

A Bound-Constrained Levenburg-Marquardt Algorithm for a Parameter Identification Problem in Electromagnetics

Johnathan M. Bardsley*

February 19, 2004

Abstract

Our objective is to solve a parameter identification inverse problem arising in electromagnetics. We give a brief discussion of the methodology used for obtaining forward solutions to Maxwell's equations in two dimensions, in the presence of a Debye medium. A careful consideration of the noise model for data generation leads to a nonlinear least squares formulation of the inverse problem, and the physical constraints on the parameters lead to bound constraints. An unconstrained Levenburg-Marquardt algorithm is presented, and modifications of this algorithm for use on bound-constrained problems are discussed. Some convergence properties of the constrained algorithm, as well as numerical results from the application of interest are also presented.

1 Introduction

In [2] the authors discuss techniques for solving Maxwell's equations in two dimensions in a Debye medium, and for solving the associated parameter estimation or inverse problem. The experimental setup assumed in the formulation of the two dimensional problem includes an infinite strip antenna which is infinite in the y direction, but is finite in the x direction. This setup is illustrated schematically in Figure 1. The resulting Maxwell system involves oblique incident waves on the dielectric slab and hence is two dimensional. Solutions of the Maxwell system are obtained via the finite difference time-domain (FDTD) algorithm [11]. Perfectly matched layer (PML) absorbing boundary conditions are implemented in order to model the (effectively) infinite spatial domain by a finite computational domain. A brief discussion of the computational methodology used to obtain forward solutions, as well as numerical results from forward simulations are presented in Section 2.1.

In Section 2.2 we discuss the associated parameter identification inverse problem. That is, given data generated from the experimental setup illustrated in Figure 1 with a Debye dielectric slab, determine the parameters that characterize the Debye medium. We begin the section with

*Department of Mathematical Sciences, University of Montana, Missoula, MT 59812-0864

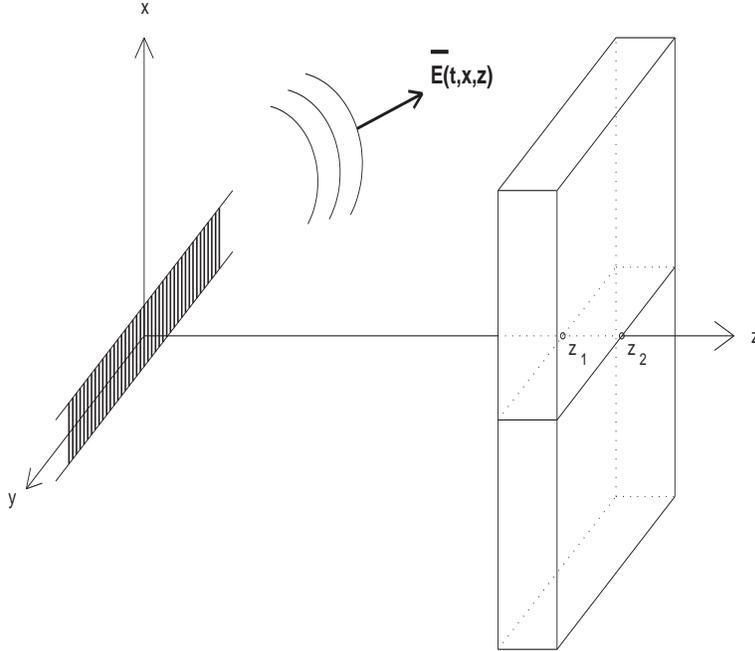


Figure 1: Infinite strip antenna and dielectric slab.

the noise model for data generation for such experiments. Using this noise model, we are led to the negative log-likelihood function, which takes a nonlinear least squares form. The inverse problem is then to minimize this function subject to the physical constraints on the parameters. To this end, we use a modified Levenburg-Marquardt method for bound constrained nonlinear least squares problems. It is this algorithm that is the subject of this paper.

Section 3 is devoted to the development of the bound-constrained Levenburg-Marquardt algorithm. To provide both motivation and a general algorithmic frame-work, we begin with an unconstrained Levenburg-Marquardt algorithm in Section 3.1, and then discuss modifications to this algorithm for use on bound constrained problems in 3.2. The major modification is in the choice of quadratic models during the trust-region subproblem at each iteration. Motivation for each of the modifications is discussed, and convergence properties of the constrained algorithm are presented. Finally, in Section 4, the algorithm is applied to the parameter identification inverse problem discussed in the previous paragraph, and is shown to converge. Conclusions are presented in Section 5

2 Problem Statement

In this section we present the computational methodology for the forward solution of Maxwell's Equations together with the associated parameter identification inverse problem of interest. Our presentation follows [2] closely.

2.1 Forward Problem

We begin with Maxwell's equations for a region with free charge density $\rho = 0$, which are given by

$$\nabla \cdot \mathbf{D} = 0 \quad (1)$$

$$\nabla \cdot \mathbf{B} = 0 \quad (2)$$

$$\nabla \times \mathbf{E} = -\partial_t \mathbf{B} \quad (3)$$

$$\nabla \times \mathbf{H} = \partial_t \mathbf{D} + \mathbf{J}, \quad (4)$$

where the vectors in (1)-(4) are functions of position $\mathbf{r} = (x, y, z)$ and time t , and $\mathbf{J} = \mathbf{J}_c + \mathbf{J}_s$, where \mathbf{J}_c is the conduction current density and \mathbf{J}_s is the source current density. We assume only free space can have a source current, and thus either $\mathbf{J}_c = \mathbf{0}$ or $\mathbf{J}_s = \mathbf{0}$, depending on whether or not the region is free space. For a Debye medium, magnetic effects can be neglected, and we assume that Ohm's law governs the electric conductivity. Hence, within the dielectric medium

$$\mathbf{B} = \mu_0 \mathbf{H}, \quad (5)$$

$$\mathbf{J}_c = \sigma \mathbf{E}. \quad (6)$$

The displacement vector \mathbf{D} , on the other hand, has a more complex representation, namely,

$$\mathbf{D} = \epsilon_0 \epsilon_\infty \mathbf{E} + \mathbf{P}, \quad (7)$$

where \mathbf{P} is the electric polarization vector, and satisfies the differential equation

$$\tau \dot{\mathbf{P}} + \mathbf{P} = \epsilon_0 (\epsilon_s - \epsilon_\infty) \mathbf{E} \quad (8)$$

inside of the dielectric and is $\mathbf{0}$ outside of the dielectric.

Provided certain assumptions are made, the system (1)-(8) can be simplified substantially. First, facilitated by the fact that we have assumed uniformity along the y -axis, the computational domain for our problem is contained in the x, z plane. Secondly, we assume that the electromagnetic pulse emitted from the antenna is polarized so that it only has a y component. These simplifying assumption are counteracted somewhat by a difficulty that arise when numerical solutions of the resulting system are sought. In particular, in order to obtain numerical solutions, a finite computational grid is necessary. For our problem, the resulting computational boundaries will generally reflect electromagnetic waves, and hence, numerical solutions will contain non-physical artifacts. One of the more effective ways to combat this difficulty is to use Perfectly Matched Layers (PMLs). The PML is a fictitious medium that is placed on the outside of the region of interest in order to absorb incoming energy without reflection.

Taking into account the simplifying assumptions discussed in the previous paragraph, and including a PDE model for the PMLs, one can obtain (see [2] for details) the following system

of PDEs:

$$\partial_t D_y^* + \frac{\hat{\sigma}(x)}{\epsilon_0} D_y^* = c_0 (\partial_x H_z - \partial_z H_x), \quad (9)$$

$$\partial_t D_y + \frac{\hat{\sigma}(z)}{\epsilon_0} D_y = \partial_t D_y^*, \quad (10)$$

$$\partial_t H_x^* + \frac{\hat{\sigma}(z)}{\epsilon_0} H_x^* = c_0 \partial_z E_y, \quad (11)$$

$$\partial_t H_x = \partial_t H_x^* + \frac{\hat{\sigma}(x)}{\epsilon_0} H_x^*, \quad (12)$$

$$\partial_t H_z^* + \frac{\hat{\sigma}(x)}{\epsilon_0} H_z^* = -c_0 \partial_x E_y, \quad (13)$$

$$\partial_t H_z = \partial_t H_z^* + \frac{\hat{\sigma}(z)}{\epsilon_0} H_z^*. \quad (14)$$

The quantities D_y^* , H_x^* , and H_z^* in (9)-(14) are defined by equations (10), (12), and (14) respectively. The function $\hat{\sigma}(z) = 0$ outside of the PML. Inside of the PML, $\hat{\sigma}(z)$ is a smooth function of z that increases from a value of zero at the free-space/PML interface to a value of σ_{max} at the boundary. The function $\hat{\sigma}(x)$ is defined similarly. Outside of the PML regions $\hat{\sigma}(x) = \hat{\sigma}(z) = 0$.

The quantity D_y in (10) is given by

$$D_y(t) = \epsilon_\infty E_y(t) + P_y(t) + C_y(t), \quad (15)$$

where the polarization P_y satisfies

$$\tau \dot{P}_y + P_y = (\epsilon_s - \epsilon_\infty) E_y, \quad (16)$$

and the conductivity term C_y satisfies

$$\dot{C}_y = (\sigma/\epsilon_0) E_y \quad (17)$$

inside of the dielectric. Outside of the dielectric, $C_y = P_y = 0$. Note that in order to simplify the implementation of the numerical algorithm, E_y was rescaled by a constant. This is not reflected in any notation change in (9)-(14), and accounts for the change in form from (8) to (16). Also, note that D_y in (15) is *not* the y component of the displacement vector \mathbf{D} of equation (7). It is, rather, the y component of the rescaled displacement vector that results from the rescaling of E_y .

Equations (9)-(16) are the full set of equations that we solve numerically. We use finite differences. In particular, we use the FDTD algorithm [10, 11], which uses forward differences to approximate the time and spatial derivatives. For implementation details, see [2].

In order to attempt the parameter identification inverse problem, we must collect data within the computational domain at a pre-specified sampling rate. We collect data at the center of the antenna at each time step. Then, our data consists of the set $\{E_y(i\Delta t, 0, \bar{z}, \mathbf{q})\}_{i=1}^{N_t}$, where E_y is the solution of Maxwell's equations given by the FDTD algorithm, $\mathbf{q} = (\sigma, \tau, \epsilon_s, \epsilon_\infty)$ is the set of parameters that characterize the Debye medium, and N_t is the total number of time

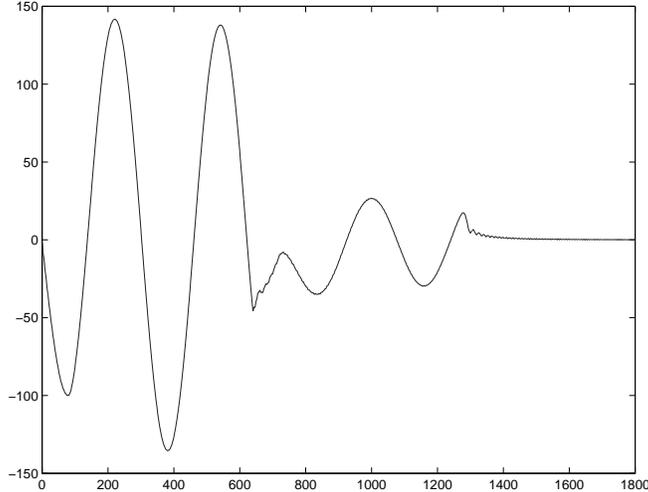


Figure 2: Data for the Debye model for water. The vertical axis gives the magnitude of the electric field. The horizontal axis gives the time in the units Δt seconds.

steps. Notice that we have now made explicit the dependence of solutions of (9)-(16) on the parameter set \mathbf{q} . We will continue to use this convention in the sequel. For the discretization mentioned above, we take $N_t = 1800$, in which case the data includes both the outgoing energy and the energy that is reflected off of the free-space/dielectric interface.

In [5], it is noted that the polarization behavior of water is reasonably modelled by (15), (17), and (16) with parameter values:

$$\begin{aligned}
 \sigma &= 1 \times 10^{-5} \text{ mhos/meter,} \\
 \tau &= 8.1 \times 10^{-12} \text{ seconds,} \\
 \epsilon_s &= 80.1 \text{ relative static permittivity,} \\
 \epsilon_\infty &= 5.5 \text{ relative high frequency permittivity.}
 \end{aligned} \tag{18}$$

With this information, numerical approximations to (9)-(16) can be obtained. The resulting data set $\{E_y(i\Delta t, 0, \bar{z}, \mathbf{q})\}_{i=1}^{N_t}$ is plotted in Figure 2. In [2], the formation of Brillouin precursors [1] is noted within the dielectric, suggesting that the FDTD solutions are accurately capturing the dispersivity of the Debye medium.

2.2 Inverse Problem

We now present the parameter identification inverse problem of interest. In the previous section, we focused our attention on the problem of solving Maxwell’s equations in the presence of a Debye polarization model. Implicit in this problem is the knowledge of the “true” parameter vector, which we will denote \mathbf{q}^* , and which is given by (18). In the inverse problem, on the other hand, we have measurements of the electric field at the point $(0, \bar{z})$ in the computational domain at uniform intervals of time t_i . Given this data, our goal is to then identify, or reconstruct, the “true” parameter vector \mathbf{q}^* .

In the laboratory setting, noise enters into the problem both through the resistor, which generates the electric pulse, and through the antenna/receiver, which reads electric field intensities at the point $(0, \bar{z})$ at discrete time steps. The noise in the data is well-approximated by a zero mean, additive Gaussian random variable with a constant variance across time (see [2] for details). Thus our data vector

$$\mathbf{E}_y^{data} = (E_{y,1}^{data}, \dots, E_{y,N_t}^{data}) \quad (19)$$

is a random vector with components

$$E_{y,i}^{data} \sim N(E_y(t_i, 0, \bar{z}, \mathbf{q}^*), \sigma^2) \quad \text{for } i = 1, \dots, N_t. \quad (20)$$

To generate synthetic, noisy data, we use the FDTD method to obtain the set $\{E_y(t_i, 0, \bar{z}, \mathbf{q}^*)\}_{i=1}^{N_t}$, together with noise model (20). We then seek the maximum likelihood estimator (MLE) of the true parameter vector \mathbf{q}^* given by (18). Assuming that \mathbf{d} is a realization from the random vector (19), (20), the MLE $\hat{\mathbf{q}}$ for \mathbf{q}^* is given by

$$\hat{\mathbf{q}} = \arg \min_{\mathbf{q} \in Q} f(\mathbf{q}), \quad \text{where} \quad f(\mathbf{q}) = \frac{1}{2} \|\mathbf{E}_y(0, \bar{z}, \mathbf{q}) - \mathbf{d}\|_2^2, \quad (21)$$

where $\mathbf{E}_y(0, \bar{z}, \mathbf{q}) = (E_y(0, \bar{z}, t_1, \mathbf{q}), \dots, E_y(0, \bar{z}, t_{N_t}, \mathbf{q}))$, and Q is the constraint set for the parameter values.

Problem (21) is a constrained, nonlinear least squares problem. A standard algorithm for solving such a problem *without* constraints is the Levenburg-Marquardt (LM) algorithm. This algorithm is very effective on a wide range of applied problems. In the next section, we present a straightforward modification of the LM algorithm for bound constrained problems. Note that the constraints on the parameters in our problem consist only of lower bounds.

3 The Optimization Algorithm

We begin by first establishing some notation. We express f in (21) as

$$f(\mathbf{q}) = \frac{1}{2} \sum_{j=1}^n r_j^2(\mathbf{q}),$$

where $r_j(\mathbf{q}) := (E_y(t_j, 0, \bar{z}, \mathbf{q}) - d_j)^2$ for $j = 1, \dots, n$. We define the vector valued function \mathbf{r} by

$$\mathbf{r}(\mathbf{q}) = (r_1(\mathbf{q}), r_2(\mathbf{q}), \dots, r_n(\mathbf{q})).$$

In order to discuss the algorithm in a more general setting, we assume that the parameter vector \mathbf{q} is a vector with m components; that is, $\mathbf{q} = (q_1, q_2, \dots, q_m)$. The Jacobian of \mathbf{r} is then given by

$$J(\mathbf{q}) = \begin{bmatrix} \frac{\partial r_j}{\partial q_i} \end{bmatrix} \begin{matrix} j = 1, \dots, n \\ i = 1, \dots, m \end{matrix}.$$

The gradient and Hessian of f can then be defined using J as follows (see [9]):

$$\text{grad } f(\mathbf{q}) = J(\mathbf{q})^T \mathbf{r}(\mathbf{q}), \quad (22)$$

$$\text{Hess } f(\mathbf{q}) = J(\mathbf{q})^T J(\mathbf{q}) + \sum_{j=1}^m r_j(\mathbf{q}) \text{Hess } r_j(\mathbf{q}). \quad (23)$$

From (22), (23) we see that once we have obtained $J(\mathbf{q})$, we can easily compute both the gradient of f and the first term in (23), i.e. $J(\mathbf{q})^T J(\mathbf{q})$, which is known as the Gauss-Newton (GN) approximation of the Hessian. The GN approximation will be accurate provided the second term in (23) is small compared to the first term. This will occur if the residuals are small, i.e. $r_j(\mathbf{q}) \approx 0$ for each j ; or if f is approximately linear, in which case $\text{Hess } r_j(\mathbf{q})$ will be approximately equal to the zero matrix for each j .

Before continuing, we will state two definitions and two theorems. First, we define what it means to be a *local solution* of (21).

Definition 3.1 *The vector $\bar{\mathbf{q}} \in Q$ is a local solution of (21) if there exists a neighborhood \mathcal{U} containing $\bar{\mathbf{q}}$ such that $f(\bar{\mathbf{q}}) \leq f(\mathbf{q})$ for all $\mathbf{q} \in \mathcal{U} \cap Q$.*

When $Q = \mathbb{R}^m$, a necessary condition for $\bar{\mathbf{q}}$ to be a local solution of f is given in the following theorem.

Theorem 3.2 *If $\bar{\mathbf{q}} \in Q$ is a local solution of (21) with $Q = \mathbb{R}^m$, then $\text{grad } f(\bar{\mathbf{q}}) = \mathbf{0}$.*

This well known result will be used to determine stopping criteria for our unconstrained method below, and the analogous result for the bound constrained case will be stated when we present the constrained method.

Since it is our objective to solve a minimization problem, it is important that we are explicit about what it means to be moving in a direction in which (at least locally) the value of our function f decreases. Such a direction is known as a *descent direction* and is defined as follows:

Definition 3.3 *The vector $\mathbf{p} \in \mathbb{R}^m$ is a descent direction for f at $\mathbf{q} \in \mathbb{R}^m$ if there exists $\bar{\alpha} > 0$ such that $f(\mathbf{q} + \alpha \mathbf{p}) < f(\mathbf{q})$ for $0 < \alpha \leq \bar{\alpha}$.*

A practical check for determining whether or not a vector \mathbf{p} is a descent direction for f at \mathbf{q} is given by the following theorem.

Theorem 3.4 *A vector $\mathbf{p} \in \mathbb{R}^m$ is a descent direction for $f : \mathbb{R}^m \rightarrow \mathbb{R}$ at $\mathbf{q} \in \mathbb{R}^m$ if and only if $\mathbf{p}^T \text{grad } f(\mathbf{q}) < 0$.*

The constrained minimization algorithm that we will present for solving (21) is a modification of the Levenburg-Marquardt algorithm for unconstrained, nonlinear least squares problems. For this reason, we begin in the next section by presenting a specific implementation of an unconstrained Levenburg-Marquardt algorithm. We then adapt this algorithm for use on bound constrained problems.

3.1 A Levenburg-Marquardt Algorithm

In order to simplify notation, in the sequel we will use the notation $f_k = f(\mathbf{q}_k)$, $g_k = \text{grad } f(\mathbf{q}_k)$, $J_k = J(\mathbf{q}_k)$, and $\mathbf{r}_k = \mathbf{r}(\mathbf{q}_k)$. The LM algorithm generates a sequence of approximate solutions of (21) when $Q = \mathbb{R}^m$ as follows:

Algorithm 1: Given the k th approximate solution \mathbf{q}_k of (21), the trust-region radius Δ_k and the positive definite, diagonal scaling matrix D_k , we generate \mathbf{q}_{k+1} , Δ_k , and D_k as follows:

1. Approximately solve the constrained least squares problem

$$\mathbf{p}_k = \arg \min_{\mathbf{p}} \|J_k \mathbf{p} + \mathbf{r}_k\|^2 \quad \text{subject to} \quad \|D_k \mathbf{p}\| \leq \Delta_k; \quad (24)$$

2. If $\mathbf{q}_k + \mathbf{p}_k$ yields a sufficient decrease in the value of the function f , set $\mathbf{q}_{k+1} = \mathbf{q}_k + \mathbf{p}_k$; Otherwise, set $\mathbf{q}_{k+1} = \mathbf{q}_k$;
3. Update Δ_k and D_k .

We will assume, until stated otherwise, that $n \geq m$ and that J_k has full column rank for all k . The important consequence of this assumption is that $J_k^T J_k$ is then positive definite and is therefore invertible. In this case, the unconstrained minimizer of $\|J_k \mathbf{p} + \mathbf{r}_k\|^2$ is also the solution of the linear system

$$J_k^T J_k \mathbf{p}_k = -J_k^T \mathbf{r}_k. \quad (25)$$

We will denote this choice of search direction by \mathbf{p}_k^{GN} , since it is the search direction used in the well-known Gauss-Newton method [9]. The following argument shows that \mathbf{p}_k^{GN} is a descent direction:

$$(\mathbf{p}_k^{GN})^T g_k = (\mathbf{p}_k^{GN})^T J_k^T \mathbf{r}_k = -(\mathbf{p}_k^{GN})^T J_k^T J_k \mathbf{p}_k^{GN} < 0. \quad (26)$$

The first equality in (26) follows from (22), the second from (25), and the final inequality is due to the fact that $J_k^T J_k$ is positive definite. Hence, by Theorem 3.4, \mathbf{p}_k^{GN} is a descent direction.

It is a straightforward exercise to show that minimizing $\|J_k \mathbf{p} + \mathbf{r}_k\|^2$ with respect to \mathbf{p} is equivalent to minimizing

$$m_k(\mathbf{p}) = f_k + \mathbf{p}^T J_k^T \mathbf{r}_k + \frac{1}{2} \mathbf{p}^T J_k^T J_k \mathbf{p} \quad (27)$$

with respect to \mathbf{p} . Hence $\|J_k \mathbf{p} + \mathbf{r}_k\|^2$ in (24) can be replaced by $m_k(\mathbf{p})$. Notice that the quadratic function (27) is the quadratic Taylor approximation of f at \mathbf{q}_k with $\text{Hess } f(\mathbf{q}_k)$ replaced by the GN approximation $J_k^T J_k$. We can therefore view LM as solving a sequence of constrained, quadratic minimization problems in which the quadratic function is a local approximation of the objective function f .

We will now be more explicit about the methodology used in each of the steps in Algorithm 1. We begin with Step 1.

3.1.1 Step 1: Compute \mathbf{p}_k .

Step 1 of Algorithm 1 is known as the trust-region subproblem and can be approximately solved using a variety of techniques (see [7, 9]). The specifics of a particular application may determine which approach is best. One of the most straightforward approaches, and one that is effective for our application is the *dogleg method*.

Before presenting the method, we establish some notation. We define $\bar{\mathbf{p}}_k$ to be the unconstrained minimizer of $m_k(\mathbf{p})$ for \mathbf{p} restricted to the path $\mathbf{p} = -\alpha g_k$ for $\alpha > 0$. Then

$$\bar{\mathbf{p}}_k = -\frac{g_k^T g_k}{g_k^T J_k^T J_k g_k} g_k. \quad (28)$$

As above, we define \mathbf{p}_k^{GN} to be the solution of (25). Finally, we define the dogleg path $\tilde{\mathbf{p}}_k(t)$ for $t \in [0, 2]$ by

$$\tilde{\mathbf{p}}_k(t) = \begin{cases} t\bar{\mathbf{p}}_k, & 0 \leq t \leq 1, \\ \bar{\mathbf{p}}_k + (t-1)(\mathbf{p}_k^{GN} - \bar{\mathbf{p}}_k), & 1 \leq t \leq 2. \end{cases} \quad (29)$$

The *dogleg method* chooses the point \mathbf{p}_k that minimizes $m_k(\mathbf{p})$ along this path, subject to the norm constraint given in (24). The results of the following lemma will suggest an algorithmic scheme for the dogleg method. The proof of this lemma requires a minor modification of the proof of Lemma 4.1 found in [9] and is therefore omitted.

Lemma 3.5 *Let $J_k^T J_k$ be positive definite. Then $\|D_k \tilde{\mathbf{p}}_k(t)\|$ is an increasing function of t , and $m_k(\tilde{\mathbf{p}}_k(t))$ is a decreasing function of t .*

The fact that $\|D_k \tilde{\mathbf{p}}_k(t)\|$ is an increasing function of t implies that the dogleg path intersects the trust-region boundary $\|D_k \mathbf{p}\| = \Delta_k$ at exactly one point provided $\|D_k \mathbf{p}_k^{GN}\| \geq \Delta_k$ and nowhere otherwise. This, together with the fact that $m_k(\tilde{\mathbf{p}}_k(t))$ is a decreasing function of t , implies that the following algorithm for executing the dogleg method is well-defined.

```

if  $\|D_k \mathbf{p}_k^{GN}\| \leq \Delta_k$ 
     $\mathbf{p}_k = \mathbf{p}_k^{GN}$ ;
else if  $\|D_k \bar{\mathbf{p}}_k\| \geq \Delta_k$ 
     $\mathbf{p}_k = \tilde{\mathbf{p}}(t^*) = t^* \bar{\mathbf{p}}_k$ , where  $t^* = \Delta_k / \|D_k \bar{\mathbf{p}}_k\|$ ;
else
     $\mathbf{p}_k = \tilde{\mathbf{p}}(t^*) = \bar{\mathbf{p}}_k + (t^* - 1)(\mathbf{p}_k^{GN} - \bar{\mathbf{p}}_k)$ , where  $t^*$ 
    solves  $\|D_k(\bar{\mathbf{p}}_k + (t-1)(\mathbf{p}_k^{GN} - \bar{\mathbf{p}}_k))\|^2 = \Delta_k^2$ .
end if

```

3.1.2 Steps 2 and 3: Update \mathbf{q}_k and Δ_k

Once an approximate solution \mathbf{p}_k of (24) is obtained in Step 1, two tests are performed. The first test determines the new iterate \mathbf{q}_{k+1} (Step 2) while the second determines the new trust region radius Δ_{k+1} (Step 3). Both tests require the following parameter:

$$\rho_k = \frac{f_k - f(\mathbf{q}_k + \mathbf{p}_k)}{m_k(\mathbf{0}) - m_k(\mathbf{p}_k)}. \quad (30)$$

The values \mathbf{q}_{k+1} and Δ_{k+1} are then determined by the value of ρ_k .

We now present a specific LM algorithm with the general form of Algorithm 1 that uses the methodology discussed in this and in the previous section.

3.1.3 Algorithm 1 Revisited

Given $\Delta_{max} > 0$, $\Delta_0 \in (0, \Delta_{max})$, $\eta_1 \in (0, \frac{1}{4})$, $\eta_1 < \eta_2 < 1$, and $\alpha_1 < 1 < \alpha_2$:

for $k = 0, 1, 2, \dots$

 Obtain \mathbf{p}_k via the dogleg algorithm of Section 3.1.1;

 Evaluate ρ_k from (30);

if $\rho_k > \eta_1$

$\mathbf{q}_{k+1} = \mathbf{q}_k + \mathbf{p}_k$,

else

$\mathbf{q}_{k+1} = \mathbf{q}_k$.

if $\rho_k > \eta_2$

$\Delta_{k+1} = \min(\alpha_2 \Delta_k, \Delta_{max})$,

else if $\rho_k > \eta_1$

$\Delta_{k+1} = \Delta_k$,

else

$\Delta_{k+1} = \alpha_1 \Delta_k$.

end (for)

Motivated by Theorem 3.2, we stop the algorithm once $\|g_k\| \leq \gamma$, where γ is some small positive number.

Remark: Provided certain conditions on the function f hold, Theorem 4.8 in [9] guarantees global convergence for this algorithm. That is, regardless of the choice of \mathbf{q}_0 , the above algorithm will generate a sequence $\{\mathbf{q}_k\}$ that converges to a local solution of (21).

3.2 A Bound-Constrained Levenburg-Marquardt Algorithm

We now address the question of how to adapt the LM algorithm of Section 3.1 for use on bound-constrained problems. Before we begin this discussion, some preliminary tools are needed.

3.2.1 Preliminaries

The projection of a vector $\mathbf{q} \in \mathbb{R}^M$ onto the feasible set Q is given by

$$\mathcal{P}_Q(\mathbf{q}) \stackrel{\text{def}}{=} \arg \min_{\mathbf{v} \in Q} \|\mathbf{v} - \mathbf{q}\| = \max\{\min\{\mathbf{q}, \mathbf{u}\}, \mathbf{l}\},$$

where \mathbf{l} and \mathbf{u} are the lower and upper bound vectors respectively for the constraint set Q ; that is, Q is defined

$$Q = \{\mathbf{q} \in \mathbb{R}^m \mid l_i \leq q_i \leq u_i, i = 1, \dots, m\}; \quad (31)$$

$\max\{\mathbf{v}, \mathbf{l}\}$ is the vector whose i th component is $\max\{v_i, l_i\}$; and $\min\{\mathbf{v}, \mathbf{u}\}$ is the vector whose i th component is $\min\{v_i, u_i\}$.

The active set for a vector $\mathbf{q} \in Q$ is given by

$$\mathcal{A}(\mathbf{q}) = \{i \mid q_i = l_i \text{ or } q_i = u_i\}. \quad (32)$$

The complementary set of indices is called the inactive set and is denoted by $\mathcal{I}(\mathbf{q})$. The inactive, or free, variables consist of the components q_i for which the index i is in the inactive set.

We now define the *projected gradient*, which can be viewed as the analogue of the gradient in the constrained setting. The projected gradient of f at $\mathbf{q} \in Q$ is the m -vector with components

$$[\text{grad}_{\mathcal{P}} f(\mathbf{q})]_i = \begin{cases} \frac{\partial f(\mathbf{q})}{\partial q_i}, & i \in \mathcal{I}(\mathbf{q}) \text{ or } (i \in \mathcal{A}(\mathbf{q}) \text{ and } \frac{\partial f(\mathbf{q})}{\partial q_i} < 0) \\ 0, & \text{otherwise.} \end{cases} \quad (33)$$

The following theorem relates the local solutions of (21) with the projected gradient.

Theorem 3.6 *If $\bar{\mathbf{q}} \in Q$ is a local solution of (21) with Q defined by (31), then $\text{grad}_{\mathcal{P}} f(\bar{\mathbf{q}}) = \mathbf{0}$.*

It is important to note the similarity between this result and Theorem 3.2, but also that $\text{grad}_{\mathcal{P}} f(\mathbf{q}) = \mathbf{0}$ does not imply that $\text{grad} f(\mathbf{q}) = \mathbf{0}$. With the result of Theorem (3.6) in mind, it is not surprising that for many constrained algorithms, the projected gradient plays the same role that the gradient plays in the analogous algorithms for unconstrained problems. We will see that this is the case for our constrained algorithm as well.

Finally, we define the *reduced Hessian* to be

$$[\text{Hess}_R f(\mathbf{q})]_{ij} = \begin{cases} \frac{\partial^2 f(\mathbf{q})}{\partial q_i \partial q_j}, & \text{if } i \in \mathcal{I}(\mathbf{q}) \text{ or } j \in \mathcal{I}(\mathbf{q}) \\ \delta_{ij}, & \text{otherwise.} \end{cases}$$

Let $D_{\mathcal{I}}$ denote the diagonal matrix with components

$$[D_{\mathcal{I}}(\mathbf{q})]_{ii} = \begin{cases} 1, & i \in \mathcal{I}(\mathbf{q}) \\ 0, & i \in \mathcal{A}(\mathbf{q}). \end{cases} \quad (34)$$

Then

$$\text{Hess}_R f(\mathbf{q}) = D_{\mathcal{I}}(\mathbf{q}) \text{Hess} f(\mathbf{q}) D_{\mathcal{I}}(\mathbf{q}) + D_{\mathcal{A}}(\mathbf{q}), \quad (35)$$

where $D_{\mathcal{A}}(\mathbf{q}) = I - D_{\mathcal{I}}(\mathbf{q})$.

3.2.2 The Algorithm

We now present the modifications of the LM algorithm that enable the incorporation of constraints.

Modification 1: Our first modification is to the quadratic function (27) that is approximately minimized in Step 1 of the algorithm. In (27) we replace $J_k^T J_k$ by

$$A_k = D_{\mathcal{I}}(\mathbf{q}_k) J_k^T J_k D_{\mathcal{I}}(\mathbf{q}_k) + D_{\mathcal{A}}(\mathbf{q}_k)$$

and g_k by $\bar{g}_k = \text{grad}_{\mathcal{P}} f(\mathbf{q}_k)$. Thus, m_k is replaced by

$$\bar{m}_k(\mathbf{p}) = f_k + \mathbf{p}^T \bar{g}_k + \mathbf{p}^T A_k \mathbf{p}. \quad (36)$$

Note that the matrix A_k is the Gauss-Newton approximation of the reduced Hessian $\text{Hess}_R f(\mathbf{q}_k)$ defined by (35).

These modifications to m_k result in corresponding changes in the dogleg path defined by (29). In particular, (28) in (29) is replaced by

$$\bar{\mathbf{p}}_k = -\frac{g_k^T \bar{g}_k}{\bar{g}_k^T A_k \bar{g}_k} \bar{g}_k, \quad (37)$$

and \mathbf{p}_k^{GN} is replaced by the solution of the linear system

$$A_k \mathbf{p} = -\bar{g}_k. \quad (38)$$

Because we have assumed that $J_k^T J_k$ is positive definite, A_k is also positive definite, and hence, (38) has a solution, which we will denote $\bar{\mathbf{p}}_k^{GN}$. An argument analogous to (26) shows that $\bar{\mathbf{p}}_k^{GN}$ is a descent direction.

The results of Lemma 3.5 will hold for \bar{m}_k . This is due to the fact that the proof of the lemma only required that $J_k^T J_k$ be positive definite. Thus, since A_k is positive definite, no change in the proof is required. In addition, replacing g_k by \bar{g}_k results in no modifications of the proof. Hence, if $\tilde{\mathbf{p}}_k(t)$ is our dogleg path, $\|D_k \tilde{\mathbf{p}}_k(t)\|$ is an increasing function of t , and $\bar{m}_k(\tilde{\mathbf{p}}_k(t))$ is a decreasing function of t . Consequently, the dogleg algorithm found in Section 3.1.1 can be used with the modified dogleg path defined above, and it will be well-defined.

Modification 2: The step generated by the dogleg method using \bar{m}_k may take us outside of the constraints. To remedy this problem, once the dogleg solution \mathbf{p}_k is obtained, we project the new step onto the constraint set Q . Then our new step \mathbf{p}_k is given by

$$\mathbf{p}_k = \mathcal{P}_Q(\mathbf{q}_k + \bar{\mathbf{p}}_k) - \mathbf{q}_k. \quad (39)$$

Modification 3: Finally, we accept or reject our step and update our trust region radius Δ_k following the frame work of the algorithm in Section 3.2.2, with the modification that m_k replaced by \bar{m}_k in the definition of ρ_k given by equation (30).

The primary modification that we have made is the replacement of m_k by \bar{m}_k . This occurs in both Modification 1 and 3. We can motivate this choice in several ways. First, both $\bar{\mathbf{p}}_k$ and $\bar{\mathbf{p}}_k^{GN}$ are *feasible* search directions; that is $\mathbf{q}_k + \alpha \bar{\mathbf{p}}_k$ and $\mathbf{q}_k + \alpha \bar{\mathbf{p}}_k^{GN}$ are in Q for α small and positive. This is not the case in general if $\bar{\mathbf{p}}_k$ is defined as in (28) and if $\bar{\mathbf{p}}_k^{GN}$ is replaced by \mathbf{p}_k^{GN} .

Secondly, using only the projected gradient as a search direction with either a trust-region or line search globalization scheme and (39) to enforce constraints, global convergence results can be attained for problems of the form (21) [6, 8]. Consequently, one expects, with the reduction in \bar{m}_k along the dogleg path from $\bar{\mathbf{p}}_k$ to $\bar{\mathbf{p}}_k^{GN}$, that in using the dogleg method, these global converge properties will be retained, while the resulting algorithm will have a much improved convergence rate.

Finally, our choice of \bar{m}_k can be motivated by the following result, which is found in [6].

Theorem 3.7 *Let $f : \mathbb{R}^m \rightarrow \mathbb{R}$ be continuously differentiable on Q , and let $\{\mathbf{q}_k\}$ be an arbitrary sequence in Q which converges to $\hat{\mathbf{q}}$. If $\{\text{grad}_{\mathcal{P}} f(\mathbf{q}_k)\}$ converges to zero and $\hat{\mathbf{q}}$ is nondegenerate, then there exists a positive number K such that $\mathcal{A}(\mathbf{q}_k) = \mathcal{A}(\hat{\mathbf{q}})$ for all $k \geq K$.*

(Note that $\hat{\mathbf{q}}$ is *nondegenerate* if $[\text{grad } f(\hat{\mathbf{q}})]_i \neq 0$ whenever $i \in \mathcal{A}(\hat{\mathbf{q}})$.) From this result we see that if our algorithm constructs a sequence that converges to a nondegenerate local solution of (21) then eventually the active set at the solution is identified. Once this occurs, the problem is no longer constrained, and our choice of \bar{m}_k guarantees that our algorithm reduces to the unconstrained Levenburg-Marquardt algorithm presented in Section 3.1, which is desirable.

Remark: It is important to note that the changes discussed above are independent of the choice of the algorithm that is used for approximately solving the trust-region subproblem. In this paper, we use the dogleg method, but other methods (see [7, 9]) can be used as well. Regardless of the choice that is made, the above modifications will yield a viable constrained LM algorithm.

In addition, we mention that a line search globalization scheme can also be used in which the search direction at each iteration is chosen to be the solution of (38). This approach follows the framework found in [4].

4 Numerical Results

In this section, we apply the bound constrained Levenburg-Marquardt algorithm from Section 3.2 to the problem of solving the bound constrained minimization problem (21). The constraint set Q is determined by the following physical constraints on the parameters: $\sigma \geq 0$, $\tau \geq 0$, $\epsilon_s \geq 1$, and $\epsilon_\infty \geq 1$. Notice that there is no upper-bound vector. The data \mathbf{d} is a realization of the random vector (19), (20) with $\{E_y(t_i, 0, \bar{z}, \mathbf{q}^*)\}_{i=1}^{N_t}$ gotten via solving Maxwell's equations in the fashion discussed in Section 2.1 with the "true" parameter vector \mathbf{q}^* defined by (18). We choose σ^2 in (20) so that the ℓ^2 -norm of the noise taken componentwise is 2% of the norm of the signal without noise. We take an initial guess $\mathbf{q}_0 = (\sigma_0, \tau_0, \epsilon_{s,0}, \epsilon_{\infty,0})$ given by $\sigma_0 = 1.5 \times 10^{-5}$, $\tau_0 = 10.0 \times 10^{-12}$, $\epsilon_{s,0} = 73.1$, and $\epsilon_{\infty,0} = 6.0$. (These range between 50% relative to 10% relative error from the true values and are typical of values used in testing algorithms [3].). The parameters that are necessary for the implementation of our algorithm (see Section 3.2.2) are chosen to be $\alpha_1 = 0.25$, $\alpha_2 = 2$, $\eta_1 = 0.25$, and $\eta_2 = 0.75$, $\Delta_0 = 1$, and $\Delta_{max} = 1000$.

The results are given in Figure 3. We see that convergence is attained. The flat spots in the convergence plot are the result of iterations in which the step \mathbf{p}_k computed in the dogleg step was not accepted. The value of the reconstructed parameters at the end of the iteration was $\tilde{\sigma} = 0$, $\tilde{\tau} = 9.989 \times 10^{-12}$, $\tilde{\epsilon}_s = 80.01$, and $\tilde{\epsilon}_\infty = 10.66$. Note that the component of \mathbf{q} corresponding to σ is active at the approximate solution. It is important to mention at this point that initially the unconstrained Levenburg-Marquardt algorithm discussed in Section 3.1 was used on this problem, and the values of the reconstructed parameters were in most cases non-physical for σ and were often non-physical for ϵ_∞ . A constrained algorithm was therefore a necessity.

Also note that, relatively speaking, the only parameters that are accurately reconstructed are ϵ_s and τ . In [2], the sensitivity of the cost function f in (21) to each of the four parameters is

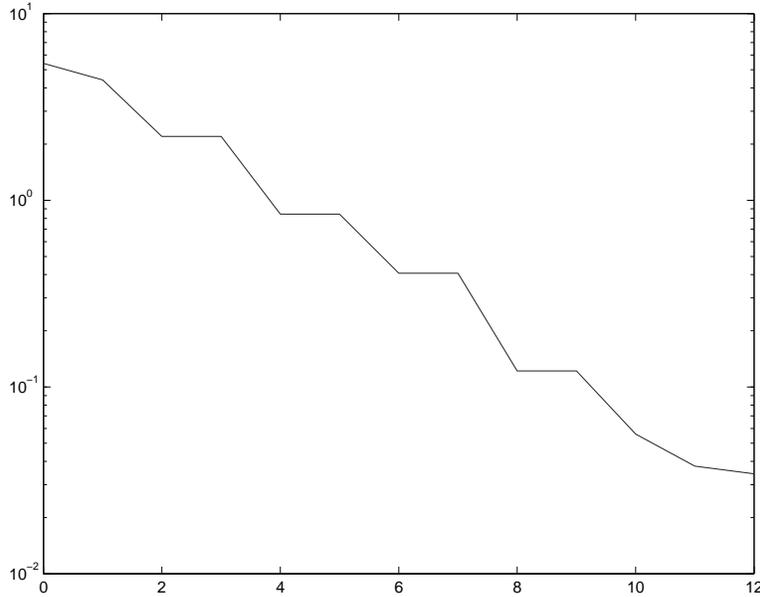


Figure 3: Norm of the Projected Gradient of f versus Iteration Count. The horizontal axis denotes iteration count k . The vertical axis denotes the natural log of the projected gradient of f at the current iterate \mathbf{q}_k .

discussed in detail and is lengthy. We will, therefore, not reproduce it in its entirety here. One important conclusion that was made, via a frequency domain analysis, was that for values of the parameters of the same order of magnitude as the true values, the solutions of the corresponding Maxwell systems are most sensitive to ϵ_s , and hence, that only accurate reconstructions of ϵ_s can be expected. This is supported by the results from the attempts at solving the inverse problem found in [2]. In fact, in other attempts, the reconstructed value of τ^* is not as accurate as it was in this case.

5 Conclusions

We present an inverse problem that arises from an application in electromagnetics. In particular, we seek to reconstruct the parameters that characterize a Debye medium, e.g. water, via the inversion of scattered, electric field data. The methodology used for solving Maxwell's equation is discussed briefly. The parameter identification inverse problem has a nonlinear least squares form with bound constraints.

After an unconstrained Levenburg-Marquardt algorithm is presented, straightforward modifications are discussed for use on bound-constrained nonlinear least squares problems. A convergence analysis of the resulting algorithm is given. In addition, it is shown that the bound-constrained algorithm reduces to the original, unconstrained Levenburg-Marquardt algorithm once the active set at the solution has been identified, which is desirable.

Finally, the algorithm is applied to the inverse problem of interest, and convergence is

attained. The Debye parameters that characterize the dispersive effects of water are used as the true parameters.

6 Acknowledgements

I would like to thank Dr. H. T. Banks of North Carolina State University for providing me with the opportunity to work on this project. This research was supported in part by the U.S. Air Force Office of Scientific Research under grant AFOSR F49620-01-1-0026 and in part by the National Science Foundation under grant DMS-0112069 to the Statistical and Applied Mathematical Sciences Institute.

References

- [1] Richard Albanese, John Penn, and Richard Medina. *Short-rise-time microwave pulse propagation through dispersive biological media*. J. Optical Society of America A, **6(9)**:1441-1446, 1989.
- [2] H. T. Banks, J. M. Bardsley. *Parameter Identification for a Dispersive Dielectric in 2D Electromagnetics: Forward and Inverse Methodology with Statistical Considerations*. Inverse Problems, submitted.
- [3] H. T. Banks and K. Kunisch. *Estimation Techniques for Distributed Parameter Systems*. Birkhauser, Boston, 1989.
- [4] Dimitri P. Bertsekas. Projected Newton Methods for Optimization Problems with Simple Constraints. SIAM Journal on Control and Optimization, **20(2)**, 1982, pp. 221-246.
- [5] J. G. Blaschak and J. Franzen. *Precursor propagation in dispersive media from short-rise-time pulses at oblique incidence*. Journal of the Optical Society of America A, **12**, 1995, pp. 1501-1512.
- [6] P. H. Calamai and Jorge J. Moré. *Projected Gradient Methods for Linearly Constrained Problems*. Mathematical Programming, **39**, 1987, pp.93-116.
- [7] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust Region Methods*, SIAM, 2000.
- [8] C. T. Kelley, *Iterative Methods for Optimization*, SIAM, Philadelphia, 1999.
- [9] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, 2000.
- [10] Dennis M. Sullivan. *Electromagnetic Simulation using the FDTD Method*. IEEE Press, 2000.
- [11] Allen Taflov and Susan C. Hagness. *Computational Electrodynamics: The Finite Difference Time Domain Method*. Artech House, 2000.