



Permeability estimation with the augmented Lagrangian method for a nonlinear diffusion equation

Trygve K. Nilssen^a, Trond Mannseth^b and Xue-Cheng Tai^a

^a *Department of Mathematics, University of Bergen, Johannes Brunsgt. 12, N-5008 Bergen, Norway*

E-mail: {TrygveKastberg.Nilssen;Xue-Cheng.Tai}@mi.uib.no

^b *Rogalandsforskning, Thormøhlensgt. 55, N-5008 Bergen, Norway*

E-mail: Trond.Mannseth@rf.no

Received 15 January 2001; accepted 15 March 2002

We consider numerical identification of the piecewise constant permeability function in a nonlinear parabolic equation, with the augmented Lagrangian method. By studying this problem, we aim at also gaining some insight into the potential ability of the augmented Lagrangian method to handle permeability estimation within the full two-phase porous-media flow setting. The identification is formulated as a constrained minimization problem. The parameter estimation problem is reduced to a coupled nonlinear algebraic system, which can be solved efficiently with the conjugate gradient method. The methodology is developed and numerical experiments with the proposed method are presented.

Keywords: inverse problems, nonlinear, parabolic, permeability

1. Introduction

The system of partial differential equations modeling two-phase immiscible flow of incompressible fluids in a porous medium with zero gravity effects is

$$\phi(x) \frac{\partial S}{\partial t} - \nabla \cdot (K(x) \mu_w^{-1}(p) k_w(S) \nabla p) = f_w(x), \quad (1.1)$$

$$-\phi(x) \frac{\partial S}{\partial t} - \nabla \cdot (K(x) \mu_n^{-1}(p) k_n(S) (\nabla p + P'(S) \nabla S)) = f_n(x). \quad (1.2)$$

Here, ϕ denotes porosity, S – wetting-phase fluid saturation, K – (absolute) permeability, μ – fluid viscosity, k – relative permeability, p – wetting-phase fluid pressure, f – fluid source term, and P denotes capillary pressure. The subscripts w and n refer to the wetting and non-wetting fluid phases, respectively, while the prime superscript denotes derivation.

Reservoir simulation based on these equations, (and more elaborate versions of them, including additional physical effects) is a standard tool to help make impor-

tant decisions regarding the management of petroleum reservoirs, including selection of the type of recovery method, fluid production and injection rates, and well locations.

The coefficient functions $\phi(x)$, $K(x)$, $k_w(S)$, $k_n(S)$ and $P(S)$ vary from one porous medium to another, and are inaccessible to direct measurements. To infer these functions from measurable reservoir quantities like well pressures and flow rates is a huge inverse problem. Usually this problem is divided in two parts – one involving estimation of the spatially dependent functions and one involving estimation of the saturation dependent functions. This paper is concerned with parameter estimation methodology for one of the spatially dependent functions $K(x)$.

The permeability $K(x)$ is usually modeled as a piecewise constant function, i.e., it is defined by a single value within each reservoir simulator grid cell. So, one is interested in inferring as many parameters as there are grid cells in the simulator. This number often exceeds 10^5 for a field simulation. There are mainly two difficulties associated with this.

The first problem is that there is far from sufficient information in objective data (well pressures and flow rates) to infer the permeability function with such a high resolution. This is usually dealt with either by substantially reducing the degrees of freedom in parameter space, or by utilizing prior knowledge about the permeability to aid in the estimation, or both. (The prior information could typically be a quantitative expression – with uncertainty bounds and parameter correlations – for the geologist/reservoir engineers opinion of what the permeability distribution looks like.) Such issues are not the subject of this paper, and we refer the interested reader to, e.g., [5,6,12,13].

The second problem is that even with a modest amount of parameters to be estimated – corresponding to the attainable resolution implicitly given by the data – the computational cost can still be very high. A single field-model reservoir simulation may take several hours to complete and many such simulator runs are required to obtain a permeability estimate with optimization methods like quasi-Newton or Gauss–Newton. Note also that due to novel data acquisition techniques, like time-lapse 3D-seismic surveys, the attainable permeability resolution from objective data is expected to increase in the near future. Hence, there will be an increasing need for parameter estimation techniques able to handle a larger number of parameters within a reasonable time span. This paper is concerned with further development of such alternative parameter estimation methodology.

Ito and Kunisch [7], and Kunisch and Tai [10], considered the augmented Lagrangian method for identification of $q(x)$ within the linear elliptic equation

$$-\nabla \cdot (q(x)\nabla u) = f(x), \quad (1.3)$$

and recently, Nilssen and Tai [11] considered the augmented Lagrangian method for recovery of $q(x)$ within the linear parabolic equation

$$u_t - \nabla \cdot (q(x)\nabla u) = f(x, t). \quad (1.4)$$

Keung and Zou [8] have also considered parameter estimation problems in (1.5), using an Armijo algorithm.

Both (1.3) and (1.4) can be viewed as describing flow processes related to those described by the coupled system (1.1), (1.2). Equation (1.3) corresponds to modeling of single-phase porous-media flow with constant fluid density and viscosity.

Equation (1.4) corresponds to modeling of slightly compressible single-phase flow with constant compressibility, viscosity and porosity. There is no difficulty in handling a nonconstant porosity, however. The time-derivative term would then have to be changed to $\phi(x)u_t$, but this would not add any new difficulties with respect to the augmented Lagrangian formalism. In the following we will assume that porosity is constant, for simplicity.

With these assumptions, the function $q(x)$ in (1.3) and (1.4), is related to the permeability function by $q(x) = \text{constant} \cdot K(x)$. In the following we will denote the function $q(x)$ the permeability.

In this paper, we attempt to take the augmented Lagrangian method further towards application to permeability identification within multi-phase porous-media flow by considering an intermediate step. That is, we extend the augmented Lagrangian method to identification of $q(x)$ within the nonlinear parabolic equation

$$u_t - \nabla \cdot (q(x)N(\nabla u, u)\nabla u) = f(x, t). \quad (1.5)$$

Here, $N(\nabla u, u)$ will model the main characteristics of the nonlinearity associated with some of the coefficient functions in equations (1.1), (1.2). These two coupled nonlinear equations for S and p contain nonlinearities associated with both of the dependent variables. The influence of S is mainly through the actual value of S , while the influence of p is mainly through ∇p . Hence, we have included both u and ∇u as independent variables in N .

Obviously, one can not expect that all aspects of the influence of the nonlinearities in the coefficient functions in equations (1.1), (1.2) will be modeled by the function N in (1.5). What we should aim for is, thus, to assess the implications of having a generic nonlinear function multiplying the function to be identified, both on the formulation of the augmented Lagrangian methodology as such, and on its computational performance.

The study of equation (1.5) also will give information about the performance of augmented Lagrangian method for the system (1.1), (1.2).

The organization of the paper is as follows. First, we give some more detail on the selected generic model and the parameter estimation framework. Next, we present the numerical scheme that is used to solve the parabolic initial-boundary condition problem. Then we present the augmented Lagrangian method for the minimization problem. Thereafter, we show how the preconditioned conjugate gradient method can be used to calculate the gradients in the augmented Lagrangian method with very low computational costs and modest need for disc space. Finally, we present results from numerical experiments with the method.

2. Generic model and parameter identification framework

The generic nonlinear parabolic equation constituting the forward model in this paper has one of the following two forms

$$u_t - \nabla \cdot (q(x)N(\nabla u)\nabla u) = f(x, t), \quad \text{in } \Omega \times (0, T) \quad (2.1)$$

or

$$u_t - \nabla \cdot (q(x)N(u)\nabla u) = f(x, t), \quad \text{in } \Omega \times (0, T) \quad (2.2)$$

with initial-boundary conditions $u(x, 0) = u_0(x)$, in Ω and $u(x, t) = 0$, in $\partial\Omega \times (0, T)$. Here Ω can be any bounded domain in \mathbb{R}^d , $d \geq 1$, with piecewise smooth boundary $\partial\Omega$, and $f(\cdot, t) \in H^{-1}(\Omega)$ is a given source term. N is some positive nonlinear function of ∇u or u . To simplify the notation we will some places write the two nonlinear equations as one, cf. (1.5).

The identification process is carried out in a way that the solution u matches its observation data $u_{\text{obs}}(x^i, t)$, $i = 1, \dots, n_p$, $t \in (0, T)$, optimally. The measurements may contain noise.

For parameter identification in elliptic systems, Ito and Kunisch [7] proposed a hybrid method, which formulate the problem as a minimization problem, combining the output least squares and the equation error formulation. This is then solved with the augmented Lagrangian method. The minimization formulation is

$$\min_{e(q,u)=0} \int_0^T \sum_{i=1}^{n_p} \frac{1}{2} (u(x^i, t) - u_{\text{obs}}(x^i, t))^2 dt + \beta R(q) \quad (2.3)$$

where $e(q, u) = 0$ is the equation constraint. $e(q, u)$ will be referred to as the equation error, and could for example be defined as the left-hand side minus the right-hand side of the equation, i.e.

$$u_t - \nabla \cdot (qN(\nabla u, u)\nabla u) - f.$$

The second term of the minimization, $\beta R(q)$, is a regularization term which will be specified later. We will assume that it is quadratic in q . Note that in our minimization formulation we do not use the interpolated version of u_{obs} . In the formulation we only calculate the distance between u and u_{obs} at the observation points. It is also interesting to notice that we minimize over both q and u . This is a flexible formulation and it will hopefully give fast and stable convergence.

The minimizer for (2.3) can now be found by the augmented Lagrangian method. We introduce the augmented Lagrangian functional

$$\begin{aligned} L_c(q, u, \lambda) = & \int_0^T \sum_{i=1}^{n_p} \frac{1}{2} (u(x^i, t) - u_{\text{obs}}(x^i, t))^2 dt + \beta R(q) \\ & + \int_0^T (e, \lambda)_V dt + \frac{c}{2} \int_0^T \|e\|_V^2 dt. \end{aligned}$$

Here the subscript V denotes a proper inner product space, which will be specified in the next section. A saddle point for L_c together with the equation constraint fulfilled is a local minima for (2.3).

With traditional methods like quasi-Newton or Gauss–Newton, the objective function is genuine nonlinear function of q . Here, L_c is quadratic in q for fixed u . In [11] L_c is also quadratic in u for fixed q , but that is not true in this paper because of the nonlinearity in the function $N(\nabla u, u)$. In the following, the presentation will be similar to that in [11], extending this approach to allow for the presence of the nonlinear function $N(\nabla u, u)$. A thorough presentation is still included, for completeness.

3. Finite element approximation and the augmented Lagrangian method

We first consider a finite element discretization for solving (1.5), and then present the augmented Lagrangian approach in a discrete setting. The numerical solver, a simple implicit Euler-scheme, serves as a basis for the minimization process for (2.3) and is used to make reference solutions to compare our results with. If we use other numerical schemes for the forward problem, the augmented Lagrangian functional should be modified correspondingly.

3.1. Finite element approximation

Let Ω be a polyhedral domain in \mathbb{R}^d , $d \geq 1$, and let \mathcal{T}^h be a regular triangulation of Ω , with simplicial elements, namely intervals in one dimension, triangles in two and tetrahedra in three dimensions (cf. [4]). The superscript h denotes the diameter of the largest simplex of the triangulation. Let V_h be the standard piecewise linear finite element space over this triangulation, with functions vanishing on the boundary $\partial\Omega$. This is the element space where u is defined.

To define the space for q , we let \mathcal{T}^H be a triangulation of Ω with either simplicial or rectangular elements. Let W_H^c denote the piecewise constant element space over this triangulation, and let W_H denote

$$W_H = \{q \in W_H^c \mid 0 < q_{\min} \leq q \leq q_{\max} < \infty\},$$

where q_{\min} and q_{\max} are given. In practical applications, the dimension of V_h is normally required to be much higher than the dimension of W_H . In the case that \mathcal{T}^h is a refined mesh of \mathcal{T}^H , the implementation is simpler.

We divide the time interval $(0, T)$ into M equal subintervals by using time levels $t^n = n\tau$, $n = 0, \dots, M$, with $\tau = T/M$. We initialize by setting

$$u_h^0 = I^h(u_0(x)) \in V_h,$$

where I^h is the interpolation operator into V_h . u_h^n is defined recursively by solving

$$\left(\frac{u_h^n - u_h^{n-1}}{\tau}, v \right) + (qN(u_h^n, \nabla u_h^n) \nabla u_h^n, \nabla v) = (f^n, v), \quad \forall v \in V_h, \quad n > 0, \quad (3.1)$$

where (\cdot, \cdot) is the $L^2(\Omega)$ -inner product and $f^n = f(x, t^n)$. To find u_h^n from u_h^{n-1} is a nonlinear problem. We use a Picard iteration to solve this (see next subsection).

This defines

$$u_h = \begin{pmatrix} u_h^0 \\ \vdots \\ u_h^M \end{pmatrix} \in (V_h)^{M+1}.$$

In the rest of the paper we drop the subscript h on u .

3.2. Picard iteration

If we write (3.1) on the form

$$A(z)z = b,$$

where $A(z)$ is a matrix depending nonlinearly on z , the Picard iteration is

$$A(z^{k-1})z^k = b,$$

till convergence for z^0 given. This iteration solves the system efficiently in the case when $N = N(u)$, but does often diverge in the case when $N = N(\nabla u)$. For the latter case we have used a more stabilized three term version of the Picard iteration

$$A(\alpha z^{k-1} + (1 - \alpha)z^{k-2})z^k = b, \quad \alpha \in (0, 1),$$

with $\alpha \in (0.4, 0.6)$.

The Picard iteration is only used to solve the forward problem (3.1).

3.3. Equation error

In the augmented Lagrangian method, we regard equation (3.1) as a constraint. To minimize the equation error, we need to use a proper norm to measure it in. In our simulations, we have used the following two inner products for this purpose

$$(u, v)_V = (u, v) + \tau(\nabla u, \nabla v), \quad (3.2a)$$

$$(u, v)_V = (u, v). \quad (3.2b)$$

The corresponding norm is $\|\cdot\|_V^2 = (\cdot, \cdot)_V$. When $\tau = O(h^2)$, the two norms induced by the two inner products are equivalent with an equivalence constant independent of h and τ for functions from V_h . In such cases, we will use the inner product (3.2b). When τ is big, we need to use the inner product (3.2a). In order to evaluate this norm, we need to solve a large linear system. This can be avoided by using equivalent norms produced by multigrid or domain decomposition methods as in [14].

For any $u \in (V_h)^{M+1}$ and $q \in W_H$, the discretized equation error $e = e(q, u) \in (V_h)^M$ is

$$(e^n, v)_V = (u^n - u^{n-1}, v) + \tau(qN(\nabla u^n, u^n)\nabla u^n, \nabla v) - \tau(f^n, v), \quad \forall v \in V_h, \quad n > 0.$$

We see that e^n depends on q , u^n and u^{n-1} . For any given $q \in W_H$ and $u \in (V_h)^{M+1}$, we say that (q, u) satisfies equation (3.1) if $e^n = 0$, $\forall n$. In an explicit form, the equation for e^n can be written in the following way

$$e^n(q, u) = (I - \tau \Delta_h)^{-1} (u^n - u^{n-1} - \tau \nabla_h \cdot (qN(\nabla u^n, u^n) \nabla_h u^n) - \tau f^n),$$

when we use the inner product (3.2a). Here the subscript h denotes that we use a discretized version of the operator. The operator $(I - \tau \Delta_h)^{-1}$ can be replaced by some corresponding operators produced by domain decomposition or multigrid methods, see [14]. If we use the inner product (3.2b), the equation error is

$$e^n(q, u) = u^n - u^{n-1} - \tau \nabla_h \cdot (qN(\nabla u^n, u^n) \nabla_h u^n) - \tau f^n.$$

3.4. Discretized minimization

We formulate the finite element problem corresponding to (2.3) as follows

$$\min_{e(q,u)=0} \tau \sum_n E(u^n) + \beta R(q), \quad (3.3)$$

subject to $q \in W_H$ and $u \in (V_h)^{M+1}$ satisfying $u^0 = I^h(u_0(x))$. Here

$$E(u^n) = \sum_{i=1}^{n_p} \frac{1}{2} (u^n(x^i) - u_{\text{obs}}^n(x^i))^2,$$

where $u_{\text{obs}}^n(x^i) = u_{\text{obs}}(x^i, t^n)$, $i = 1, \dots, n_p$, $n = 1, \dots, M$, and the regularization term $\beta R(q)$ will be specified in section 5.

3.5. Minimization algorithm

The constrained minimization problem can be solved with an augmented Lagrangian method. The discretized augmented Lagrange functional $L_c : W_H \times (V_h)^{M+1} \times (V_h)^M \rightarrow \mathbb{R}$ is now written

$$L_c(q, u, \lambda) = \tau \sum_{n=1}^M E(u^n) + \beta R(q) + \sum_{n=1}^M (\lambda^n, e^n)_V + \sum_{n=1}^M \frac{c}{2} \|e^n\|_V^2.$$

Here $c > 0$ is a penalization constant, which is determined experimentally. In the discrete setting, it is known that L_c has a saddle point and that this point is a minimizer for (3.3), see [3,7,10].

We will use the following modified Uzawa algorithm to find saddle-points for this functional.

Algorithm 3.1 (The global minimization algorithm).

- (1) Choose initial values for $\lambda_0 \in (V_h)^M$, $u_0 \in (V_h)^{M+1}$ and set $k = 1$.

(2) Find q_k from

$$L_c(q_k, u_{k-1}, \lambda_{k-1}) = \min_{q \in W_H} L_c(q, u_{k-1}, \lambda_{k-1}).$$

(3) Set $u_k^0 = u^0$ and find $\{u_k^n\}_{n=1}^M$ from

$$L_c(q_k, u_k, \lambda_{k-1}) = \min_{u \in (V_h)^{M+1}} L_c(q_k, u, \lambda_{k-1}). \quad (3.4)$$

(4) Update the Lagrange-multiplier by

$$\lambda_k = \lambda_{k-1} + ce(q_k, u_k).$$

If not converged: Set $k = k + 1$ and go to step 2.

We call algorithm 3.1 the global minimization algorithm because it searches for the global minimizer in step (3). Later we will present an algorithm which searches for an approximate solution in this step.

4. Implementation issues with the conjugate gradient method

In this section, we study an efficient method to solve the two subminimization problems in the modified Uzawa algorithm. We will use the notations

$$L'_c \cdot p = \frac{\partial L_c(q, u, \lambda)}{\partial q} \cdot p \quad \text{and} \quad L'_c \cdot w = \frac{\partial L_c(q, u, \lambda)}{\partial u} \cdot w$$

to denote the Gateaux derivatives of the functional $L_c(q, u, \lambda)$. Note that when writing $L'_c \cdot p$, the p indicates that we take the derivative with respect to q in the direction p . Similarly the w in $L'_c \cdot w$ indicates that we take the derivative with respect to u in the direction w . The notations

$$L''_c \cdot (p, p) = \frac{\partial^2 L_c(q, u, \lambda)}{\partial q^2} \cdot (p, p) \quad \text{and} \quad L''_c \cdot (w, w) = \frac{\partial^2 L_c(q, u, \lambda)}{\partial u^2} \cdot (w, w)$$

are used for the second order derivatives.

4.1. The nonlinear conjugate gradient method

In this subsection we look at methods to solve

$$\min_{z \in \mathbb{R}^l} F(z),$$

where F is smooth function and we have its gradients available. For large scale problems (l large), conjugate gradient methods are an important class of optimization algorithms. These methods have the following form

$k = 1$, z_0 given, $g_1 = -\nabla F(z_0)$,
while $\|\nabla F(z_k)\| > \varepsilon$,
 $z_k = z_{k-1} + \alpha_k g_k$,
 $g_{k+1} = -\nabla F(z_k) + \beta_k g_k$,
 $k = k + 1$,
end

where α_k is a step size, which can be found with a one-dimensional line search

$$\alpha_k = \arg \min_{\alpha} F(z_k + \alpha g_k).$$

The scalar β_k can be chosen as either β_k^1 or β_k^2 , (see [1]):

$$\beta_k^1 = \frac{\|\nabla F(z_k)\|^2}{\|\nabla F(z_{k-1})\|^2},$$

$$\beta_k^2 = \frac{(\nabla F(z_k), \nabla F(z_k) - \nabla F(z_{k-1}))}{\|\nabla F(z_{k-1})\|^2},$$

where $\|\cdot\|$ and (\cdot, \cdot) denotes the l^2 -norm and -inner product, correspondingly. The latter choice β_k^2 is most stable with respect to non-optimal line search. If F is a quadratic functional, we have that $\nabla F(z_k)$ and $\nabla F(z_{k-1})$ are orthogonal. Therefore $\beta_k^1 = \beta_k^2$ in that case.

For quadratic functions F , the exact solution of the line search is

$$\alpha_k = \frac{\|\nabla F(z_{k-1})\|^2}{g_k^T H_k g_k},$$

where H_k is the Hessian of F in the point z_k , $H_k = \nabla^2 F(z_k)$. As an approximation to the line search this can also be done in the nonquadratic cases. This will be done in this paper.

To use the conjugate gradient method, we need to calculate $\nabla F(z_k)$ and $g_k^T \nabla^2 F(z_k) g_k$ for given z_k and g_k . We do not need to form the Hessian. In the next subsection we will show how this can be done.

For fixed (u, λ) , the functional $L_c(q, u, \lambda)$ is quadratic with respect to q , but for fixed (q, λ) , the functional $L_c(q, u, \lambda)$ is nonquadratic with respect to u . In order to use the conjugate gradient method we need to calculate the Gateaux derivatives of L_c . The next subsection shows these calculations when $N = N(\nabla u)$, and the subsection after shows these calculations when $N = N(u)$. The calculations in this section are similar to the corresponding calculations in [11]. The only difference is the nonlinear function N . For more detailed calculations, see [11].

4.2. The Gateaux derivatives of L_c when $N = N(\nabla u)$

In this subsection we first calculate the Gateaux derivatives of L_c with respect to q , and comment how these calculations are used in the implementation. Thereafter the same is done for the derivatives with respect to u .

We define $d_1^n = (e^n)' \cdot p$. We see that d_1 satisfies

$$(d_1^n, v)_V = \tau(pN(\nabla u^n)\nabla u^n, \nabla v) \quad \forall v \in V_h.$$

The Gateaux derivative of L_c with respect to q in direction of p is

$$\begin{aligned} L'_c \cdot p &= \beta R'(q) \cdot p + \sum_n (d_1^n, \lambda^n + ce^n)_V \\ &= \beta R'(q) \cdot p + \sum_n \tau(pN(\nabla u^n)\nabla u^n, \nabla(\lambda^n + ce^n)). \end{aligned}$$

In the implementation we need a formula for calculating $\partial L_c / \partial q$. Assume that $\{\phi_j\}$ are the basis functions for W_H^c . Since entry j of $\partial L_c / \partial q$ is $(\partial L_c / \partial q)(j) = L'_c \cdot \phi_j$, we see that

$$\frac{\partial L_c}{\partial q}(j) = \beta R'(q) \cdot \phi_j + \sum_n \tau(\phi_j N(\nabla u^n)\nabla u^n, \nabla(\lambda^n + ce^n)).$$

For the second order derivative with respect to q we get that

$$L''_c \cdot (p, p) = \beta R''(q) \cdot (p, p) + c \sum_n \|d_1^n\|_V^2.$$

As above we define $d_2^n = (e^n)' \cdot w$, and see that it satisfies

$$(d_2^n, v)_V = (w^n - w^{n-1}, v) + \tau(qN'(\nabla u^n) \cdot \nabla w^n \nabla u^n + qN(\nabla u^n)\nabla w^n, \nabla v) \quad \forall v \in V_h.$$

The Gateaux derivative of L_c with respect to u is

$$\begin{aligned} L'_c \cdot w &= \tau \sum_n E'(u^n) \cdot w^n + \sum_n (d_2^n, \lambda^n + ce^n)_V \\ &= \tau \sum_{n,i} (u^n(x^i) - u_{\text{obs}}^n(x^i)) w^n(x^i) + \sum_n (w^n, (\lambda^n + ce^n) - (\lambda^{n+1} + ce^{n+1})) \\ &\quad + \sum_n \tau(qN'(\nabla u^n) \cdot \nabla w^n \nabla u^n + qN(\nabla u^n)\nabla w^n, \nabla(\lambda^n + ce^n)), \end{aligned}$$

where we have defined $\lambda^{M+1} = e^{M+1} = 0$ to simplify the notation.

Assume that $\{\psi_j\}$ are the basis functions for V_h . We can now calculate entry j in time level n of $\partial L_c / \partial u$

$$\begin{aligned} \frac{\partial L_c}{\partial u}(n, j) &= \tau \sum_i (u^n(x^i) - u_{\text{obs}}^n(x^i)) \psi_j(x^i) + (\psi_j, (\lambda^n + ce^n) - (\lambda^{n+1} + ce^{n+1})) \\ &\quad + \tau(qN'(\nabla u^n) \cdot \nabla \psi_j \nabla u^n + qN(\nabla u^n)\nabla \psi_j, \nabla(\lambda^n + ce^n)). \end{aligned}$$

The second order derivative with respect to u is

$$\begin{aligned}
L_c'' \cdot (w, w) &= \tau \sum_n E''(u^n) \cdot (w, w) + c \sum_n ((e^n)' \cdot w, (e^n)' \cdot w)_V \\
&\quad + \sum_n ((e^n)'' \cdot (w, w), \lambda^n + ce^n)_V \\
&= \tau \sum_{n,i} (w^n(x^i))^2 + c \sum_n \|d_2^n\|_V^2 \\
&\quad + 2\tau \sum_n (qN'(\nabla u^n) \cdot \nabla w^n \nabla w^n, \nabla(\lambda^n + ce^n)) \\
&\quad + \tau \sum_n (qN''(\nabla u^n) \cdot (\nabla w^n, \nabla w^n) \nabla u^n, \nabla(\lambda^n + ce^n)).
\end{aligned}$$

4.3. The Gateaux derivatives of L_c when $N = N(u)$

The calculations are similar to those in the previous subsection and are omitted. Like above we define $d_1^n = (e^n)' \cdot p$ and $d_2^n = (e^n)' \cdot w$. We see that d_1 and d_2 satisfy

$$\begin{aligned}
(d_1^n, v)_V &= \tau (pN(u^n) \nabla u^n, \nabla v) & \forall v \in V_h, \\
(d_2^n, v)_V &= (w^n - w^{n-1}, v) + \tau (qN'(u^n) \cdot w^n \nabla u^n + qN(u^n) \nabla w^n, \nabla v) & \forall v \in V_h.
\end{aligned}$$

The Gateaux derivatives of L_c are

$$L_c' \cdot p = \beta R'(q) \cdot p + \sum_n \tau (pN(u^n) \nabla u^n, \nabla(\lambda^n + ce^n))$$

and

$$\begin{aligned}
L_c' \cdot w &= \tau \sum_{n,i} (u^n(x^i) - u_{\text{obs}}^n(x^i)) w^n(x^i) + \sum_n (w^n, (\lambda^n + ce^n) - (\lambda^{n+1} + ce^{n+1})) \\
&\quad + \sum_n \tau (qN'(u^n) \cdot w^n \nabla u^n + qN(u^n) \nabla w^n, \nabla(\lambda^n + ce^n)),
\end{aligned}$$

where we have defined $\lambda^{M+1} = e^{M+1} = 0$. The second order derivatives are

$$L_c'' \cdot (p, p) = \beta R''(q) \cdot (p, p) + c \sum_n \|d_1^n\|_V^2$$

and

$$\begin{aligned}
L_c'' \cdot (w, w) &= \tau \sum_{n,i} (w^n(x^i))^2 + c \sum_n \|d_2^n\|_V^2 \\
&\quad + 2\tau \sum_n (qN'(u^n) w^n \nabla w^n, \nabla(\lambda^n + ce^n)) \\
&\quad + \tau \sum_n (qN''(u^n) \cdot (w^n, w^n) \nabla u^n, \nabla(\lambda^n + ce^n)).
\end{aligned}$$

4.4. An efficient minimization algorithm

The most time consuming part of algorithm 3.1 is step (3), i.e. the minimization with respect to u . This is because u is a function of both space and time, and therefore have most degrees of freedom. In addition the dimension of the space V_h is usually larger than the dimension of W_H . In this subsection we present an alternative minimization algorithm (see also [11]). This minimization algorithm will not search for the exact minimizer like algorithm 3.1. Algorithm 3.1 minimizes over all time levels simultaneously. In the new algorithm we try to split it up, and do the minimization for each time level separately.

4.4.1. Marching minimization algorithm

The idea behind this algorithm is that we go through all the time levels recursively, and at a given time level we assume that the minimizer from the previous time levels are correct. Then we calculate the minimizer for this time level based on observations from this time level only.

We shall use the notation

$$L_c(q, u, \lambda) = \sum_n \tilde{L}_c(u^n, u^{n-1}) + \beta R(q),$$

where

$$\tilde{L}_c(u^n, u^{n-1}) = \tau E(u^n) + (\lambda^n, e^n)_V + \frac{c}{2} \|e^n\|_V^2.$$

We see that $\tilde{L}_c(u^n, u^{n-1})$ also depends on λ and q , but since we only use this notation for the solving of (3.4), we will omit q and λ in $\tilde{L}_c(u^n, u^{n-1})$ for notational simplicity.

The idea for the marching minimization algorithm is that the solution at time level n of the minimizer for $\min_u L_c(q, u, \lambda)$, can be approximated by

$$u^n \approx \arg \min_v \tilde{L}_c(v, u^{n-1}),$$

where u^{n-1} is the minimizer from the previous time level.

The following algorithm will be used as a replacement for algorithm 3.1.

Algorithm 4.1 (The marching minimization algorithm).

- (1) Choose initial values for $\lambda_0 \in (V_h)^M$, $u_0 \in (V_h)^{M+1}$ and set $k = 1$.
- (2) Find q_k from

$$q_k = \arg \min_{q \in W_H} L_c(q, u_{k-1}, \lambda_{k-1}).$$

- (3) Set $u_k^0 = u^0$ and find $\{u_k^n\}_{n=1}^M$ sequentially for $n = 1, 2, \dots, M$ such that

$$u_k^n = \arg \min_{v \in V_h} \tilde{L}_c(v, u_k^{n-1}). \quad (4.1)$$

(4) Update the Lagrange-multiplier by

$$\lambda_k = \lambda_{k-1} + ce(q_k, u_k).$$

If not converged: Set $k = k + 1$ and go to step 2.

When solving (4.1), the newest values for q and λ are used. Step (3) in algorithm 4.1 defines $u_k = \{u_k^n\}_{n=0}^M$ sequentially for each time level.

To use the conjugate gradient method to solve this new minimization problem we should do some calculations similar to those in the previous subsection. The difference is that we now take the Gateaux derivative with respect to one time level u^n instead of in all time levels.

First we do the calculation with $N = N(\nabla u)$. We define $d_3^n = (\partial e^n / \partial u^n) \cdot w^n$, which satisfies

$$(d_3^n, v)_V = (w^n, v) + \tau(qN'(\nabla u^n) \cdot \nabla w^n \nabla u^n + qN(\nabla u^n) \nabla w^n, \nabla v) \quad \forall v \in V_h.$$

The Gateaux derivative of \tilde{L}_c with respect to u^n in direction w^n is

$$\begin{aligned} \tilde{L}'_c(u^n, u^{n-1}) \cdot w^n &= \tau \sum_i (u^n(x^i) - u_{\text{obs}}^n(x^i)) w^n(x^i) + (w^n, \lambda^n + ce^n) \\ &\quad + \tau(qN'(\nabla u^n) \cdot \nabla w^n \nabla u^n + qN(\nabla u^n) \nabla w^n, \nabla(\lambda^n + ce^n)), \end{aligned}$$

and the second order derivative is

$$\begin{aligned} \tilde{L}''_c(u^n, u^{n-1}) \cdot (w^n, w^n) &= \tau \sum_i (w^n(x^i))^2 + c \|d_3^n\|_V^2 + \tau(qN''(\nabla u^n) \cdot (\nabla w^n, \nabla w^n) \nabla u^n \\ &\quad + 2qN'(\nabla u^n) \cdot \nabla w^n \nabla w^n, \nabla(\lambda^n + ce^n)). \end{aligned}$$

Then the calculation with $N = N(u)$. We define $d_3^n = (\partial e^n / \partial u^n) \cdot w^n$, which satisfies

$$(d_3^n, v)_V = (w^n, v) + \tau(qN'(u^n) \cdot w^n \nabla u^n + qN(u^n) w^n, \nabla v) \quad \forall v \in V_h.$$

The Gateaux derivative of \tilde{L}_c is

$$\begin{aligned} \tilde{L}'_c(u^n, u^{n-1}) \cdot w^n &= \tau \sum_i (u^n(x^i) - u_{\text{obs}}^n(x^i)) w^n(x^i) + (w^n, \lambda^n + ce^n) \\ &\quad + \tau(qN'(u^n) \cdot w^n \nabla u^n + qN(u^n) w^n, \nabla(\lambda^n + ce^n)), \end{aligned}$$

and the second order derivative is

$$\begin{aligned} \tilde{L}''_c(u^n, u^{n-1}) \cdot (w^n, w^n) &= \tau \sum_i (w^n(x^i))^2 + c \|d_3^n\|_V^2 + \tau(qN''(u^n) \cdot (w^n, w^n) \nabla u^n \\ &\quad + 2qN'(u^n) \cdot w^n \nabla w^n, \nabla(\lambda^n + ce^n)). \end{aligned}$$

5. Numerical experiments

We now show some numerical experiments with the proposed method for parameter identification. The test problem is

$$\begin{aligned} u_t - \nabla \cdot (q(x)N(\nabla u, u)\nabla u) &= f(x, t) && \text{in } \Omega \times (0, T), \\ u(x, 0) &= u_0(x) && \text{in } \Omega, \\ u(x, t) &= 0 && \text{on } \partial\Omega \times (0, T), \end{aligned}$$

with several choices for the function N . Here $\Omega = [0, 1] \times [0, 1]$, $T = 0.01$, $u_0(x) = \sin(\pi x_1) \cdot \sin(\pi x_2)$ and the true permeability, $q(x)$, is piecewise constant

$$q(x) = \begin{cases} q_1^c, & x \in [0, 0.5] \times [0, 0.5], \\ q_2^c, & x \in [0.5, 1] \times [0, 0.5], \\ q_3^c, & x \in [0, 0.5] \times [0.5, 1], \\ q_4^c, & x \in [0.5, 1] \times [0.5, 1]. \end{cases} \quad (5.1)$$

In the examples $q_i^c = i$, $i = 1, \dots, 4$, unless otherwise defined. The source function is

$$f(x, t) = \sum_{i=1}^4 \delta(x - y_i) - 4\delta(x - y_5),$$

where y_i for $i = 1, \dots, 4$ are the corners and y_5 is the center of Ω and δ is Dirac's delta function. This corresponds to having one producer in the center of Ω and one injector in each corner of Ω .

The domain is triangulated by first dividing it into $h \times h$ squares. Then each square is divided into two triangles, by the diagonal with positive slope, to get \mathcal{T}^h . The element function $u^n(x)$ is defined over this triangulation with linear elements. The element triangulation where $q(x)$ is defined, \mathcal{T}^H , is built up of $H \times H$ squares. The number of time steps is $M = T/\Delta t$.

With the Euler-scheme described in section 3, the forward problem can be solved. The solution from this, u , will then be used as a source for the observations that our algorithms will use to recover the permeability q .

In example 5.5 we will add normally distributed noise to the observations in a multiplicative way, i.e.

$$u_{\text{obs}}^n(x^i) = u^n(x^i) + \sigma u^n(x^i) \text{rand}(i, n). \quad (5.2)$$

Here $\text{rand}(i, n)$ is a vector of normally distributed numbers with expectation 0 and standard deviation 1, and $\sigma \in \mathbb{R}$ is the noise level.

As a stopping criteria for the conjugate gradient method, we have used that the relative L^2 -norm of the gradient of the functional, $\nabla F(z_k)$, is below a certain level:

$$\frac{\|\nabla F(z_k)\|}{\|\nabla F(z_0)\|} \leq \varepsilon. \quad (5.3)$$

In all the examples we have set $\varepsilon = 10^{-6}$.

In the figures in the following examples we illustrate the convergence rates of the Uzawa algorithms. In all examples we plot $\|q_k - q\|_{L^2}$ with increasing k -value, where q is the true permeability. In example 5.1 we also illustrate the convergence rates of u by plotting $\sum_n \|u_k^n - u^n\|_{L^2}$. In the examples the Uzawa algorithm has been stopped by inspection of these plots.

The initial value for u is the spatial linear interpolation of $u_{\text{obs}}^n(x^i)$. The Lagrange multiplier is initially $\lambda_0 = 0$. The c -value is determined experimentally.

The initial values for q and u in the conjugate gradient method are the solution from the previous iteration of the Uzawa algorithm. In the first step of the Uzawa algorithm we use the spatial linear interpolation of $u_{\text{obs}}(x^i, t)$ as an initial value for u , and for q we use the constant $q_0 = (1/|\Omega|) \int_{\Omega} q \, dx$, where q is the true permeability.

In the three first examples we have tested three different N -functions: first $N(\nabla u) = 1 + 0.1|\nabla u|^2$ then $N(\nabla u) = 1/(1 - 0.1|\nabla u|^2)$ and then $N(u) = 1 + \frac{1}{2}u + u^2$. With these choices of N , the nonlinearity is approximately of the same magnitude. In the fourth example we test the marching minimization, and in the fifth we test how our algorithms handle noise in the observations. In the last example we have tested our algorithm in an example with strong discontinuities. In the last three examples the nonlinear function $N(\nabla u) = 1 + 0.1|\nabla u|^2$ is used.

In the following examples we have observed the convergence rates when either the inner products (3.2a) or (3.2b) are used. We have seen that the conjugate gradient method converges in fewer iterations when (3.2b) is used, but since (3.2a) is computationally less expensive this has been preferable in our examples.

In the examples we use the regularization term $R(q) = \|q\|_{L^2}^2$. However, the results seems to be best when the regularization parameter is chosen $\beta = 0$. The reason for this is probably that the dimension on \mathcal{T}^H is relatively small in our examples.

In all examples we have used $H = \frac{1}{4}$, $h = \frac{1}{8}$ and $M = 20$. In the center of each square element of \mathcal{T}^H , there is one observation point. That means we have 16 observation points for u_{obs} and 16 parameters representing q .

In the experiments the constraints on W_H : $q_{\min} \leq q \leq q_{\max}$, were never active.

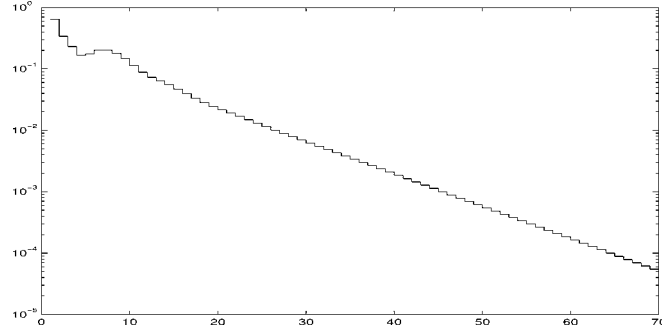
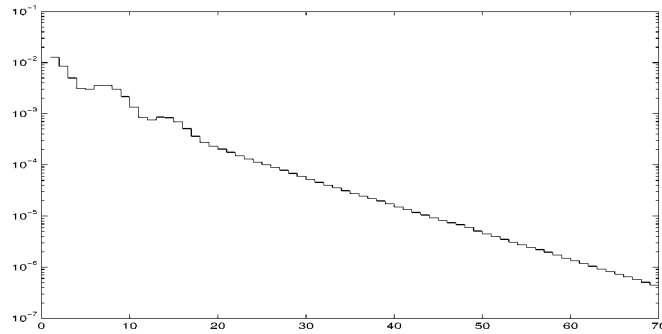
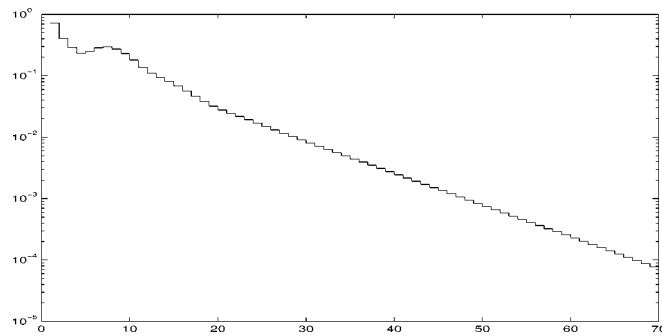
Example 5.1. In the first example we use algorithm 3.1. The nonlinear function N is $N(\nabla u) = 1 + 0.1|\nabla u|^2$, with Gateaux derivatives

$$\begin{aligned} N'(\nabla u) \cdot \nabla w &= 0.2\nabla u \cdot \nabla w, \\ N''(\nabla u) \cdot (\nabla w, \nabla w) &= 0.2|\nabla w|^2. \end{aligned}$$

In this example the c -value was set to $7 \cdot 10^{-7}$. The convergence rate of q is shown in figure 1, and the convergence rate of u is shown in figure 2.

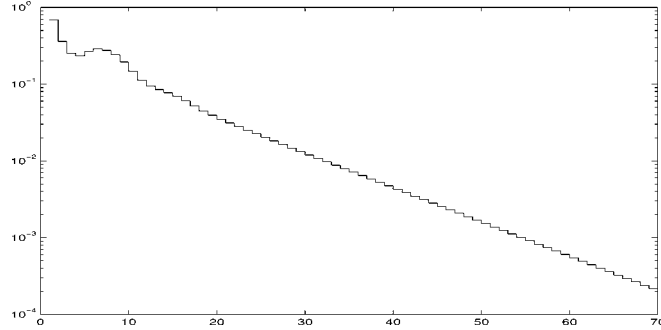
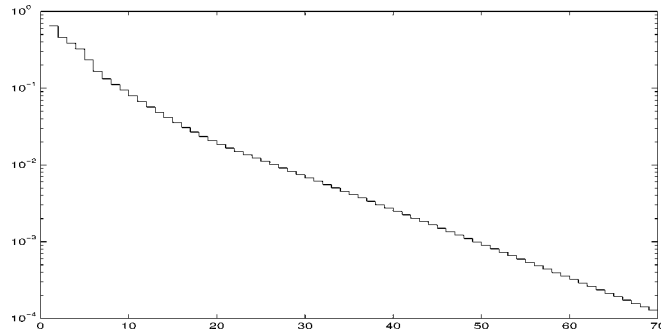
Example 5.2. In this example we also use algorithm 3.1. The nonlinear function N is $N(\nabla u) = 1/(1 - 0.1|\nabla u|^2)$, with Gateaux derivatives

$$N'(\nabla u) \cdot \nabla w = \frac{0.2\nabla u \cdot \nabla w}{(1 - 0.1|\nabla u|^2)^2},$$

Figure 1. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis.Figure 2. $\sum_n \|u_k^n - u^n\|_{L^2}$ versus k . Logarithmic scale on the vertical axis.Figure 3. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis.

$$N''(\nabla u) \cdot (\nabla w, \nabla w) = \frac{0.08(\nabla u \cdot \nabla w)^2 + 0.2|\nabla w|^2(1 - 0.1|\nabla u|^2)}{(1 - 0.1|\nabla u|^2)^3}.$$

In this example the c -value was set to $6 \cdot 10^{-7}$. The convergence rate of q is shown in figure 3.

Figure 4. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis.Figure 5. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis.

Example 5.3. In this example we also use algorithm 3.1. The nonlinear function N is $N(u) = 1 + \frac{1}{2}u + u^2$, with Gateaux derivatives

$$N'(u) \cdot w = \frac{1}{2}w + 2uw,$$

$$N''(u) \cdot (w, w) = 2w^2.$$

In this example the c -value was set to $8 \cdot 10^{-7}$. The convergence rate of q is shown in figure 4.

The three different N -functions tested seems to give about the same convergence rate. However, if the nonlinearity strength is increased (i.e. the coefficients in front of the nonlinear terms in N), the nonlinear conjugate gradient methods (see section 4.1) may diverge.

Example 5.4. In this example we use algorithm 4.1. The nonlinear function N is $N(\nabla u) = 1 + 0.1|\nabla u|^2$. In this example the c -value was set to $1.4 \cdot 10^{-5}$. The convergence rate of q is shown in figure 5.

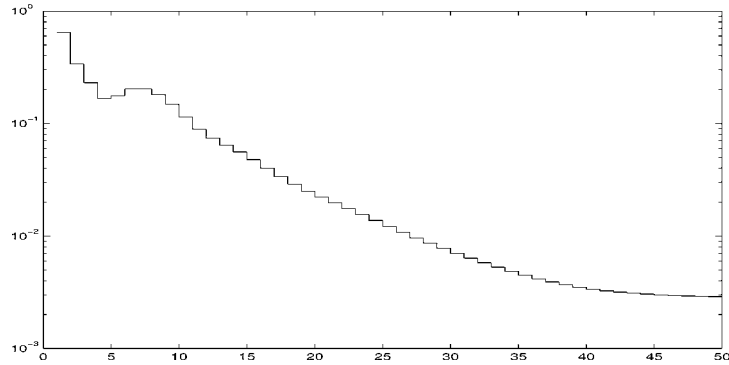


Figure 6. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis. The noise level is $\sigma = 10^{-3}$.

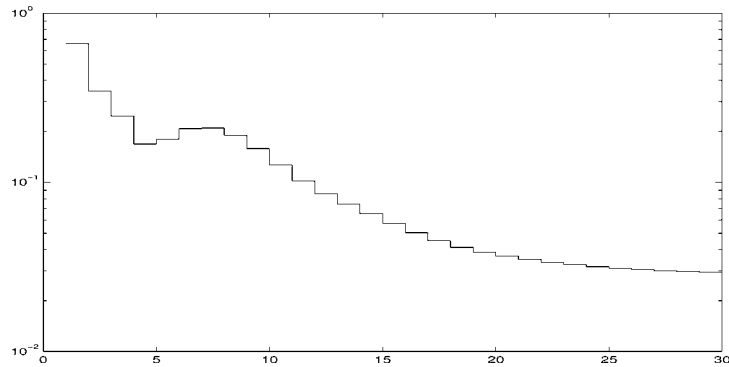


Figure 7. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis. The noise level is $\sigma = 10^{-2}$.

We see that the convergence rate with algorithm 4.1 is almost as good as with algorithm 3.1. The computational time in this example is about 65% of the time in example 5.1 due to reduced cost per iteration. The reduction of computational cost will be larger when working with problems involving more time steps.

Example 5.5. This example is as described in examples 5.1 and 5.4, except that we have introduced multiplicative noise (see (5.2)). The convergence rates of q when the noise level is $\sigma = 10^{-3}$ and $\sigma = 10^{-2}$ are shown in figures 6 and 7, respectively. Here algorithm 3.1 is used.

In figure 8 we show the convergence rate when algorithm 4.1 is used, and the noise level is $\sigma = 10^{-2}$.

To show the influence of a noise level of magnitude 10^{-2} on the data, we have in figure 9 plotted the pressure with and without noise in a point $x^1 = (\frac{1}{8}, \frac{1}{8})$ as a function of time. In the plot we have linearly interpolated the discrete functions to get continuous functions of time.

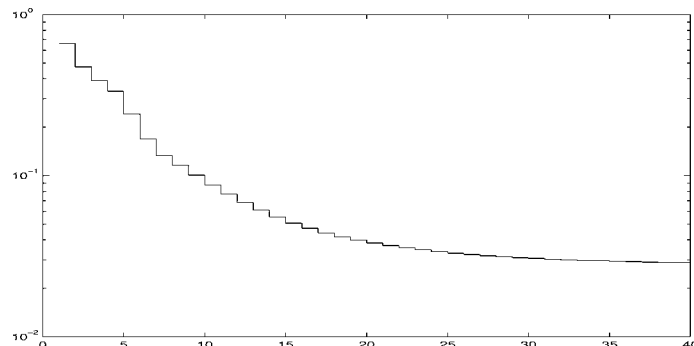


Figure 8. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis. The noise level is $\sigma = 10^{-2}$ and algorithm 4.1 is used.

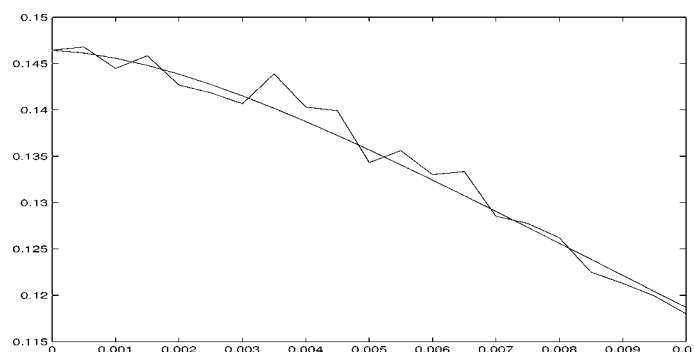


Figure 9. The pressure with and without noise in position $x^1 = (\frac{1}{8}, \frac{1}{8})$, i.e. $u(x^1, t)$ and $u_{\text{obs}}(x^1, t)$, $t \in (0, T)$. The noise level is $\sigma = 10^{-2}$.

Figure 10 shows the result after 30 iterations, when $\sigma = 10^{-2}$ and algorithm 3.1 is used (cf. figure 7). The relative L^2 error in q is $\|q_{30} - q\|/\|q\| \approx 0.011$.

Example 5.6. In oil reservoirs the permeability often has very large jumps. In this example we shall test algorithm 3.1 with permeability as described in (5.1) with $q_i^c = 10^{i-3}$, $i = 1, \dots, 4$.

In this example the c -value was set to $2.9 \cdot 10^{-7}$. The convergence rate of q is shown in figure 11. We see that the convergence rate is a little bit slower than in the previous examples. In addition every iteration is about twice as costly as in example 5.1, because the conjugate gradient method converges slower.

6. Concluding remarks

In this paper we have further developed the augmented Lagrangian method to solve a parameter estimation problem associated with a nonlinear parabolic equation. This

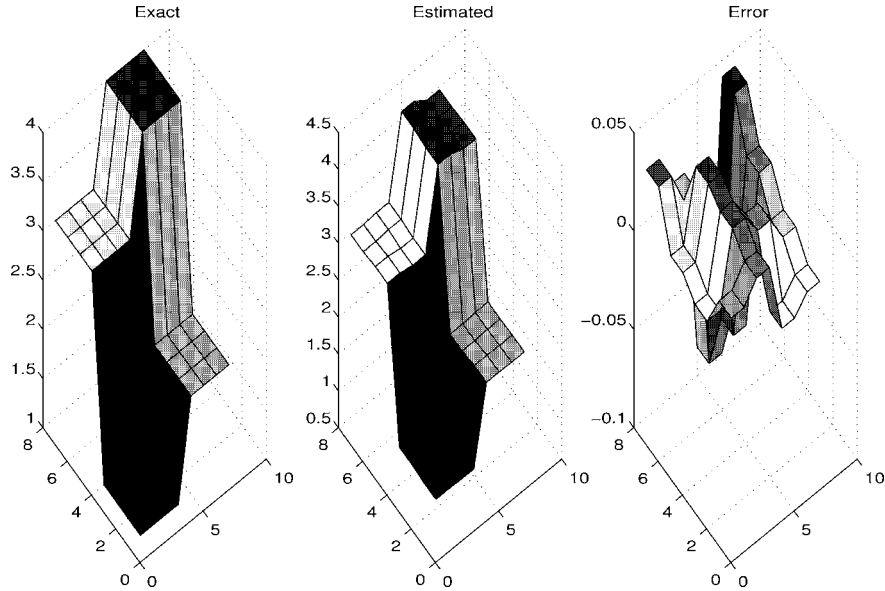


Figure 10. The exact, the estimated and the error in permeability $q(x)$ with noise level $\sigma = 10^{-2}$.

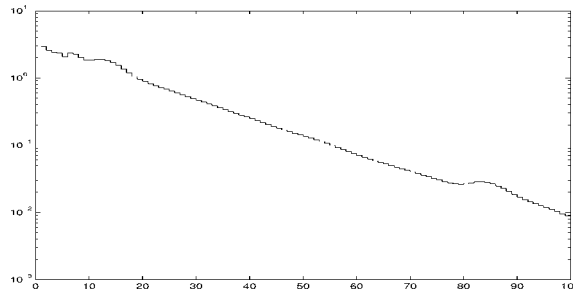


Figure 11. $\|q_k - q\|_{L^2}$ versus k . Logarithmic scale on the vertical axis.

problem can be viewed as a simplified form of the permeability estimation within multiphase porous-media flow.

The convergence rate for the Uzawa algorithm is approximately the same as in [11], which consider parameter estimation within a linear parabolic equation. The model equation in that paper is the same as in this paper if $N = 1$. Each iteration of the Uzawa algorithm is about 60% more expensive than in [11]. This is because the nonlinear conjugate gradient method converges slower than the linear conjugate gradient method. Our code could probably be optimized some by doing the line search in the nonlinear conjugate gradient method more accurate.

The increase in computational cost when going from linear to nonlinear parabolic equations is sufficiently small that we find that it would be of interest to develop the augmented Lagrangian method further to solve permeability estimation problems in the multiphase flow equations.

Acknowledgements

This work was partially supported by the Research Council of Norway (NFR), under grant 128224/431.

References

- [1] D.P. Bertsekas, *Nonlinear Programming* (Athena Scientific, 1990).
- [2] T.F. Chan and X.-C. Tai, Identification of discontinuous coefficient from elliptic problems using total variation regularization, Technical Report CAM-97-35, University of California at Los Angeles, Department of Mathematics (1997).
- [3] Z. Chen and J. Zou, An augmented Lagrangian method for identifying discontinuous parameters in elliptic systems, *SIAM J. Control Optim.* 37 (1999) 892–910.
- [4] P.G. Ciarlet, *The Finite Element Method for Elliptic Problems* (North-Holland, Amsterdam, 1978).
- [5] M. Cuypers, O. Dubrule, P. Lamy and R. Bissell, Optimal choice of inversion parameters for history matching with the pilot point method, in: *Proc. of the 6th European Conf. on the Mathematics of Oil Recovery*, Peebles, UK (1998).
- [6] A.A. Grimstad, T. Mannseth, J.E. Nordtvedt and G. Nævdal, Reservoir characterization through scale splitting, in: *Proc. of the 7th European Conf. on the Mathematics of Oil Recovery*, Baveno, Italy (2000).
- [7] K. Ito and K. Kunisch, The augmented Lagrangian method for parameter estimation in elliptic systems, *SIAM J. Control Optim.* 28 (1990) 113–136.
- [8] Y.L. Keung and J. Zou, Numerical identifications of parameters in parabolic equations, *Inverse Problems* (1998) 83–100.
- [9] K. Kunisch and G. Peichl, Estimation of a temporally and spatially varying diffusion coefficient in a parabolic system by an augmented Lagrangian technique, *Numer. Math.* 59 (1991) 473–509.
- [10] K. Kunisch and X.-C. Tai, Sequential and parallel splitting methods for bilinear control problems in Hilbert spaces, *SIAM J. Numer. Anal.* 34 (1997) 91–118.
- [11] T.K. Nilssen and X.-C. Tai, Parameter estimation with the augmented Lagrangian method for a parabolic equation, see URL: <http://www.mi.uib.no/~tai/> (2001).
- [12] A.C. Reynolds, N. He, L. Chu and D. Oliver, Reparameterization techniques for generating reservoir descriptions conditioned to variograms and well-test pressure data, SPE 30588, in: *Proc. of SPE Annual Technical Conference and Exhibition*, Dallas, USA (1995).
- [13] P.C. Shah, G.R. Gavalas and J.H. Seinfeld, Error analysis in history matching, The optimum level of parameterization, *Soc. Petrol. Engrg. J.* 6 (1978).
- [14] X.-C. Tai, J. Frøyen, M.E. Espedal and T.F. Chan, Overlapping domain decomposition and multigrid methods for inverse problems, in: *Domain Decomposition Methods, 10th Internat. Conf. on Domain Decomposition Methods*, Contemporary Mathematics, Vol. 218 (Amer. Math. Soc., Providence, RI, 1998) pp. 523–529, see also URL: <http://www.mi.uib.no/~tai/>.