PARAMETER ESTIMATION IN DENSITY DEPENDENT GROUNDWATER FLOW AND SOLUTE TRANSPORT MODELLING

Luit J. SLOOTEN, **Juan J. HIDALGO**, and **Jesús CARRERA**

Dept. of Geotechnical Engineering & Geosciences.
School of Civil Engineering.
Technical University of Catalonia,
Campus Nord, Edif. D-2, Barcelona E-08034 SPAIN.

ABSTRACT

A Newton-Raphson scheme to solve the coupled seawater flow and transport equations is proposed, which has proved to be robust and suitable for optimizing coupled non-linear problems (Galarza *et al.* 1999). Also, two alternative ways of solving the inverse problem are discussed, both of which we believe to be faster and more reliable than the parameter perturbation techniques. Furthermore, we show that the matrices required for solving the inverse problem are equal to the Jacobian matrix appearing in the Newton-Raphson formulation of the forward problem.

INTRODUCTION

Motivation

Saltwater intrusion has received much attention in recent years, both because it threatens coastal aquifers and because advances in computational speed have made it feasible to solve the coupled equations of flow and transport. This coupling arises from the fact that concentration differences in the domain may be large enough to produce nontrivial density differences that affect groundwater flow.

Many numerical approaches are available for solving sets of non-linear equations. Most are variations of the Picard scheme, which solves the equations sequentially and then iterates to convergence, and the Newton-Raphson scheme, which solves the flow and transport equations simultaneously but which require knowledge of the derivatives of the governing equations with respect to the state variables.

As knowledge of the hydrological parameters is a prerequisite for solving the flow and transport equations, the inverse problem naturally appears at this stage. From an inverse problem point of view, saltwater intrusion could be viewed as an aquifer-scale tracer test. Especially, the movement of the saltwater-freshwater interface gives a huge amount of valuable information about the parameter fields. However, solving the inverse problem using the classical parameter perturbation approach is extremely expensive. Therefore, alternative methods should be sought.

Background

The general theory of density dependent flow is described by Bear (1972). On this basis a lot of work has been done and different expressions for balance equations for mass of fluid and mass of solute have been proposed. Voss (1984) proposes a mass balance in terms of pressure while Huyakorn *et al.* (1987) derive the equations in terms of equivalent freshwater heads. The solute transport equations can be expressed in terms of concentration (Bear, 1979, Kinzelbach, 1986) or mass fraction (Olivella *et al.*, 1994). A more extensive review of equations can be found in Kolditz *et al.* (1998), or Holzbecher (1998).

Parameter identification in groundwater is based on finding the values of model parameters that minimize an objective function that measures the differences between the calculated solutions and the observed values. In order to minimize the objective function it is necessary to evaluate the derivatives of this function with respect to parameters, i.e. to evaluate the Jacobian matrix. Mainly two methods to calculate the Jacobian have been proposed: sensitivity equations and adjoint state.

Experience in inverting seawater intrusion data is limited. Iribar *et al.*, 1996, made a first approximation of calibrating aquifer parameters with saltwater intrusion considering negligible the effects of concentration in density. Acceptable results were obtained. Parameter identification in density-dependent flow has been treated by Piggott *et. al* (1993) who implemented an inversion algorithm for SUTRA (Voss, 1984) called SUTRA^{-1} (or SUTRAINV). It is based on generalized least squares. The error function is minimized via a downhill simplex method.

The code CALIF (Häfner *et al.*, 1998) solves the inverse problem minimizing the objective function by a combination of steepest descent and Gauss-Newton methods and by the method according to Powel and Levenberg-Marquardt (Press *et al.* 1994). The Boussinesq approximation is made. Johansenn *et al.* 2002, solve least square problem applying a Newton method. The Jacobian matrix is approximated by incremental ratios.

There are other algorithms that can be used to calculate the Jacobian. PEST (based on Vecchia *et al.*, 1987) and UCODE, (Poeter *et al.*, 1998) are generic codes for minimization that can be coupled to any software that solves the flow and transport problem. Both algorithms calculate the Jacobian matrix by incremental rates, which is expensive and inaccurate.

Objectives

The objective of this work is to discuss alternative ways of computing Jacobian matrices exactly at a low cost.

EQUATIONS FOR SALTWATER INTRUSION

The flow equation is derived from the conservation of fluid mass neglecting the product of density gradient times flux and assuming that the density of the fluid is dependent only on solute mass fraction. This leads to:

$$S_p \frac{\partial p}{\partial t} + \phi \beta_\omega \frac{\partial \omega}{\partial t} = -\nabla \cdot \mathbf{q} + Q \qquad (1)$$

where p is fluid pressure, ρ is fluid density, ϕ is porosity, ω is the solute mass fraction (mass of solute per unit of mass of fluid), S_p is the specific storage coefficient, $\rho\beta_\omega$ is the derivative of fluid density respect to concentration, **q** is Darcy's velocity and Q are the external sink/source terms. This equation is solved subject to standard boundary and initial conditions. Darcy's Velocity can be calculated according to the generalized Darcy's law:

$$\mathbf{q} = -\frac{k}{\mu}\left[\nabla p + \rho g \nabla \mathbf{z}\right] \qquad (2)$$

where k is the intrinsic permeability, μ is the fluid viscosity, g is the acceleration of gravity and z is a unit vector pointing upwards.

Solute mass conservation leads to the transport equation by subtracting the flow equation multiplied by ω.

$$\phi \frac{\partial \omega}{\partial t} = \nabla \cdot \left(\mathbf{D} \nabla \boldsymbol{\omega}\right) - \mathbf{q} \cdot \nabla \boldsymbol{\omega} + Q(\omega^* - \omega) \qquad (3)$$

Density is taken to depend on solute mass fraction according to:

$$\rho(\omega) = \rho_0 e^{\beta_\omega(\omega-\omega_0)} \tag{4}$$

where ρ_0 is the density for solute mass fraction ω_0.

Formulation of Numerical Equations

A Galerkin formulation is chosen for solving equations (1) and (3). The resulting system consists of two coupled algebraic systems of equations.

$$\mathbf{f}_F^{k+1} = \left[\theta\mathbf{A}_F + \frac{1}{\Delta t}\mathbf{D}_F\right]\mathbf{p}^{k+1} + \frac{1}{\Delta t}\mathbf{E}_F\boldsymbol{\omega}^{k+1} - \left[(1-\theta)\mathbf{A}_F + \frac{1}{\Delta t}\mathbf{D}_F\right]\mathbf{p}^{k} + \frac{1}{\Delta t}\mathbf{E}_F\boldsymbol{\omega}^{k} + \mathbf{b}_F^{k+\theta} = 0 \tag{5}$$

$$\mathbf{f}_T^{k+1} = \left[\theta\mathbf{A}_T^{k+1} + \frac{1}{\Delta t}\mathbf{D}_T^{k+1}\right]\boldsymbol{\omega}^{k+1} - \left[(1-\theta)\mathbf{A}_T^{k} + \frac{1}{\Delta t}\mathbf{D}_T^{k}\right]\boldsymbol{\omega}^{k} + \mathbf{b}_T^{k+\theta} = 0 \tag{6}$$

where $\mathbf{p}^k$ and $\boldsymbol{\omega}^k$ are the state variables vector in time step *k*, θ is a time weighting parameter, $\mathbf{A_F}$, $\mathbf{D_F}$, $\mathbf{E_F}$, $\mathbf{A_T}$, $\mathbf{D_T}$, are matrices resulting from Galerkin's method and $\mathbf{b_F}$, $\mathbf{b_T}$, are the right hand side terms. These matrices are obtained by assembling element matrices, whose components can be found in, e.g. Voss (1984),or Huyakorn *et al.*, (1987). The point to stress here is that $\mathbf{A_F}$, $\mathbf{D_F}$, and $\mathbf{E_F}$ are linearly dependent on k, S_p and ϕ, respectively. $\mathbf{A_T}$ is the sum of advective and dispersive matrices. The former is a linear function of **q** and the latter is linear in D which in turn depends on k and ρ. $\mathbf{D_T}$ is also a linear function of ϕ. Density dependence is included in $\mathbf{b_F}$ through the buoyancy term and of course of **q**. All these terms are evaluated in k+θ.

Solution of the Numerical Equations

A coupled non-linear system like (5) and (6) is obtained, regardless the numeric method applied to discretize the continuous partial differential equations (1) and (3). The most usual method for solving coupled non-linear systems is the Picard iterative scheme. For the *m*-th iteration within the time step, the values of state variables at the previous iteration, $\mathbf{p}^{k+1,m-1}$ and $\boldsymbol{\omega}^{k+1,m-1}$, are used to evaluate the matrices in (5) and (6) as well as Darcy's velocity. The resulting linear system can then be solved for , $\mathbf{p}^{k+1}$ and $\boldsymbol{\omega}^{k+1}$, which become the values for the next iteration. The process is repeated until convergence.

Another suitable method for non-linear systems of equations is Newton–Raphson method. This method is based on a first order approximation of the system of equations (5) and (6).

$$\mathbf{f}^{k+1}(\mathbf{u}^{(m+1)}) = \mathbf{f}^{k+1}(\mathbf{u}^{(m)}) + \mathbf{R}^{k+1}(\mathbf{u}^{(m)})(\mathbf{u}^{k+1,(m+1)} - \mathbf{u}^{k+1,(m)}) = \mathbf{0} \tag{7}$$

where *m* is the iteration index, **f** is a vector of functions depending on **u** and **R** is the Jacobian of the system. In the case of the density-dependent flow, the vector of equations is $\mathbf{f}^t = (\mathbf{f}_F^t, \mathbf{f}_T^t)$ and that of state variables is $\mathbf{u}^t = (\mathbf{p}^t, \boldsymbol{\omega}^t)$. Considering the dependences of the matrices involved in (5) and (6), matrix **R** can be spelled out as:

$$\mathbf{R}^{k+1}(\mathbf{p},\boldsymbol{\omega}) = \begin{bmatrix} \dfrac{\partial \mathbf{f}_F^{k+1}}{\partial \mathbf{p}^{k+1}} & \dfrac{\partial \mathbf{f}_F^{k+1}}{\partial \boldsymbol{\omega}^{k+1}} \\ \dfrac{\partial \mathbf{f}_T^{k+1}}{\partial \mathbf{p}^{k+1}} & \dfrac{\partial \mathbf{f}_T^{k+1}}{\partial \boldsymbol{\omega}^{k+1}} \end{bmatrix} = \begin{bmatrix} \theta\mathbf{A}_F + \dfrac{1}{\Delta t}\mathbf{D}_F & \dfrac{1}{\Delta t}\mathbf{E}_F - \beta_\omega\theta\mathbf{b}_F^{k+\theta} \\ \left(\dfrac{\partial \mathbf{A}_T^{k+1}}{\partial \mathbf{q}^{k+\theta}} - \dfrac{\partial \mathbf{b}_T^{k+\theta}}{\partial \mathbf{q}^{k+\theta}}\right)\dfrac{\partial \mathbf{q}^{k+\theta}}{\partial \mathbf{p}^{k+1}} & \theta\mathbf{A}_T^{k+1} + \left(\dfrac{\partial \mathbf{A}_T^{k+1}}{\partial \mathbf{q}^{k+\theta}} + \dfrac{\partial \mathbf{b}_T^{k+\theta}}{\partial \mathbf{q}^{k+\theta}}\right)\dfrac{\partial \mathbf{q}^{k+\theta}}{\partial \boldsymbol{\omega}^{k+1}} \end{bmatrix} \tag{8}$$

Solving (7) iteratively until convergence leads to the solution of the problem at time step *k*. A partial Newton scheme (i.e. neglecting off-diagonal terms in **R**) has proven to be more robust and efficient than Picard iteration (Putti *et al.*, 1995).

INVERSE PROBLEM METHODS FOR DENSITY DEPENDENT FLOW AND SOLUTE TRANSPORT

Many formulations of the indirect inverse problem are available, e.g. maximum likelihood (Carrera *et al.*, 1986a), weighted least squares (e.g. Hill, 1998), etc. They lead to minimizing an objective function. Most commonly, this objective function is defined as:

$$F(\mathbf{x},\mathbf{p},\boldsymbol{\omega}) = (\mathbf{p}^* - \mathbf{p})^t \mathbf{V}_p^{-1} (\mathbf{p}_i^* - \mathbf{p}) + (\boldsymbol{\omega}^* - \boldsymbol{\omega})^t \mathbf{V}_\omega^{-1} (\boldsymbol{\omega}^* - \boldsymbol{\omega}) + (\mathbf{x}^* - \mathbf{x})^t \mathbf{V}_x^{-1} (\mathbf{x}^* - \mathbf{x}) \quad (9)$$

where $\mathbf{p}^*$, $\boldsymbol{\omega}^*$and $\mathbf{x}^*$ are vectors of measured pressure, solute mass fraction and parameters respectively, $\mathbf{p}$, $\boldsymbol{\omega}$ and $\mathbf{x}$ are their simulated counterparts and the matrices $\mathbf{V}_p$, $\mathbf{V}_\omega$, $\mathbf{V}_x$ are proportional to the corresponding covariance matrices. The objective function is evaluated after each forward model run as shown in Figure 1. Minimization of this function is achieved by iteratively modifying the vector of model parameters

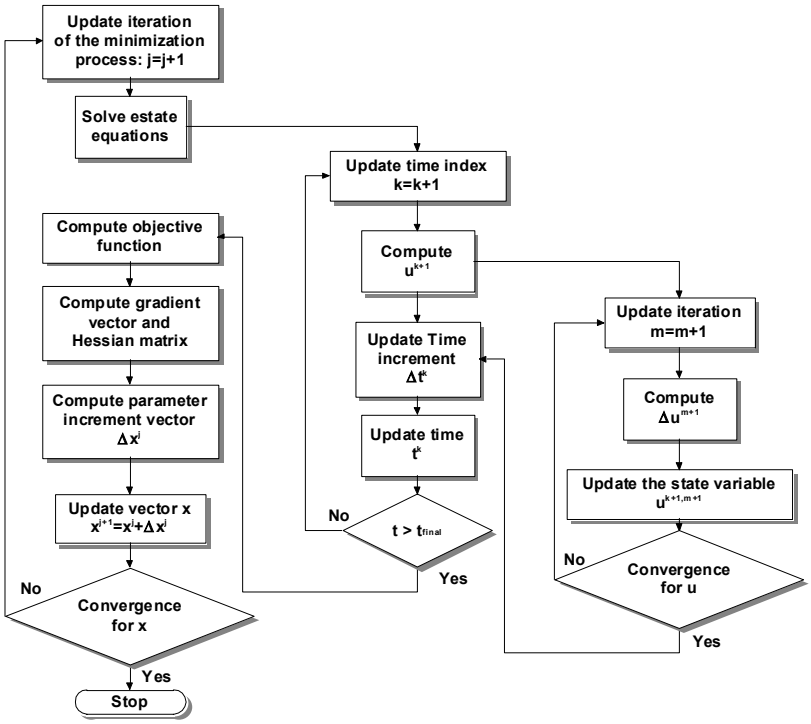

Figure 1 *The different iterative layers of an automatic parameter calibration process. Inspired in Galarza et al.(1999).*

As the governing equations relate simulated pressure and parameter values, the objective function is an N_p dimensional function of the parameters, where N_p is the number of parameters. Figure 2 shows an example of an objective surface in 2 parameters. Unlike in this example, the minimum of the objective function is in general not zero, due to measurement errors and due to a number of parameters that is smaller than the number of observations, making the system to be optimized over-determined.

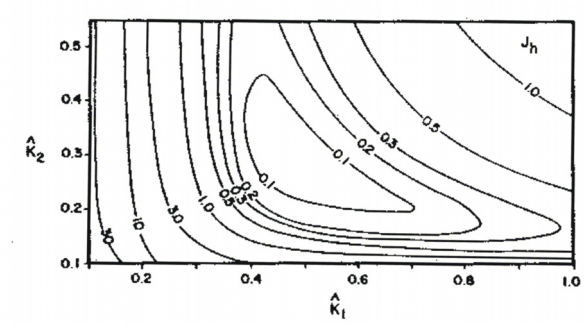

Figure 2 *Contour lines of an objective function depending on two parameters (from Carrera* et al.*, 1986b).*

In order to minimize the objective function, both gradient based and Newton based methods can be used. Gradient-based methods only require knowledge of the derivative of *F* with respect to the parameters, *dF/dx*, whereas the Newton method also requires knowledge of the Hessian matrix, (second derivatives with respect to the parameters). Actually, evaluating the Hessian is so expensive that for quasi-gradient functions, Gauss-Newton methods are normally preferred. These are based on the first order approximation of **p** and ω as a function of **x**, so that the Hessian matrix becomes:

$$\mathbf{H} \approx \mathbf{J}^t \mathbf{J} \tag{10}$$

where **J** is the Jacobian matrix, $\partial(\boldsymbol{p}, \boldsymbol{\omega})/\partial \mathbf{x}$.

The Gauss-Newton method preserves the second order convergence of the pure Newton method although it is less accurate for large residuals. In order to evaluate the sensitivities *∂p/∂x* and *∂w/∂x* again several options are available. These include parameter perturbation, adjoint state and direct derivation. The parameter perturbation technique calculates the forward problem N_p+1 times per inverse iteration: the first with the previous iteration parameters and then N_p problems, perturbing one parameter every time. The resulting pressure and concentration perturbations are used to calculate the derivatives in the Jacobian. This has several drawbacks: it is quite expensive and it is subject to round off errors at small perturbations and to over-linearization at large perturbations. These reasons make direct derivation and adjoint state attractive for calculating the Jacobian matrix.

DIRECT DERIVATION

Direct derivation method is based on calculating the Jacobian matrix from the system of equation for the state variables (Carrera *et al*, 1990). The derivative of (9) with respect to vector of parameters **x** is

$$\frac{dF}{d\mathbf{x}} = \frac{\partial F}{\partial \mathbf{x}} + \sum_{k=0}^{N_t} \left(\frac{\partial F}{\partial \mathbf{u}^k} \frac{\partial \mathbf{u}^k}{\partial \mathbf{x}} \right) \tag{11}$$

where the derivatives with respect to a vector indicate the derivative with respect to each component and N_t is the number of time steps. To obtain the derivatives $\partial \mathbf{p}^k / \partial \mathbf{x}$ and $\partial \boldsymbol{\omega}^k / \partial \mathbf{x}$, we derive the state equations (5) and (6) with respect to **x**.

$$\frac{d\mathbf{f}_F^{k+1}}{d\mathbf{x}} = \frac{\partial \mathbf{f}_F^{k+1}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_F^{k+1}}{\partial \mathbf{p}^{k+1}} \frac{\partial \mathbf{p}^{k+1}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_F^{k+1}}{\partial \boldsymbol{\omega}^{k+1}} \frac{\partial \boldsymbol{\omega}^{k+1}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_F^{k+1}}{\partial \mathbf{p}^{k}} \frac{\partial \mathbf{p}^{k}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_F^{k+1}}{\partial \boldsymbol{\omega}^{k}} \frac{\partial \boldsymbol{\omega}^{k}}{\partial \mathbf{x}} = \mathbf{0} \tag{12}$$

$$\frac{d\mathbf{f}_T^{k+1}}{d\mathbf{x}} = \frac{\partial \mathbf{f}_T^{k+1}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_T^{k+1}}{\partial \mathbf{p}^{k+1}} \frac{\partial \mathbf{p}^{k+1}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_T^{k+1}}{\partial \boldsymbol{\omega}^{k+1}} \frac{\partial \boldsymbol{\omega}^{k+1}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_T^{k+1}}{\partial \mathbf{p}^{k}} \frac{\partial \mathbf{p}^{k}}{\partial \mathbf{x}} + \frac{\partial \mathbf{f}_T^{k+1}}{\partial \boldsymbol{\omega}^{k}} \frac{\partial \boldsymbol{\omega}^{k}}{\partial \mathbf{x}} = \mathbf{0} \tag{13}$$

This is a system consisting of N_p right hand sides with $2N_n$ unknowns each, which has to be solved N_t times. Solving it for $\partial\mathbf{p}^{k+1}/\partial\mathbf{x}$ and $\partial\boldsymbol{\omega}^{k+1}/\partial\mathbf{x}$ requires knowledge of $\partial\mathbf{p}^{k}/\partial\mathbf{x}$ and $\partial\boldsymbol{\omega}^{k}/\partial\mathbf{x}$. To initiate the calculation we need $\partial\mathbf{p}^{0}/\partial\mathbf{x}$ and $\partial\boldsymbol{\omega}^{0}/\partial\mathbf{x}$. If the initial condition is a fixed pressure and solute mass fraction field then $\partial\mathbf{p}^{0}/\partial\mathbf{x}=\mathbf{0}$ and $\partial\boldsymbol{\omega}^{0}/\partial\mathbf{x}=\mathbf{0}$. If the initial condition is a steady state flow and concentration field, then

$$\frac{\partial\mathbf{f}_F^0}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_F^0}{\partial\mathbf{p}^0}\frac{\partial\mathbf{p}^0}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_F^0}{\partial\boldsymbol{\omega}^0}\frac{\partial\boldsymbol{\omega}^0}{\partial\mathbf{x}}=\mathbf{0} \tag{14}$$

$$\frac{\partial\mathbf{f}_T^0}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_T^0}{\partial\mathbf{p}^0}\frac{\partial\mathbf{p}^0}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_T^0}{\partial\boldsymbol{\omega}^0}\frac{\partial\boldsymbol{\omega}^0}{\partial\mathbf{x}}=\mathbf{0} \tag{15}$$

Notice that system (12), (13) can be written in matrix form as

$$\mathbf{R}^{k+1}\begin{pmatrix}\dfrac{\partial\mathbf{p}^{k+1}}{\partial\mathbf{x}}\\[2ex]\dfrac{\partial\boldsymbol{\omega}^{k+1}}{\partial\mathbf{x}}\end{pmatrix}=\begin{pmatrix}-\dfrac{\partial\mathbf{f}_F^{k+1}}{\partial\mathbf{x}}-\dfrac{\partial\mathbf{f}_F^{k+1}}{\partial\mathbf{p}^k}\dfrac{\partial\mathbf{p}^k}{\partial\mathbf{x}}-\dfrac{\partial\mathbf{f}_F^{k+1}}{\partial\boldsymbol{\omega}^k}\dfrac{\partial\boldsymbol{\omega}^k}{\partial\mathbf{x}}\\[2ex]-\dfrac{\partial\mathbf{f}_T^{k+1}}{\partial\mathbf{x}}-\dfrac{\partial\mathbf{f}_T^{k+1}}{\partial\mathbf{p}^k}\dfrac{\partial\mathbf{p}^k}{\partial\mathbf{x}}-\dfrac{\partial\mathbf{f}_T^{k+1}}{\partial\boldsymbol{\omega}^k}\dfrac{\partial\boldsymbol{\omega}^k}{\partial\mathbf{x}}\end{pmatrix} \tag{16}$$

where matrix of **R** of (16) is the same that in (8).

ADJOINT STATE

The adjoint state method uses the method of Lagrange multipliers to eliminate the terms $\partial\boldsymbol{p}/\partial\mathbf{x}$ and $\partial\boldsymbol{\omega}/\partial\mathbf{x}$ in (12)y (13) (Carrera *et al*, 1990). We can add to the objective function $2N_t$ vectors of time varying Lagrange multipliers multiplying the state equations, since these are zero:

$$\mathrm{F}(\mathbf{x},\mathbf{p},\boldsymbol{\omega})+\sum_{\mathrm{i}=0}^{\mathrm{N_t}}\boldsymbol{\lambda}^{i^t}\mathbf{f}^i \tag{17}$$

where $\boldsymbol{\lambda}^{i^t}=(\boldsymbol{\lambda}_F^{i\ t},\boldsymbol{\lambda}_F^{i\ t})$ are the transposed column vectors of Lagrange multipliers associated with $\mathbf{f}_F^i$ and $\mathbf{f}_T^i$, respectively, at time step *i*. The derivative of the objective function with respect to **x** can then be written as

$$\frac{dF}{d\mathbf{x}}=\frac{\partial F}{\partial\mathbf{x}}+\frac{\partial F}{\partial\mathbf{p}}\frac{\partial\mathbf{p}}{\partial\mathbf{x}}+\frac{\partial F}{\partial\boldsymbol{\omega}}\frac{\partial\boldsymbol{\omega}}{\partial\mathbf{x}}+\sum_{\mathrm{i}=0}^{\mathrm{N_t}}\left(\boldsymbol{\lambda}_F^{i\ t}\left(\frac{\partial\mathbf{f}_F^{\ i}}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_F^{\ i}}{\partial\mathbf{p}}\frac{\partial\mathbf{p}}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_F^{\ i}}{\partial\boldsymbol{\omega}}\frac{\partial\boldsymbol{\omega}}{\partial\mathbf{x}}\right)+\boldsymbol{\lambda}_T^{i\ t}\left(\frac{\partial\mathbf{f}_T^{\ i}}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_T^{\ i}}{\partial\mathbf{p}}\frac{\partial\mathbf{p}}{\partial\mathbf{x}}+\frac{\partial\mathbf{f}_T^{\ i}}{\partial\boldsymbol{\omega}}\frac{\partial\boldsymbol{\omega}}{\partial\mathbf{x}}\right)\right) \tag{18}$$

This can be rearranged by factoring $\partial\mathbf{p}/\partial\mathbf{x}$ and $\partial\boldsymbol{\omega}/\partial\mathbf{x}$ out:

$$\frac{dF}{d\mathbf{x}}=\frac{\partial F}{\partial\mathbf{x}}+\frac{\partial\mathbf{p}}{\partial\mathbf{x}}\left[\frac{\partial F}{\partial\mathbf{p}}+\sum_{i=0}^{N_t}\left(\boldsymbol{\lambda}_F^{i\ t}\frac{\partial\mathbf{f}_F^i}{\partial\mathbf{p}}+\boldsymbol{\lambda}_T^{i\ t}\frac{\partial\mathbf{f}_T^i}{\partial\mathbf{p}}\right)\right]+\frac{\partial\boldsymbol{\omega}}{\partial\mathbf{x}}\left[\frac{\partial F}{\partial\boldsymbol{\omega}}+\sum_{i=0}^{N_t}\left(\boldsymbol{\lambda}_F^{k\ t}\frac{\partial\mathbf{f}_F^k}{\partial\boldsymbol{\omega}}+\boldsymbol{\lambda}_T^{k\ t}\frac{\partial\mathbf{f}_T^k}{\partial\boldsymbol{\omega}}\right)\right]+\sum_{i=0}^{N_t}\left(\boldsymbol{\lambda}^{i^t}\frac{\partial\mathbf{f}^i}{\partial\mathbf{x}}\right) \tag{19}$$

Now we can choose the values of the Lagrange multipliers so as to eliminate $\partial\boldsymbol{p}/\partial\mathbf{x}$ and $\partial\boldsymbol{\omega}/\partial\mathbf{x}$:

$$\frac{\partial F}{\partial \mathbf{p}^j} + \sum_{i=0}^{N_t} \left(\boldsymbol{\lambda}_F^{i\,t} \frac{\partial \mathbf{f}_F^i}{\partial \mathbf{p}^j} + \boldsymbol{\lambda}_T^{i\,t} \frac{\partial \mathbf{f}_T^i}{\partial \mathbf{p}^j} \right) = \mathbf{0} \tag{20}$$

$$\frac{\partial F}{\partial \boldsymbol{\omega}^j} + \sum_{i=0}^{N_t} \left(\boldsymbol{\lambda}_F^{i\,t} \frac{\partial \mathbf{f}_F^i}{\partial \boldsymbol{\omega}^j} + \boldsymbol{\lambda}_T^{i\,t} \frac{\partial \mathbf{f}_T^i}{\partial \boldsymbol{\omega}^j} \right) = \mathbf{0} \tag{21}$$

for j= 1 to N_t. This reduces (19) to

$$\frac{dF}{d\mathbf{x}} = \frac{\partial F}{\partial \mathbf{x}} + \sum_{i=0}^{N_t} \left(\boldsymbol{\lambda}_F^{i\,t} \frac{\partial \mathbf{f}_F^i}{\partial \mathbf{x}} + \boldsymbol{\lambda}_T^{i\,t} \frac{\partial \mathbf{f}_T^i}{\partial \mathbf{x}} \right) \tag{22}$$

Eliminating all the zero derivatives and transposing all matrices of (20) and (21) for j=k+1 yields the system of equations

$$\left(\frac{\partial F}{\partial \mathbf{p}^k} \right)^t + \left(\frac{\partial \mathbf{f}_F^{k+1}}{\partial \mathbf{p}^k} \right)^t \boldsymbol{\lambda}_F^{k+1} + \left(\frac{\partial \mathbf{f}_T^{k+1}}{\partial \mathbf{p}^k} \right)^t \boldsymbol{\lambda}_T^{k+1} + \left(\frac{\partial \mathbf{f}_F^{k}}{\partial \mathbf{p}^k} \right)^t \boldsymbol{\lambda}_F^{k} + \left(\frac{\partial \mathbf{f}_T^{k}}{\partial \mathbf{p}^k} \right)^t \boldsymbol{\lambda}_T^{k} = \mathbf{0}^t \tag{23}$$

$$\left(\frac{\partial F}{\partial \boldsymbol{\omega}^k} \right)^t + \left(\frac{\partial \mathbf{f}_F^{k+1}}{\partial \boldsymbol{\omega}^k} \right)^t \boldsymbol{\lambda}_F^{k+1} + \left(\frac{\partial \mathbf{f}_T^{k+1}}{\partial \boldsymbol{\omega}^k} \right)^t \boldsymbol{\lambda}_T^{k+1} + \left(\frac{\partial \mathbf{f}_F^{k}}{\partial \boldsymbol{\omega}^k} \right)^t \boldsymbol{\lambda}_F^{k} + \left(\frac{\partial \mathbf{f}_T^{k}}{\partial \boldsymbol{\omega}^j} \right)^t \boldsymbol{\lambda}_T^{k} = \mathbf{0}^t \tag{24}$$

it should be noticed that this system is solved backwards in time starting from k+1=Nt and subject to the final condition $\boldsymbol{\lambda}_{\mathrm{F}}^{\mathrm{N_t}} = \boldsymbol{\lambda}_{\mathrm{T}}^{\mathrm{N_t}} = \mathbf{0}$.

The system (23), (24) can be written as:

$$\mathbf{R}^{\mathrm{k\,t}} \boldsymbol{\lambda}^{\mathrm{k}} = \begin{pmatrix} -\left(\frac{\partial \mathrm{F}}{\partial \mathbf{p}^{\mathrm{k}}} \right)^{\mathrm{t}} - \left(\frac{\partial \mathbf{f}_{\mathrm{F}}^{\mathrm{k+1}}}{\partial \mathbf{p}^{\mathrm{k}}} \right)^{\mathrm{t}} \boldsymbol{\lambda}_{\mathrm{F}}^{\mathrm{k+1}} - \left(\frac{\partial \mathbf{f}_{\mathrm{T}}^{\mathrm{k+1}}}{\partial \mathbf{p}^{\mathrm{k}}} \right)^{\mathrm{t}} \boldsymbol{\lambda}_{\mathrm{T}}^{\mathrm{k+1}} \\ -\left(\frac{\partial \mathrm{F}}{\partial \boldsymbol{\omega}^{\mathrm{k}}} \right)^{\mathrm{t}} - \left(\frac{\partial \mathbf{f}_{\mathrm{F}}^{\mathrm{k+1}}}{\partial \boldsymbol{\omega}^{\mathrm{k}}} \right)^{\mathrm{t}} \boldsymbol{\lambda}_{\mathrm{F}}^{\mathrm{k+1}} - \left(\frac{\partial \mathbf{f}_{\mathrm{T}}^{\mathrm{k+1}}}{\partial \boldsymbol{\omega}^{\mathrm{k}}} \right)^{\mathrm{t}} \boldsymbol{\lambda}_{\mathrm{T}}^{\mathrm{k+1}} \end{pmatrix} \tag{25}$$

where it should be noticed that the coefficient matrix is the transpose of that in (8) and (16). The resulting algorithm is shown in Figure 3. It should be apparent that computations associated to the adjoint state are limited to one system of equations (25) per time step.

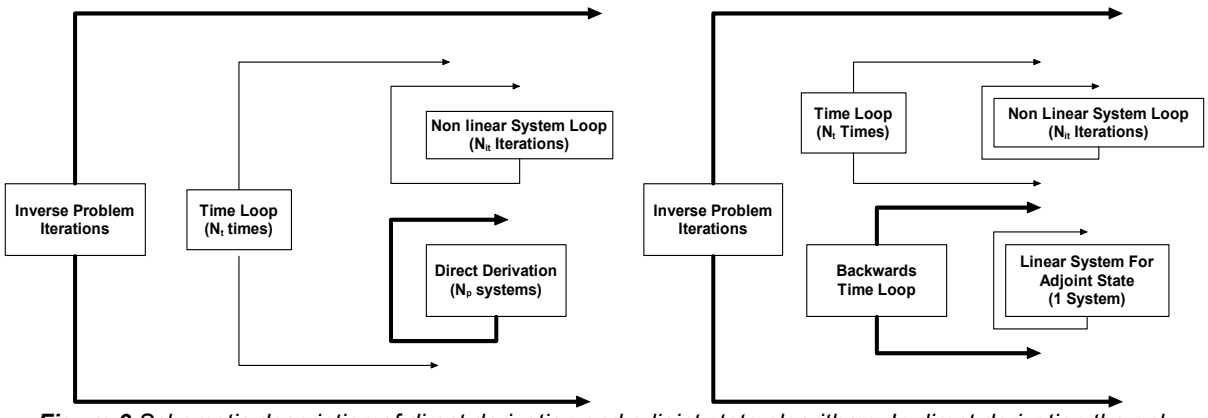

Figure 3 *Schematic description of direct derivation and adjoint state algorithms. In direct derivation the only modifications with respect to a forward problem simulations are the need for the solution of N_p (non-iterative systems at each time step, and the inverse problem iterations. Instead, in the adjoint state method, the forward simulation time loop remains unaltered, but a noniterative backwards in time loop need to be added. (See Figure 1 for details on each loop).*

DISCUSSION

Three systems of equations have been obtained. One for the Newton-Raphson method applied to the forward problem and two for the inverse problem. The coefficient matrix **R i**n all of them is the same (transposed when using adjoint state). This is an additional advantage of using Newton-Raphson for solving forward problems with automatic parameter estimation.

When using the parameter perturbation technique, the forward problem must be computed N_p+1 times (Figure 3). The system size of the forward problem is the same as the system size of the inverse problem. Therefore, it can be compared to solving the system (N_p+1) times in each time step. Iterating to a solution is necessary, which costs N_{it} times solving the system and N_{it} times building a matrix, where N_{it} is the number of iterations required.

When the direct derivation technique is used, the system (12) and (13) should be solved N_t times (not forgetting that it is composed by N_p sub-systems of for $2N_n$ unknowns). Solving it does not require the construction of a system matrix, since this matrix is already known from the resolution of the forward problem. Nor does it require iterating to a solution. Therefore, direct derivation approximately N_{it} times cheaper than parameter perturbation.

When using the adjoint state method, system (23) and (24) should be solved Nt times, each time for 2Nn unknowns. As it must be solved backwards in time, it requires the one to store pressure and solute mass fraction fields at every time step. Solving the system then also requires to build the matrix $\mathbf{R}^t$, already calculated when solving the forward problem. Iterating to a solution is not necessary and the matrix $\mathbf{R}^t$ can be build directly using the stored values of pressure and solute mass fraction.

Since an iterative process in every time step is necessary when calculating the forward problem, solving the adjoint state equations will take less time than solving the forward problem (approximately N_{it} times less, assuming that the number of iterations is the same for all time steps.

When comparing computational costs it becomes obvious that both direct derivation and adjoint state are faster than parameter perturbation. Moreover, they are exact.

ACKNOWLEDGMENTS

This work was performed at the Technical University of Catalonia in the framework of the SALTRANS and FEBEX projects. The first one is funded by the European Union (contract n°: EVK1-CT-2000-00062) and the second one by ENRESA (Spanish nuclear waste disposal company) through grant number 774313.

REFERENCES

Bear, J. (1972), Dynamics of Fluids in Porous Media, 764 pp., Elsevier, New York.

Bear, J (1979), Hydraulics of Groundwater, 559 pp., McGraw-Hill, New York.

Carrera, J. & Neuman, S. P. (1986a) Estimation of aquifer parameters under transient and steady state conditions. 1. Maximum likelihood method incorporating prior information, Wat. Resour. Res. 22(2). 199-210.

Carrera, J. & Neuman, S. P. (1986b) Estimation of aquifer parameters under transient and steady state conditions: 2. Uniqueness, stability, and solution algorithms, Wat. Resour. Res. 22(2). 211-227.

Carrera, J., (1988) State of the art of the inverse problem applied to the flow and solute transport equations, in Groundwater flow and quality modelling, Custodio, E. et al. (eds,.) pp. 549-583, Reidel Publishing Company, Dordrecht

Carrera, J., Navarrina, F., Vives, J., Heredia, J. Medina, A. (1990), Computational aspects of the inverse problem, , in Computational methods in subsurface hydrology (Proceedings of the eight International Conference on Computational Methods in Water Resources), Gambolatti et al. (eds.), 514-522, Springer-Verlag, Berlin.

Galarza, G. A., Carrera, J. & Medina, A., (1999), Computational techniques for optimization of problems involving non-linear transient simulations, Int. J. Numer. Meth. Engng. 45, 319-334.

Häfner, F., Boy, S., Behr, A., Kelbe, B. & Germeshuyse, T., Parameter calibration using CALIF for density-dependent groundwater flow and transport in Groundwater qualty: Remediation and protection, Herbert et al. (eds.), 521-528 , IAHS Publication nº 250,

Hill, M., (1998), Methods and guidelines for effective model calibration, U.S. Geological Survey, Water –resources investigations report 98-4005, Denver, Colorado (USA)

Holzbecher, E., (1998) Modelling Density-Driven Flow in Porous Media, 305 pp., Springer-Verlag, Berlin.

Huyakorn, P. S., Andersen, P. F., Mercer, J. W., With, H. O. Jr., (1987) Saltwater intrusion in aquifers: Developing and testing of a three-dimensional finite element model, Water Resour. Res., 23, 293-312.

Iribar, V., Carrera, J., Custodio, E. & Medina, A., 1996, Inverse modelling of seawater intrusion in the Llobregat delta deep aquifer, J. Hydrol. 198 (1-4) (1997) pp. 226-244

Johannsen, K., Kinzelbach, W., Oswald, S. & Wittum, G., 2002, The saltpool benchmark problem – numerical simulation of saltwater upconing in a porous medium, Adv. Water Res., 25 (2002), 335-348.

Kinzelbach, W., (1986) Groundwater Modelling, 333 pp., Elsevier, Amsterdam.

Kolditz, O., Ratke, R., Diersch, H. J. & Zielke, W. (1998) Coupled groundwater flow and transport. I. Verification of variable density flow and transport models. Adv. Water Res. 21(1), 27-46.

Piggott, a. R., & Bobba A. G., Inverse analysis implementation of the SUTRA groundwater flow and transport model and user's guide for SUTRA-1, Environment Canada, National Water Research Institute, Burlington, Saskatoon, NWRI Contribution No. 93-115.

Poeter, E.P. and Hill, M.C., (1998), Documentation of UCODE, a computer code for universal inverse modelling. U.S. Geolog. Surv. Water Resour. Invest. Report 98-4080, 116 p.

Putti, M. and Paniconi, C., (1995), Picard and Newton linearization for the coupled model of saltwater intrusion in aquifers, Adv. Water Res., 18(3), 159-170.

Press, W., Flannery, B. P., Teukolsky, S. A. & Vetterling, W. T. (1994) Numerical Recipes (Second edn.) Cambridge university Press, Cambridge, UK.

Olivella, S., Carrera, J., Gens, A. & Alonso, E. E., (1994) Nonisothermal multiphase flow of brines and gas through saline media, Transport in Porous Media, 15: 271-293.

Vecchia, A.V. and Cooley, R.L., (1987). Simultaneous Confidence and Prediction Intervals for Non-linear Regression Models with Application to a Groundwater Flow Model. Water Resour. Res., 23 (7),1237-1250.

Voss, C. I. (1984) SUTRA: A finite element simulation model for saturated-unsaturated fluid-density-dependent groundwater flow with energy transport of chemically-reactive simple-species solute transport, U. S. Geolg. Surv. Water Resour. Invest., 84-4369, 409.