Parallel Algorithms for Computational Models of Geophysical Systems

By

Antonio Carrillo-Ledesma, Ismael Herrera[*]*, and Luis M. de la Cruz*

Instituto de Geofísica

Universidad Nacional Autónoma de México (UNAM)

Apdo. Postal 22-582, México, 14000 D.F.

Email: iherrera@unam.mx

Abstract

Mathematical models of many geophysical systems are based on the computational processing of large-scale algebraic systems. The most advanced computational tools are based on massively parallel processors. The most effective software for solving partial differential equations in parallel intends to achieve the *DDM-paradigm*. A set of four algorithms, the *DVS-algorithms*, which achieve it, and of very general applicability, has recently been developed and here they are explained. Also, their application to problems that frequently occur in Geophysics is illustrated.

Keywords: Computational-geophysics, Computational-PDEs, non-overlapping DDM, BDDC; FETI-DP

Resumen

Los modelos matemáticos de muchos sistemas geofísicos requieren el procesamiento de sistemas algebraicos de gran escala. Las herramientas computacionales más avanzadas están masivamente paralelizadas. El software más efectivo para resolver ecuaciones diferenciales parciales en paralelo intenta alcanzar el *paradigma de los métodos de descomposición de dominio*, que hasta ahora se había mantenido como un anhelo no alcanzado. Sin embargo, un grupo de cuatro algoritmos –los *algoritmos DVS*- que lo alcanzan y que tiene aplicabilidad muy general se ha desarrollado recientemente. Este artículo está dedicado a presentarlos y a ilustrar su aplicación a problemas que se presentan frecuentemente en la investigación y el estudio de la Geofísica.

Keywords: Computational-geophysics, Computational-PDEs, non-overlapping DDM, BDDC; FETI-DP

1. Introduction

Mathematical models of many systems of interest, including very important continuous systems of Earth Sciences and Engineering, lead to a great variety of partial differential equations (PDEs) whose solution methods are based on the computational processing of large-scale algebraic systems. Furthermore, the incredible expansion experienced by the existing computational hardware and software has made amenable to effective treatment problems of an ever increasing diversity and complexity, posed by scientific and engineering applications [PITAC, 2006].

Parallel computing is outstanding among the new computational tools and, in order to effectively use the most advanced computers available today, massively parallel software is required. Domain decomposition methods (DDMs) have been developed precisely for effectively treating PDEs in parallel [DDM Organization, 2012]. Ideally, the main objective of domain decomposition research is to produce algorithms capable of '*obtaining the global*

solution by exclusively solving local problems', but up-to-now this has only been an aspiration; that is, a strong desire for achieving such a property and so we call it *'the DDM-paradigm'*. In recent times, numerically competitive DDM-algorithms are *non-overlapping*, *preconditioned* and necessarily incorporate *constraints* [Dohrmann, 2003; Farhat et al, 1991; Farhat et al, 2000; Farhat et al, 2001; Mandel, 1993; Mandel et al, 1996; Mandel and Tezaur, 1996; Mandel et al, 2001; Mandel et al, 2003; Mandel et al, 2005; J. Li et al, 2005; Toselli et al, 2005], which pose an additional challenge for achieving the *DDM-paradigm*.

Recently a group of four algorithms, referred to as the *'DVS-algorithms'*, which fulfill the *DDM-paradigm*, was developed [Herrera et al , 2012; L.M. de la Cruz et al, 2012; Herrera and L.M. de la Cruz et al , 2012; Herrera and Carrillo Ledesma et al , 2012]. To derive them a new discretization method, which uses a non-overlapping system of nodes (the *derived-nodes*), was introduced. This discretization procedure can be applied to any boundary-value problem, or system of such equations. In turn, the resulting system of discrete equations can be treated using any available DDM-algorithm. In particular, two of the four *DVS-algorithms* mentioned above were obtained by application of the well-known and very effective algorithms BDDC and FETI-DP [Dohrmann, 2003; Farhat et al , 1991; Farhat et al, 2000; Farhat et al, 2001; Mandel et al, 1993; Mandel et al, 1996; Mandel and Tezaur, 1996; Mandel et al, 2001; Mandel et al, 2003; Mandel et al, 2005; J. Li et al, 2005; Toselli et al, 2005]; these will be referred to as the *DVS-BDDC* and *DVS-FETI-DP* algorithms. The other two, which will be referred to as the *DVS-PRIMAL* and *DVS-DUAL* algorithms, were obtained by application of two new algorithms that had not been previously reported in the literature [Herrera et al, 2011; Herrera et al, 2010; Herrera et al, 2009; Herrera et al, 2009; Herrera, 2008; Herrera, 2007]. As said before, the four *DVS-algorithms* constitute a group of preconditioned and constrained algorithms that, for the first time, fulfill the *DDM-paradigm* [Herrera et al , 2013; L.M. de la Cruz et al, 2012].

Both, BDDC and FETI-DP, are very well-known [Dohrmann, 2003; Farhat et al, 1991; Farhat et al, 2000; Farhat et al, 2001; Mandel et al, 1993; Mandel et al, 1996; Mandel and Tezaur, 1996; Mandel et al, 2001]; and both are highly efficient. Recently, it was established that these two methods are closely related and its numerical performance is quite similar [Mandel et al, 2003; Mandel et al, 2005]. On the other hand, through numerical experiments, we have established that the numerical performances of each one of the members of *DVS-algorithms* group (*DVS-BDDC*, *DVS-FETI-DP*, *DVS-PRIMAL* and *DVS-DUAL*) are very similar too. Furthermore, we have carried out comparisons of the performances of the standard versions of BDDC and FETI-DP with *DVS-BDDC* and *DVS-FETI-DP*, and in all such numerical experiments the DVS algorithms have performed significantly better.

Each *DVS-algorithm* possesses the following conspicuous features:

- It fulfills the *DDM-paradigm*;
- It is applicable to symmetric, non-symmetric and indefinite matrices (i.e., neither positive, nor negative definite); and
- It is preconditioned and constrained, and has update numerical efficiency.

Furthermore, the uniformity of the algebraic structure of the matrix-formulas that define each one of them is remarkable.

This article is organized as follows. In Section 2 the basic definitions for the DVS framework are given; here we define the set of 'derived-nodes', internal, interface, primal and dual nodes, the 'derived-vector-space', among others. Section 3 is devoted to define the new set of vector spaces that conforms the DVS framework; the Euclidean inner product, is also defined here. In Section 4 the 'transformed-problem' on the derived-nodes is explained in detail, and this is our starting point to define the DVS algorithms. Section 5 presents a summary of the four DVS-algorithms: DVS-BDDC, DVS-FETI-DP, DVS-PRIMAL and DVS-DUAL. In Section 6 we give the numerical procedures we use to fulfilling the DDM-paradigm, and we explain in detail the implementation issues. Finally, in Section 7 we show some numerical results obtained after the application of the DVS-algorithms in the solution of several boundary values problems of interest in Geophysics. We studied examples for a single-equation, for the cases of symmetric, non-symmetric and indefinite problems. We also present results for an elasticity problem, where a system of PDE equations is solved.

2.-DVS Framework: A Summary

The *'derived-vector-space framework (DVS-framework)'* is applied to the discrete system of equations that is obtained after the partial differential equation, or system of such equations, has been discretized. The procedure is independent of the method of discretization that is used. Thus, the DVS-framework's starting point is a system of linear algebraic equations that is referred to as the *'original problem'*:

$$\underline{\underline{\widehat{A}}}\,\underline{\widehat{u}} = \underline{\widehat{f}} \tag{2.1}$$

However, in the *DVS* setting one does not work with the set of nodes originally used for discretizing the problem the *original-nodes'* (Figure 1). Instead, one uses an auxiliary set of nodes: the *'derived-nodes'*. Each one of such nodes has the property that it belongs to one and only one subdomain of the *coarse mesh*.

Indeed, generally after a *coarse-mesh* has been introduced, some *original-nodes* belong to more than one subdomain of the *coarse-mesh* (Figure 2), which is inconvenient for achieving the *DDM-paradigm*. Therefore, in the *DVS-framework*, each *original-node* that belongs to more than one subdomain is divided into as many new nodes –the *derived-nodes* (Figure 3) - as subdomains it belongs to. Then, the *derived-nodes* so obtained are distributed into the *coarse-mesh* subdomains so that each *derived-node* is assigned to one and only one subdomain of the *coarse-mesh* (Figure 4). Once this has been done, a convenient notation is

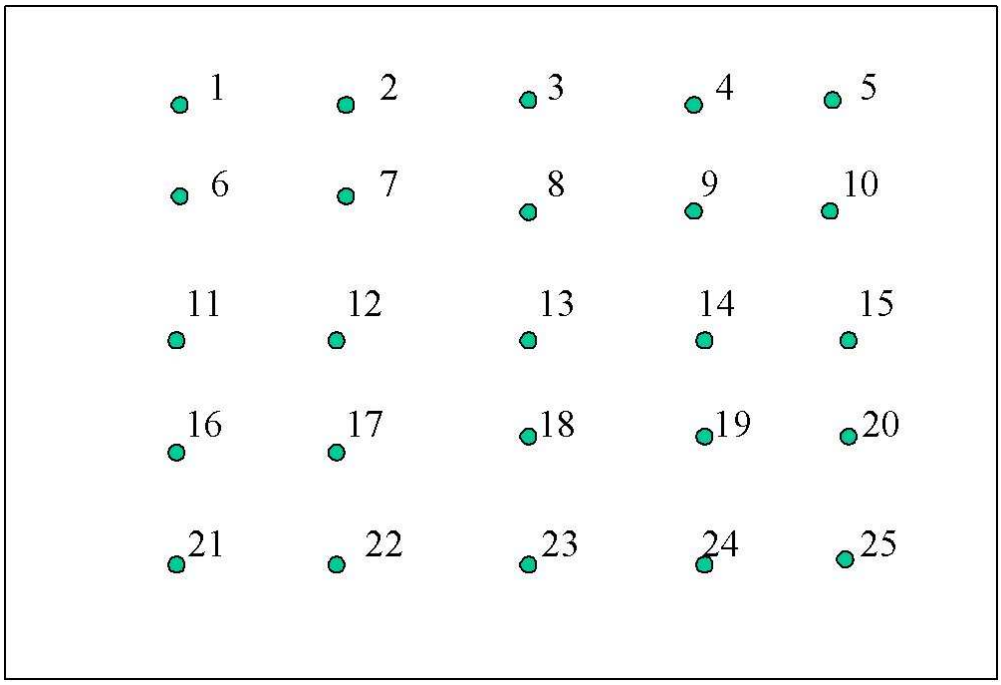

Figure 1. The *'original nodes'*

to label each *derived-node* by a pair of natural numbers: the first one indicating the *original-node* from which it derives and the second one, the subdomain to which it is assigned.

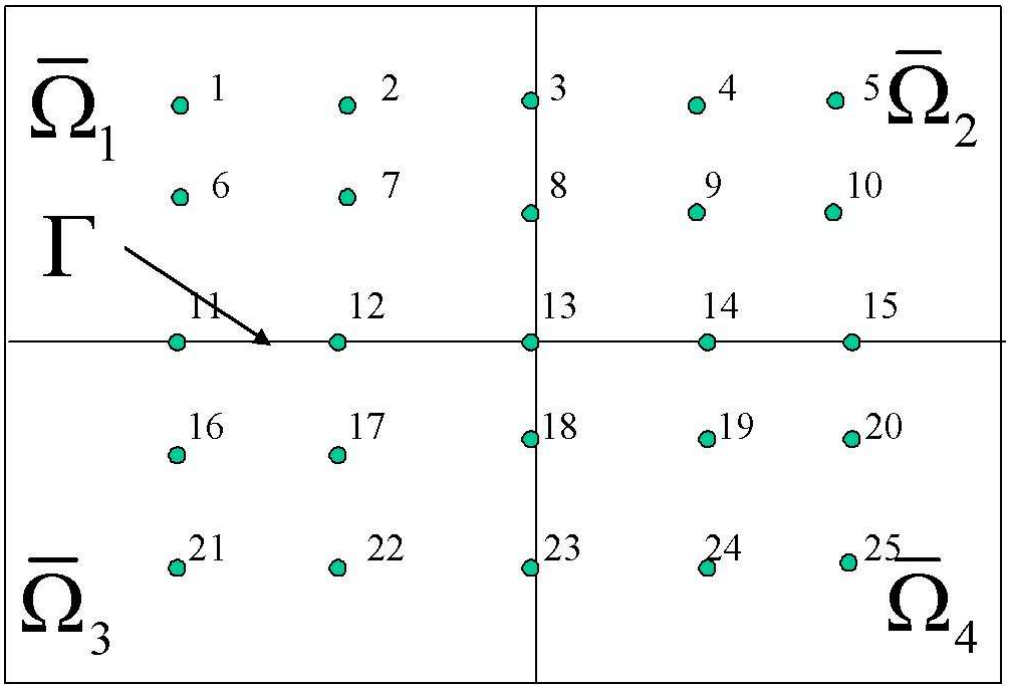

Figure 2. The *original nodes* in the *coarse-mesh*

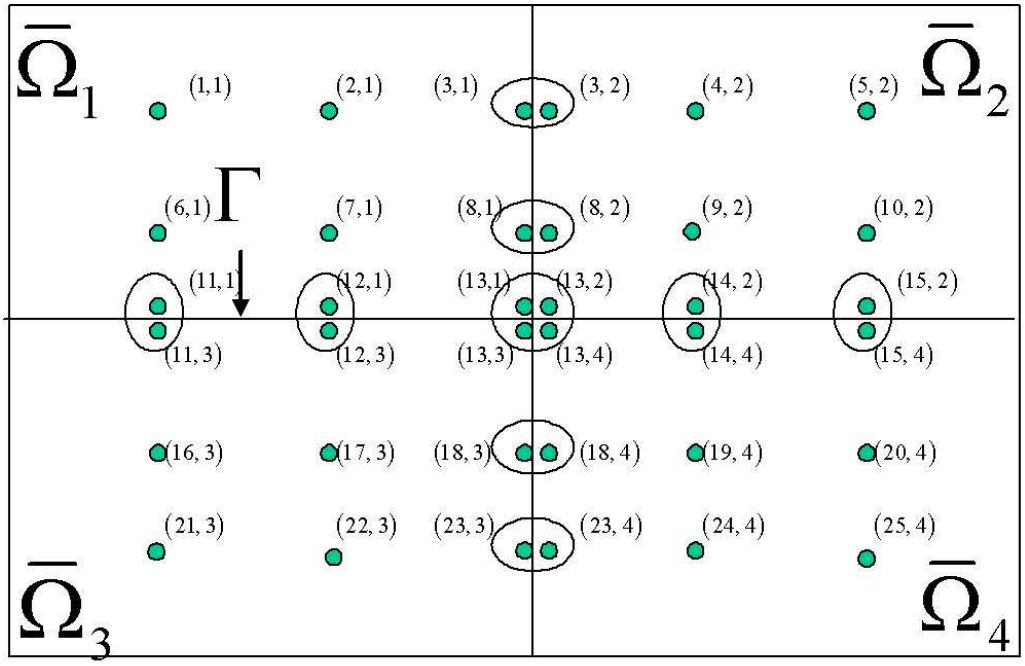

Figure 3. The *mitosis*

The real-valued functions defined in the set of *derived-nodes* constitute a vector-space: the *'derived-vector-space'*, W. This space becomes a finite-dimensional Hilbert-space when it is supplied with the inner-product that is usually introduced when dealing with real-valued functions defined in a set of nodes; this is referred to as the *Euclidean inner-product*.

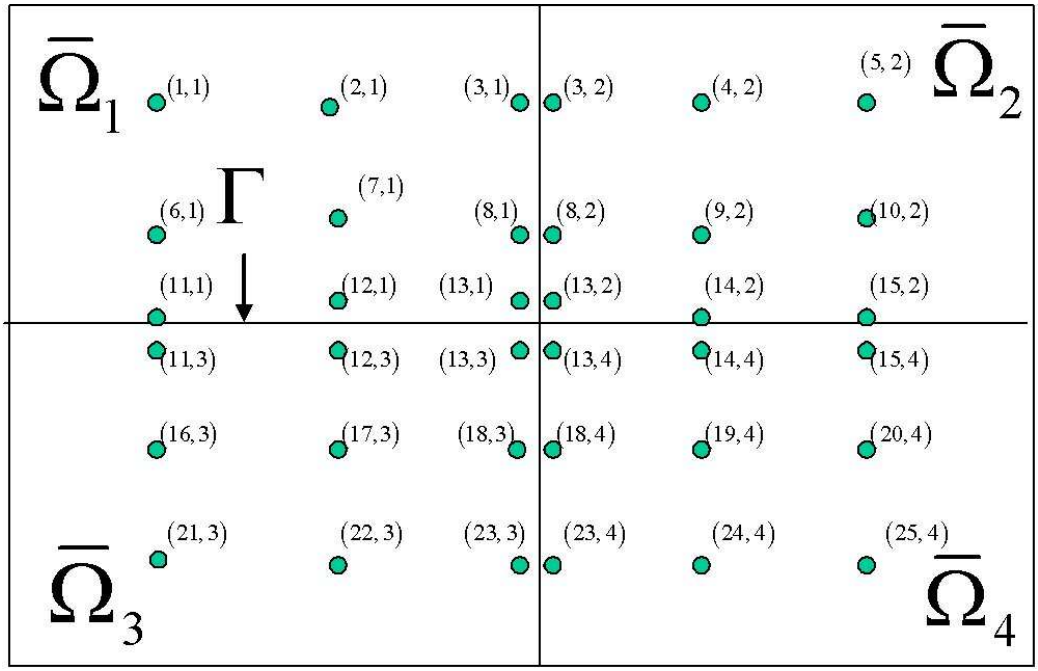

Figure 4. The *derived-nodes* distributed in the *coarse-mesh*

Afterwards, a new problem (referred to as the *'transformed problem'*) is defined in the *derived-vector-space*, which is equivalent to the original system of discrete equations. Thereafter, all the numerical and computational work is carried out in the *DVS-space*.

Before leaving this Section, we dwell a little further on the meaning of a *coarse-mesh*. By it, we mean a partition of Ω into a set of non-overlapping subdomains, $\{\Omega_1,...,\Omega_E\}$, such that for each $\alpha=1,...,E$, Ω_α is open and:

$$\Omega_\alpha \cap \Omega_\beta = \varnothing \; and \; \Omega \subset \bigcup_{\alpha=1}^{E} \bar{\Omega}_\alpha \tag{2.2}$$

Where $\bar{\Omega}_\alpha$ stands for the closure of Ω_α. The set of *'subdomain-indices'* will be

$$\hat{\mathrm{E}} \equiv \{1,...,E\} \tag{2.3}$$

$\hat{\mathrm{N}}^\alpha$, $\alpha=1,...,E$, will be used for the subset of *original-nodes* that correspond to nodes pertaining to $\bar{\Omega}_\alpha$. As usual, nodes will be classified into *'internal'* and *'interface-nodes'*: a node is *internal* if it belongs to only one partition-subdomain closure and it is an *interface-node*, when it belongs to more than one. For the application of *dual-primal* methods, *interface-nodes* are classified into *'primal'* and *'dual'* nodes. We define:

- $\hat{\mathrm{N}}_\mathrm{I} \subset \hat{\mathrm{N}}$ as the set of *internal-nodes*;
- $\hat{\mathrm{N}}_\Gamma \subset \hat{\mathrm{N}}$ as the set of *interface-nodes*;
- $\hat{\mathrm{N}}_\pi \subset \hat{\mathrm{N}}_\Gamma \subset \hat{\mathrm{N}}$ as the set of *primal-nodes*[1]; and
- $\hat{\mathrm{N}}_\Delta \subset \hat{\mathrm{N}}$ as the set of *dual-nodes*.

The set of *primal-nodes* is required to be a subset of $\hat{\mathrm{N}}_\Gamma$ and, in principle, could be otherwise chosen arbitrarily. However, the algorithms considered by *domain decomposition methods* are iterative-algorithms and their rate of convergence depends crucially on the selection of the set $\hat{\mathrm{N}}_\pi$. Thus, criteria for selecting $\hat{\mathrm{N}}_\pi$ have been studied extensively (see [Toselli et al, 2005], for detailed discussions of this topic). Each one of the following two families of node-subsets is disjoint: $\{\hat{\mathrm{N}}_\mathrm{I}, \hat{\mathrm{N}}_\Gamma\}$ and $\{\hat{\mathrm{N}}_\mathrm{I}, \hat{\mathrm{N}}_\pi, \hat{\mathrm{N}}_\Delta\}$. Furthermore, these node subsets fulfill the relations:

$$\hat{\mathrm{N}} = \hat{\mathrm{N}}_\mathrm{I} \cup \hat{\mathrm{N}}_\Gamma = \hat{\mathrm{N}}_\mathrm{I} \cup \hat{\mathrm{N}}_\pi \cup \hat{\mathrm{N}}_\Delta \; and \; \hat{\mathrm{N}}_\Gamma = \hat{\mathrm{N}}_\pi \cup \hat{\mathrm{N}}_\Delta \tag{2.4}$$

Throughout our developments the *original matrix* $\underline{\underline{\hat{A}}}$ is assumed to be non-singular (i.e., it defines a bijection of $\widehat{W}$ into itself). The following assumption (*'axiom'*) is also adopted in throughout the *DVS-framework*: "When the indices $p \in \hat{\mathrm{N}}^\alpha$ and $q \in \hat{\mathrm{N}}^\beta$ are *internal original-nodes*, while $\alpha \neq \beta$, then $p \in \hat{\mathrm{N}}^\alpha$ and $q \in \hat{\mathrm{N}}^\beta$ are unconnected". We recall that

[1] In order to mimic standard notations, we should have used Π instead of the *low-case* π. However, the modified definitions given here yield some convenient algebraic properties.

unconnected means:

$$\widehat{A}_{pq} = \widehat{A}_{qp} = 0 \qquad (2.5)"$$

3.- The Derived-Vector Space (DVS)

In order to have at hand a sufficiently general framework, we consider functions defined on the set X of *derived-nodes* whose value at each *derived-node* is a $dD-vector$. The numerical applications that will be discussed in this paper correspond to two possible choices of d: when the application refers to a single partial differential equation (PDE), $d=1$, and for the problems of elasticity that will be considered, which are governed by a three-equations system, $d=3$.

Independently of the chosen value for d, the set of such functions constitute a vector space, W, referred to as the *'derived-vector space'*. When $\underline{u} \in W$, we write $u(p,\alpha)$ for the value of $\underline{u}$ at the *derived-node* (p,α). We observe that, in general, $u(p,\alpha)$ itself is a $d-vector$ and we adopt the notation $u(p,\alpha,i)$, $i=1,...,d$. for the $i-th$ component of $u(p,\alpha)$. When $d=1$ the index i is irrelevant and, in such a case, will deleted throughout.

For every pair of functions, $\underline{u} \in W$ and $\underline{w} \in W$, the *'Euclidean inner product'* is defined to be

$$\underline{u} \bullet \underline{w} = \sum_{(p,\alpha)\in \mathrm{X}} u(p,\alpha) \odot w(p,\alpha) \qquad (3.1)$$

Here, $u(p,\alpha) \odot w(p,\alpha)$ stands for the inner-product of the $dD-vectors$ involved; thus,

$$u(p,\alpha) \odot w(p,\alpha) \equiv \sum_{i=1}^{n} u(p,\alpha,i)\, w_i(p,\alpha,i) \qquad (3.2)$$

A fundamental property of the *derived-vector space* W, is that it constitutes a finite dimensional *Hilbert-space* with respect to the *Euclidean inner-product*.

Let $W' \subset W$ be a linear subspace and assume $\mathrm{M} \subset \mathrm{X}$ is a subset of *derived-nodes*. Then, the notation $W'(\mathrm{M})$ will be used to represent the vector subspace of W', whose elements vanish at every *derived-node* that does not belong to M. Furthermore, corresponding to each *local subset of derived-nodes*, X^{α}, there is a *'local subspace of derived-vectors'*, W^{α}, which is defined by

$$W^{\alpha} \equiv W\left(X^{\alpha}\right) \qquad (3.3)$$

Clearly, when $\underline{u} \in W^{\alpha} \subset W$, $\underline{u}(p,\beta) = 0$ whenever $\beta \neq \alpha$. We observe that

$$W = W^1 \oplus ... \oplus W^E \qquad (3.4)$$

A derived-vector $\underline{u} \in W$ is said to be *continuous* when $\underline{u}(p,\alpha)$ is independent of α. The set of *continuous vectors* constitute the linear subspace, W_{12}.

The orthogonal complement (with respect to the Euclidean inner-product) of $W_{12} \subset W$ is $W_{11} \subset W$. Then $W = W_{11} \oplus W_{12}$. Two projection-matrices $\underline{\underline{a}} : W \to W$ and $\underline{\underline{j}} : W \to W$ are here introduced; they are the projection-operators, with respect to the *Euclidean inner-product* on W_{12} and W_{11}, respectively. When $\underline{u} \in W$, one has

$$\underline{u} = \underline{u}_{11} + \underline{u}_{12} \; with \begin{cases} \underline{u}_{11} \equiv \underline{\underline{j}}\,\underline{u} \in W_{11} \\ \underline{u}_{12} \equiv \underline{\underline{a}}\,\underline{u} \in W_{12} \end{cases} \tag{3.5}$$

the vectors $\underline{\underline{j}}\,\underline{u}$ and $\underline{\underline{a}}\,\underline{u}$ are said to be the *'jump'* and the *'average'* of $\underline{u}$, respectively. Therefore, W_{11} is the *'zero-average'* subspace, while W_{12} is the *'zero-jump'* subspace.

Original-nodes are classified into *'internal'* and *'interface-nodes'*: a node is *internal* if it belongs to only one subdomain-closure of the *coarse-mesh*, and it is an *interface-node* when it belongs to more than one of such closure-subdomains. Some subspaces, significant for our developments, are listed next:

- $W_I \equiv W(\mathrm{I})$;
- $W_\Gamma \equiv W(\Gamma)$;
- $W_\pi \equiv W(\pi)$;
- $W_\Delta \equiv W(\Delta)$;and
- $W_\Pi \equiv W(\Pi)$.

At present, numerically competitive algorithms need to incorporate *restrictions* and to this end, in the *DVS-framework*, a *'restricted subspace'* $W_r \subset W$ is selected. In the developments that follow, it is assumed that:

$$W_r \equiv W_\mathrm{I} + \underline{\underline{a}} W_\pi + W_\Delta \tag{3.6}$$

The matrix $\underline{\underline{a}}^r$ will be the projection-operator on W_r. We observe that when $\underline{u} \in (W_\mathrm{I} + W_\Delta)$, one has $\underline{\underline{a}}^r \underline{u} = \underline{u}$. We also notice that

$$W = W_\mathrm{I} \oplus W_\Gamma = W_\mathrm{I} \oplus W_\pi \oplus W_\Delta \tag{3.7}$$

4.- The Transformed Problem

The *transformed-problem* consists in finding $\underline{u} \in W$ such that

$$\underline{\underline{a}}\,\underline{\underline{A}}^t \underline{u} = \underline{f} \; and \; \underline{\underline{j}}\,\underline{u} = 0 \tag{4.1}$$

Where:

$$\underline{\underline{A}}^t \equiv \sum_{\alpha=1}^{E} \underline{\underline{A}}^\alpha \tag{4.2}$$

and

$$\underline{\underline{\widehat{A}}}^\alpha \equiv \left(\widehat{A}^\alpha_{pq} \right) with \; \widehat{A}^\alpha_{pq} \equiv \frac{\widehat{A}_{pq} \delta^\alpha_{pq}}{s(p,q)} \tag{4.3}$$

together with

$$m(p,q) \equiv \sum_{\alpha=1}^{E} \delta_{pq}^{\alpha} \text{ and } s(p,q) \equiv \begin{cases} 1, \text{when } m(p,q) = 0 \\ m(p,q), \text{when } m(p,q) \neq 0 \end{cases} \tag{4.4}$$

The function $m(p,q)$ is said to be the '*multiplicity*' of the pair (p,q). The '*derived-nodes*' are created after a *coarse-mesh* has been introduced, by dividing the *original-nodes* as explained in the Overview (Section 2), and then with each '*derived-node*' we associate a unique pair of numbers (p,α) such that $\alpha \in \hat{\mathrm{E}}$ and $p \in \hat{\mathrm{N}}^{\alpha}$. In what follows, we identify *derived-nodes* with such pairs.

Then, in order to incorporate the constraints, we define

$$W_r \equiv W(\mathrm{I}) + W(\Delta) + \underline{\underline{a}} W(\pi) \tag{4.5}$$

then, the matrix $\underline{\underline{A}} : W_r \to W_r$ defined by

$$\underline{\underline{A}} \equiv \underline{\underline{a}}^r \, \underline{\underline{A}}^t \, \underline{\underline{a}}^r \tag{4.6}$$

has the property that

$$\underline{\underline{a}} \, \underline{\underline{A}} \underline{\underline{a}} = \underline{\underline{a}} \, \underline{\underline{A}}^t \, \underline{\underline{a}} \tag{4.7}$$

Hence, Eq.(4.1) is replaced by

$$\underline{\underline{a}} \, \underline{\underline{A}} \underline{u} = \underline{f} \text{ and } \underline{\underline{j}} \underline{u} = 0 \tag{4.8}$$

For matrices and vectors the following notation is adopted:

$$\underline{\underline{A}} \equiv \begin{pmatrix} \underline{\underline{A}}_{\Pi\Pi} & \underline{\underline{A}}_{\Pi\Delta} \\ \underline{\underline{A}}_{\Delta\Pi} & \underline{\underline{A}}_{\Delta\Delta} \end{pmatrix}; \begin{cases} \underline{u} \equiv \begin{pmatrix} \underline{u}_{\Pi} \\ \underline{u}_{\Delta} \end{pmatrix} \text{ for any } \underline{u} \in W \\ \underline{u} \equiv \begin{pmatrix} \underline{u}_{\mathrm{I}} \\ \underline{u}_{\pi} \end{pmatrix} \text{ for any } \underline{u} \in W(\Pi) \end{cases} \tag{4.9}$$

where the matrices

$$\begin{aligned} &\underline{\underline{A}}_{\Pi\Pi} : W_r(\Pi) \to W_r(\Pi), \ \underline{\underline{A}}_{\Pi\Delta} : W_r(\Delta) \to W_r(\Pi) \\ &\underline{\underline{A}}_{\Delta\Pi} : W_r(\Pi) \to W_r(\Delta), \ \underline{\underline{A}}_{\Delta\Delta} : W_r(\Delta) \to W_r(\Delta) \end{aligned} \tag{4.10}$$

furthermore,

$$\begin{aligned} \underline{\underline{A}}_{\Pi\Pi} &\equiv \begin{pmatrix} \left(\underline{\underline{A}}^t\right)_{\Pi} & \left(\underline{\underline{A}}^t\right)_{\mathrm{I}\pi} \underline{\underline{a}}^r \\ \underline{\underline{a}}^r \left(\underline{\underline{A}}^t\right)_{\pi\mathrm{I}} & \underline{\underline{a}}^r \left(\underline{\underline{A}}^t\right)_{\pi\pi} \underline{\underline{a}}^r \end{pmatrix}, \ \underline{\underline{A}}_{\Pi\Delta} \equiv \begin{pmatrix} \left(\underline{\underline{A}}^t\right)_{\mathrm{I}\Delta} \\ \underline{\underline{a}}^r \left(\underline{\underline{A}}^t\right)_{\pi\Delta} \end{pmatrix} \\ \underline{\underline{A}}_{\Delta\Pi} &\equiv \begin{pmatrix} \left(\underline{\underline{A}}^t\right)_{\Delta\mathrm{I}} & \left(\underline{\underline{A}}^t\right)_{\Delta\pi} \underline{\underline{a}}^r \end{pmatrix}, \ \underline{\underline{A}}_{\Delta\Delta} \equiv \left(\left(\underline{\underline{A}}^t\right)_{\Delta\Delta}\right) \end{aligned} \tag{4.11}$$

The matrix $\underline{\underline{A}} : W \to W$ will be referred to as the '*transformed-matrix*'. We observe that $\underline{\underline{A}} = \underline{\underline{A}}^t$ when $\pi = \varnothing$.

In turn, the *transformed problem* of (4.8) can be reduced, see [Herrera et al, 2010; Herrera et al, 2009; Herrera, 2008; Herrera, 2007; Farhat et al, 2000] for details, into the following problem, which is expressed in terms of the values of the solution at $dual-nodes$, exclusively: "Find $\underline{u}_{\Delta} \in W(\Delta)$ that satisfies

$$\underline{\underline{a}}\underline{\underline{S}}\underline{u}_{\Delta} = \underline{f}_{\Delta} \; and \; \underline{\underline{j}}\underline{u}_{\Delta} = 0 \tag{4.12}$$"

Here, $\underline{f}_{\Delta} \in \underline{\underline{a}}W(\Delta)$ and the *'Schur-complement matrix with constraints'* are defined by

$$\overline{\underline{f}}_{\Delta} \equiv \underline{f}_{\Delta} - \underline{\underline{A}}_{\Delta\Pi}\left(\underline{\underline{A}}_{\Pi\Pi}\right)^{-1}\underline{f}_{\Pi} \tag{4.13}$$

and

$$\underline{\underline{S}} \equiv \underline{\underline{A}}_{\Delta\Delta} - \underline{\underline{A}}_{\Delta\Pi}\left(\underline{\underline{A}}_{\Pi\Pi}\right)^{-1}\underline{\underline{A}}_{\Pi\Delta} \tag{4.14}$$

respectively.

5.- The DVS-Algorithms

Generally two kinds of approaches are distinguished: primal –these are direct approaches, which do not resort to Lagrange multipliers- and dual –indirect approaches that use Lagrange multipliers-. However, when DDMs are formulated using a setting as general as that supplied by the *DVS-framework*, such a distinction is irrelevant. The feature that is conspicuous for different options is the information that the algorithm seeks. Indeed, four algorithms will be obtained by seeking successively for the vectors: $\underline{u}_{\Delta}$, $\underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\underline{u}_{\Delta}$, $\underline{\underline{j}}\underline{\underline{S}}\underline{u}_{\Delta}$ and $\underline{\underline{S}}\underline{u}_{\Delta}$. However, in the presentation that follows we stick to the *'primal vs. dual-algorithms'* classification.

5.1 Primal Formulations

THE DVS VERSION OF BDDC

This is a primal algorithm which seeks directly for $\underline{u}_{\Delta}$. Pre-multiplying Eq.(4.12) by $\underline{\underline{a}}\underline{\underline{S}}^{-1}$, one gets:

$$\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\underline{S}}\underline{u}_{\Delta} = \underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{f}_{\Delta} \; and \; \underline{\underline{j}}\underline{u}_{\Delta} = 0 \tag{5.1}$$

In [Farhat et al, 2000], it was shown that Eq.(5.1) is equivalent to Eq.(4.12). This equation is the DVS-version of BDDC.

THE DVS-PRIMAL ALGORITHM

For this algorithm, the *sought-information* is:

$$\underline{v}_{\Delta} \equiv -\underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\underline{u}_{\Delta} \tag{5.2}$$

Applying $\underline{\underline{a}}\underline{\underline{S}}$ to Eq.(5.2) it is seen that $\underline{\underline{a}}\underline{\underline{S}}\underline{v}_{\Delta} = 0$. Furthermore,

$$\underline{\underline{j}}\left(\underline{\underline{S}}^{-1}\underline{f}_{\Delta} + \underline{v}_{\Delta}\right) = \underline{\underline{j}}\left(\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\underline{S}}_{\Delta} + \underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\right)\underline{u}_{\Delta} = \underline{\underline{j}}\underline{u}_{\Delta} = 0 \tag{5.3}$$

Therefore

$$\underline{\underline{j}}\underline{v}_{\Delta} = -\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_{\Delta} \; and \; \underline{\underline{a}}\underline{\underline{S}}\underline{v}_{\Delta} = 0 \tag{5.4}$$

Eq.(5.4) does not define an iterative algorithm. In order to obtain such an algorithm, we

project on $\underline{\underline{j}}\underline{\underline{S}}^{-1}W_\Delta$, to obtain:

$$\underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{v}_\Delta = -\underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_\Delta \text{ and } \underline{\underline{a}}\underline{\underline{S}}\underline{v}_\Delta = 0 \tag{5.5}$$

This algorithm is referred to as the *'DVS-primal algorithm'*. The solution is given by

$$\underline{u}_\Delta = \underline{\underline{a}}\left(\underline{v}_\Delta + \underline{\underline{S}}^{-1}\underline{f}_\Delta\right) \tag{5.6}$$

We observe that we could have written $\underline{u}_\Delta = \underline{v}_\Delta + \underline{\underline{S}}^{-1}\underline{f}_\Delta$ instead of Eq.(5.6). However, the application of the projection operator $\underline{\underline{a}}$ is important when $\underline{v}_\Delta$ and $\underline{\underline{S}}^{-1}\underline{f}_\Delta$ are not computed with exact arithmetic, as it is the case when using numerical methods, because when it is applied it replaces $\underline{v}_\Delta + \underline{\underline{S}}^{-1}\underline{f}_\Delta$ by the continuous-vector closest (with respect to the Euclidean distance) to it.

5.2 Dual Formulations

THE DVS VERSION OF FETI-DP

For this algorithm the *sought-information* is defined to be: $\underline{\lambda}_\Delta \equiv -\underline{\underline{j}}\underline{\underline{S}}\underline{u}_\Delta$. This algorithm can be easily derived from the *DVS-primal* formulation that has just been presented. We observe that $\underline{v}_\Delta = \underline{\underline{S}}^{-1}\underline{\lambda}_\Delta$, $\underline{\lambda}_\Delta = \underline{\underline{S}}\underline{v}_\Delta$, in view of Eq.(5.2), and $\underline{\underline{a}}\underline{\lambda}_\Delta = 0$. This permits transforming Eq.(5.5) into

$$\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{\lambda}_\Delta = \underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_\Delta \text{ and } \underline{\underline{a}}\underline{\lambda}_\Delta = 0 \tag{5.7}$$

Applying $\underline{\underline{S}}^{-1}$ to the first of these equations, it is obtained:

$$\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{\lambda}_\Delta = \underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_\Delta \text{ and } \underline{\underline{a}}\underline{\lambda}_\Delta = 0 \tag{5.8}$$

As for Eq.(5.6), it becomes:

$$\underline{u}_\Delta = \underline{\underline{a}}\underline{\underline{S}}^{-1}\left(\underline{f}_\Delta - \underline{\underline{j}}\underline{\lambda}_\Delta\right) \tag{5.9}$$

THE DVS-DUAL ALGORITHM

In this algorithm, the *sought-information* is: $\underline{\mu}_\Delta \equiv \underline{\underline{S}}\underline{u}_\Delta$. Then, $\underline{u}_\Delta = \underline{\underline{S}}^{-1}\underline{\mu}_\Delta$. Replacing this in Eq.(5.1), one gets:

$$\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\mu}_\Delta = \underline{\underline{S}}\,\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{f}_\Delta \text{ and } \underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{\mu}_\Delta = 0 \tag{5.10}$$

Finally, multiplying by $\underline{\underline{S}}$ the first of these equalities, it is obtained:

$$\underline{\underline{S}}\,\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\mu}_\Delta = \underline{\underline{S}}\,\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{f}_\Delta \text{ and } \underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{\mu}_\Delta = 0 \tag{5.11}$$

When $\underline{\mu}_\Delta$ is known, $\underline{u}_\Delta$ can be recovered by means of

$$\underline{u}_\Delta = \underline{\underline{a}}\underline{\underline{S}}^{-1}\left(\underline{f}_\Delta + \underline{\mu}_\Delta\right) \tag{5.12}$$

A comment similar to that made immediately after Eq.(5.6), goes here: we have applied the projection matrix $\underline{\underline{a}}$, in Eq.(5.12) because we are assuming that exact arithmetic generally will not be used.

6.- Numerical Procedures Fulfilling the DDM-Paradigm

Summarizing, the preconditioned *DVS-algorithms* with constraints are:

$$\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\underline{S}}\underline{u}_{\Delta} = \underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{f}_{\Delta} \;\; and \;\; \underline{\underline{j}}\underline{u}_{\Delta} = 0; \; DVS\text{ - }BDDC \tag{6.1}$$

$$\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{\lambda}_{\Delta} = \underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_{\Delta} \;\; and \;\; \underline{\underline{a}}\underline{\lambda}_{\Delta} = 0; \; DVS\text{ - }FETI\text{ - }DP$$
$$where \; \underline{u}_{\Delta} = \underline{\underline{a}}\underline{\underline{S}}^{-1}\left(\underline{f}_{\Delta} - \underline{\underline{j}}\underline{\lambda}_{\Delta}\right) \tag{6.2}$$

$$\underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{v}_{\Delta} = \underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_{\Delta} \;\; and \;\; \underline{\underline{a}}\underline{\underline{S}}\underline{v}_{\Delta} = 0; \;\; DVS\text{ - }PRIMAL$$
$$where \; \underline{u}_{\Delta} = \underline{\underline{a}}\underline{\underline{S}}^{-1}\left(\underline{f}_{\Delta} - \underline{\underline{j}}\underline{\underline{S}}\underline{v}_{\Delta}\right) \tag{6.3}$$

$$\underline{\underline{S}}\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\mu}_{\Delta} = \underline{\underline{S}}\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\underline{S}}\,\underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{f}_{\Delta} \;\; and \;\; \underline{\underline{j}}\underline{\underline{S}}^{-1}\underline{\mu}_{\Delta} = 0; \quad DVS\text{ - }DUAL$$
$$where \; \underline{u}_{\Delta} = \underline{\underline{a}}\underline{\underline{S}}^{-1}\left(\underline{f}_{\Delta} + \underline{\mu}_{\Delta}\right) \tag{6.4}$$

6.1- Comment on the DVS Numerical Procedures

The outstanding uniformity of the formulas given in Eqs.(6.1) to (6.4) yields clear advantages for code development, especially when such codes are built using object-oriented programming techniques. Such advantages include:

- **I.** The construction of very robust codes. This is an advantage of the *DVS-algorithms*, which stems from the fact the definitions of such algorithms exclusively depend on the discretized system of equations, obtained after discretization of the partial differential equations considered (referred to as the *original problem*), but which is otherwise independent of the problem that motivated it. In this manner, for example, essentially the same code was applied to treat 2-D and 3-D problems; indeed, only the part defining the geometry had to be changed, and that was a very small part of it;
- **II.** The codes may use different local solvers, which can be direct or iterative solvers;
- **III.** Minimal modifications are required for transforming sequential codes into parallel ones; and
- **IV.** Such formulas also permit developing codes which fulfill the *DDM-paradigm*; i.e., in which "the solution of the global problem is obtained by exclusively solving local problems".

This last property makes the DVS-algorithms very suitable as a tool to be used in the construction of massively-parallelized software, so much needed for efficiently programming the most powerful parallel computers available at present. In the next Subsection, procedures for constructing codes possessing Property IV are explained with some detail.

All the DVS-algorithms of Eqs.(6.1) to (6.4) are iterative and can be implemented with recourse to Conjugate Gradient Method (CGM), when the matrix is definite and symmetric, or some other iterative procedure such as GMRES, when that is not the case. At each

iteration step, depending on the *DVS-algorithm* that is applied, one has to compute the action on a *derived-vector* of one of the following matrices: $\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}\underline{\underline{S}}$, $\underline{\underline{j}}\underline{\underline{S}}\underline{\underline{j}}\underline{\underline{S}}^{-1}$, $\underline{\underline{S}}^{-1}\underline{\underline{j}}\underline{\underline{S}}\underline{\underline{j}}$ or $\underline{\underline{S}}\underline{\underline{a}}\underline{\underline{S}}^{-1}\underline{\underline{a}}$. Such matrices in turn are different permutations of the matrices $\underline{\underline{S}}$, $\underline{\underline{S}}^{-1}$, $\underline{\underline{a}}$ and $\underline{\underline{j}}$. Thus, to implement any of the preconditioned *DVS-algorithms*, one only needs to separately develop codes capable of computing the action of each one of the matrices $\underline{\underline{S}}$, $\underline{\underline{S}}^{-1}$, $\underline{\underline{a}}$ or $\underline{\underline{j}}$ on an arbitrary *derived-vector*, of W.

Therefore, next we present numerical procedures for computing the application of each one of the matrices $\underline{\underline{S}}$, $\underline{\underline{S}}^{-1}$, $\underline{\underline{a}}$ and $\underline{\underline{j}}$, which fulfill the *DDM-paradigm*. It will be seen that only $\underline{\underline{a}}$ requires exchange of information between derived-nodes belonging to different subdomains; actually, between *derived-nodes* that are descendants of the same *original-node* (the exchange of information is minimal). As for $\underline{\underline{j}} = \underline{\underline{I}} - \underline{\underline{a}}$, once the action of $\underline{\underline{a}}$ has been computed, no further exchange of information is required.

6.2- Application of $\underline{\underline{S}}$

From Eq.(4.13), we recall the definition of the matrix $\underline{\underline{S}} \equiv \underline{\underline{A}}_{\Delta\Delta} - \underline{\underline{A}}_{\Delta\Pi}\left(\underline{\underline{A}}_{\Pi\Pi}\right)^{-1}\underline{\underline{A}}_{\Pi\Delta}$. In order to evaluate the action of $\underline{\underline{S}}$ on any *derived-vector*, we need to successively evaluate the action of the following matrices $\underline{\underline{A}}_{\Pi\Delta}$, $\underline{\underline{A}}_{\Pi\Pi}^{-1}$, $\underline{\underline{A}}_{\Delta\Pi}$ and $\underline{\underline{A}}_{\Delta\Delta}$. Nothing special is required except for $\left(\underline{\underline{A}}_{\Pi\Pi}\right)^{-1}$. A procedure for evaluating the action of this matrix, which fulfills the *DDM-paradigm* is explained next.

We have

$$\underline{\underline{A}}_{\Pi\Pi} \equiv \begin{pmatrix} \underline{\underline{A}}_{\mathrm{II}} & \underline{\underline{A}}_{\mathrm{I}\pi} \\ \underline{\underline{A}}_{\pi\mathrm{I}} & \underline{\underline{A}}_{\pi\pi} \end{pmatrix} = \begin{pmatrix} \underline{\underline{A}}_{\mathrm{II}}^{t} & \underline{\underline{A}}_{\mathrm{I}\pi}^{t}\underline{\underline{a}}^{r} \\ \underline{\underline{a}}^{r}\underline{\underline{A}}_{\pi\mathrm{I}}^{t} & \underline{\underline{a}}^{r}\underline{\underline{A}}_{\pi\pi}^{t}\underline{\underline{a}}^{r} \end{pmatrix} \tag{6.5}$$

Let $\underline{v} \in W$, be an arbitrary *derived-vector*, and write

$$\underline{w} \equiv \left(\underline{\underline{A}}_{\Pi\Pi}\right)^{-1}\underline{v} \tag{6.6}$$

Then, $\underline{w} = \underline{w}_{\mathrm{I}} + \underline{w}_{\pi} \in W$ is characterized by

$$\begin{aligned} &\sigma_{\pi\pi}\left(\underline{\underline{A}}_{\Pi\Pi}\right)\underline{w}_{\pi} = \left\{\underline{v}_{\pi} - \underline{\underline{A}}_{\pi\mathrm{I}}\left(\underline{\underline{A}}_{\mathrm{II}}\right)^{-1}\underline{v}_{\mathrm{I}}\right\}, \textit{subjected to } \underline{\underline{j}}^{\pi}\underline{w}_{\pi} = 0 \\ &\underline{w}_{\mathrm{I}} = \left(\underline{\underline{A}}_{\mathrm{II}}\right)^{-1}\left\{\underline{v}_{\mathrm{I}} - \underline{\underline{A}}_{\mathrm{I}\pi}\underline{w}_{\pi}\right\} \end{aligned} \tag{6.7}$$

and can obtained iteratively. Here,

$$\sigma_{\pi\pi}\left(\underline{\underline{A}}_{\Pi\Pi}\right)\equiv\left\{\underline{\underline{A}}_{\pi\pi}-\underline{\underline{A}}_{\pi\mathrm{I}}\left(\underline{\underline{A}}_{\mathrm{II}}\right)^{-1}\underline{\underline{A}}_{\mathrm{I}\pi}\right\} \tag{6.8}$$

and, with $\underline{\underline{a}}^{\pi}$ as the projection-matrix into $W_r(\pi)$, $\underline{\underline{j}}^{\pi}\equiv\underline{\underline{\mathrm{I}}}-\underline{\underline{a}}^{\pi}$.

We observe that fulfilling the *DDM-paradigm* when computing the action of $\left(\underline{\underline{A}}_{\mathrm{II}}\right)^{-1}$ is straightforward because

$$\left(\underline{\underline{A}}_{\mathrm{II}}\right)^{-1}=\sum_{\alpha=1}^{E}\left(\underline{\underline{A}}_{\mathrm{II}}^{\alpha}\right)^{-1} \tag{6.9}$$

is parallelizable. Once $\underline{v}_{\pi}\in W_r(\pi)$ has been obtained, to derive $\underline{v}_{\mathrm{I}}$ one can apply:

$$\underline{v}_{\mathrm{I}}=\left(\underline{\underline{A}}_{\mathrm{II}}\right)^{-1}\left(\underline{w}_{\mathrm{I}}-\underline{\underline{A}}_{\mathrm{I}\pi}\underline{v}_{\pi}\right) \tag{6.10}$$

this completes the evaluation of $\underline{\underline{S}}$.

6.3- Application of $\underline{\underline{S}}^{-1}$

We define

$$\Sigma\equiv\mathrm{I}\cup\Delta \tag{6.11}$$

and observe that

$$\Sigma\cup\pi=\mathrm{X}\ \textit{and}\ \Sigma\cap\pi=\varnothing \tag{6.12}$$

Therefore, the matrix $\underline{\underline{A}}^{-1}$ can be written as:

$$\underline{\underline{A}}^{-1}=\begin{pmatrix}\left(\underline{\underline{A}}^{-1}\right)_{\Pi\Pi}&\left(\underline{\underline{A}}^{-1}\right)_{\Pi\Delta}\\ \left(\underline{\underline{A}}^{-1}\right)_{\Delta\Pi}&\left(\underline{\underline{A}}^{-1}\right)_{\Delta\Delta}\end{pmatrix}=\begin{pmatrix}\left(\underline{\underline{A}}^{-1}\right)_{\Sigma\Sigma}&\left(\underline{\underline{A}}^{-1}\right)_{\Sigma\pi}\\ \left(\underline{\underline{A}}^{-1}\right)_{\pi\Sigma}&\left(\underline{\underline{A}}^{-1}\right)_{\pi\pi}\end{pmatrix} \tag{6.13}$$

Furthermore, $\underline{\underline{S}}:W_{\Delta}\rightarrow W_{\Delta}$ fulfills

$$\underline{\underline{S}}^{-1}=\left(\underline{\underline{A}}^{-1}\right)_{\Delta\Delta} \tag{6.14}$$

Another property that is relevant for the following discussion is:

$$W_r(\Sigma)=W(\Sigma) \tag{6.15}$$

for any $\underline{v}\in W$, let us write

$$\underline{w}\equiv\underline{\underline{A}}^{-1}\underline{v} \tag{6.16}$$

then, $\underline{w}_{\pi}$ fulfills

$$\sigma_{\pi\pi}\left(\underline{\underline{A}}\right)\underline{v}_{\pi}=\underline{w}_{\pi}-\underline{\underline{A}}_{\pi\Sigma}\left(\underline{\underline{A}}_{\Sigma\Sigma}^{t}\right)^{-1}\underline{w}_{\Sigma},\ \textit{subjected to}\ \underline{\underline{j}}^{r}\underline{v}_{\pi}=0 \tag{6.17}$$

Here, $\underline{\underline{j}}^{r}\equiv\underline{\underline{\mathrm{I}}}-\underline{\underline{a}}^{r}$, where the matrix $\underline{\underline{a}}^{r}$ is the projection operator on W_r, while

$$\sigma_{\pi\pi}\left(\underline{\underline{A}}\right)\equiv\underline{\underline{A}}_{\pi\pi}-\underline{\underline{A}}_{\pi\Sigma}\left(\underline{\underline{A}}_{\Sigma\Sigma}^{t}\right)^{-1}\underline{\underline{A}}_{\Sigma\pi} \tag{6.18}$$

Furthermore, we observe that

$$\left(\underline{\underline{A}}_{\Sigma\Sigma}^{t}\right)^{-1}=\sum_{\alpha=1}^{E}\left(\underline{\underline{A}}_{\Sigma\Sigma}^{\alpha}\right)^{-1} \tag{6.19}$$

In order to use Eq.(6.19) as a means of parallelizing the DVS-algorithms, however, the detailed discussion of such procedures will be presented separately [Herrera et al, 2013; L.M. de la Cruz et al, 2013]. It is necessary that the local matrices, $\underline{\underline{A}}^{\alpha}_{\Sigma\Sigma}$, be invertible. This is granted when $\underline{\underline{A}}$ invertible in W_r, which generally is achieved by taking a sufficiently large number of *primal-nodes*.

Eq.(6.17) is solved iteratively. Once $\underline{v}_{\pi}$ has been obtained, we apply:

$$\underline{v}_{\Sigma} = \left(\underline{\underline{A}}^{t}_{\Sigma\Sigma}\right)^{-1}\left(\underline{w}_{\Sigma} - \underline{\underline{A}}_{\Sigma\pi}\underline{v}_{\pi}\right) \tag{6.20}$$

This procedure permits obtaining $\underline{\underline{A}}^{-1}\underline{w}$ in full; however, we only need $\left(\underline{\underline{A}}^{-1}\right)_{\Delta\Delta}\underline{w}$. We observe that

$$\left(\underline{\underline{A}}^{-1}\right)_{\Delta\Delta}\underline{w} = \left(\underline{\underline{A}}^{-1}\underline{w}_{\Delta}\right)_{\Delta} \tag{6.21}$$

The vector $\underline{\underline{A}}^{-1}\underline{w}_{\Delta}$ can be obtained by the general procedure presented above. Thus, take $\underline{w} \equiv \underline{w}_{\Delta} \in W_{\Delta} \subset W$ and

$$\underline{v} \equiv \underline{\underline{A}}^{-1}\underline{w}_{\Delta} \tag{6.22}$$

Therefore,

$$\underline{v}_{\mathrm{I}} + \underline{v}_{\Delta} = \underline{v}_{\Sigma} = -\left(\underline{\underline{A}}^{t}_{\Sigma\Sigma}\right)^{-1}\underline{\underline{A}}_{\Sigma\pi}\underline{v}_{\pi} = -\left(\underline{\underline{A}}^{t}_{\Sigma\Sigma}\right)^{-1}\underline{\underline{A}}^{t}_{\Sigma\pi}\underline{\underline{a}}^{r}\underline{v}_{\pi}. \tag{6.23}$$

6.4- Application of $\underline{\underline{a}}$ and $\underline{\underline{j}}$.

We use the notation

$$\underline{\underline{a}} = \left(a_{(i,\alpha)(j,\beta)}\right) \tag{6.24}$$

then [Herrera et al, 2010]:

$$a_{(i,\alpha)(j,\beta)} = \frac{1}{m(i)}\delta_{ij}, \forall \alpha \in \mathrm{Z}(i) \; and \; \forall \beta \in \mathrm{Z}(j) \tag{6.25}$$

while $\underline{\underline{j}} = \underline{\underline{\mathrm{I}}} - \underline{\underline{a}}$ therefore,

$$\underline{\underline{j}}\underline{w} = \underline{w} - \underline{\underline{a}}\underline{w}, \; for \; every \; \underline{w} \in W \tag{6.26}$$

Therefore, only the evaluation of $\underline{\underline{a}}\underline{u}$ requires exchange of information between subdomains. In general, such numbers are very small; for example in application to single-equation problem, when an orthogonal grid is used, they are at most: 4, for problems in 2D, and 8 for problems in 3D.

As for the right hand-sides of Eqs. (4.14), all they can be obtained by successively applying to $\underline{f}_{\Delta}$ some of the operators that have already been discussed. Recalling Eq. (4.14), we have

$$\underline{f}_{\Delta} \equiv \left(R\underline{\hat{f}}\right)_{\Delta} - \underline{\underline{A}}_{\Delta\Pi}\underline{\underline{A}}^{-1}_{\Pi\Pi}\left(R\underline{\hat{f}}\right)_{\Pi} \tag{6.27}$$

The computation of $R\underline{\widehat{f}}$ does not present any difficulty and the evaluation of the actions of $\left(\underline{\underline{A}}_{\Pi\Pi}\right)^{-1}$ and $\underline{\underline{A}}_{\Delta\Pi}$ were already analyzed.

7. – Numerical Results

Taking into account the general description of the DVS-framework given of Section 2, it can be seen that each one of the DVS-algorithms is uniquely defined by:

1. The original-matrix;
2. The partition of the set of original-nodes, which is induced by the *coarse-mesh* that is applied; and
3. The set of constraints.

In turn, the original-matrix is determined by the partial differential equation, or system of such equations, the discretization method chosen and the *fine-mesh* adopted. As explained in Section 2, the partition of the set of original-nodes depends when the *fine-mesh* has already been defined, on the *coarse-mesh* (i.e., the domain decomposition) used. The *coarse-mesh* is constituted by a family of non-overlapping subdomains $\{\Omega_1,...,\Omega_E\}$ of Ω, the domain of definition of the boundary-value problem to be solved. In all the examples that are presented in this article, the *constraints* are fully determined by the *primal-nodes* and consist in requiring continuity of the *derived-vectors* at them.

Several codes were developed to treat the examples, which were written in C++ language, using the MPI library for the communications. In the computational implementations, the methods of solution used to treat the *original-problems* are: CGM, when such a linear system is symmetric and positive-definite and GMRES when the discrete system is non-symmetric or indefinite. Both are applied with a tolerance of 10^{-6}. Each *DVS-algorithm* was applied to each one of the examples considered, except for that referring to elasticity.

The results obtained for Examples 1 to 5 are summarized in Tables 1 to 5, respectively. In them, the acronym *dof* stands for to the number of degrees of freedom of the *original problem*, but it should be mentioned that the procedures used to treat such examples are such that the nodes that lie on the external boundary do not contribute to the *dof* . The notation to indicate the meshes that were adopted is as follows: In 2D cases, we use *(n×m)×(q×r)*, where *(n×m)* refers to the *coarse-mesh*, while *(q×r)* to the *fine-mesh*; and similarly, in 3D cases, we use *(n×m×p)* × *(q×r×s)*, where *(n×m×p)* define the *coarse-mesh* and *(q×r×s)* the *fine-mesh*. The constrains are imposed on the primal nodes, in all of our experiments the primal nodes were located at vertex in 2D and at edges in 3D of the subdomains, this coinciding with the algorithm "D" in [Toselli et al, 2005].

Each Table contains at most ten columns. The first four indicate respectively: 1) the meshes used, 2) the number of subdomains of the *coarse-mesh*, 3) the *dof*, and 4) the number of *primal-nodes* used. The figures appearing in columns 5 to 9 correspond to the number of iterations that were required for convergence of each one of the algorithms applied. Columns 9 and 10 were only included in Table 3. For Example 3, in order to cover a wide range of values of the Peclet-number, the diffusion coefficient in Eq.(7.3), ν, was varied and the tenth column in Table 3 indicates the different values of ν for which the

corresponding boundary-value problem was solved. Furthermore, the results obtained when the DVS-algorithms were applied were compared with those obtained in [Da Conceição et al, 2006] for the same problem, using the standard version of BDDC.

7.1 **Application of the DVS-algorithms to a Single-Equation**

The applicability of the *DVS-algorithms* is wide, as previously said it can be applied to general equation systems. In Section 3, it was announced that in this paper we present examples for which d, the number of equations of the system, is one and three. In this Subsection the examples for which $d=1$ will be discussed, leaving for the next Subsection the treatment of static-elasticity models, for which $d=3$.

Four boundary value problems corresponding to a single-equation will be presented. The first two are symmetric and positive definite boundary-value problems, whose definition involves the Laplace differential operator. The other two correspond to advection-diffusion transport, and the corresponding boundary-value problems are non-symmetric and indefinite. The discretization methods used in this Subsection are based on central finite differences (CFD), which are directly applicable to the symmetric problems. To apply CFD to the advection-diffusion problems it was necessary to stabilize the advection-diffusion differential-operator and to this end artificial diffusion was incorporated.

Despite the simplicity of the examples presented in this Subsection, they are very important because a wide range of geophysical systems give rise to similar problems [Herrera and Pinder, 2012]. The diversity of physical interpretations of the boundary-value problems here discussed is enormous. All the differential operators involved can be classified as advection-diffusion operators, since Laplace operator is obtained from the general advection-diffusion differential-operator when the transport-velocity vanishes. Transport processes of heat and solutes occur in a great diversity of geophysical systems. However, the physical processes governed by such differential-equations go far beyond transport phenomena.

Example 1. Poisson equation in two-dimensions.

$$-\Delta u = 2\pi^2 n^2 \sin(\pi n x)\sin(\pi n y),\ (x,y)\in[-1,1]\times[-1,1],\ n=4$$
$$u = 0 \ on\ \partial\Omega \tag{7.1}$$

PARTITION	SUBDOMAINS	DOF	PRIMALS	DVS-BDDC	DVS-FETI-DP	DVS PRIMAL	DVS DUAL
(2x2) X (2x2)	4	9	1	1	1	1	1
(4x4) X(4x4)	16	225	9	1	5	5	4
(6x6) X (6x6)	36	1225	25	8	8	8	7
(8x8) X (8x8)	64	3969	49	10	10	10	9
(10x10) X (10x10)	100	9801	81	11	11	12	10
(12x12) X (12x12)	144	20449	121	12	11	12	11
(14x14) X (14x14)	196	38025	169	12	12	12	11
(16x16) X (16x16)	256	65025	225	13	11	13	11
(18x18) X (18x18)	324	104329	289	13	11	13	11
(20x20) X (20x20)	400	159201	361	13	11	13	11
(22x22) X (22x22)	484	233289	441	13	12	14	11
(24x24) X (24x24)	576	330625	529	13	12	13	11
(26x26) X (26x26)	676	455625	625	13	12	14	11
(28x28) X (28x28)	784	613089	729	13	12	14	11
(30x30) X (30x30)	900	808201	841	13	12	14	11

Table 1. Number of iterations made by the four DVS algorithms. The primal nodes were located at the vertices of subdomains.

We can see from Table 1, that the four algorithms perform very well as the number of subdomains and the degrees of freedom (dof) are increased. In this example, the DVS-DUAL algorithm presents the best performance, requiring only 11 iterations from 12×12 until 30×30 subdomains, and the same number of dof. All other algorithms show similar behavior. The numerical solution of this example can be seen in the Figure 5.

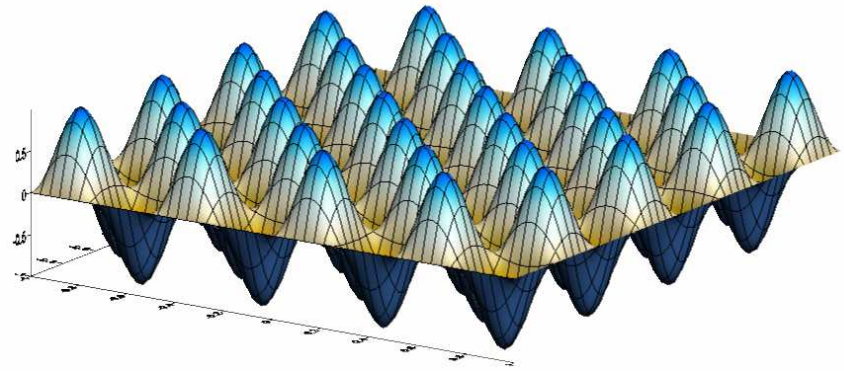

Figure 5 The numerical solution for the 2D case, here we use n=4.

Example 2. Similar to Example 1, but it is formulated in a 3D domain.

$$-\Delta u = 3\pi^2 n^2 \sin(\pi n x)\sin(\pi n y)\sin(\pi n z),\ (x,y,z)\in[-1,1]\times[-1,1]\times[-1,1],\ n=4$$
$$u = 0 \ on\ \partial\Omega \tag{7.2}$$

PARTITION	SUBDOMAINS	DOF	PRIMALS	DVS-BDDC	DVS-FETI-DP	DVS-PRIMAL	DVS-DUAL
(2x2x2) X (2x2x2)	8	27	7	1	1	1	1
(3x3x3) X (3x3x3)	27	512	80	4	4	4	3
(4x4x4) X (4x4x4)	64	3375	351	5	4	4	3
(5x5x5) X (5x5x5)	125	13824	1024	6	5	6	5
(6x6x6) X (6x6x6)	216	42875	2375	7	6	7	5
(7x7x7) X (7x7x7)	343	110592	4752	7	6	7	5
(8x8x8) X (8x8x8)	512	250047	8575	8	6	8	5
(9x9x9) X (9x9x9)	729	512000	14336	8	6	8	6
(10x10x10) X (10x10x10)	1000	970299	22599	9	6	9	6

Table 2. Number of iterations made by the four DVS algorithms. The primal nodes were located at edge.

In Table 2, we observe a similar performance of the algorithms as in the two-dimensional case. One more time the DVS-DUAL algorithm presents a little better behavior with respect all others.

Example 3. The boundary-value problem treated is:

$$-\nu\Delta u + \underline{b}\bullet\nabla u = 0;\quad (x,y)\in[0,1]\times[0,1],\ \underline{b}\equiv(1,3)$$
$$u(x,y) = \begin{cases} 0, & (x,y)\in\psi_1 \\ 1, & (x,y)\in\psi_2 \end{cases} \tag{7.3}$$

This is an advection-diffusion transport problem in 2D, for which the differential operator is not self-adjoint.

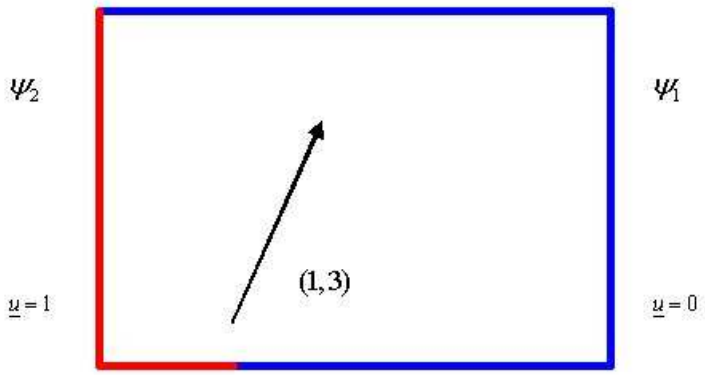

Figure 6, $\partial\Omega$

This example is very interesting because it contains diffusion and advection terms, which are common in several complex geophysics phenomena. In this example, the Péclet number is defined as $\mathrm{P}e = \|\underline{b}\| L / \nu$, where L is a characteristic length (in this case L = 1). We also define a local Péclet number as $\mathrm{P}e_h = \|\underline{b}\| h / \nu$. Using these definitions, fixing the global partition to h=1/512, and the varying the viscosity from 0.01 to 0.0001, we have that the Péclet number varies from 316 to 316,227, and the local Péclet number varies from 0.617 to 617. In this case the linear system is non-symmetric, therefore we choose the GMRES method with a tolerance of 10^{-6}.

PARTITION	SUB-DOMAINS	DOF	PRIMALS	DVS-BDDC	DVS-FETI-DP	DVS-PRIMAL	DVS-DUAL	BDDC	ν
(8x8) X (64x64)	64	261121	49	12	11	11	11	12	0.01
(8x8) X (64x64)	64	261121	49	8	8	8	7	9	0.001
(8x8) X (64x64)	64	261121	49	7	7	7	7	9	0.0001
(8x8) X (64x64)	64	261121	49	7	7	7	7	9	0.00001
(16x16) X (32x32)	256	261121	255	19	17	17	18	20	0.01
(16x16) X (32x32)	256	261121	255	14	14	13	13	17	0.001
(16x16) X (32x32)	256	261121	255	13	13	13	13	15	0.0001
(16x16) X (32x32)	256	261121	255	13	13	13	13	16	0.00001
(32x32) X (16x16)	1024	261121	961	33	29	29	31	33	0.01
(32x32) X (16x16)	1024	261121	961	26	25	25	25	30	0.001
(32x32) X (16x16)	1024	261121	961	25	25	25	25	28	0.0001
(32x32) X (16x16)	1024	261121	961	25	25	25	26	29	0.00001
(64x64) X (8x8)	4096	261121	3969	53	52	53	59	52	0.01
(64x64) X (8x8)	4096	261121	3969	46	46	46	47	53	0.001
(64x64) X (8x8)	4096	261121	3969	45	47	45	47	53	0.0001
(64x64) X (8x8)	4096	261121	3969	45	47	45	48	54	0.00001

Table 3. Comparison of the DVS-algorithms against the BDDC implemented in [Mandel et al, 1996].

In Table 3 presents the results that the *DVS-algorithms* yielded and compares them with those obtained in [Da Conceição et al, 2006]. We observe that, with fixed *coarse* and *fine meshes*, as the viscosity coefficient is reduced, so that the Péclet number increases, generally the iterations required for convergence reduce. Increasing the Péclet number implies that the effect of the advection term enlarges, and the numerical solution generally becomes unstable. However, the performance of the discretization strategy based on CFD combined with stabilization of the numerical-scheme by means of artificial viscosity is resilient to Péclet-number variations. For comparison purposes, the examples presented here were chosen to be the same as those presented in [Da Conceição et al, 2006], where the standard BDDC algorithm was applied with the same set of constraints; namely, the same set of subdomains and vertex nodes were chosen to be *primal*. As can be seen in Table 3, when the comparison criterion is based on the number of iterations required for convergence, the observed performance of the *DVS-algorithms* in these examples is slightly better than that of the standard BDDC algorithm. Finally, an illustration of the kind of numerical solution obtained is shown in Figure 7.

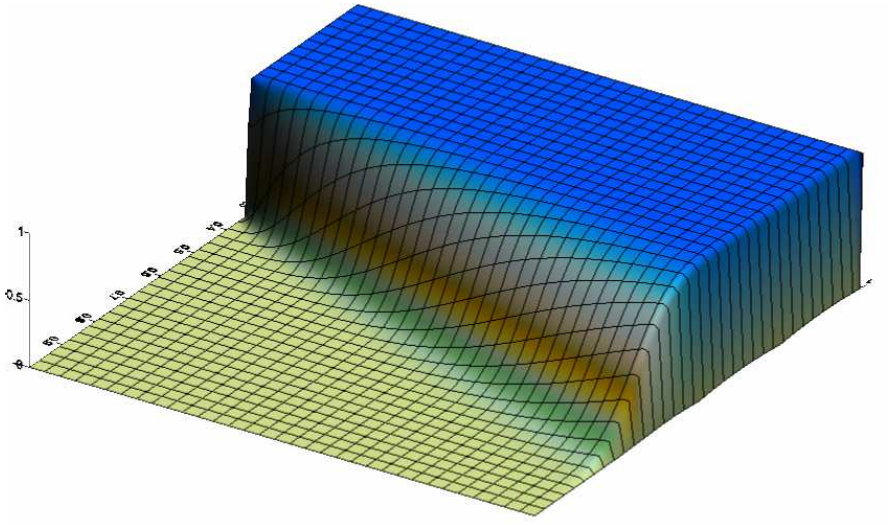

Figure 7 The numerical solution for $\nu = 0.01$.

The relative-residual decay for a coarse mesh (16×16) and several fine meshes is presented in Figure 8. We consider in these computations b=(1,3) and $\nu = 0.00001$, in such a way that $Pe = 3.16e+5$. We observe that the best convergence is obtained when the fine mesh is increased, and the convergence slows when the dof occurring in the subdomains is reduced.

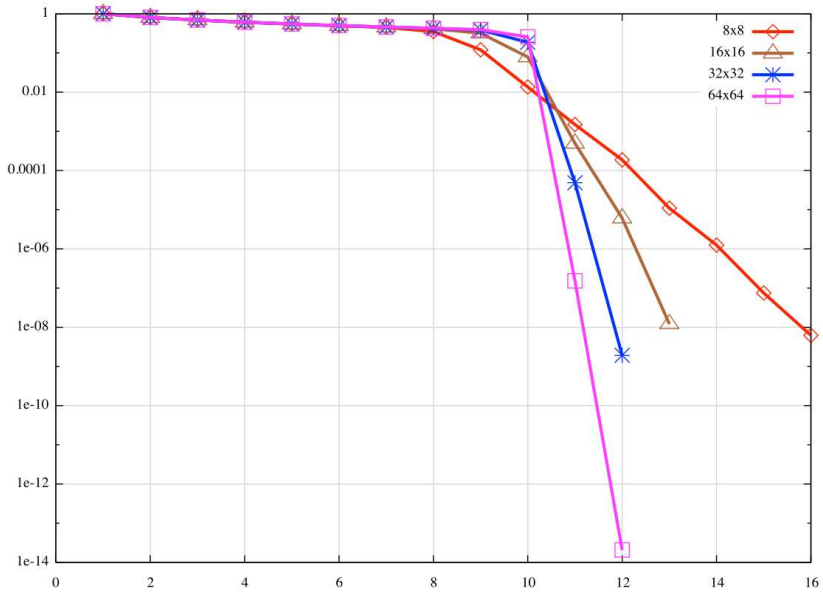

Figure 8. Relative residual decay for the local mesh (16×16).

Example 4. The boundary-value problem treated is:

$$-\Delta u + \underline{b} \bullet \nabla u = 0; \quad (x, y, z) \in [0,1] \times [0,1] \times [0,1], \ \underline{b} = (1,1,1)$$
$$u(x, y, z) = \exp(x + y + z) \text{ on } \partial\Omega \tag{7.4}$$

This is an advection-diffusion transport problem in 3D, for which the differential operator is not self-adjoint.

PARTITION	SUBDOMAINS	DOF	PRIMALS	DVS-BDDC	DVS-FETI-DP	DVS-PRIMAL	DVS-DUAL
(2x2x2) X (2x2x2)	8	27	7	4	3	3	4
(3x3x3) X (3x3x3)	27	512	80	7	5	6	5
(4x4x4) X (4x4x4)	64	3375	351	9	6	7	6
(5x5x5) X (5x5x5)	125	13824	1024	10	7	8	7
(6x6x6) X (6x6x6)	216	42875	2375	11	7	9	8
(7x7x7) X (7x7x7)	343	110592	4752	12	8	10	8
(8x8x8) X (8x8x8)	512	250047	8575	|13	8	11	8
(9x9x9) X (9x9x9)	729	512000	14336	14	8	11	9
(10x10x10) X (10x10x10)	1000	970299	22599	15	9	12	9

Table 4. Number of iterations made by the four DVS algorithms. The primal nodes were located at edges of the subdomains.

The diffusion and advection-diffusion differential-operator appears in the equations of the examples presented above. They are very important in natural and industrial phenomena. For example, the flow and transport of solutes in subsurface groundwater, the movement of aerosol and trace gases in the atmosphere, mixing of fluids in processes of crystal growth, among many other important applications [Tood, 1980; Pinder et al, 2006; Herrera et al, 1969; Herrera et al, 1973; Herrera et al, 1977; Herrera G.S. et al, 2005; L.M. de la Cruz et al, 2006]. In all our examples, we have shown that the *DVS-algorithms* obtain the numerical solution efficiently on parallel machines. In this respect, we remark that for advection-diffusion problems the matrices of the discrete linear systems are non-symmetric.

7.2- Application to a System-Equations

We use the *DVS-framework* to solve a Dirichlet boundary value problem, where displacements are zero over the boundary of the elastic body that occupies the domain Ω of the physical space. Over each one of such subdomains is solved a local problem by FEM, using linear functions as basis. On each node α of the mesh is defined a vector valued function $\underline{u}_\alpha$ with each component identified as $u_{\alpha i}$ for $i = 1,2,3$.

Because our operators are symmetric and positive definite, we use CGM as an iterative procedure to solve those linear systems of equations that we have defined in the DVS framework.

The code used in the previous section, which was originally developed to solve a single equation using finite differences, was adapted for solving systems of equations with FEM. We added the corresponding functionality in order to be able to solve systems of equations, in this case the elasticity problem.

Example 5. A system of partial differential equations in three-dimensions has also been treated. This is the system of differential equations of static elasticity; namely:

$$(\lambda+\mu)\nabla\nabla\bullet\underline{u}+\mu\Delta\underline{u}=\underline{f}_{\Omega},\ \text{in}\ \Omega \tag{7.5}$$

which was subject to the following Dirichlet boundary conditions:

$$\underline{u}=0,\ \text{on}\ \partial\Omega \tag{7.6}$$

The domain of study for our numerical experiments is a homogeneous isotropic linearly elastic unitary cube. In all of our experiments the primal nodes were located at edges of the subdomains, which is enough for $\underline{\underline{A}}^{t}$ not being singular.

We consider constant coefficients λ and μ equal to one. With these conditions we have a problem that has analytical solution, and is written as follows:

$$\underline{u}=\left(\sin\pi x\sin\pi y\sin\pi z,\sin\pi x\sin\pi y\sin\pi z,\sin\pi x\sin\pi y\sin\pi z\right) \tag{7.7}$$

The Tables 5, summarizes the numerical results obtained using the DVS methods with a tolerance of 10^{-7}.

PARTITION	SUBDOMAINS	DOF	PRIMALS	DVS-BDDC	DVS-FETIDP	DVS-PRIMAL	DVS-DUAL
(5x5x5) X (5x5x5)	125	41,472	1,024	8	7	9	9
(6x6x6) X (6x6x6)	216	128,625	2,375	8	8	10	10
(7x7x7) X (7x7x7)	343	331,776	4,752	8	8	11	11
(8x8x8) X (8x8x8)	512	750,141	8,575	8	8	12	12

Table 5. Results for DVS Algorithms

8. Conclusions

Mathematical models of many geophysical systems lead to a great variety of partial differential equations (PDEs) whose solution methods are based on the computational processing of large-scale algebraic systems [Herrera and Pinder, 2012]. Parallel computing is outstanding among the new computational tools and, in order to effectively use the most advanced computers available today, massively parallel software is required. Domain decomposition methods (DDMs) have been developed precisely for effectively treating PDEs in parallel [DDM Organization, 2012]. What domain decomposition methods ideally intend to do has been summarized in this paper in the *"DDM-paradigm"*: to develop algorithms that *'obtain the global solution by exclusively solving local problems'*.

In conclusion, in this paper:

1. We have presented a *non-overlapping discretization* method (the *DVS-discretization*) -in the sense that it uses a system of nodes such that each one of them belongs to one and only one subdomain of the *coarse-mesh*- applicable to a wide class of well-posed boundary problems associated with elliptic systems of equations. In particular, the differential operators may be symmetric, non-symmetric or indefinite (non-positive-definite);

2. Four algorithms –the *DVS-algorithms* [Herrera et al, 2011]-, which were derived using the *DVS-discretization* and achieve the *DDM-paradigm* have been explained. Two of them

are the result of using the BDDC and FETI-DP algorithms after applying *DVS-discretization* to the boundary value problem considered. The other two are obtained when two new algorithms, which had not been reported previously in the literature, were used instead;

3. Numerical procedures that permit achieving the *DDM-paradigm* with each one of the *DVS-algorithms* have been also presented;

4. Codes were developed and applied to several boundary values problems that occur in the modeling of certain geophysical phenomena, such as transport of solutes by both, free-fluids and fluids in a porous medium. We also present results for a static elasticity problem, which thereby illustrates the application of the algorithms to systems of differential equations; and

5. Besides their attractive parallelization properties, in the numerical examples the *DVS-algorithms* exhibited significantly improved numerical performance with respect to standard versions of BDDC and FETI-DP.

Acknowledgement. The authors express their gratitude to Alberto Rosas-Medina e Iván Contreras-Trejo, both PhD students of the Earth-Sciences Graduate Program at UNAM, for having permitted us to reproduce some numerical results of their research work.

References

PITAC, 2005, Computational Science: Ensuring america's competiveness, Report to the President of the United States, President Information, Technology Advisory Committee, Executive Office of the President of the United States, June.

DDM Organization, 2012, Proceedings of 21 International Conferences on Domain Decomposition Methods. www.ddm.org.

Mandel J., 1993, Balancing domain decomposition, Commun. Numer. Methods Engrg. 233-241.

Mandel J. and Brezina M., 1996, Balancing domain decomposition for problems with large jumps in coefficients, Math. Comput. 65, pp 1387-1401.

Farhat Ch. and Roux F., 1991, A method of finite element tearing and interconnecting and its parallel solution algorithm, Internat. J. Numer. Methods Engrg. 32:1205-1227.

Mandel J. and Tezaur R., 1996, Convergence of a substructuring method with Lagrange multipliers, Numer. Math 73(4): 473-487.

Farhat C., Lessoinne M. LeTallec P., Pierson K. and Rixen D., 2001, FETI-DP a dual-primal unified FETI method, Part I: A faster alternative to the two-level FETI method, Int. J. Numer. Methods Engrg. 50, pp 1523-1544.

Farhat C., Lessoinne M. and Pierson K., 2000, A scalable dual-primal domain decomposition method, Numer. Linear Algebra Appl. 7, pp 687-714.

Mandel J. and Tezaur R., 2001, On the convergence of a dual-primal substructuring method, SIAM J. Sci. Comput., 25, pp 246-258, 2001.

Dohrmann C.R., 2003, A preconditioner for substructuring based on constrained energy minimization, SIAM J. Sci. Comput. 25(1):246-258.

Mandel J. and C. R. Dohrmann, 2003, Convergence of a balancing domain decomposition by constraints and energy minimization, Numer. Linear Algebra Appl., 10(7):639-659, 2003.

Mandel J., Dohrmann C.R. and Tezaur R., 2005, An algebraic theory for primal and dual substructuring methods by constraints, Appl. Numer. Math., 54: 167-193.

J. Li and O. Widlund, 2005, FETI-DP, BDDC and block Cholesky methods, Int. J. Numer. Methods Engrg. 66, 250-271.

Toselli A. and O. Widlund, , 2005, Domain decomposition methods- algorithms and Theory, Springer Series in Computational Mathematics, Springer-Verlag, Berlin, 2005, 450p.

Herrera I. and Alberto A. Rosas-Medina, 2013, The Derived-Vector Space Framework and Four General Purposes Massively Parallel DDM Algorithms, Engineering Analysis with Boundary Elements, in press.

L.M. de la Cruz and Herrera I., 2013, Generic and Parallel Software based on DVS algorithms for engineering, Submitted at Advances in Engineering Software.

Herrera I., L.M. de la Cruz, Carrillo-Ledesma A., Rosas-Medina A. and Contreras, I., 2012, Foundations of the DVS-framework: Theory and algorithms, Memoria No.8 del Grupo de Modelación Matemática y Computacional del Instituto de Geofísica, UNAM.

Herrera, I., Carrillo-Ledesma A. & Rosas-Medina A., 2012, “Four general purposes massively parallel DDM algorithms”, Available as Memoria No.7 del Grupo de Modelación Matemática y Computacional del Instituto de Geofísica, UNAM.

Herrera, I., Carrillo-Ledesma A. & Rosas-Medina Alberto, 2011, A brief overview of non-overlapping domain decomposition methods”, Geofísica Internacional, Vol. 50(4), pp 445-463.

Herrera, I. & Yates R. A., 2009, The multipliers-free dual primal domain decomposition methods for non-symmetric Matrices, NUMER. METH. PART D. E. 2009, DOI 10.1002/Num. 20581.

Herrera, I. & Yates R. A., 2010, The multipliers-free domain decomposition methods, NUMER. METH. PART D. E. 26: 874-905 July 2010, DOI 10.1002/num. 20462.

Herrera I. and R. Yates, 2009, Unified multipliers-free theory of dual-primal domain decomposition methods, NUMER. METH. PART D. E. Eq. 25:552-581, May 2009, (Published on line May 13, 08) DOI 10.1002/num. 20359.

Herrera, I., 2008, New formulation of iterative substructuring methods without Lagrange Multipliers: Neumann-Neumann and FETI, NUMER METH PART D E 24(3) pp 845-878, DOI 10.1002 NO. 20293.

Herrera, I., 2007, Theory of differential equations in discontinuous piecewise-defined-functions, NUMER METH PART D E, **23**(3), pp 597-639, DOI 10.1002 NO. 20182.

Da Conceição, D. T. Jr., 2006, Balancing domain decomposition preconditioners for non-symmetric problems, Instituto Nacional de Matemática Pura e Aplicada, Agência Nacional do Petróleo PRH-32, Rio de Janeiro, May. 9.

Herrera I. and G.F. Pinder, 2012, Mathematical modeling in science and engineering: An axiomatic approach, Wiley, 243p.

Todd, D.K., 1980, Groundwater hydrology, 2nd ed. Wiley.

Pinder, G.F. and M.A. Celia, 2006, Subsurface hydrology, Wiley, 468p.

Herrera I. and G.E. Figueroa V., 1969, A correspondence principle for the theory of leaky aquifers, Water Resources Research, Vol. 5, NO. 4, P. 900.

Herrera, I. and Rodarte, L., 1973, Integrodifferential equations for systems of leaky aquifers and applications, Part 1: The nature of approximate Theories, Water Resources Research, **9**(4), pp. 995-1005.

Herrera, I. and Yates, R., 1977, Integrodifferential equations for systems of leaky aquifers. Part 3. A numerical method of unlimited applicability, Water Resources Research, **13**(4), pp. 725-732.

Herrera G.S. and G.F. Pinder, 2005, Space-time optimization of groundwater quality sampling networks, WATER RESOURCES RESEARCH, VOL. 41, W12407, 15 PP.

L.M. de la Cruz and E. Ramos, 2006, Mixing with time dependent natural convection, Int. Comm. in Heat and Mass Transfer, Vol. 33/2, pp 191-198.