrbx(idel,jdel,del,v,tt)
    t=t+tt
  END IF
  matvec=matvec+1
!----------------------- Computation of a new omega ----------------
    ns = SQRT(DOT_PRODUCT(r,r))
    nt = SQRT(DOT_PRODUCT(t,t))
    rho = DABS(DOT_PRODUCT(t,r)/(nt*ns))
    omega=DOT_PRODUCT(t,r)/(nt*nt)
    IF(s/=1)THEN
      IF ( rho < angle ) omega = omega*angle/rho
    END IF
```

```
    IF (omega==zero)THEN
      WRITE (*,'(A)')"omega = 0.IDR fails!"
      RETURN      !fail
    END IF
!------------------------------------------------------------------
  r = r - omega*t
  x = x + omega*v
  normr = SQRT(DOT_PRODUCT(r,r))
  iters = iters + 1
  IF (normr <tolb .OR. iters == maxit )THEN
    rhs=x
    relres=normr/normb
    RETURN
  END IF
END DO iteration
RETURN
END SUBROUTINE idrs_r

!-----------------DIRECT LU decomposition and solver----------------
SUBROUTINE ludcmp(a,n,indx,d)
!------------------------------------------------------------------
! This subroutine form the LU decomposition of a square matrix
! On input
!    a          : square matrix, output as LU decomposition
!    n          : dimension of [a]
! On output
!    indx : vector recording the row permutation effected by
!                  the partial pivoting
!    d          = 1 if the number of row interchanges is even
!               = -1 if the number of row interchanges is odd
! Reference: Numerical recipes
!                by w.h. press, b. p. flannery, s.a. teukolsky and
!                w.t. vetterling, cambridge university press, 1986
!------------------------------------------------------------------
INTEGER,PARAMETER::iwp=SELECTED_REAL_KIND(15)
REAL(iwp),INTENT(INOUT)::a(:,:)
REAL(iwp)::amax,dum,sum,tiny=1.5e-16_iwp
REAL(iwp),ALLOCATABLE::vv(:)
INTEGER,INTENT(IN)::n
INTEGER,INTENT(OUT)::indx(:),d
INTEGER:: i,j,k,ii,ll,imax

ALLOCATE(vv(n))
d=1
 DO i=1,n
   amax=0.0_iwp
   DO j=1,n
     IF (abs(a(i,j))>= amax) amax=abs(a(i,j))
```

```
    END DO ! j loop
    IF(amax .LT. 0.0_iwp) THEN
      d = 1  !fail
      RETURN
    END IF
    vv(i) = 1.0_iwp / amax
  END DO ! i loop

  DO j=1,n
    DO i=1,j-1
      sum = a(i,j)
      DO k=1,i-1
        sum = sum - a(i,k)*a(k,j)
      END DO ! k loop
      a(i,j) = sum
    END DO ! i loop
    amax = 0.0_iwp
    DO i=j,n
      sum = a(i,j)
      DO k=1,j-1
        sum = sum - a(i,k)*a(k,j)
      END DO ! k loop
      a(i,j) = sum
      dum = vv(i)*dabs(sum)
      IF(dum .ge. amax) THEN
        imax = i
        amax = dum
      END IF
    END DO ! i loop

    IF(j .ne. imax) THEN
      DO k=1,n
        dum = a(imax,k)
        a(imax,k) = a(j,k)
        a(j,k) = dum
      END DO ! k loop
      d = -d
      vv(imax) = vv(j)
    END IF

    indx(j) = imax
    IF(dabs(a(j,j)) < tiny) a(j,j) = tiny

    IF(j .ne. n) THEN
      dum = 1.0_iwp / a(j,j)
      DO i=j+1,n
        a(i,j) = a(i,j)*dum
      END DO ! i loop
```

```
    END IF
  END DO ! j loop
  RETURN
  END SUBROUTINE ludcmp
!-------------------------------------------------------------------
SUBROUTINE lubksb(a,n,b)
!-------------------------------------------------------------------
! This subroutine solves the linear system [a]{x}={b}
! On input
!     a    : LU decomposition from the "ludcmp" subroutine
!     n    : dimension of [a]
!     indx: permutation vector returned by "ludcmp" subroutine
!     b    : right-hand-side vector
! On output
!     b    : solution {x}
!-------------------------------------------------------------------
INTEGER,PARAMETER::iwp=SELECTED_REAL_KIND(15)
REAL(iwp),INTENT(INOUT)::a(:,:),b(:)
REAL(iwp)::sum
INTEGER,ALLOCATABLE::indx(:)
INTEGER,INTENT(IN):: n
INTEGER::d,ii,i,ll,j
!
ALLOCATE(indx(n))
CALL ludcmp(a,n,indx,d)
!
ii = 0
DO i=1,n
     ll = indx(i)
     sum = b(ll)
     b(ll) = b(i)
     IF(ii .NE. 0) THEN
     DO j=ii,i-1
               sum = sum - a(i,j)*b(j)
     END DO ! j loop
     ELSE IF(sum .NE. 0.0_iwp) THEN
     ii = i
     END IF
     b(i) = sum
END DO ! i loop
DO i=n,1,-1
     sum = b(i)
     IF(i < n) THEN
     DO j=i+1,n
               sum = sum - a(i,j)*b(j)
     END DO ! j loop
     END IF
     b(i) = sum / a(i,i)
```

```
END DO ! i loop
RETURN
END SUBROUTINE lubksb
!-------------------- END OF DIRECT LU --------------------------

!-----------------------------------------------------------------
SUBROUTINE lusol(n, y, x, alu, jlu, ju)
IMPLICIT NONE
INTEGER,PARAMETER::iwp=SELECTED_REAL_KIND(15)
REAL(iwp),INTENT(IN)::y(:), alu(:)
REAL(iwp),INTENT(OUT)::x(:)
INTEGER,INTENT(IN):: n, jlu(:), ju(:)
!-----------------------------------------------------------------
! This routine solves the system (LU) x = y,
! given an LU decomposition of a matrix stored in (alu, jlu, ju)
! modified sparse row format
!-----------------------------------------------------------------
! on entry:
! n   = dimension of system
! y   = the right-hand-side vector
! alu, jlu, ju
!     = the LU matrix as provided from the ILU routines.
!
! on return
! x   = solution of LU x = y.
!-----------------------------------------------------------------
! Note: routine is in place: call lusol (n, x, x, alu, jlu, ju)
!       will solve the system with rhs x and overwrite the result on
x .
!-----------------------------------------------------------------
! local variables
!
        integer:: i,k
!
! forward solve
!
        do i = 1, n
           x(i) = y(i)
           do k=jlu(i),ju(i)-1
              x(i) = x(i) - alu(k)* x(jlu(k))
           end do
        end do
!
!     backward solve.
!
     do i = n, 1, -1
        do k=ju(i),jlu(i+1)-1
              x(i) = x(i) - alu(k)*x(jlu(k))
```

```
       end do
           x(i) = alu(i)*x(i)
    end do
!
RETURN
END SUBROUTINE lusol
```

B.2 Subroutine for preconditioned IDR(*s*) to solve 2-by-2 block nonsymmetric linear system

```
!-----------------------------------------------------------------
SUBROUTINE  idrs_blkp(pindx,n,ns,jcsra,icsra,csra,jdel,idel,del,rhs,s,
&
  maxit,tol,matvec,relres,icsrs,jcsrs,csrs,slu,sjlu,sju,da1,schrlu,
&
  schrjlu,schrju,gjal,icsrbt,jcsrbt,csrbt)
!-----------------------------------------------------------------
! This subroutine uses preconditioned IDR method solve
! (A + delta) x=b
! nonsymmetric linear system with a right block preconditioner.
! pindx = 4: Block diagonal preconditioner is of the form
!          |K~       0|      |v1|
!     M =  |0       S~|  v = |v2|
!     Solve Mv = u by: v1 = invert(K~)*u1
!                      v2 = invert(S~)*u2
! pindx = 5: Block constrained preconditioner is of the form
!          |K~       B|      |v1|
!     M =  |B'      S~|  v = |v2|
!     Solve Mv = u by: w = (K~)^-1*u1
!                      z = (S~)^-1*(B'*w - u2)
!                      v1 = (K~)^-1*(v1-B*z)
!                      v2 = z
! Parameters:
!   On input:
!          pindx: index indicating which block preconditioner is
!                  used
!          = 421: use K2 = ILU0 of K,  S1 = diag(B'*diag(K)^-1*B + C)
!          = 422: use K2 = ILU0 of K,  S1 = ILU0(B'*diag(K)^-1*B + C)
!          = 431: use K3 = SSOR of K,  S1 = diag(B'*diag(K)^-1*B + C)
!          = 432: use K3 = SSOR of K,  S2 = ILU0(B'*diag(K)^-1*B + C)
!              n: dimension of coefficient matrix A
!             ns: number of soil dof, n-ns = number of fluid dof
!   jcsra,icsra,
!           csra: CSR storage of coefficient matrix A
!   jdel,idel,del:CSR storage of matrix delta
!            rhs: at input, it is right hand vector b
!                 at output,it is returned approximate solution x
!            pre: preconditioner, from "form_preconditioner"
!              s: input for IDR method
!          maxit: user-defined maximum iteration count;
!            tol: it is the user-defined stopping tolerance;
!                 relative residual norm criterion (x0=.0) for
!                 convergence
!       alu,jlu : matrix stored in Modified Sparse Row (MSR) format
!                 containing the L and U factors together. The
```

```
!                  diagonal (stored in alu(1:n) ) is inverted. Each
!                  i-th row of the alu,jlu matrix contains the i-th
!                  row of L (excluding the diagonal entry=1) followed
!                  by the i-th row of U.
!             ju : pointer to the diagonal elements in alu, jlu.!
!   On output:
!            rhs: approximate solution x
!          iters: the iterative count when PCG converges;
!         relres: the relative residual when PCG converges.
!-----------------------------------------------------------------
! Reference:
!   Sonneveld, P., & Gijzen, M. B. V. (2008). IDR(s): A family of
!   simple and fast algorithms for solving large nonsymmetric
!   systems of linear equations. SIAM Journal on Scientific
!   Computing, 31(2), 1035-1062.
!-----------------------------------------------------------------
IMPLICIT NONE
INTEGER,PARAMETER::iwp=SELECTED_REAL_KIND(15)
INTEGER,INTENT(IN):: n,ns,jcsra(:),icsra(:),maxit,s,pindx
INTEGER,OPTIONAL,INTENT(IN)::jdel(:),idel(:),icsrs(:),jcsrs(:),    &
  sjlu(:),sju(:),schrju(:),schrjlu(:),icsrbt(:),jcsrbt(:)
REAL(iwp),INTENT(IN):: csra(:),tol
REAL(iwp),OPTIONAL,INTENT(IN)::da1(:),del(:),csrs(:),slu(:),       &
  schrlu(:),gjal,csrbt(:)
INTEGER,INTENT(OUT):: matvec
REAL(iwp),INTENT(OUT):: relres
REAL(iwp),INTENT(IN OUT):: rhs(:)
INTEGER::i,j,d,k1,k2,k,sd=2,seed(2),iters
INTEGER,ALLOCATABLE::indx(:)
REAL(iwp),ALLOCATABLE::x(:),r(:),P(:,:),G(:,:),U(:,:),M(:,:),f(:),&
   v(:),temp(:,:),p1(:),p2(:),c(:),t(:),q(:),qi(:),c1(:),v1(:),    &
   tt(:),v2(:),v3(:)
REAL(iwp)::angle,normr,tolb,zero=0.0_iwp,omega,nr,nt,ts,rho,   &
    alpha,beta,r0,one=1.0_iwp,normb
!--------------------- Generate Random Matrix P --------------------
IF(s==1)THEN
    angle=zero
ELSE
    angle=0.7_iwp
END IF
ALLOCATE(x(n),r(n),P(n,s),f(s),v(n),t(n),v1(n),v2(ns),v3(n),tt(n))
! generate random matrix P
seed(1)=2147483560
seed(2)=1
CALL RANDOM_SEED (SIZE = sd)
CALL RANDOM_SEED(PUT=seed(1:sd))
CALL RANDOM_NUMBER(P)
IF(s==1)P(:,1)=rhs
```

```
!----------------------- Compute initial residual ------------------
x=zero
normb=SQRT(DOT_PRODUCT(rhs,rhs))
!---------------------------- Relative tolerance -------------------
tolb = tol*normb
r=rhs
normr=normb
!------------------------------------------------------------------
IF (normr <= tolb) THEN          ! Initial guess is a good enough
solution
   rhs=x
   iters = 0
   matvec=0
   relres = normr/normb
   RETURN
END IF
!------------------------------------------------------------------
ALLOCATE(G(n,s),U(n,s),M(s,s))
G=zero
U=zero
M=zero
DO i=1,s
    M(i,i)=one
END DO
omega=one
iters = 0
matvec=0
ALLOCATE(p1(n),p2(n),c(s),c1(s))
k=1
ALLOCATE(temp(s-k+1,s-k+1),indx(s-k+1))
iteration:DO WHILE ( normr > tolb .AND. iters < maxit )
    f=MATMUL(TRANSPOSE(P),r)
    DO k = 1,s
!------------------ Solve Mc=f using LU decomposition --------------
        temp(1:s-k+1,1:s-k+1)=M(k:s,k:s)
        ! Solve Mc=f using LU decomposition
        c(k:s)=f(k:s)
        CALL lubksb(temp(1:s-k+1,1:s-k+1),s-k+1,c(k:s))
!------------------------------------------------------------------
        v = r - MATMUL(G(:,k:s),c(k:s))
!-------------- Preconditioning: v=invert(preconditioner)*v --------
        SELECT CASE(pindx)
          CASE(400:500)
!--------------------- Compute v(1:ns)=invert(K~)*v ----------------
            SELECT CASE(MOD(pindx,100)-MOD(pindx,10))
              CASE(10)! K1 = diag(K)
                v(1:ns)=da1(1:ns)*v(1:ns)
              CASE(20)! K2 = ILU0(K)
```

```
                CALL lusol(ns,v(1:ns),v(1:ns),slu,sjlu,sju)
              CASE(30)! K3 = SSOR(K)
              ! Solve (L+D)*(D^-1)*(U+D)*v1(1:ns) = v(1:ns)
              ! v1 = (D+L)^-1*v --> Solve (D+L)*v1 = v
              ! v1 = D*v1
              ! v = (D+U)^-1*v1 --> Solve (D+U)*v = v1
                CALL
lsolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v(1:ns),v1(1:ns))
                v1(1:ns)=v1(1:ns)/da1(1:ns)
                CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v1(1:ns),v(1:ns))
            END SELECT
!----------------- Compute v(ns+1:n)=invert(S~)*v(ns+1:n) ----------
            SELECT CASE(MOD(pindx,10))
              CASE(1)! S1 = diag(S)
                v(ns+1:n)=da1(ns+1:n)*v(ns+1:n)
              CASE(2,3)! S2 = ILU0(B'*diag(K)^-1*B + C)
                CALL lusol(n-
ns,v(ns+1:n)/gjal,v(ns+1:n),schrlu,schrjlu, &
                  schrju)
            END SELECT
!-------------------------------------------------------------------
!     Solve Mv = v by: v1(1:ns) = (K~)^-1*v(1:ns)
!                      z = (S~)^-1*(B'*v1(1:ns) - v(ns+1:n))
!                      v1(1:ns) = (K~)^-1*(v(1:ns)-B*z)
!                      v(ns+1:n) = z
          CASE(501:)
!---------------------- v1(1:ns) = (K~)^-1*v(1:ns) ----------------
            SELECT CASE(MOD(pindx,100)-MOD(pindx,10))
              CASE(10)! K1 = diag(K)
                v1(1:ns)=da1(1:ns)*v(1:ns)
              CASE(20)! K2 = ILU0
                CALL lusol(ns,v(1:ns),v1(1:ns),slu,sjlu,sju)
              CASE(30)! K3 = SSOR(K)
              ! Solve (L+D)*(D^-1)*(U+D)*v1(1:ns) = v(1:ns)
              ! v2 = (D+L)^-1*v --> Solve (D+L)*v2 = v
              ! v2 = D*v2
              ! v1 = (D+U)^-1*v2 --> Solve (D+U)*v1 = v2
                CALL
lsolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v(1:ns),v2(1:ns))
                v2(1:ns)=v2(1:ns)/da1(1:ns)
                !CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v1(1:ns),v(1:ns))
                CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v2(1:ns),v1(1:ns))
                !v1(1:ns)=v(1:ns)
            END SELECT
!------------ v1(ns+1:n) = (S~)^-1*(B'*v1(1:ns) - v(ns+1:n)) -------
```

```
            CALL
csrbx(icsrbt,jcsrbt,csrbt,v1(1:ns),v1(ns+1:n)) !B'*v1(1:ns)
            v1(ns+1:n)=v1(ns+1:n)-v(ns+1:n)
            SELECT CASE(MOD(pindx,10))
              CASE(1)
                v1(ns+1:n)=da1(ns+1:n)*v1(ns+1:n)
              CASE(2,3)
                 CALL lusol(n-ns,v1(ns+1:n),v3(1:n-
ns),schrlu,schrjlu,     &
                   schrju)
                 v1(ns+1:n)=v3(1:n-ns)
            END SELECT
!--------------- v1(1:ns) = (K~)^-1*(v(1:ns)-B*v1(ns+1:n)) ---------
            v(ns+1:n)=v1(ns+1:n) !final vector
            CALL csrbtx(icsrbt,jcsrbt,csrbt,v1(ns+1:n),v1(1:ns))
            v1(1:ns)=v(1:ns)-v1(1:ns)
            SELECT CASE(MOD(pindx,100)-MOD(pindx,10))
              CASE(10)
                v(1:ns)=da1(1:ns)*v1(1:ns)
              CASE(20)
                CALL lusol(ns,v1(1:ns),v(1:ns),slu,sjlu,sju) ! final
vector
              CASE(30)! K3 = SSOR(K)
              ! Solve (L+D)*(D^-1)*(U+D)*v(1:ns) = v1(1:ns)
              ! v = (D+L)^-1*v1 --> Solve (D+L)*v = v1
              ! v = D*v
              ! v1 = (D+U)^-1*v --> Solve (D+U)*v1 = v
                CALL
lsolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v1(1:ns),v(1:ns))
                v(1:ns)=v(1:ns)/da1(1:ns)
                CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v(1:ns),v1(1:ns))
                !CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v(1:ns),v(1:ns))
                v(1:ns)=v1(1:ns)
            END SELECT
        END SELECT
!-------------------------------------------------------------------
        U(:,k) = MATMUL(U(:,k:s),c(k:s)) + omega*v
!------- G(:,k) = MATMUL(A,U(:,k)) -- matrix-vector multiplication -
        CALL csrbx(icsra,jcsra,csra,U(:,k),G(:,k))
        IF (PRESENT(del))THEN
          CALL csrbx(idel,jdel,del,U(:,k),tt)
          G(:,k)=G(:,k)+tt
        END IF
        matvec=matvec+1      ! after matvec, then count
!-------------------------------------------------------------------
-------
```

```
        DO i = 1,k-1
            p1=P(:,i)
            p2=G(:,k)
            alpha=DOT_PRODUCT(p1,p2)/M(i,i)
            G(:,k) = G(:,k) - alpha*G(:,i)
            U(:,k) = U(:,k) - alpha*U(:,i)
        END DO
        M(k:s,k) = MATMUL(TRANSPOSE(P   (:,k:s)),G(:,k))
        IF (M(k,k)==zero)THEN
            WRITE (*,'(A)')"M(k,k) = 0. IDR fails!"
            RETURN      !fail
        END IF
        beta = f(k)/M(k,k)
        r = r - beta*G(:,k)
        x = x + beta*U(:,k)
        normr=SQRT(DOT_PRODUCT(r,r))
        iters=iters+1        ! update x then count iters
        IF (normr < tolb.OR. iters == maxit)THEN
            rhs=x
            relres=normr/normb
            RETURN
        END IF
        IF (k < s)f(k+1:s)=f(k+1:s)-beta*M(k+1:s,k)
    END DO
    IF (normr <tolb .OR. iters == maxit )THEN
        rhs=x
        relres=normr/normb
        RETURN
    END IF
!-------------- Preconditioning: v=invert(preconditioner)*r --------
    SELECT CASE(pindx)
      CASE(400:500)
!--------------------- Compute v(1:ns)=invert(K~)*r ----------------
        SELECT CASE(MOD(pindx,100)-MOD(pindx,10))
          CASE(10)! K1 = diag(K)
            v(1:ns)=da1(1:ns)*r(1:ns)
          CASE(20)! K2 = ILU0(K)
            CALL lusol(ns,r(1:ns),v(1:ns),slu,sjlu,sju)
          CASE(30)! K3 = SSOR(K)
          ! Solve (L+D)*(D^-1)*(U+D)*v1(1:ns) = r(1:ns)
          ! v1 = (D+L)^-1*r --> Solve (D+L)*v1 = r
          ! v1 = D*v1
          ! v = (D+U)^-1*v1 --> Solve (D+U)*v = v1
            CALL
lsolve(ns,da1(1:ns),icsrs,jcsrs,csrs,r(1:ns),v1(1:ns))
            v1(1:ns)=v1(1:ns)/da1(1:ns)
            CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v1(1:ns),v(1:ns))
```

```
        END SELECT
!------------------ Compute v(ns+1:n)=invert(S~)*r(ns+1:n) ----------
        SELECT CASE(MOD(pindx,10))
          CASE(1)! S1 = diag(S)
            v(ns+1:n)=da1(ns+1:n)*r(ns+1:n)
          CASE(2,3)! S = ILU0
            CALL lusol(n-
ns,r(ns+1:n)/gjal,v(ns+1:n),schrlu,schrjlu,schrju)
        END SELECT
!-------------------------------------------------------------------
!     Solve Mv = r by: v1(1:ns) = (K~)^-1*r(1:ns)
!                      v1(ns+1:n) = (S~)^-1*(B'*v1(ns+1:n) -
!                                   r(ns+1:n))
!                      v(1:ns) = (K~)^-1*(r(1:ns)-B*v1(ns+1:n))
!                      v(ns+1:n) = v1(ns+1:n)
      CASE(501:)
!------------------ Compute v1(1:ns)=invert(K~)*r(1:ns) ------------
        SELECT CASE(MOD(pindx,100)-MOD(pindx,10))
          CASE(10)
            v1(1:ns)=da1(1:ns)*r(1:ns)
          CASE(20)
            CALL lusol(ns,r(1:ns),v1(1:ns),slu,sjlu,sju)
          CASE(30)! K3 = SSOR(K)
          ! Solve (L+D)*(D^-1)*(U+D)*v1(1:ns) = r(1:ns)
          ! v2 = (D+L)^-1*r --> Solve (D+L)*v2 = r
          ! v2 = D*v2
          ! v1 = (D+U)^-1*v2 --> Solve (D+U)*v1 = v2
            CALL
lsolve(ns,da1(1:ns),icsrs,jcsrs,csrs,r(1:ns),v2(1:ns))
            v2(1:ns)=v2(1:ns)/da1(1:ns)
            !CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v1(1:ns),v(1:ns))
            CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v2(1:ns),v1(1:ns))
        END SELECT
! ----------- v1(ns+1:n) = (S~)^-1*(B'*v1(ns+1:n) - r(ns+1:n)) -----
        CALL csrbx(icsrbt,jcsrbt,csrbt,v1(1:ns),v1(ns+1:n))
        v1(ns+1:n)=v1(ns+1:n)-r(ns+1:n)
        SELECT CASE(MOD(pindx,10))
          CASE(1)
            !v1(ns+1:n)=da1(ns+1:n)*(v1(ns+1:n)-r(ns+1:n))
            v1(ns+1:n)=da1(ns+1:n)*v1(ns+1:n)
          CASE(2,3)
            !CALL lusol(n-
ns,v1(ns+1:n),v1(ns+1:n),schrlu,schrjlu,schrju)
            CALL lusol(n-ns,v1(ns+1:n),v3(1:n-
ns),schrlu,schrjlu,schrju)
            v1(ns+1:n)=v3(1:n-ns)
```

```
        END SELECT
!---------------- v(1:ns) = (K~)^-1*(r(1:ns)-B*v1(ns+1:n))-----------
        v(ns+1:n)=v1(ns+1:n) !final vector
        CALL csrbtx(icsrbt,jcsrbt,csrbt,v1(ns+1:n),v1(1:ns))
        v1(1:ns)=r(1:ns)-v1(1:ns)
        SELECT CASE(MOD(pindx,100)-MOD(pindx,10))
          CASE(10)
            v(1:ns)=da1(1:ns)*v1(1:ns)
          CASE(20)
            CALL lusol(ns,v1(1:ns),v(1:ns),slu,sjlu,sju) ! final
vector
          CASE(30)! K3 = SSOR(K)
          ! Solve (L+D)*(D^-1)*(U+D)*v(1:ns) = v1(1:ns)
          ! v2 = (D+L)^-1*v1 --> Solve (D+L)*v2 = v1
          ! v2 = D*v2
          ! v = (D+U)^-1*v2 --> Solve (D+U)*v = v2
            CALL
lsolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v1(1:ns),v2(1:ns))
            v2(1:ns)=v2(1:ns)/da1(1:ns)
            CALL
usolve(ns,da1(1:ns),icsrs,jcsrs,csrs,v2(1:ns),v(1:ns))
        END SELECT
    END SELECT
!------------ t=MATMUL(A,v)-- matrix vector multiplication ---------
    CALL csrbx(icsra,jcsra,csra,v,t)
    IF (PRESENT(del))THEN
      CALL csrbx(idel,jdel,del,v,tt)
      t=t+tt
    END IF
    matvec=matvec+1
!----------------------- Computation of a new omega ----------------
    nr = SQRT(DOT_PRODUCT(r,r))
    nt = SQRT(DOT_PRODUCT(t,t))
    !ts = DOT_PRODUCT(t,r)
    rho = ABS(DOT_PRODUCT(t,r)/(nt*nr))
    !om=ts/(nt*nt)
    omega=DOT_PRODUCT(t,r)/DOT_PRODUCT(t,t)
    IF ( rho < angle ) omega = omega*angle/rho
    IF (omega==zero)THEN
        WRITE (*,'(A)')"omega = 0. IDR fails!"
        RETURN     !fail
    END IF
!-------------------------------------------------------------------
    r = r - omega*t
    x = x + omega*v
    normr = SQRT(DOT_PRODUCT(r,r))
    iters = iters + 1
    WRITE(*,*)iters,"  ",normr/normb
```

```
    IF (normr <tolb .OR. iters == maxit )THEN
        rhs=x
        relres=normr/normb
        RETURN
    END IF
END DO iteration
RETURN
END SUBROUTINE idrs_blkp
```