INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING
Volume 1, Number 1, Pages 1–20



# APPLICATION OF AN ADAPTIVE PRECONDITIONER FOR SYSTEMS OF PDES IN RESERVOIR SIMULATION

CAROLINA DIAZ-GOANO, OLABANJO AJEGBOMOGUN, YVES ACHDOU, FREDERIC NATAF, ROLAND MASSON, AND PHILIPPE QUANDALLE

**Abstract.** Reservoir simulation has become an important tool for managing oil and gas reservoirs. Reservoir simulation involves solving systems of non-linear, time dependent partial differential equations. The discretization of the governing equations produces large systems of linear equations that have to be solved within the non-linear solver. This is a very computational intensive task, therefore efficient preconditioning techniques are essential for achieving solutions within reasonable times. In our work we investigate the construction of an efficient block incomplete factorization of a sparse block-tridiagonal matrix that arises from the discretization of the partial differential equations. Our preconditioner is based on a Tangential Frequency Filtering Decomposition (TFFD) method [1]. The problem grid is partitioned in $n$ planes according to a given direction, each plane constituting a block in the preconditioner. The preconditioner is built applying a filtering condition subject to a test vector and used in an adaptive and multiplicative way in the context of a preconditioned Krylov subspace method. In this paper we present the results of incorporating the TFFD preconditioning method to our finite volume scheme. The performance of the TFFD preconditioner for different restarting criteria is analyzed and compared to that of the ILU0 preconditioner.

**Key Words.** Preconditioning, Tangential Filtering, Reservoir Simulation.

## 1. Introduction

Reservoir simulation has become a very important tool for managing oil and gas reservoirs. Simulations are used to forecast production and optimize the design schemes in order to enhance oil and gas recovery. Reservoir simulation involves solving coupled system of nonlinear, time dependent partial differential equations. The discretization of the governing equations produces a large system of linear equations that has to be solved at each time step on the non-linear solver. This is a very computational intensive task, therefore a fast and robust solver is essential for achieving solutions within reasonable times. To meet the high requirements of these numerical simulations efficient preconditioning techniques have to be implemented. The aim of our research is to develop an efficient block incomplete factorization of a sparse block-tridiagonal matrix that arises from the discretization of the partial differential equations. The preconditioner is based on a Tangential Frequency Filtering Decomposition (TFFD) proposed by Wagner [10] [11] and Achdou and Nataf [1] but extended to non-symmetric matrices that represent three dimensional problems arising from reservoir simulations. The original system matrix $\mathbf{K}$ admits an exact $\mathbf{LDU}$ block factorization where the off diagonal blocks $L$ and $U$ are the

2000 *Mathematics Subject Classification.* 15A12 , 65F15, 35R20, 35J60 .

same as those in the original matrix. Since the diagonal blocks $D$ are dense, these are replaced by blocks that attempt to preserve the sparsity of the original matrix. A block factorization is performed by a modified induction formula for a given test vector $\mathbf{t}$. The resulting factorization is such that it coincides with the original matrix when the test vector is applied. The preconditioner $\mathbf{M}$ is constructed such that $\mathbf{Mt} = \mathbf{Kt}$. For a given grid partition representing a 3D problem with $m$ cells distributed in $n$ adjacent planes, the preconditioner is constructed in $n$ steps. At each step an approximation of the diagonal block is constructed by imposing the filtering condition for that plane. The preconditioner has a highly recursive structure. The choice of the filtering vector $\mathbf{t}$ affects the convergence. A discussion of this topic can be found in [1]. In the current work we obtain the filtering vector $\mathbf{t}$ from the singular value decomposition applied to the solver (GMRES [8]) working matrix. The main idea for this choice is that after a few iterations of the linear solver, many components of the error have been damped with the remaining eigenmodes being responsible for the slow down in the convergence rate. Choosing a vector from this subspace, we can then construct a preconditioner that eliminates the subspaces in question. The preconditioner $\mathbf{M}$ is said to have a filtering property because in one Richardson iteration $e^{(i+1)} = (\mathbf{I} - \mathbf{M}^{-1}\mathbf{K})e^{(i)}$ filters out the component of the error $e^{(i)}$ parallel to the vector $\mathbf{t}$ [1]. The built preconditioner is then used in a multiplicative way within the iterative method chosen, GMRES in our case. Different test vectors are also combined at different stages of the solving procedure. From our numerical experiments this preconditioner technique gives very good results.

The rest of this document is organized as follows: Section 2 presents some background in filtering factorization. Section 3 describes the numerical construction of the adaptive preconditioner, Section 4 presents a brief description of the computational implementation. Section 5 shows the results of the numerical experiments. Finally, conclusions are presented in Section 6.

## 2. Background

**2.1. The Appeared-Cheshire Nested Factorization.** Nested Factorization was first introduced and described by Appleyard et al. [4], [3]. In their work the authors present this technique for the solution of linear equations arising from finite difference reservoir simulation problems. Their method differ from the incomplete Cholesky factorization in that instead of building the preconditioner from a product of strictly lower and upper triangular factors, they factorize the problem matrix into block lower and block upper matrices.
The nested factorization they proposed starts by defining the problem matrix as:

$$\mathbf{A} = \mathbf{D} + \mathbf{L}_1 + \mathbf{U}_1 + \mathbf{L}_2 + \mathbf{U}_2 + \mathbf{L}_3 + \mathbf{U}_3 \tag{1}$$

where $\mathbf{A}$ is a banded matrix from a three dimensional finite difference problem, and can be regarded as a block tridiagonal matrix because of the gaps in $\mathbf{L}_2$ and $\mathbf{U}_2$ corresponding to the edges of the reservoir. $\mathbf{L}_1, \mathbf{L}_2$ and $\mathbf{L}_3$ are lower bands and $\mathbf{U}_1, \mathbf{U}_2$ and $\mathbf{U}_3$ are the corresponding upper bands [3]. Then, they define the preconditioner $\mathbf{B}$ as [3]:

$$\mathbf{B} = (\mathbf{P} + \mathbf{L}_3)(\mathbf{I} + \mathbf{P}^{-1}) \tag{2}$$

$$\mathbf{P} = (\mathbf{T} + \mathbf{L}_2)(\mathbf{I} + \mathbf{T}^{-1}) \tag{3}$$

$$\mathbf{T} = (\mathbf{M} + \mathbf{L}_1)(\mathbf{I} + \mathbf{M}^{-1}) \tag{4}$$

where $\mathbf{M}$ is a diagonal matrix, $\mathbf{T}$ is tridiagonal and $\mathbf{P}$ is a block diagonal matrix. In their work Appleyard et al. [3] proposed different variations for calculating the

preconditioner based on modifying the definition of $\mathbf{T}$ with the purpose of reducing the effect of the error matrix

**2.2. Filtering Factorization.** Let $\Omega$ be a bounded domain in $\Re^3$. Given a second order elliptic operator $\mathbf{L}$ in $\Omega$, and a set of elliptic partial differential equations

$$\mathcal{L}(v) = f \tag{5}$$

we would like to solve the linear system of equations that arises from discretizing the elliptic operator. Let $\mathbf{K}$ be the matrix obtained from such a discretizations by means of either finite element, volume or differences methods on a structured grid. Then we are interested in solving

$$\mathbf{Kv} = \mathbf{f} \tag{6}$$

The frequency filtering decomposition can be used to construct a preconditioner $\mathbf{M}^{-1}$ to be used in an iterative algorithm

$$\mathbf{u}^{i+1} = \mathbf{u}^i + \mathbf{M}^{-1}(\mathbf{f} - \mathbf{K}\mathbf{u}^i) \tag{7}$$

to solve the system shown in eq. (6). If we partition our domain grid into $n_x$ planes of $n_{y_i} \times n_{z_i}$ cells, then $\mathbf{K}$ has clearly a block tridiagonal structure.

$$\mathbf{K} = \begin{bmatrix} D_1 & U_1 & & \\ L_1 & D_2 & ... & ... \\ & & ... & U_{n_x-1} \\ & & L_{n_x-1} & D_{n_x} \end{bmatrix} \tag{8}$$

Each $D_i$ block being of size $n_{y_i} \times n_{z_i}$, according to the geometry of the problem. Note that in most cases the dimension of the different $D_i$ matrices is not the same. The blocks in $\mathbf{D}$, $\mathbf{L}$ and $\mathbf{U}$ have also block diagonal structure according to the connectivity of the elements in the grid. The matrix $\mathbf{K}$ can be written according to the following factorization:

$$\mathbf{K} = (\mathbf{L} + \mathbf{T})\mathbf{T}^{-1}(\mathbf{U} + \mathbf{T}) \tag{9}$$

where $\mathbf{T} = (T_1, T_2, ...T_{n_x})$ , and $\mathbf{L}$ and $\mathbf{U}$ are defined as:

$$\mathbf{L} = \begin{bmatrix} 0 & & & \\ L_1 & & & \\ & ... & & \\ & & L_{n_x-1} & 0 \end{bmatrix} \mathbf{U} = \begin{bmatrix} 0 & U_1 & & \\ & 0 & ... & ... \\ & & ... & U_{n_x-1} \\ & & & 0 \end{bmatrix} \tag{10}$$

Then $\mathbf{K}$ has the form of:

$$\mathbf{K}= \begin{bmatrix} T_1 & & & \\ L_1 & T_2 & & \\ & ... & & \\ & & L_{n_x-1} & T_{n_x} \end{bmatrix} \begin{bmatrix} T_1^{-1} & & & \\ & T_2^{-1} & & \\ & & ... & \\ & & & T_{n_x}^{-1} \end{bmatrix} \begin{bmatrix} T_1 & U_1 & & \\ & T_2 & & \\ & & ... & U_{n_x-1} \\ & & & T_{n_x} \end{bmatrix} \tag{11}$$

The blocks of the $\mathbf{T}$ matrix, $T_i$ with $i = 1...n_x$ are matrices of order $n_{y_i} \times n_{z_i}$. $T_i$ satisfies the induction formula

$$T_i = \begin{cases} D_1, & i = 1 \\ D_i - L_{i-1}T_{i-1}^{-1}U_{i-1}, & 1 < i \leq n_x \end{cases} \tag{12}$$

In order to build an incomplete block factorization of $\mathbf{K}$ the idea is to replace the blocks from $\mathbf{T}$ by a suitable sparse approximation $\tilde{\mathbf{T}}$. The factorization obtained can then be used as a preconditioner. The preconditioner $\mathbf{M}$ is then built as

$$\mathbf{M} = (\mathbf{L} + \tilde{\mathbf{T}})\tilde{\mathbf{T}}^{-1}(\mathbf{U} + \tilde{\mathbf{T}}) \tag{13}$$

(14)

$$\mathbf{M}= \begin{bmatrix} \tilde{T}_1 & & & \\ L_1 & \tilde{T}_2 & & \\ & & & L_{n_x-1} \quad \tilde{T}_{n_x} \end{bmatrix} \begin{bmatrix} \tilde{T}_1^{-1} & & & \\ & \tilde{T}_2^{-1} & & \\ & & ... & \\ & & & \tilde{T}_{n_x}^{-1} \end{bmatrix} \begin{bmatrix} \tilde{T}_1 & U_1 & & \\ & \tilde{T}_2 & ... & \\ & & ... & U_{n_x-1} \\ & & & \tilde{T}_{n_x} \end{bmatrix}$$

To apply the preconditioner by solving $\mathbf{Mx} = \mathbf{f}$ no matrix needs to be inverted. Instead the procedure requires a block-wise forward and a backward substitution.
1- Block-forward substitution

$$(\mathbf{L} + \tilde{\mathbf{T}})\mathbf{y} = \mathbf{f} \tag{15}$$

2- Block-backward substitution

$$(\mathbf{U} + \tilde{\mathbf{T}})\mathbf{x} = \tilde{\mathbf{T}}\mathbf{y} \tag{16}$$

Different filtering methods exist for the construction of each $T_i$ block. These will be discuss in the following section.

**2.3. Filtering methods.** *Point-wise tangential filtering* and *filtering in the average* are two of the possible methods for constructing the preconditioner. They differ in the transfer matrix $\theta$ chosen to construct the blocks $\tilde{T}_i$ of the preconditioner. A detailed discussion on this topic can be found in [1].

**2.3.1. Point-wise filtering.** Wagner and Wittum [10] [11] proposed a filtering decomposition where the $T_i$ blocks are constructed according to:

$$T_i = \begin{cases} D_1, & i = 1 \\ D_i - \Theta_{i,i-1}T_{i-1}^{-1}\Theta_{i-1,i} - \Theta_{i,i-1}L_{i-1,i} - L_{i,i-1}\Theta_{i-1,i}, & 1 < i \end{cases} \tag{17}$$

They restricted the choice of the transfer matrix to fulfill the filter condition:

$$(\Theta_{i,i-1}T_{i-1}\Theta_{i-1,i} - \Theta_{i,i-1}L_{i-1,i} - L_{i,i-1}\Theta_{i-1,i})t_i = -L_{i,i-1}T_{i-1}^{-1}L_{i-1,i}t_i \tag{18}$$

and suggested that the transfer matrix $\Theta_{i,j}$ be chosen so that the pattern in $T_i$ does not fill up [10]. Wagner [10] defined then the point-wise tangential filtering factorization for symmetric matrices as:

$$\tilde{T}_i = \begin{cases} D_1, & i = 1 \\ D_i - \Theta_i\tilde{T}_{i-1}\Theta_i - \Theta_i L_{i-1}^T - L_{i-1}\Theta_i, & 1 < i \leq n_x \end{cases} \tag{19}$$

with the transfer matrix as:

$$\theta_i t_i = \tilde{T}_{i-1}L_{i-1}^T t_i \tag{20}$$

The choice of such a transfer matrix is due to the fact that if we denote the preconditioner $\mathbf{M}$ as a function of the matrix $\mathbf{K}$ and the test vector $\mathbf{t}$ equation so that $\mathbf{M} = \mathcal{M}(\mathbf{K}, \mathbf{t})$ then equation (20) is equivalent to:

$$\mathcal{M}(\mathbf{K}, \mathbf{t})\mathbf{t} = \mathbf{K}\mathbf{t} \tag{21}$$

**2.4. Adaptive filtering.** Wagner and Wittum [10] introduced and analyzed the adaptive filtering method. Given an iterative algorithm approximating the solution of a linear system,

$$\mathbf{u}^{(i+1)} = \mathbf{u}^{(i)} + \mathbf{M}^{-1}(\mathbf{f} - \mathbf{M}\mathbf{u}^{(i)}) \tag{22}$$

it would produce a sequence of solution vectors $\mathbf{u}^{(i)}$. After a few iterations only a few eigenvectors govern the error $\mathbf{u} - \mathbf{u}^{(i+1)}$. If the corresponding eigenvalues are sufficiently far from one another then the vector $\mathbf{u}^{(i+1)} - \mathbf{u}^{(i)}$ represents that subspace. So, after a few iterations a new preconditioner matrix $\mathbf{M}$ can be built using a filtering vector that is a function of the remaining error. The new $\mathbf{M}$ is then used in the following iteration steps damping the remaining error components.

## 3. Numerical algorithm

In our work we combine ideas from adaptive filtering [9] and iterated tangential filtering [3], [2] and apply them to sytems of PDEs by constructing an adaptive preconditioner. The idea behind the tangential filtering is similar to that presented by Achdou and Nataf [1] but extended to cases of non-symmetric matrices. We also deal with cases where the domain mesh is non-conforming.

**3.1. Preconditioner Definition.** For a given problem matrix $\mathbf{K}$ its exact factorization can be written as shown in equation (11). To construct the preconditioner we replace the blocks from $\mathbf{T}$ by a sparse approximation that would make the preconditioner more cost effective. Extending the work of Achdou and Nataf [1] the preconditioner is built by blocks, with each $\tilde{T}_i$ block following the induction formula:

$$\tilde{T}_i = \begin{cases} D_1, & i = 1 \\ D_i - \Theta_{i,i-1}\tilde{T}_{i-1}^{-1}\Theta_{i-1,i} - \Theta_{i,i-1}U_{i-1} - L_{i-1}\Theta_{i-1,i}, & 1 < i \leq n_x \end{cases} \tag{23}$$

In our case we define the transfer matrix $\Theta$ as:

$$\Theta_{i,i-1} = L_{i-1}\gamma_{i-1}^{-1} \tag{24}$$

$$\Theta_{i-1,i} = \gamma_{i-1}^{-1}U_{i-1} \tag{25}$$

where $\gamma_{i-1}$ is a diagonal matrix of the same size as $T_{i-1}$ and substituting in eq. (23) and rearranging we obtain [2]

$$\tilde{T}_i = \begin{cases} D_1, & i = 1 \\ D_i + L_{i-1}(\gamma_{i-1}^{-1}\tilde{T}_{i-1}\gamma_{i-1}^{-1} - 2\gamma_{i-1}^{-1})U_{i-1} & 1 < i \leq n_x \end{cases} \tag{26}$$

$\gamma$ is calculated after applying the filtering condition with an adequate filtering vector $\mathbf{t}$. The filter condition is derived from analyzing the error at the *ith* block:

$$\text{Error}^{(i)} = M_i - K_i = (L_{i-1}\gamma_{i-1}^{-1}\tilde{T}_{i-1} - L_{i-1})\tilde{T}_{i-1}^{-1}(\tilde{T}_{i-1}\gamma_{i-1}^{-1}U_{i-1} - U_{i-1}) \tag{27}$$

Rearranging equation (27), multiplying by the test vector $\mathbf{t}$ and imposing that the error equals zero:

$$L_{i-1}(\gamma_{i-1}^{-1}\tilde{T}_{i-1} - I)\ \tilde{T}_{i-1}^{-1}\ (\tilde{T}_{i-1}\gamma_{i-1}^{-1} - I)U_{i-1}\ t_i = 0 \tag{28}$$

From equation (28) we obtain the filtering condition to calculate $\gamma$:

$$\tilde{T}_{i-1}\gamma_{i-1}^{-1}U_{i-1}\ t_i = U_{i-1}t_i \tag{29}$$

Then, $\gamma$ is calculated by blocks according to equations (29), solving the linear systems: $\tilde{T}_{i-1}\mathbf{x} = U_{i-1}t_i$ and then computing an element-wise division with $U_{i-1}t_i$. The preconditioner is built within the context of a Krylov subspace method. In

this study we use GMRES as our linear solver. For a chosen dimension of the Krylov space, determined upon the maximum number of iterations allowed $N$. If convergence is not achieved after the first pass of the ILU0 preconditioner, then a new filtering vector is computed based on the current GMRES solution and the TFFD preconditioner is computed.

**3.2. Filtering vector.** As mentioned before, the preconditioner is built within the context of a Krylov subspace method. A Krylov subspace is defined as

$$\mathbf{K}_N(\mathbf{A}, \mathbf{r}^0) = span(\mathbf{r}^{(0)}, \mathbf{A}\mathbf{r}^{(0)}, \mathbf{A}^2\mathbf{r}^{(0)}, .., \mathbf{A}^{N-1}\mathbf{r}^{(0)}) \tag{30}$$

Krylov subspace methods form an orthogonal basis from a sequence of successive matrix powers times the initial residual. Then, the approximations to the solution are computed by minimizing the residual over the subspace formed. After a few iterations, the error is concentrated in a small subspace. It would be desirable to damp the dominating error components (eigenvectors corresponding to the smallest eigenvalues). The idea is to build the filtering vector with the information extracted from the Krylov subspace computed from a first attempt to solve the problem with a simpler preconditioner, for example using ILU0. The proposed filtering decomposition would eliminate the components of the error in the neighborhood of the filtering vector. The choice of the filtering vector is then based on the the fact that ILU0 works on the eigenvectors corresponding to the large eigenvalues but does a poor job on the small ones, and in many cases the solver will stagnate. Then, the filtering vector should filter out the eigenvectors corresponding to those small eigenvalues. In this way, the new combined TFFD preconditioner will not stagnate.

In the Generalized Minimal Residual (GMRES) [8] method, the orthogonal basis of the Krylov space are formed explicitly:

**Algorithm 1** GMRES [8]

1: Solve for $w$ from $\mathbf{M}w = \mathbf{K}a^{(i)}$
2: for $k = 1, ..., i$
3: $\quad h_{k,i} = (w, a^{(k)})$
4: $\quad w = w - h_{k,i}\, a^{(k)}$
5: end
6: $h_{i+1,i} = ||w||_2$
7: $a^{(i+1)} = w/h_{i+1,i}$

The inner products coefficients of the orthogonalization (Arnoldi method) $(w, v^{(k)})$ and $||w^{(i)}||$ are stored in an upper Hessenberg matrix $\mathbf{H}$. The GMRES iterates are calculated as:

$$\mathbf{x}^{(N)} = \mathbf{x}^{(0)} + \mathbf{A}_N \mathbf{y}_N \tag{31}$$

Where $\mathbf{A}_N$ is the orthogonal basis for the Krylov space of dimension $N$ as defined by equation (30) and obtained via the Arnoldi process so that the orthogonal projection of $\mathbf{K}$ onto $\mathbf{K}_N$ results in the upper Hessenberg matrix $\mathbf{H}_N = \mathbf{A}_N^T \mathbf{K} \mathbf{A}_N$. The vector $\mathbf{y}_N$ is the solution to the minimization of the norm of the residual

$$\min ||\beta_N \mathbf{e}_1 - \mathbf{H}_N \mathbf{y}_N||_2 \tag{32}$$

where $\beta_N = \left|\left|\mathbf{r}^{(N)}\right|\right|_2$ and $\mathbf{r}^N$ is the residual vector after $N$ GMRES iterations, $\mathbf{e}_1$ is the canonical basis vector. The filtering vector is then computed from a linear combination of the singular value decomposition of the Hessenberg matrix $\mathbf{H}$ and

Arnoldi vectors from the GMRES iterations. Given $\mathbf{H}$ we obtain its singular value decomposition as

$$\mathbf{H} = \mathbf{U}\Sigma\mathbf{V}^T \tag{33}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices and $\Sigma$ is a diagonal matrix. Then we use the matrix $\mathbf{V}$ containing the right singular vectors (the columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{H}^T\mathbf{H}$).

$$\mathbf{t} = \mathbf{A}_N\mathbf{v} \tag{34}$$

here $\mathbf{v}$ is one of the right singular vectors of $\mathbf{H}$ and $\mathbf{A}_N$ is the matrix with the Arnoldi vectors. The filtering vector is calculated in an adaptive way, this means, a new filtering vector is computed each time GMRES is restarted.

**3.3. Multiplicative combinative preconditioner.** The TFFD preconditioner is used in a multiplicative way. Given a combined preconditioner $\mathbf{M}$ solving

$$\mathbf{M}\mathbf{y} = \mathbf{f} \tag{35}$$

can be written as

$$(\mathbf{M}_1 + \mathbf{M}_2 + \mathbf{M}_2\mathbf{K}\mathbf{M}_1)\mathbf{y} = \mathbf{f} \tag{36}$$

Where $\mathbf{M}_1$ is the first preconditioner (for example ILU0) and $\mathbf{M}_2$ is the preconditioner from the frequency filtering (TFFD). Then the multiplicative combinative algorithm is:

$$\mathbf{y} = \mathbf{M}_1^{-1}\mathbf{f} \tag{37}$$

$$\mathbf{z} = \mathbf{b} - \mathbf{K}\mathbf{y} \tag{38}$$

$$\mathbf{y} = \mathbf{y} + \mathbf{M}_2^{-1}\mathbf{z} \tag{39}$$

## 4. Computational Solution

**4.1. Flexible GMRES.** For a chosen dimension of the Krylov space, determined upon the maximum number of iterations allowed $N$, the numerical solver can be described as: If convergence is not achieved after the first pass of the TFFD preconditioner, then a new filtering vector is computed based on the current GMRES solution and the TFFD preconditioner is recomputed. Recomputing the TFFD preconditioner involves calculating the new $T_i$ blocks using the new filtering vector. The blocks corresponding to the $L$ and $U$ matrices remain unchanged.

**Algorithm 2** TFFD + Solver

1: Build the ILU0 preconditioner.
2: Compute GMRES ([8],[7]) with a given tolerance and $N$ maximum number of iterations.
3: If convergence is not achieved then calculate the filtering vector.
4: Build the TFFD preconditioner.
5: Re-run GMRES with the new preconditioner used in a multiplicative combinative way.

**4.2. Implementation details.** A Fortran 90 code was developed in order to test the tangential frequency filtering decomposition. Different modules or classes were created. Such modules include the preconditioner class, the grid class, the superblock matrix class and the GMRES class. GMRES was implemented following the presentation by Saad et al. [6].

A pre-processing step calls a grid manipulation algorithm within the module that takes care of the grid (the grid is considered an object by itself). The grid module keeps track of the grid, containing the information related to cell connectivity. This module provides a procedure for partitioning the grid according to a selected direction. The information concerning each plane and how a plane is related to the others is computed and stored by the module. This is a very important step previous to the filtering procedure since the $T_i$ blocks of the preconditioners represent the part of the PDEs that relates cells from a given plane and the $L_i$ and $U_i$ matrix blocks contain the terms that relate a given cell with neighboring cells in adjacent planes. For storing matrices in Harwell-Boeing format but retaining the block information a "superblock" class was developed.

As mentioned before, the grid partition has a direct relationship with the different blocks of the preconditioner. In some particular grid domains, adjacent planes had different number of cells. When translating the grid geometry into matrix form this would imply that there would be some $L_i$ and $U_i$ blocks where the matrix would be rank deficient. For those cases special consideration had to be taken to avoid division by zero when imposing the filtering condition.

## 5. Results and discussion

Three cases were used to test the algorithm and its code. The first two correspond to a finite volume Black Oil Model solver used for reservoir simulation problems with two different meshes. The third test case presents the results obtained for an elliptic problem involving flow in the porous media. The results from the code execution and the description of each test case follows.

**5.1. Case 1: Black Oil Model.** To describe the multiphase flow in the porous media we use a Black Oil Model as described by Aziz and Settari [5]. The model assumes that there are three distinct phases: oil, water and gas, with water being the wetting phase. The reservoir fluid is composed of three different chemical components: one heavy hydrocarbon, a light hydrocarbon, and water. Both the light and heavy component represent mean ideal hydrocarbons. Water and oil are assumed immiscible and they do not exchange mass or change phase and gas is assumed to be soluble in oil. At standard or storage tank conditions (STC) the solubility of the gas phase is zero, that means that the oil phase is composed of the heavy hydrocarbon and the gas by the light hydrocarbon. Under these conditions each phase can be identified by only one principal component. At reservoir conditions both heavy and light hydrocarbon can be present in the oil phase. The mass transfer between the phases is governed by thermodynamic relationships (normally given as a function of the pressure in the reservoir) that determine how the chemical components redistribute in the different phases and what amount of the porous volume is occupied by each phase. The complete description of the governing equations can be found in [5].

For this test case, the reservoir is described by a rectangular box shape domain. The size of the reservoir is $(0, 1000) \times (0, 1000) \times (0, 100)$ m with a Cartesian mesh of $15 \times 15 \times 8$ cells. There is one production well (vertical) and 4 vertical injector wells. The permeability is log-normal distributed and set to $\kappa_x = \kappa_y$. and the

anisotropy is in the z axis given by $\kappa_z/\kappa_x = 1/5$. Figure 1 shows the distribution of $x$ component of the permeability tensor, $\kappa_x$, for a plane at middle height of the domain. To test the solver the convergence criteria was set to a relative residual

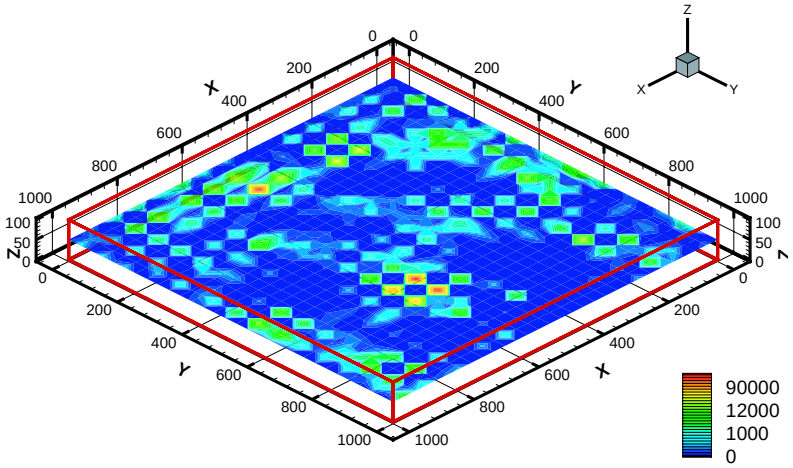


FIGURE 1. Permeability values in the x direction for the center plane.

of $10^{-5}$ and the restart of GMRES fixed to 20 iterations with a maximum of 5 restarts. The relative residual at the $k-th$ iteration was calculated as:

$$\text{Relative residual } = \frac{\left|\left|\mathbf{r}^k\right|\right|_2}{||\mathbf{f}||_2} \tag{40}$$

with $\mathbf{f}$ the right hand side of the system to be solved.

For the construction of this new preconditioner the filtering vector was computed from the singular value decomposition of the Hessenberg matrix of the previous Krylov subspace after the maximum number of iteration were completed, in this case after 20 iterations. Figures 2 and 3 show the eigenvector for the cells in the center plane of the reservoir. These two plots presents the eigenvector corresponding to the largest and smallest eigenvalue for the problem matrix and ILU0 preconditioner respectively. Note that the irregularity is due to random values of the permeability, $\kappa$.

The results obtained for this test case are presented in Figure 4. From the plot we can see that the number of iterations required for convergence is much smaller when using the TFFD preconditioner than when using ILU0 alone. When using ILU0 alone the number of calls to the preconditioner was 70. This 70 iterations meant that GMRES had to be restarted 3 times, the first 3 passes the maximum number of iterations was reached (set to 20 for this case), and then convergence was achieved in the fourth pass after 10 more iterations. On the other hand, the total number of iterations required using the combinative TFFD was only 35. In this last case, in order to make a fair comparison, we should note that the cost of the combinative TFFD preconditioner is approximately double that the cost of using ILU0 alone. We should consider that those 35 iterations represent 20 calls to ILU0 and 15 calls to the combinative preconditioner. Each call to the combinative

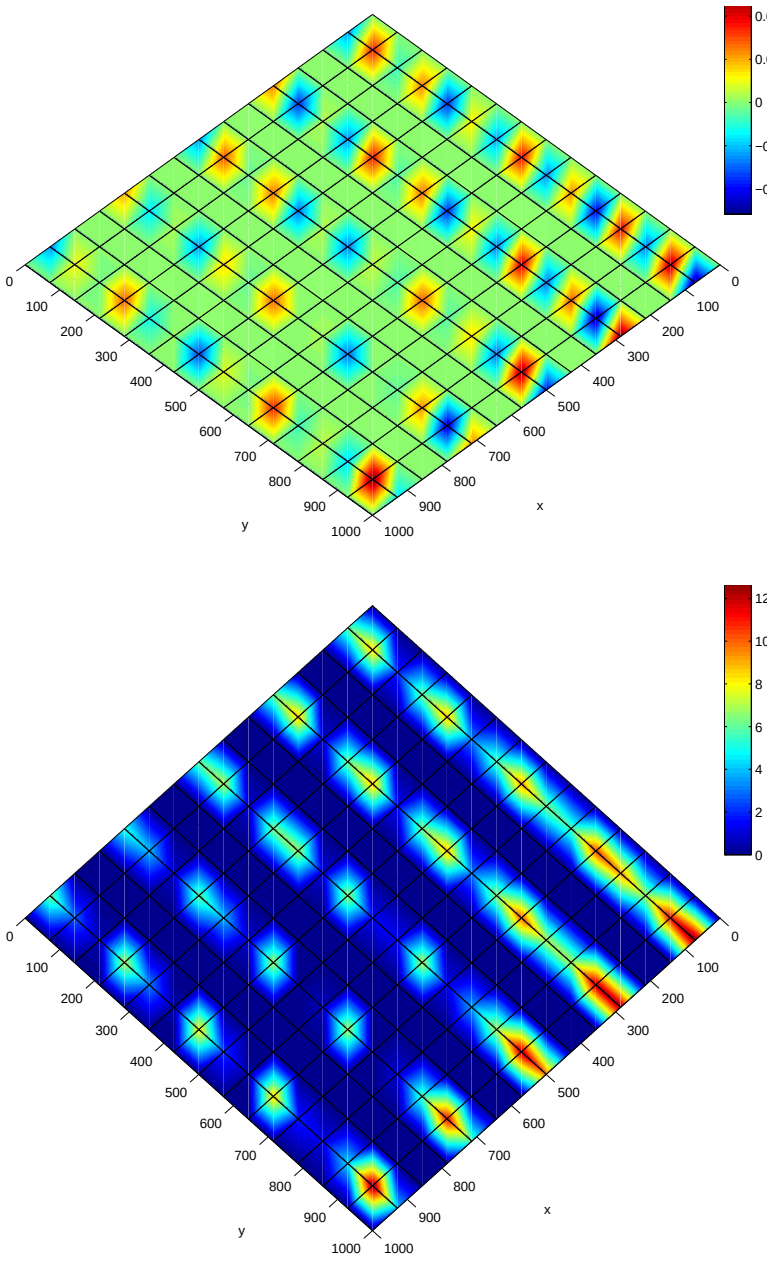


FIGURE 2. Section of the eigenvector of the problem matrix corresponding to the largest (top) and smallest (bottom) eigenvalues.

preconditioner involves solving with ILU0 and with TFFD. So, the 15 calls should be counted double. To take this into account, the following formula will be applied to calculate the number of **equivalent** iterations and will be used for the rest of the comparisons:

$$N_{Ts} = N_{ILU0} + 2 \cdot N_{TFFDc} \tag{41}$$

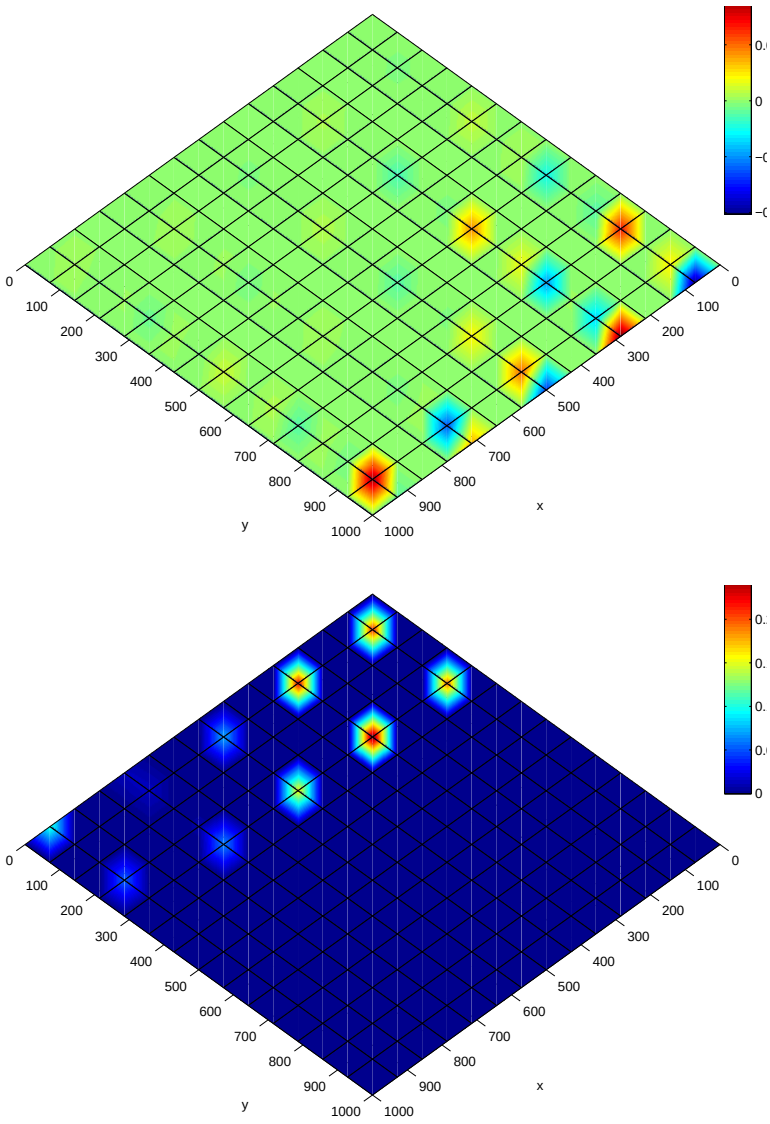


FIGURE 3. Section of the eigenvector of the ILU0 preconditioner corresponding to the largest (top) and smallest (bottom) eigenvalues.

where $N_Ts$ is the total number of times that any single preconditioner was applied; $N_{ILU0}$ is the number of calls to the ILU0 preconditioner and $N_{TFFDc}$ is the number of calls to the combinative preconditioner. Using equation (41) for case 1, the first test using ILU0 needed 70 iterations vs 50 equivalent iterations for the test case where TFFD was used.

**5.2. Case 2.** Case 2 presents the results of the same Black Oil Model solver but in this test case the mesh had $30 \times 30 \times 16$ finite volumes. The physical properties of the reservoir are the same as in Case 1. Figure 5 and 6 show the results obtained for these conditions. Figure 5 presents the comparison of the number of iterations

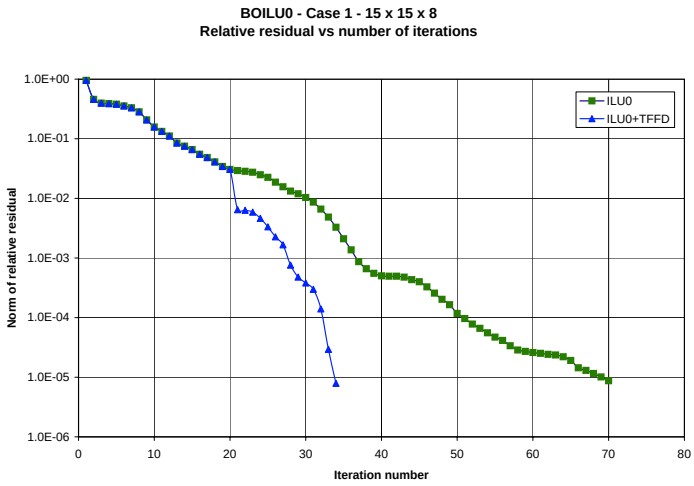


FIGURE 4. Norm of the relative residual vs number of iterations (semilog) for Case 1 with restart set to 20.

required by the combinative TFFD preconditioner vs the number of iterations required if ILU0 is used as the only preconditioner. From this figure we can observe that using ILU0 as the only preconditioner required 226 GMRES iterations vs 70 GMRES iterations using the combinative preconditioner. The 70 combinative iterations are equivalent to 100 single preconditioner calls (using the formula given above, the first 40 correspond to ILU0 alone and then 30 iterations using TFFD should be counted double). For this case we saved 126 GMRES iterations. Figure

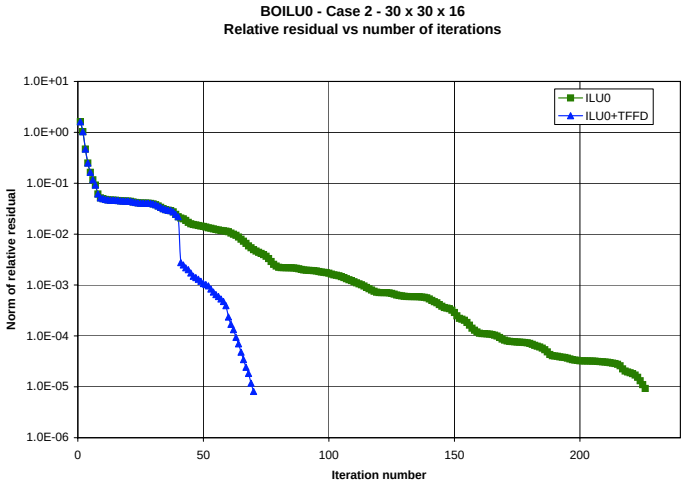


FIGURE 5. Norm of the relative residual vs number of iterations (semilog) for Case 2 with restart set to 40.

6 shows the contours corresponding to the filtering vector for the center plane of the domain. The filtering vector was computed after 40 iterations of GMRES using ILU0 alone.

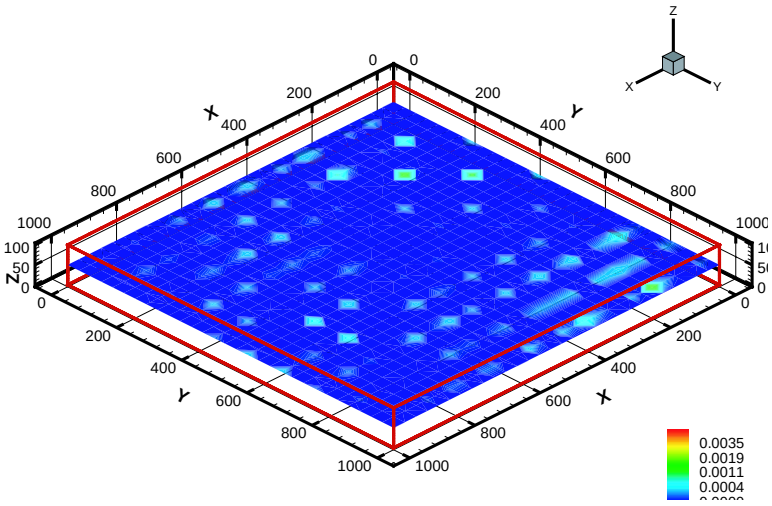


FIGURE 6. Filtering vector for the center plane.

To test the selection of the restart criteria (dimension of Krylov space) we varied the number of iterations allowed for the first pass of GMRES, when ILU0 was the only preconditioner used. In this way we could further investigate the performance of the filtering vector and the TFFD preconditioner. The following table shows the results obtained for this problem when the dimension of the initial GMRES space was limited to $N_{ILU0} = 3, 5, 7, 10, 16, 20, 30, 40$ respectively. For all the cases, GMRES was run with $N_{ILU0}$ iterations where the matrix was preconditioned with ILU0 only. If convergence was not achieved, GMRES was restarted and the maximum number of iterations per GMRES pass was set to 40, with the TFFD preconditioner being used in a combinative way. The maximum number of total iterations was set to 200. The relative residual for convergence was set to $10^{-5}$.

| | Iteration counts (Tolerance= $10^{-5}$) | | |
|---|---|---|---|
| $N_{ILU0}$ | $N_T$ | $N_{ILU0+TFFD}$ | $N_{Ts}$ |
| 3 | 70 | $3 + 2 \cdot 40 + 2 \cdot 27$ | 134 |
| 5 | 39 | $5 + 2 \cdot 34$ | 73 |
| 7 | 40 | $7 + 2 \cdot 33$ | 70 |
| 10 | 40 | $10 + 2 \cdot 30$ | 70 |
| 16 | 40 | $16 + 2 \cdot 24$ | 64 |
| 20 | 47 | $20 + 2 \cdot 27$ | 74 |
| 30 | 54 | $30 + 2 \cdot 24$ | 78 |
| 40 | 70 | $40 + 30 \cdot 2$ | 100 |

TABLE 1. Case 2: Total iterations for different first GMRES restart criteria.

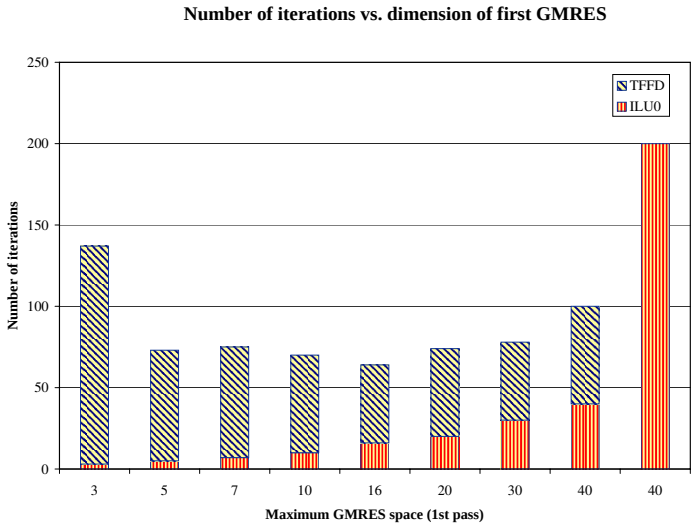


FIGURE 7. Number of equivalent single preconditioner iterations vs restart for the first GMRES pass.

In this table $N_{ILU0}$ is the number of iterations with ILU0, $N_T$ is the number of GMRES iterations, $N_{ILU0+TFFD}$ is the number of iterations with the combined preconditioner and $N_{Ts}$ is the number of equivalent iterations as defined by equation (41). So, for example, for the first run the initial number of maximum iteration for the first pass of GMRES was set to 3. Then, the new preconditioner was built and TFFD was applied in a combinative way and the maximum number of iterations set to 40. After 40 more iterations of GMRES, the preconditioner was rebuilt once again and after 27 more iterations convergence was achieved. To compute the equivalent single preconditioner iterations we have to consider that after the first restart every iterations should count double. Figure 7 presents this information in graphical form. From the plot it is clear that even with a very small initial number of ILU0 iterations, the combinative TFFD preconditioner required considerable less iterations to achieve convergence. The minimum number of iterations was when the restart was set to 16, but even restarting after only 5 iterations gave much better results than using ILU0 alone. For the best case scenario using TFFD only 64 equivalent iterations were required. For completeness of this comparison, a last test was performed using ILU0 as the only preconditioner and setting the restart for GMRES at 40 iterations and letting the calculations continue till convergence was achieved. For that case, ILU0 required 226 iterations.

**5.3. Case 3.** Our third numerical experiment deals with flow in the porous media. We consider a boundary value problem discretized on a regular Cartesian grid using finite volumes. The equation that describe the flow in the porous media can be written as:

$$\begin{array}{ll} -\nabla \cdot (\kappa \nabla u) = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega_D, \\ \frac{\partial u}{\partial n} = 0 & \text{on } \partial\Omega_N \end{array} \tag{42}$$

With $\Omega = [0,1]^2$, $\partial\Omega_D = [0,1] \times \{0,1\}$ and $\partial\Omega_N = \partial\Omega \backslash \partial\Omega_D$. For this case the tensor $\kappa$ is isotropic and discontinuous. The domain contains many zones of high permeability which are isolated. Figure 8 shows the filtering vector computed after the initial 70 iterations of GMRES (with ILU0) and used to calculate the new TFFD preconditioner.

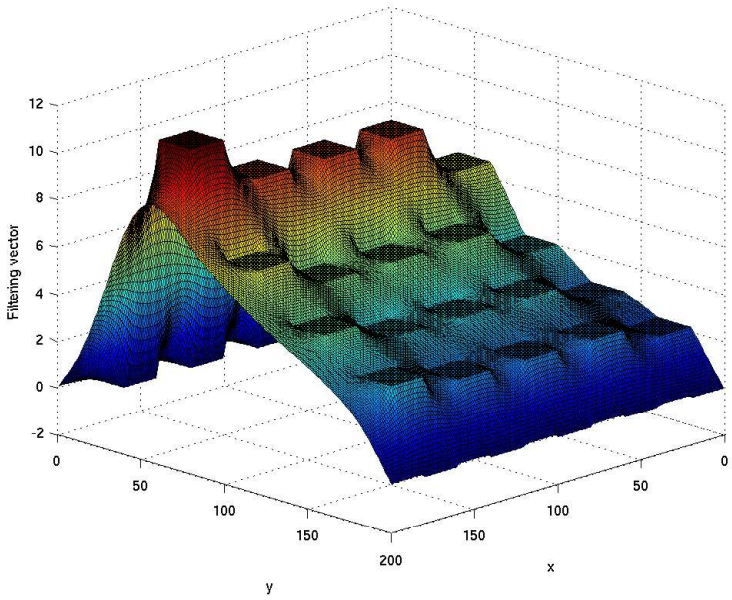


FIGURE 8. Filtering vector for Case 3 after 70 iterations.

Figure 9 shows the results obtained. In this case the combinative preconditioner required only 108 equivalent iterations while using ILU0 the maximum number of iterations was reached (3 restarts) but convergence was not achieved.

To compare different restarting criteria we conducted several tests where we varied the maximum number of iterations of the GMRES routine. The comparison was performed for restarts set to 70, 100 and 150 iterations respectively. Table 2 and Figure 10 show these results.

| | Iteration counts (Tolerance= $10^{-5}$) | | | | |
|---|---|---|---|---|---|
| *Restart* | $N_T$ | $N_{ILU0}$ | $N_{TFFDc}$ | $N_{Ts}$ | Converged? |
| 70 | 210 | 210 | 0 | 210 | No |
| 100 | 300 | 300 | 0 | 300 | No |
| 150 | 450 | 450 | 0 | 450 | No |
| 70 | 89 | 70 | 38 | 108 | Yes |
| 100 | 117 | 100 | 34 | 134 | Yes |
| 150 | 167 | 150 | 34 | 184 | Yes |

TABLE 2. Results for Case 3: Total iterations for different GMRES restart criteria.

For these numerical experiments the convergence criteria was not met using ILU0 alone. In all the three cases we can see that the combinative TFFD outperforms

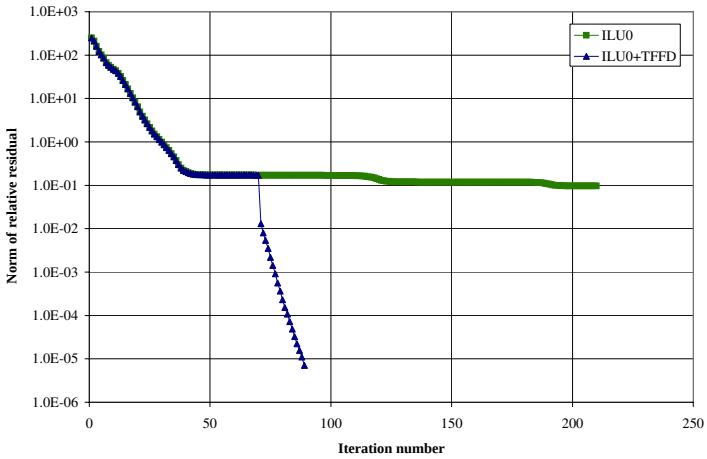


FIGURE 9. Norm of the relative residual vs number of iterations (semilog) for Case 3 with restart set to 70.

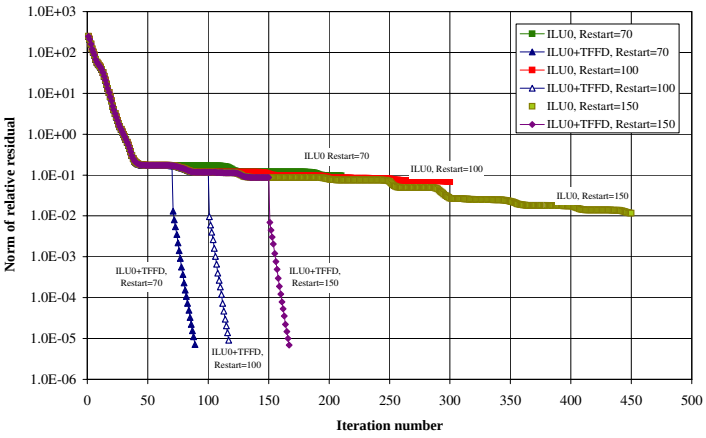


FIGURE 10. Results for different restart criteria for Case 3.

ILU0. The minimum number of iterations was obtained for the smallest restart criteria. We can also note that when the restart was set to 100 and 150 the combinative TFFD required the same number of iterations: 34. That means that all

the extra ILU0 iterations did not provide any further critical information when computing the filtering vector and building the TFFD preconditioner.

This last elliptic case perfectly illustrates the advantages of the new TFFD preconditioner. For hard cases when using only ILU0, a plateau is reached since the preconditioner is only efficient on one part of the spectrum of the problem matrix. When using the combinative TFFD preconditioner this problem can be overcome and convergence achieved.

## 6. Conclusions

This paper shows the results obtained from applying a new Tangential Frequency Filtering Decomposition (TFFD) method for the construction of an efficient preconditioner. The preconditioner is built in stages according to the problem grid and used in an adaptively and multiplicative way within the linear solver in those cases where convergence was not achieved with ILU0 preconditioner.
Details of the programming implementation as well as results of several test cases were presented. For all our test cases, the TFFD preconditioner outperforms ILU0, with the difference in the number of iteration being quite impressive. For some cases, the number of iterations using TFFD was less than 25% than in those where ILU0 was used as the only preconditioner.
This new technique seems promising. Future work should focus on testing with more matrices arising from complex problems and comparing with other preconditioners and solvers. Ultimately, if the techniques continues to prove as efficient as in these cases, it would be desirable to parallelize the entire algorithm.

## Acknowledgments

The authors would like to thank the Natural Sciences and Engineering Research Council (NSERC) of Canada and the Institut Français du Pétrole for their financial support.

## References

[1] Y. Achdou and F. Nataf. An iterated tangential filtering decomposition. *Num. Lin. Alg. and Appl.*, 10(5-6):511–539, July/September 2003.
[2] Y. Achdou and F. Nataf. Low frequency tangential filtering decomposition. *Num. Lin. Alg. and Appl.*, 2006. Submitted for publication.
[3] J.R. Appleyard and I.M. Cheshire. Nested factoriaztion. *Society of Petroleum Engineering of AIME*, pages 315–320, 1983.
[4] J.R. Appleyard, I.M. Cheshire, and R.K. Pollard. Spectral techniques for fully implicit simulators. pages 295–408, Bournemouth, England, 1981. Proc. European Symposium on Enhanced Oil Recovery.
[5] Aziz and Settari. *Petroleum Reservoir Simulation.* Blitzprint Ltd., Calgary, second edition, 2002.
[6] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. M. Donato, Jack Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. V. der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* Philadalphia: Society for Industrial and Applied Mathematics., 1994.
[7] Y. Saad. *Iterative Methods for Sparse Linear Systems.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2003.
[8] Y. Saad and M. H. Schultz. GMRES: A Generalized Minimum Residual algorithm for solving non nonsymmetric linear systems. *J-SCI-STAT-COMP*, 7:856–869, 1986.
[9] C. Wagner. Adaptive filtering. *Numerische Mathematik*, 78(2):305–328, 1997.
[10] C. Wagner. Tangential frequency filtering decompositoin for symmetric matrices. *Numerische Mathematik*, 78(1):119–142, 1997.
[11] C. Wagner. Tangential frequency filtering decompositoin for unsymmetric matrices. *Numerische Mathematik*, 78(1):143–163, 1997.

Department of Chemical and Material Engineering, University of Alberta, Edmonton, AB T6G 2G6, Canada
*E-mail*: carolina@ualberta.ca

UFR Mathématiques, Université Paris 7, Paris, France

Centre de Mathématiques Appliquées, Ecole Polytechnique, Paris, France

Dept. Inf. Sci.et Math. Appl., Institut Français du Pétrole, Rueil Malmason, France