2 Flow through porous media

Petroleum reservoirs are layers of sedimentary rock, which vary in terms of their grain size, mineral and clay contents. These rocks contain grains and empty space, the void is called pore space. The pore space allows the rock to store and transmit fluids. The volume fraction of the rock that is void is called *rock porosity* (ϕ).
Some rocks are compressible and their porosity depends on the pressure, the dependence is called *rock compressibility* (c_r). The ability of the rock to transmit a single fluid when the void space is completely filled with fluid is known as *rock permeability* (K).
Reservoir simulation is a way to analyze and predict the fluid behavior in a reservoir by the analysis of its behavior in a model. The description of subsurface flow simulation involves two types of models: mathematical and geological models. The geological model is used to describe the reservoir, i.e., the porous rock formation. The mathematical modeling of porous media flow is performed taking into account mass conservation and Darcy's law, corresponding to the momentum conservation. The equations used to describe single-phase flow through a porous medium are:

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho v) = q, \qquad v = -\frac{K}{\mu}(\nabla p - \rho g \nabla d), \tag{2.1}$$

or

$$\frac{\partial(\rho\phi)}{\partial t} - \nabla \cdot \left(\frac{\rho K}{\mu}(\nabla p - \rho g \nabla d) \right) = q. \tag{2.2}$$

Where the primary unknown is the pressure p, g is the constant of gravity, d is the reservoir depth, ρ and μ are the fluid density and viscosity and q are the sources. The fluid density $\rho = \rho(p)$ and the rock porosity $\phi = \phi(p)$ can be pressure dependent. Rock porosity is related to the pressure through the rock compressibility, the relation is given by the following expression:

$$c_r = \frac{1}{\phi}\frac{d\phi}{dp} = \frac{dln(\phi)}{dp},$$

If the rock compressibility is constant, the previous equation is integrated as:

$$\phi(p) = \phi_0 e^{c_r(p-p_0)}. \tag{2.3}$$

The fluid density and the pressure are related via the fluid compressibility c_f, the relation is given by:

$$c_f = \frac{1}{\rho}\frac{d\rho}{dp} = \frac{dln(\rho)}{dp}.$$

If the fluid compressibility is constant, the previous equation is integrated as:

$$\rho(p) = \rho_0 e^{c_f(p-p_0)}. \tag{2.4}$$

Incompressible fluid
If the density and the porosity do not depend on the pressure in Equation (2.2), we have

an incompressible model. Assuming no gravity terms and a fluid with constant viscosity, Equation (2.2) becomes:

$$-\frac{\rho}{\mu}\nabla \cdot (K\nabla p) = q. \tag{2.5}$$

Discretization
The spatial differentials are approximate using a finite difference scheme with cell central derivatives. For a 3D model, taking a mesh with a uniform grid size Δx, Δy, Δz where (i, j, l) is the center of the cell in the position i in the x direction and j in the y direction and l in the z direction (x_i, y_j, z_l) and $p_{i,j,l} = p(x_i, y_j, z_l)$ is the pressure at this point.
For the x direction, we have (see [16]):

$$\begin{aligned}\frac{\partial}{\partial x}\left(k\frac{\partial p}{\partial x}\right) &= \frac{\Delta}{\Delta x}\left(k\frac{\Delta p}{\Delta x}\right) + \mathcal{O}(\Delta x^2)\\ &= \frac{k_{i+\frac{1}{2},j}(p_{i+1,j,l} - p_{i,j,l}) - k_{i-\frac{1}{2},j,l}(p_{i,j,l} - p_{i-1,j,l})}{(\Delta x)^2} + \mathcal{O}(\Delta x^2),\end{aligned}$$

where $k_{i-\frac{1}{2},j,l}$ is the harmonic average of the permeability for the cells $(i-1, j, l)$ and (i, j, l):

$$k_{i-\frac{1}{2},j,l} = \frac{1}{\frac{1}{k_{i-1,j,l}} + \frac{1}{k_{i,j,l}}}. \tag{2.6}$$

After discretization, Equation 2.5 can be written as:

$$\mathbf{Tp} = \mathbf{q}, \tag{2.7}$$

Where $\mathbf{T}$ is known as the transmissibility matrix with elements in adjacent grid cells, e.g., the *transmissibility* $(T_{i-\frac{1}{2},j,l})$ between grid cells $(i-1, j, l)$ and (i, j, l) is defined as:

$$T_{i-\frac{1}{2},j,l} = \frac{\Delta y}{\Delta x}\frac{d}{\mu}k_{i-\frac{1}{2},j,l}, \tag{2.8}$$

System (2.7) is a linear system that can be solved with iterative or direct methods.

Compressible fluid
If the fluid is compressible with a constant compressibility, the density depends on the pressure (see Equation 2.4). Therefore, Equations 2.1 become:

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho(p)v) = q, \qquad v = -\frac{K}{\mu}(\nabla p - \rho(p)g\nabla d), \tag{2.9}$$

Discretization
Using backward Euler time discretization, Equations (2.9) are approximated by:

$$\frac{(\phi\rho(p))^{n+1} - (\phi\rho(p))^n}{\Delta t^n} + \nabla \cdot (\rho(p)v)^{n+1} = q^{n+1}, \qquad v^{n+1} = -\frac{K}{\mu^{n+1}}(\nabla(p^{n+1}) - g\rho^{n+1}\nabla d). \tag{2.10}$$

Assuming no gravity terms, constant fluid viscosity and constant rock porosity, Equations (2.10) become:

$$\phi \frac{\rho(p^{n+1}) - \rho(p^n)}{\Delta t^n} - \frac{1}{\mu} \nabla \cdot (\rho(p^{n+1}) K \nabla p^{n+1}) + q^{n+1} = 0. \tag{2.11}$$

Due to the dependence of ρ on the pressure, the latter is a nonlinear equation for p that can be linearized with, e.g., the Newton-Raphson (NR) method. Equation 2.11 can be discretized in space, using, e.g., finite differences schemes. After discretization Equation 2.11 reads:

$$\phi \frac{\rho(\mathbf{p}^{n+1}) - \rho(\mathbf{p}^n)}{\Delta t^n} - \frac{1}{\mu} \nabla \cdot (\rho(\mathbf{p}^{n+1}) \mathbf{K} \nabla \mathbf{p}^{n+1}) + \mathbf{q}^{n+1} = 0. \tag{2.12}$$

Well model

In reservoirs, wells are typically drilled to extract or inject fluids. Fluids are injected into a well at constant surface rate or constant bottom-hole pressure (bhp) and are produced at constant bhp or a constant surface rate.

When the bhp is known, some models are developed to accurately compute the flow rate into the wells. A widely used model is Peaceman's model, that takes into account the bhp and the average grid pressure in the block containing the well. This model is a linear relationship between the bhp and the surface flow rate in a well, for a cell (i, j, l) that contains a well, this relationship is given by:

$$q_{(i,j,l)} = I_{(i,j,l)}(p_{R(i,j,l)} - p_{bhp(i,j,l)}), \tag{2.13}$$

where $I_{(i,j,l)}$ is the productivity or injectivity index of the well, $p_{R(i,j,l)}$ is the reservoir pressure in the cell where the well is located, and $p_{bhp(i,j,l)}$ is the pressure inside the well.

Incompressible fluid

Using the well model for an incompressible fluid, Equation 2.7 transforms into:

$$\mathbf{T}\mathbf{p} = \mathbf{I}_w(\mathbf{p} - \mathbf{p}_{bhp}). \tag{2.14}$$

Where $\mathbf{I}_w$ is a vector containing the productivity or injetivity indices of the wells present in the reservoir. It is zero for cells without wells and the value of the well index for each cell containing a well.

Compressible fluid

For a compressible fluid, using the well model, Equation 2.12 reads:

$$\phi \frac{\rho(\mathbf{p}^{n+1}) - \rho(\mathbf{p}^n)}{\Delta t^n} - \frac{1}{\mu} \nabla \cdot (\rho(\mathbf{p}^{n+1}) \mathbf{K} \nabla \mathbf{p}^{n+1}) + \mathbf{I}_w^{n+1}(\mathbf{p}^{n+1} - \mathbf{p}_{bhp}^{n+1}) = 0. \tag{2.15}$$

Solution procedure

To solve Equations (2.14) and (2.15), it is necessary to define boundary conditions. These

conditions can be prescribed pressures (Dirichlet conditions), flow rates (Neumann conditions) or a combination of these (Robin conditions).
For Equation (2.15) we also need to specify the initial conditions that are the pressure values of the reservoir at the beginning of the simulation.
As mentioned before, for the compressible problem, we have a nonlinear system that depends on the pressure at the time step n and the pressure at the time step $n+1$. Therefore, Equation (2.15) can be seen as a function that depends on $\mathbf{p}^{n+1}$ and $\mathbf{p}^n$, i.e.,

$$\mathbf{F}(\mathbf{p}^{n+1}; \mathbf{p}^n) = 0. \tag{2.16}$$

This nonlinear system can be solved with the NR method, the system for the $(k+1)$-th NR iteration is:

$$\mathbf{J}(\mathbf{p}^k)\delta\mathbf{p}^{k+1} = -\mathbf{F}(\mathbf{p}^k), \qquad \mathbf{p}^{k+1} = \mathbf{p}^k + \delta\mathbf{p}^{k+1},$$

where $\mathbf{J}(\mathbf{p}^k) = \frac{\partial \mathbf{F}(\mathbf{p}^k)}{\partial \mathbf{p}^k}$ is the Jacobian matrix, and $\delta\mathbf{p}^{k+1}$ is the NR update at iteration step $k+1$.
Therefore, the linear system to solve is:

$$\mathbf{J}(\mathbf{p}^k)\delta\mathbf{p}^{k+1} = \mathbf{b}(\mathbf{p}^k). \tag{2.17}$$

with $\mathbf{b}(\mathbf{p}^k)$ being the function evaluated at iteration step k, $\mathbf{b}(\mathbf{p}^k) = -\mathbf{F}(\mathbf{p}^k)$.
The procedure to solve this problem consists of three stages. During the first stage, we select a time and solve Equation 2.15 for this particular time, i.e., we have a solution for each time step. In the second stage, we linearize the equations with the NR method, i.e., we perform a series of iterations to find the zeros of Equation (2.16). For every NR iteration the linear system in Equation (2.17) is solved. A summary of this procedure is presented in Algorithm 1.

Algorithm 1

```
for t = 0, ...,                                   %Time integration
    Select time step
    for NR_iter = 0, ...,                         %NR iteration
        Find zeros of F(p^{n+1}; p^n) = 0
        for lin_iter = 0, ...,                    %Linear iteration
            Solve J(p^k) δp^{k+1} = b(p^k) for each NR iteration
        end
    end
end
```

3 Iterative solution methods

When simulating single-phase flow through a porous medium, we obtain a linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{3.1}$$

for both compressible and incompressible models. Since $\mathbf{A}$ is SPD we choose as iterative method the Conjugate Gradient (CG) method accelerated by the Incomplete Cholesky preconditioner. In this work, we will also study the acceleration with deflation techniques. In this section we give a brief overview of the methods.

Conjugate Gradient Method

Given a starting solution $\mathbf{x}^0$ and the residual defined by $\mathbf{r}^k = \mathbf{b} - \mathbf{A}\mathbf{x}^k$, we define the Krylov subspace $\mathcal{K}_k(\mathbf{A}, \mathbf{r}^0) = span\{\mathbf{r}^0, \mathbf{A}\mathbf{r}^0, \ldots, \mathbf{A}^{k-1}\mathbf{r}^0\}$ and $\mathbf{x}^k \in \mathbf{x}^0 + \mathcal{K}_k(\mathbf{A}, \mathbf{r}^0)$ is minimal for all approximations contained in $\mathbf{x}^0 + \mathcal{K}_k(\mathbf{A}, \mathbf{r}^0)$. The error of this approximation is bounded by:

$$||\mathbf{x} - \mathbf{x}^{k+1}||_{\mathbf{A}} \leq 2||\mathbf{x} - \mathbf{x}^0||_{\mathbf{A}} \left(\frac{\sqrt{\kappa_2(\mathbf{A})} - 1}{\sqrt{\kappa_2(\mathbf{A})} + 1} \right)^{k+1}{}^{1}. \tag{3.2}$$

The pseudo code for CG is given in Algorithm 2.

Algorithm 2 Conjugate Gradient (CG) method, solving $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Give an initial guess $\mathbf{x}^0$.
Compute $\mathbf{r}^0 = \mathbf{b} - \mathbf{A}\mathbf{x}^0$ and set $\mathbf{p}^0 = \mathbf{r}^0$.
for $k = 0, \ldots,$ until convergence
$\alpha^k = \frac{(\mathbf{r}^k, \mathbf{r}^k)}{(\mathbf{A}\mathbf{p}^k, \mathbf{p}^k)}$
$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha^k \mathbf{p}^k$
$\mathbf{r}^{k+1} = \mathbf{r}^k - \alpha^k \mathbf{A}\mathbf{p}^k$
$\beta^k = \frac{(\mathbf{r}^{k+1}, \mathbf{r}^{k+1})}{(\mathbf{r}^k, \mathbf{r}^k)}$
$\mathbf{p}^{k+1} = \mathbf{r}^{k+1} + \beta^k \mathbf{p}^k$
end

Preconditioning

To accelerate the convergence of a Krylov method, one can transform the system into another one containing an iteration matrix with a better spectrum , i.e, a smaller condition number. This can be done by multiplying the system (3.1) by a matrix $\mathbf{M}^{-1}$.

$$\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}. \tag{3.3}$$

[1]The condition number $\kappa_2(\mathbf{A})$ is defined as $\kappa_2(\mathbf{A}) = \frac{\sqrt{\lambda_{max}(\mathbf{A}^T\mathbf{A})}}{\sqrt{\lambda_{min}(\mathbf{A}^T\mathbf{A})}}$. If $\mathbf{A}$ is SPD, $\kappa_2(\mathbf{A}) = \frac{\lambda_{max}(\mathbf{A})}{\lambda_{min}(\mathbf{A})}$.

The new system has the same solution but can provide a substantial improvement on the spectrum. For this preconditioned system, the error is bounded by:

$$||\mathbf{x} - \mathbf{x}^k||_{\mathbf{A}} \leq 2||\mathbf{x} - \mathbf{x}^0||_{\mathbf{A}} \left(\frac{\sqrt{\kappa(\mathbf{M}^{-1}\mathbf{A})} - 1}{\sqrt{\kappa(\mathbf{M}^{-1}\mathbf{A})} + 1} \right)^k. \tag{3.4}$$

$\mathbf{M}$ is chosen as an SPD matrix such that $\kappa(\mathbf{M}^{-1}\mathbf{A}) \leq \kappa(\mathbf{A})$, and $\mathbf{M}^{-1}b$ is cheap to compute.

Deflation

Deflation is used to annihilate the effect of extreme eigenvalues on the convergence of an iterative method ([12]). Given an SPD matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, for a given matrix $\mathbf{Z} \in \mathbb{R}^{n\times m}$ the deflation matrix $\mathbf{P}$ is defined as follows ([17, 15]):

$$\mathbf{P} = \mathbf{I} - \mathbf{AQ}, \qquad \mathbf{P} \in \mathbb{R}^{n\times n}, \qquad \mathbf{Q} \in \mathbb{R}^{n\times n},$$

where

$$\mathbf{Q} = \mathbf{ZE}^{-1}\mathbf{Z}^T, \qquad \mathbf{Z} \in \mathbb{R}^{n\times m}, \qquad \mathbf{E} \in \mathbb{R}^{m\times m},$$

with

$$\mathbf{E} = \mathbf{Z}^T\mathbf{AZ}.$$

The matrix $\mathbf{E}$ is known as the *Galerkin* or *coarse* matrix that has to be invertible. If $\mathbf{A}$ is SPD and $\mathbf{Z}$ is full rank then $\mathbf{E}$ is invertible. The full rank matrix $\mathbf{Z}$ is called the $deflation - subspace$ matrix, and it's columns are the $deflation$ vectors or *projection* vectors.

Some properties of the previous matrices are [15]:

a) $\mathbf{P}^2 = \mathbf{P}$.

b) $\mathbf{AP}^T = \mathbf{PA}$.

c) $(\mathbf{I} - \mathbf{P}^T)\mathbf{x} = \mathbf{Qb}$.

d) $\mathbf{PAQ} = \mathbf{0}^{n\times n}$.

We can split the vector $\mathbf{x}$ as:

$$\mathbf{x} = \mathbf{x} - \mathbf{P}^T\mathbf{x} + \mathbf{P}^T\mathbf{x} = (\mathbf{I} - \mathbf{P}^T)\mathbf{x} + \mathbf{P}^T\mathbf{x}. \tag{3.5}$$

Multiplying expression (3.5) by $\mathbf{A}$, using the properties above, we have:

$$\begin{aligned} \mathbf{Ax} &= \mathbf{A}(\mathbf{I} - \mathbf{P}^T)\mathbf{x} + \mathbf{AP}^T\mathbf{x}, && Property: \\ \mathbf{Ax} &= \mathbf{AQb} + \mathbf{AP}^T\mathbf{x}, && c) \\ \mathbf{b} &= \mathbf{AQb} + \mathbf{PAx}, && b), \end{aligned}$$

multiplying by $\mathbf{P}$ and using the properties $\mathbf{PAQ} = \mathbf{0}^{n\times n}$ and $\mathbf{P}^2 = \mathbf{P}$, properties d) and a), we have:

$$\begin{aligned} \mathbf{PAQb} + \mathbf{P}^2\mathbf{Ax} &= \mathbf{Pb}, \\ \mathbf{PAx} &= \mathbf{Pb}, \end{aligned} \tag{3.6}$$

where $\mathbf{PAx} = \mathbf{Pb}$ is the deflated system. Since $\mathbf{PA}$ is singular, the solution of Equation (3.6) can contain components of the null space of $\mathbf{PA}$. A unique solution of this system, called the deflated solution, is denoted by $\hat{\mathbf{x}}$. The deflated system for $\hat{\mathbf{x}}$ is:

$$\mathbf{PA}\hat{\mathbf{x}} = \mathbf{Pb}. \tag{3.7}$$

The relation between $\mathbf{x}$ and $\hat{\mathbf{x}}$ is (see Appendix D):

$$\mathbf{x} = \mathbf{Qb} + \mathbf{P}^T\hat{\mathbf{x}}. \tag{3.8}$$

Deflated CG Method

To obtain the solution of linear system (3.1), we solve the deflated system:

$$\mathbf{PA}\hat{\mathbf{x}} = \mathbf{Pb}.$$

with the CG method, for a deflated solution $\hat{\mathbf{x}}$. Thereafter, the solution $\mathbf{x}$ to the original system is obtained from Equation (3.8):

$$\mathbf{x} = \mathbf{Qb} + \mathbf{P}^T\hat{\mathbf{x}}.$$

Deflated PCG Method

The deflated linear system can also be preconditioned by an SPD matrix $\mathbf{M}$.
The deflated preconditioned system to solve with CG is [15]:

$$\tilde{\mathbf{P}}\tilde{\mathbf{A}}\hat{\tilde{\mathbf{x}}} = \tilde{\mathbf{P}}\tilde{\mathbf{b}},$$

where:

$$\tilde{\mathbf{A}} = \mathbf{M}^{-\frac{1}{2}}\mathbf{A}\mathbf{M}^{-\frac{1}{2}}, \qquad \hat{\tilde{\mathbf{x}}} = \mathbf{M}^{\frac{1}{2}}\hat{\mathbf{x}}, \qquad \tilde{\mathbf{b}} = \mathbf{M}^{-\frac{1}{2}}\mathbf{b}$$

This method is called the Deflated Preconditioned Conjugate Gradient $DPCG$ method.
In practice $\mathbf{M}^{-1}\mathbf{PAx} = \mathbf{M}^{-1}\mathbf{Pb}$ is computed and the error is bounded by:

$$||\mathbf{x} - \mathbf{x}^{i+1}||_{\mathbf{A}} \leq 2||\mathbf{x} - \mathbf{x}^0||_{\mathbf{A}} \left(\frac{\sqrt{\kappa_{eff}(\mathbf{M}^{-1}\mathbf{PA})} - 1}{\sqrt{\kappa_{eff}(\mathbf{M}^{-1}\mathbf{PA})} + 1} \right)^{i+1},$$

were $\kappa_{eff} = \frac{\lambda_{max}(M^{-1}PA)}{\lambda_{min}(M^{-1}PA)}$ is the effective condition number and $\lambda_{min}(M^{-1}PA)$ is the smallest non-zero eigenvalue of $M^{-1}PA$.

3.1 Choices of Deflation Vectors

The deflation method is used to remove the effect of the most unfavorable eigenvalues of $\mathbf{A}$. If the matrix $\mathbf{Z}$ contains eigenvectors corresponding to the unfavorable eigenvalues, the convergence of the iterative method is achieved faster. However, to obtain and to apply the eigenvectors is costly in view of memory and CPU time. Therefore, a good choice of the matrix $\mathbf{Z}$ that efficiently approximate the eigenvectors is essential for the applicability of the method.
A good choice of the deflation vectors is usually problem-dependent. Available information on the system is, in general, used to obtain these vectors. Most of the techniques used to choose deflation vectors are based on approximating eigenvectors, recycling [18], subdomain deflation vectors [2] or multigrid and multilevel based deflation techniques [15, 19]. A summary of these techniques is given below.

Recycling Deflation. A set of search vectors previously used is reused to build the deflation-subspace matrix [18]. The vectors could be, for example, $q-1$ solution vectors of the linear system with different right-hand sides or of different time steps. The matrix $\mathbf{Z}$ containing these solutions is:

$$\mathbf{Z} = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(q-1)}].$$

Subdomain Deflation. The domain is divided into several subdomains, using domain decomposition techniques or taking into account the properties of the problem. For each subdomain, there is a deflation vector that contains ones for cells in the subdomain and zeros for cells outside [2].

Multi Grid and Multilevel Deflation. For the multigrid and multilevel methods, the prolongation and restriction matrices are used to pass from one level or grid to another. These matrices can be used as the deflation-subspace matrices $\mathbf{Z}$ [15].

4 Proper Orthogonal Decomposition (POD)

As mentioned before, in this work we want to combine deflation techniques and Proper Orthogonal Decomposition method (POD) to reduce the number of iterations necessary to solve the linear system obtained from reservoir simulation in a cheap and automatic way. In this section, we give a brief overview of the POD method.
The POD method is a Model Order Reduction (MOR) method, where a high-order model is projected onto a space spanned by a small set of orthonormal basis vectors. The high dimensional variable $\mathbf{x} \in \mathbb{R}^n$ is approximated by a linear combination of l orthonormal basis vectors [8]:

$$\mathbf{x} \approx \sum_{i=1}^{l} c_i \psi_i, \tag{4.1}$$

where $\psi_i \in \mathbb{R}^n$ are the basis vectors and c_i are their corresponding coefficients. In matrix notation, equation (4.1) is rewritten as :

$$\mathbf{x} \approx \Psi \mathbf{c},$$

where $\Psi = [\psi_1 \ \psi_2 \ .. \ \psi_l]$, $\Psi \in \mathbb{R}^{n \times l}$ is the matrix containing the basis vectors, and $\mathbf{c} \in \mathbb{R}^l$ is the vector containing the coefficients of the basis vectors.
The basis vectors ψ_i are computed from a set of 'snapshots' $\{\mathbf{x_i}\}_{i=1,..,m}$, obtained by simulation or experiments [9]. In POD, the basis vectors $\{\psi_j\}_{j=1}^{l}$, are l eigenvectors corresponding to the largest eigenvalues $\{\sigma_j\}_{j=1}^{l}$ of the data snapshot correlation matrix $\mathbf{R}$.

$$\mathbf{R} := \frac{1}{m}\mathbf{X}\mathbf{X}^T \equiv \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}_i \mathbf{x}_i^T, \qquad \mathbf{X} := [\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_m], \tag{4.2}$$

where $\mathbf{X} \in \mathbb{R}^{n \times m}$ is an SPSD matrix containing the previously obtained snapshots. The l eigenvectors should contain almost all the variability of the snapshots. Usually, they are chosen as the eigenvectors of the maximal number (l) of eigenvalues satisfying [9]:

$$\frac{\sum_{j=1}^{l} \sigma_j}{\sum_{j=1}^{m} \sigma_j} \leq \alpha, \qquad 0 < \alpha \leq 1, \tag{4.3}$$

with α close to 1. The eigenvalues σ_j are ordered from large to small with σ_1 the largest eigenvalue of $\mathbf{R}$. It is not necessary to compute the eigenvalues from $\mathbf{X}\mathbf{X}^T$, instead, it is possible to compute the eigenvalues of the much smaller matrix $\mathbf{X}^T\mathbf{X}$ (see Appendix C).

include possible x_i have length one.

5 Deflation vector analysis.

As mentioned in Section 3, it is important to choose 'good' deflation vectors if we want to speed up an iterative method.
We can use solutions of systems slightly different from the original (snapshots) as deflation vectors. For this, we need to choose the way of selecting these snapshots. The idea behind this selection is to obtain a small number of snapshots and, at the same time, obtain the largest amount of information from the system.
In this section, two lemmas are proved. The lemmas are helpful to select the systems to solve to obtain the snapshots.

Lemma 1. Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a non-singular matrix, and $\mathbf{x}$ is a solution of:

$$\mathbf{A}\mathbf{x} = \mathbf{b}. \tag{5.1}$$

Let $\mathbf{x}_i, \mathbf{b}_i \in \mathbb{R}^n$, $i = 1, ..., m$, be vectors linearly independent $(l.i.)$ and

$$\mathbf{A}\mathbf{x}_i = \mathbf{b}_i. \tag{5.2}$$

The following equivalence holds

$$\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{x}_i \qquad \Leftrightarrow \qquad \mathbf{b} = \sum_{i=1}^{m} c_i \mathbf{b}_i. \tag{5.3}$$

Proof $\Rightarrow$

$$\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{x}_i \Rightarrow \mathbf{b} = \sum_{i=1}^{m} c_i \mathbf{b}_i. \tag{5.4}$$

Substituting $\mathbf{x}$ from (5.4) into $\mathbf{A}\mathbf{x} = \mathbf{b}$ leads to:

$$\mathbf{A}\mathbf{x} = \sum_{i=1}^{m} \mathbf{A}c_i\mathbf{x}_i = \mathbf{A}(c_1\mathbf{x}_1 + ... + c_m\mathbf{x}_m).$$

Using the linearity of $\mathbf{A}$ the equation above can be rewritten as:

$$\mathbf{A}c_1\mathbf{x}_1 + ... + \mathbf{A}c_m\mathbf{x}_m = c_1\mathbf{b}_1 + ... + c_m\mathbf{b}_m = \mathbf{B}\mathbf{c}. \tag{5.5}$$

where $\mathbf{B} \in \mathbb{R}^{n\times m}$, $\mathbf{c} \in \mathbb{R}^m$, and the columns of $\mathbf{B}$ are the vectors $\mathbf{b}_i$.
From (5.1) and (5.5) we get:

$$\mathbf{A}\mathbf{x} = \mathbf{b} = c_1\mathbf{b}_1 + ... + c_m\mathbf{b}_m = \sum_{i=1}^{m} c_i\mathbf{b}_i.$$

Proof $\Leftarrow$

$$\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{x}_i \Leftarrow \mathbf{b} = \sum_{i=1}^{m} c_i \mathbf{b}_i. \tag{5.6}$$

Substituting $\mathbf{b}$ from (5.6) into $\mathbf{A}\mathbf{x} = \mathbf{b}$ leads to:

$$\mathbf{A}\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{b}_i. \tag{5.7}$$

Since $\mathbf{A}$ is non-singular, multiplying (5.2) and (5.6) by $\mathbf{A}^{-1}$ we obtain:

$$\mathbf{x}_i = \mathbf{A}^{-1}\mathbf{b}_i,$$

$$\mathbf{x} = \mathbf{A}^{-1} \sum_{i=1}^{m} c_i \mathbf{b}_i = \sum_{i=1}^{m} c_i \mathbf{A}^{-1}\mathbf{b}_i,$$

then

$$\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{x}_i. \tag{5.8}$$

$\boxtimes$

Lemma 2. If the the deflation matrix $\mathbf{Z}$ is constructed with a set of m vectors

$$\mathbf{Z} = \begin{bmatrix} \mathbf{x}_1 & \dots & \dots & \mathbf{x}_m \end{bmatrix}, \tag{5.9}$$

such that $\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{x}_i$, with $\mathbf{x}_i$ *l.i.*, then the solution of system (5.1) is obtained with one iteration of DCG.

Proof.

The relation between $\hat{\mathbf{x}}$ and $\mathbf{x}$ is given in Equation (3.8):

$$\mathbf{x} = \mathbf{Q}\mathbf{b} + \mathbf{P}^T\hat{\mathbf{x}}.$$

For the first term $\mathbf{Q}\mathbf{b}$, taking $\mathbf{b} = \sum_{i=1}^{m} c_i \mathbf{b}_i$ we have:

$$\begin{aligned}
\mathbf{Q}\mathbf{b} &= \mathbf{Z}\mathbf{E}^{-1}\mathbf{Z}^T \left(\sum_{i=1}^{m} c_i \mathbf{b}_i \right) \\
&= \mathbf{Z}(\mathbf{Z}^T\mathbf{A}\mathbf{Z})^{-1}\mathbf{Z}^T \left(\sum_{i=1}^{m} c_i \mathbf{A}\mathbf{x}_i \right) \qquad \text{using Lemma 1} \\
&= \mathbf{Z}(\mathbf{Z}^T\mathbf{A}\mathbf{Z})^{-1}\mathbf{Z}^T \left(\mathbf{A}\mathbf{x}_1 c_1 + \dots + \mathbf{A}\mathbf{x}_m c_m \right) \\
&= \mathbf{Z}(\mathbf{Z}^T\mathbf{A}\mathbf{Z})^{-1}\mathbf{Z}^T(\mathbf{A}\mathbf{Z}\mathbf{c}) \\
&= \mathbf{Z}(\mathbf{Z}^T\mathbf{A}\mathbf{Z})^{-1}(\mathbf{Z}^T\mathbf{A}\mathbf{Z})\mathbf{c} \\
&= \mathbf{Z}\mathbf{c} = c_1\mathbf{x}_1 + c_2\mathbf{x}_2 + c_3\mathbf{x}_3 + c_4\mathbf{x}_4 + c_5\mathbf{x}_5 \\
&= \sum_{i=1}^{m} c_i \mathbf{x}_i = \mathbf{x}.
\end{aligned}$$

Therefore,

$$\mathbf{x} = \mathbf{Q}\mathbf{b}, \tag{5.10}$$

is the solution to the original system.
For the second term of Equation (3.8), $\mathbf{P}^T\hat{\mathbf{x}}$, we compute $\hat{\mathbf{x}}$ from Equation (3.7):

$$\begin{aligned}
\mathbf{P}\mathbf{A}\hat{\mathbf{x}} &= \mathbf{P}\mathbf{b} \\
\mathbf{A}\mathbf{P}^T\hat{\mathbf{x}} &= (\mathbf{I} - \mathbf{A}\mathbf{Q})\mathbf{b} \qquad \text{using 3 f) and definition of } \mathbf{P}, \\
\mathbf{A}\mathbf{P}^T\hat{\mathbf{x}} &= \mathbf{b} - \mathbf{A}\mathbf{Q}\mathbf{b} \\
\mathbf{A}\mathbf{P}^T\hat{\mathbf{x}} &= \mathbf{b} - \mathbf{A}\mathbf{x} = 0 \qquad \text{taking } \mathbf{Q}\mathbf{b} = \mathbf{x} \text{ from above}, \\
\mathbf{P}^T\hat{\mathbf{x}} &= 0 \qquad \text{as } \mathbf{A} \text{ is invertible.}
\end{aligned}$$

Then we have obtain the solution

$$\mathbf{x} = \mathbf{Q}\mathbf{b} + \mathbf{P}^T\hat{\mathbf{x}} = \mathbf{Q}\mathbf{b},$$

in one step of DCG.

⊠

5.1 Accuracy of the snapshots.

If we use an iterative method to obtain an approximate solution $\mathbf{x}^k$ for the system $\mathbf{A}\mathbf{x} = \mathbf{b}$, we cannot compute the relative error e_r (Equation 5.11) of the approximation with respect to the true solution because the true solution is unknown,

$$e_r = \frac{||\mathbf{x} - \mathbf{x}^k||_2}{||\mathbf{x}||_2}. \tag{5.11}$$

Instead, we compute the relative residual r_r (Equation 5.12),

$$r_r = \frac{||\mathbf{r}^k||_2}{||\mathbf{b}||_2} \leq \epsilon, \tag{5.12}$$

and we set a stopping criterium ϵ or tolerance, that is related to the relative error as follows [20] (see Appendix B),

$$\frac{||\mathbf{x} - \mathbf{x}^k||_2}{||\mathbf{x}||_2} \leq \kappa_2(\mathbf{A})\epsilon = r_r.$$

Various tolerance values can be used in the experiments for the snapshots as well as for the solution of the original system.
If the maximum relative residual for the snapshots is $\epsilon = 10^{-\eta}$ then the error in the snapshots is given by

$$\frac{||\mathbf{x}_i - \mathbf{x}_i^k||_2}{||\mathbf{x}_i||_2} \leq \kappa_2(\mathbf{A}) \times 10^{-\eta} = r_r.$$

From Equation (5.8), if we compute m snapshots with an iterative method such that the solution of $\mathbf{x}$ is a linear combination of these vectors, after one iteration of DCG we obtain

$$\mathbf{x}^1 = \sum_{i=1}^{m} c_i \mathbf{x}_i^{1(i)},$$

where $\mathbf{x}_i^{1(i)}$ is the $i-th$ approximated solution of the snapshot i after 1 iteration.
The error of this solution is given by:

$$\frac{||\mathbf{x} - \mathbf{x}^1||_2}{||\mathbf{x}||_2} = \frac{||\sum_{i=1}^{m} c_i(\mathbf{x}_i - \mathbf{x}_i^1)||_2}{||\sum_{i=1}^{m} c_i \mathbf{x}_i||_2} \leq \frac{\sum_{i=1}^{m} |c_i| \times \kappa_2(\mathbf{A}) \times 10^{-\eta}}{||\sum_{i=1}^{m} c_i \mathbf{x}_i||_2}.$$

Which means that the approximation has an error of the order $\kappa_2(\mathbf{A}) \times 10^{-\eta}$.
From Lemma 2 we know that if we use the snapshots $\mathbf{x}_i$ as deflation vectors, for the deflation method the solution is given by (Equation 5.10):

$$\mathbf{x} = \mathbf{Q}\mathbf{b}.$$

If the approximation of $\mathbf{x}$ has an error of the order $\kappa_2(\mathbf{A}) \times 10^{-\eta}$, then the solution achieved with the deflation method will have the same error,

$$\mathbf{Q}\mathbf{b} - \mathbf{x}^1 = \kappa_2(\mathbf{A}) \times 10^{-\eta}.$$

Therefore, it is important to take into account the condition number of the matrix to estimate the accuracy of the deflation vectors.

5.2 Boundary conditions.

From Lemma 2, we know that if we use as deflation vectors a set of m snapshots

$$\mathbf{Z} = [\mathbf{x}_1 \quad ... \quad \mathbf{x}_m],$$

such that $\mathbf{x} = \sum_{i=1}^{m} c_i \mathbf{x}_i$, where $\mathbf{x}$ is the solution of the system $\mathbf{A}\mathbf{x} = \mathbf{b}$, the solution of the latter system is achieved with one DCG iteration.
In our application, only a small number (m) of elements of the right-hand side vector $\mathbf{b}$ can be changed. This implies that every $\mathbf{b}$ can be written as $\mathbf{b} = \sum_{i=1}^{m} c_i \mathbf{b}_i$. Using Lemma 1, this implies that $\mathbf{x}$ is such that $\mathbf{x} \in span\{\mathbf{x}_1, ..., \mathbf{x}_m\}$, which is called the solution span. Therefore, it is necessary to find the solution span of the system, such that the sum of the elements in the solution span and the sum of right-hand sides give as result the original system. In this section we explore the subsystems that should be chosen, depending on the boundary conditions of the original system.

Neumann Boundary conditions

When we have Neumann boundary conditions everywhere, the resulting matrix $\mathbf{A}$ is singular, and $\mathbf{A}[1 \quad 1 \quad ... \quad 1 \quad 1]^T = \mathbf{0}$, $Ker(\mathbf{A}) = span([1 \quad 1 \quad ... \quad 1 \quad 1]^T)$. Note that $\mathbf{Ax} = \mathbf{b}$ has only a solution if $\mathbf{b} \in span\{\mathbf{a}_1, ..., \mathbf{a}_n\}$ (with $\mathbf{a}_i$ the $i-th$ column of $\mathbf{A}$), which is equivalent to $\mathbf{b} \perp Ker(\mathbf{A})$ [21]. This implies that if we have m sources with value s_i for the vector $\mathbf{b}_i$, we need that

$$\sum_{j=1}^{m} s_i^j = 0.$$

Then, for each nonzero right-hand side we need to have at least two sources. Therefore, we can have at most $m-1$ linearly independent right-hand sides $\mathbf{b}_i$ containing two sources. This means that the solution space has dimension $m-1$ and it can be spanned by $span\{\mathbf{x}_1, ..., \mathbf{x}_{m-1}\}$. Each of these subsystems will have the same no-flux conditions (Neumann) in all the boundaries. As the original system is a linear combination of the subsystems (Lemma 1), the deflation vectors can be chosen as the solutions corresponding to the subsystems. Therefore, the deflation matrix will be given by:

$$\mathbf{Z} = [\mathbf{x}_1 \qquad ... \qquad \mathbf{x}_{m-1}],$$

and if the accuracy of the snapshots used as deflation vectors is high enough (see Section 5.1), the solution is expected to be achieved in one iteration.

Dirichlet Boundary conditions

In this case, the right-hand side of the system can contain the values of the boundary $\mathbf{b}_b$ and the sources of the system $\mathbf{s}_i$. If we have m sources, as in the previous case, the right-hand side will be given by:

$$\mathbf{b} = \sum_{i=1}^{m} c_i \mathbf{s}_i + \mathbf{b}_b.$$

The subsystems will be $m+1$, where one of them corresponds to the boundary conditions $\mathbf{Ax}_b = \mathbf{b}_b$, and the other m will correspond to the sources $\mathbf{Ax}_i = \mathbf{s}_i$. Therefore, snapshot $m+1$ will be the solution $\mathbf{x}_b$ of the system with no sources and the Dirichlet boundary conditions of the original system. The other m snapshots will correspond to the m sources with homogeneous Dirichlet boundary conditions. Then, the solution space will be given by $span\{\mathbf{x}_1, ..., \mathbf{x}_m, \mathbf{x}_b\}$. If we use the solution of the $m+1$ snapshots as deflation vectors, with the correct accuracy, we will obtain the solution within one iteration.