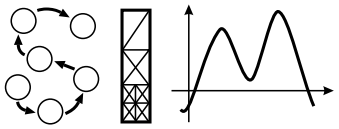
FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN-NÜRNBERG
INSTITUT FÜR INFORMATIK (MATHEMATISCHE MASCHINEN UND DATENVERARBEITUNG)

Lehrstuhl für Informatik 10 (Systemsimulation)



Comparison of Solvers for a Bioelectric Field Problem

Marcus Mohr

Lehrstuhlbericht 01-2

Abstract

The model-based reconstruction of electrical brain activity from electroencephalographic measurements is of constantly growing importance in the fields of Neurology and Neurosurgery. Algorithms for this task involve the solution of a 3D Poisson problem on a complicated geometry and with non-continuous coefficients for a considerable number of different right hand sides. Thus efficient solvers for this subtask are required.

In this paper we will give a very brief introduction into the inverse EEG problem. We will describe a discretisation approach based on cell-centered finite differences and report on our experiences with different iterative solvers for the forward 3D Poisson problem.

1 Introduction

The electroencephalogram (EEG) and the magnetoencephalogram (MEG) are major diagnostical tools to determine the state of the brain. In recent years the model-based analysis of such measurements has substantially gained in importance. The task is to reconstruct from the measured potential and/or magnetic field data either the sources inside the brain responsible for the fields, or to compute the current density inside the brain or on its surface. The results are then used in the planning of brain surgery and even during the surgery itself. Figure 1 shows two applications, one from epilepsy surgery [VHB+98] and one from Neuronavigation [GSK+97]. In the former case an epileptic focus to be removed by surgery was reconstructed and in the latter important brain areas, which must not be hurt during the operation are inserted into the surgeon's microscope. While EEG measurement devices are comparatively cheap to build, easy to use and thus are widespread, this is not the case for MEG devices and our work so far has been restricted to the EEG case.

The reconstruction is performed by numerical simulation based on Maxwell's equations. A major distinction of this problem from other fields of simulation involving partial differential equations is, that the reconstruction is not a classical boundary value problem with prescribed source and boundary terms. Instead it is an inverse problem and the reconstruction of the sources and/or boundary terms is the central task. This inverse problem is ill-posed, thus inevitable measurement errors in the data make a sensible solution impossible, if no auxiliary conditions are posed. This is denoted regularization and usually involves a priori assumptions on the smoothness and local or temporal behaviour of the solution.

As will be explained in the following section all solution approaches to the inverse problem of EEG involve the solution of a considerable number of so called forward problems. A single forward problem consists in computing the potentials at the measurement electrodes for a given source distribution. This constitutes a classical elliptic boundary value problem to be solved on a 3D representation of the patient's head.

Medical applications naturally require a high level of precision. Therefore it is necessary to take the individual patient anatomy into account for the simulation. While attempts so far have mostly employed head models consisting of concentric spheres, which were locally deformed according to the head shape, it is becoming standard nowadays to base the numerical computations on the individual geometry of a patient's head, as can be acquired from magnetic resonance imaging (MRI). The conductivities needed for the computation of the electric field are usually assumed to be constant and isotropic for the different tissues. The values are taken from in vivo and in vitro measurements reported in literature. This constitutes a great simplification since the conductivities can vary from one individual to another and, especially inside the brain, are clearly anisotropic. Therefore research is undertaken to determine patient specific isotropic values in combination with the standard EEG [FT99] or even the full anisotropic information from diffusion weighted MRI [TWD+99]. Thus there is also growing interest in volume based discretisation techniques which are about to replace the traditional boundary element approach that was well suited for the spherical head models. This transition to volume discretised realistic head models is encumbered by the fact that there is still a lack of fast solvers for the forward problem, see e. g. [EMBL01]. While with multigrid methods there exist, at least theoretically, highly efficient solvers for this type of problem, the complicated geometry and the jumping coefficients make their application challenging to some extent.

In this paper we will report on results with different iterative solvers for a forward problem discretised with a cell-centered finite difference scheme. The list of solvers includes Successive-Overrelaxation (SOR) as an example of a stationary iteration method, Conjugate Gradients (CG)

and Preconditioned Conjugate Gradients (PCG) for Krylov subspace methods and a variant of Algebraic Multigrid (AMG).

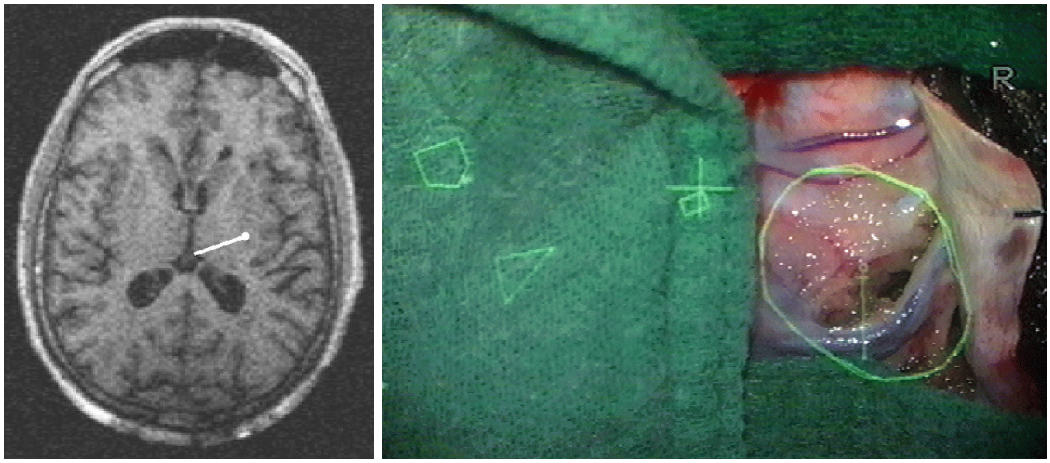


Figure 1: Localization of an epileptic focus from EEG data (left), view through an operation microscope with markers showing functional brain areas determined from MEG data and the contours of a tumor (right).

2 Problem Formulation

This section is intended as a brief introduction into some of the basic concepts of inverse bioelectric field simulation. We begin with a description of the physical model and the derivation of its central equation. This is followed by a short explanation of the three standard approaches for the solution of the inverse EEG problem and an explanation of the two important concepts of lead fields and reciprocity. This section will also demonstrate the need for efficient forward solvers in the inverse context.

Physical Model

In bioelectric field simulation the patient's head is modeled as a volume conductor, i. e. a contiguous, passively conducting medium. Inside the head we have neural current densities which constitute the cause for the electrical and potential field. Due to the temporal behaviour of the sources, which is typically < 1 kHz and the physiological conductivities (e.g. brain 0.2 S/m, skull 0.015 S/m [OD99]) one can assume a quasi-static behaviour. Thus displacement currents can be neglected and the law of Ohm takes the following form

$$I = \sigma E + I_V \quad . \tag{1}$$

Here I represents the total current density, while I_V is the current density of the neural sources. E denotes the electrical field and σ the local conductivity tensor. Since the potential field can be derived from the electrical field via

$$E = -\nabla\Phi \tag{2}$$

and the total current density is divergence free

$$\nabla \cdot I = 0 \tag{3}$$

we can reformulate (1) and arrive at the central equation

$$\nabla \cdot (\sigma \nabla\Phi) = \nabla \cdot I_V \quad . \tag{4}$$

Together with boundary conditions on the current flow through the scalp

$$\sigma \frac{\partial\Phi}{\partial\vec{n}} = g \tag{5}$$

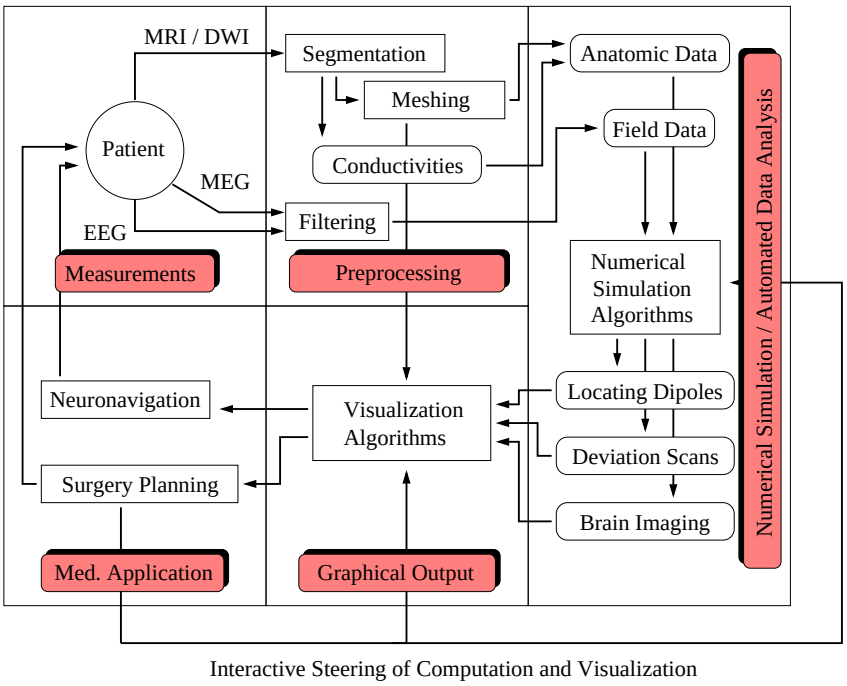


Figure 2: Dataflow in bioelectromagnetic field simulation for clinical application.

this constitutes an elliptic boundary value problem. The classical task of numerical simulation would be to compute the potential field Φ for given source $\nabla \cdot I_V$ and boundary terms g. This is referred to as the *forward problem*. The related *inverse EEG problem* is to reconstruct from g and measurements of Φ at some electrodes either the sources or the potential at some internal interfaces, esp. on the cortex.

Standard Approaches

There are three standard ways to approach the inverse EEG problem. The first is *dipole localization*. Here one tries to reconstruct the source terms from the measurements. Neural centers of activity consist of a large number of neurons at close quarters, usually several thousands, which are synchronously electrically active. It is common to model this in form of dipoles. The easiest model, that can only be applied in the case of very dominant and highly localized activity, is to assume that there is a single center of activity inside the brain that explains the measurements. Thus one searches for a single dipole such that the virtual measurements it creates match the recorded ones best in some norm.

A dipole is given by its position, orientation and strength/momentum. In order to find these free parameters an optimization procedure is employed. This typically is some variant of the downhill simplex method by Nelder and Mead [NM65]. Like most optimization algorithms it works iteratively. It employs a simplex whose vertices are repeatedly moved according to some strategy and that will finally contract around a local minimum. The property that makes the method interesting in the context of dipole localization is that it only employs functional evaluations, but no gradient information or the like. An evaluation of the functional implicates the computation of the potentials at the measurement electrodes for a virtual dipole. Thus the forward problem (4, 5) has to be solved.

Looking for only one dipole as the cause of the measurements leads to an overdetermined system and usually a least squares approach without further regularization is employed. This is different for multi-pole localization since here the illposed-ness of the problem is more significant.

The single-pole localization approach is often performed in concert with the second approach, a so called *deviation scan*. Here for a finite but large number of positions in the brain resp. on the cortex the optimal dipole orientation is computed. The difference between the virtual potentials and the recorded potentials at the electrodes constitutes a measure, how well a single dipole at

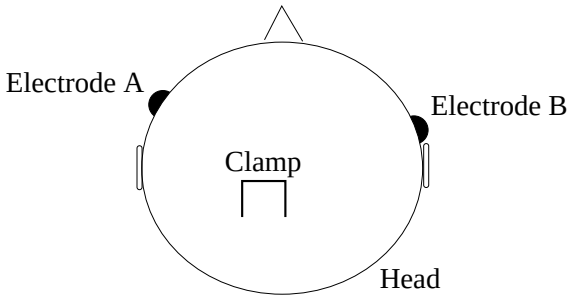


Figure 3: Schematic representation of problem configuration.

the given position will explain the recordings. Thus the deviation scan gives an indication on the suitability of the single dipole model. If there is a single notedly confined region, where it yields a high measure, the single dipole approach is justified. If there are, on the other hand, several regions or the region is large and blurred, a multi-pole model will be more suitable.

The third approach, which is denoted as *brain imaging* or *current density analysis*, consists of reconstructing the potential field in the whole brain or more often only on its surface, the cortex. For this the cortex has to be discretised and the potential be reconstructed at the discretisation points. Since these are usually more numerous than the electrodes, the problem is highly underdetermined. Due to its illposed-ness regularization techniques have to be employed. We refer the reader e. g. to [HH93, Köh98, DSB01, PM99, ML99].

All three approaches involve in one way or the other the setup up a so called *lead field matrix*.

Lead Field Matrix

Assume that a dipole is given with position $\vec{r}$ and orientation $\vec{d}$ and let $\mathcal{E}$ be a set of N electrodes. The vector $M(\vec{r},\vec{d})$ of potentials generated by the dipole at the electrodes in $\mathcal{E}$ is given by

$$M(\vec{r},\vec{d}) = \hat{L}(\vec{r}) \cdot \vec{d} = R \cdot L(\vec{r}) \cdot \vec{d} \tag{6}$$

Here $\hat{L} \in \mathbb{R}^{N\times 3}$ is the so called lead field matrix, that maps the dipole orientation onto the electrode potentials. It can be split into two parts. A matrix $L \in \mathbb{R}^{(N-1)\times 3}$ that maps the dipole orientation onto potential differences between the electrodes and a referencing matrix $R \in \mathbb{R}^{N\times(N-1)}$ that turns these $(N-1)$ differences into N scalar potential values. Each row of L is of the following form

$$(\Phi^x_{\text{AB}}(\vec{r}), \Phi^y_{\text{AB}}(\vec{r}), \Phi^z_{\text{AB}}(\vec{r})) \tag{7}$$

where $\Phi^k_{\text{AB}}(\vec{r})$ denotes the potential difference between electrode A and B induced by a unit dipole at position $\vec{r}$ which is oriented in k-direction. Thus, if we have computed the lead field matrix for the point $\vec{r}$, the solution of the forward problem, i. e. finding the electrode potentials for a dipole $(\vec{r},\vec{d})$, is simply achieved by computing the matrix vector product $\hat{L}(\vec{r}) \cdot \vec{d}$.

The potentials generated by several dipoles are simply a superposition of the potentials generated by the individual dipoles. So, if we allow for an increased number of dipoles, like for multipole localization or current density analysis, we only have to add the corresponding number of columns to the matrix $\hat{L}$.

Reciprocity Theorem

A major part of the computational work in solving the inverse EEG problem goes into the setup of the lead field matrix/matrices. This amount of work can (in some cases) drastically be reduced with the help of the reciprocity theorem of Helmholtz, see e. g. [VHB+98, WZJ00]. Assume that we have a dipole with a fixed position and orientation and two electrodes A and B and let us model the dipole as a small clamp, see Fig. 3. A current I_{clamp} flowing through the clamp will lead to a potential difference Φ_{AB} between the two electrodes. Vice versa, if we let a current I_{AB} flow from one electrode to the other this will result in a potential difference Φ_{clamp} between the ends of the

clamp. According to the reciprocity theorem these values are coupled by

$$\frac{I_{AB}}{\Phi_{\text{clamp}}} = \frac{I_{\text{clamp}}}{\Phi_{AB}} \ . \tag{8}$$

Thus in order to compute the row entries of L that connect a dipole $(\vec{p_1}, \vec{r_1})$ to the potential difference between A and B we can do the following. We compute the potential distribution inside the head if we have a current I_{AB} entering the head at electrode A and leaving at electrode B. Assuming a small clamp at $\vec{p_1}$ that is oriented in x-direction we can determine Φ^x_{clamp} from these values by simple interpolation. Choosing I^x_{clamp} to be a unit current we can compute the resulting potential difference Φ^x_{AB} via

$$\Phi^x_{AB} = \frac{I^x_{\text{clamp}}}{I_{AB}} \Phi^x_{\text{clamp}} \ . \tag{9}$$

We can proceed similarly for Φ^y_{AB} and Φ^z_{AB}. The advantage of this approach is, that, if we have another dipole $(\vec{p_2}, \vec{r_2})$, we can again utilize (9), but we do not have to solve another forward problem.

This suggests to proceed in the following fashion. First we choose $(n-1)$ representative pairs from the set of electrodes. For each such pair we solve a forward problem where one electrode appears as source and the second as sink and store the resulting potential distribution. If we model the source and sink as Dirac delta functions the problem can according to (4, 5) be formulated as

$$\begin{aligned} \nabla \cdot (\sigma \nabla \Phi(x)) &= -I\delta(x_{\text{source}} - x) + I\delta(x_{\text{sink}} - x) \quad , x \in \Omega \\ \sigma \frac{\partial \Phi(x)}{\partial \vec{n}} &= 0 \qquad\qquad\qquad\qquad\qquad\qquad\quad , x \in \partial\Omega \ . \end{aligned} \tag{10}$$

Here I denotes the current flowing between the two electrodes. Now we can setup $\hat{L}$ for an arbitrary number of dipoles or for one dipole at an arbitrary number of positions by a mere interpolation process. This reduces the amount of computational work considerably. Nevertheless the solution of $(N-1)$ forward problems on a realistic head model, especially when N is large, remains a very costly task.

Note that the decision which electrode is source and sink and the strength of the current I_{AB} is unimportant, since the quotient in (8) remains unaffected. So both can be chosen to be most convenient for the numerical computation. The choice of a specific I_{clamp} on the other hand leads to a scaling of the lead field matrix that has to be taken into account in its later use.

3 Discrete Model

In order to solve the boundary value problem (10) in terms of the bioelectric potentials we need to pose the problem in a computationally tractable way. This involves approximating the boundary value problem on a finite dimensional subspace that approximates the solution space, and reformulating it in terms of a linear matrix equation. In this paper we will apply a cell-centered finite difference (FD) formulation, that can also be regarded as a kind of finite volume (FV) approach.

Anatomical Representation

Before one can start the discretisation process, the problem domain Ω and the conductivity tensor σ have to be specified. With current technology it has become feasible to employ the real geometry of an individual patient's head in the setup of Ω. The necessary information is obtained from magnetic resonance imaging (MRI scans). In some cases this is supplemented by computer tomography (CT) in order to improve the determination of the skull. But this is not always possible due to the radiation connected with the CT, see e. g. [Has99]. The data are used to form a 3D model (e.g. a mesh of tetrahedra, or a rectangular cube of voxels). The next step is the segmentation, i.e. tissues with different physical properties have to be identified. The latter can then be used to assign the local conductivities. Today mostly experimental values taken from literature are used in this process and the conductivities are assumed to be constant and isotropic in each tissue compartment. Of course this is a gross simplification. Current is flowing along the dendrites and axons of neurons, which do not form a regular network. E. g. there are a lot more connections from the cortex down

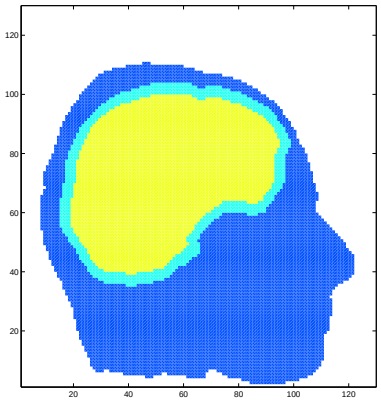


Figure 4: Sagittal slice through a 3-compartment voxel model of a human head.

into the brain than to neighbouring neurons on the cortex. Therefore the conductivity is clearly anisotropic and can vary from subject to subject. However there exist techniques to determine constant, isotropic conductivities at least individually for each patient [FT99]. Soon the technique of diffusion weighted MRI will allow to approximate the locally varying and anisotropic conductivity tensors on a patient basis [TWD$^+$99].

For our experiments we take the following approach. MRI scans return a collection of 2D slices containing pixel information. Thus a simple and obvious approach to get to a volume representation is to basically extend pixels from each 2D slice to 3D voxels. In the segmentation process we assign each such voxel a tissue information. We restrict ourselves to the three compartments scalp, skull and brain, see Fig. 4, and assume constant, isotropic conductivities in each of them. The conductivity of the surrounding air is assumed to be zero. Note however, that the described approach is not limited to a certain number of compartments and in principle can also handle anisotropic conductivities. For details on the creation of the employed 3D voxel model see [Van01] and for a more sophisticated segmentation approach, see e. g. [WRB$^+$01].

Discretisation

The next step is to find a discrete version of (10). Using a derivation similar to finite volume theory, we assume that the approximate solution Φ_h of (10) is a constant Φ_l within each voxel V_l for $l = 1, \ldots, N$. Note that we only want to compute Φ_l for voxels that belong to the head and not to the surrounding air. Using the technique of box integration, see e. g. [MG87], we replace (10) for every voxel V_l with a difference equation in the following way. A solution Φ of (10) must satisfy

$$\int_{V_l} \nabla \cdot \sigma \nabla \Phi = \sum_{k=1}^{3} \int_{V_l} \frac{\partial}{\partial x_k} \left(\sigma \frac{\partial \Phi}{\partial x_k} \right) = -I \int_{V_l} \left(\delta(x_{\text{source}} - x) - \delta(x_{\text{sink}} - x) \right) \tag{11}$$

Assume for simplicity that the edges of V_l are aligned with the coordinate axes, then partial integration gives us

$$\sum_{k=1}^{3} \int_{V_l} \frac{\partial}{\partial x_k} \left(\sigma \frac{\partial \Phi}{\partial x_k} \right) = \sum_{k=1}^{3} \left(\int_{F_{l,k}^+} \sigma \frac{\partial \Phi}{\partial x_k} - \int_{F_{l,k}^-} \sigma \frac{\partial \Phi}{\partial x_k} \right) \tag{12}$$

where $F_{l,k}^+$ and $F_{l,k}^-$ represent the top and bottom face of the voxel V_l in direction x_k. We now approximate the integrals on the right hand side of (12) using the mid-point rule

$$\int_{V_l} \nabla \cdot \sigma \nabla \Phi \approx \sum_{k=1}^{3} \left[A_{l,k}^+ \left(\sigma \frac{\partial \Phi}{\partial x_k} \right) (p_{l,k}^+) - A_{l,k}^- \left(\sigma \frac{\partial \Phi}{\partial x_k} \right) (p_{l,k}^-) \right] \quad . \tag{13}$$

Here $A_{l,k}^{+/-}$ and $p_{l,k}^{+/-}$ denote the surface and center of the face $F_{l,k}^{+/-}$. Since $\sigma \nabla \Phi$ is a continuous function the above formulation is well-defined. What remains is to approximate the values of this

function in the face centers. Let us denote by $V_{l'}$ the voxel that shares the face $F^+_{l,k}$ with V_l, by c_l and $c_{l'}$ the centers of the two cells and by e_k the unit vector in direction x_k. We then have the following two relationships

$$\left(\sigma\frac{\partial\Phi}{\partial x_k}\right)(p^+_{l,k}) = \lim_{h\uparrow 0}\left(\sigma\frac{\partial\Phi}{\partial x_k}\right)(p^+_{l,k}+he_k) \approx \sigma_l\frac{\Phi(p^+_{l,k})-\Phi(c_l)}{h_k/2} \tag{14}$$

$$\left(\sigma\frac{\partial\Phi}{\partial x_k}\right)(p^+_{l,k}) = \lim_{h\downarrow 0}\left(\sigma\frac{\partial\Phi}{\partial x_k}\right)(p^+_{l,k}+he_k) \approx \sigma_{l'}\frac{\Phi(c_{l'})-\Phi(p^+_{l,k})}{h_k/2} \tag{15}$$

From these two equations we can eliminate $\Phi(p^+_{l,k})$ and get that

$$\left(\sigma\frac{\partial\Phi}{\partial x_k}\right)(p^+_{l,k}) \approx \frac{2}{h_k}\frac{\sigma_l\cdot\sigma_{l'}}{\sigma_l+\sigma_{l'}}\left(\Phi(c_{l'})-\Phi(c_l)\right) \tag{16}$$

Proceeding similarly we get that

$$\left(\sigma\frac{\partial\Phi}{\partial x_k}\right)(p^-_{l,k}) \approx \frac{2}{h_k}\frac{\sigma_l\cdot\sigma_{l''}}{\sigma_l+\sigma_{l''}}\left(\Phi(c_{l''})-\Phi(c_l)\right) \tag{17}$$

where l'' denotes quantities in the voxel $V_{l''}$ that shares the face $F^-_{l,k}$ with V_l. We see that the voxel V_l is coupled to its six neighbours via a 7-point-stencil. Let us denote by $\mathcal{N}_l$ the set of indices of these six voxels. Then we can write (13) as

$$\int_{V_l}\nabla\cdot\sigma\nabla\Phi \approx \sum_{j\in\mathcal{N}_l}\gamma_j\Phi_j - \left(\sum_{j\in\mathcal{N}_l}\gamma_j\right)\Phi_l \ . \tag{18}$$

The stencil coefficients γ_j are basically the harmonic average of the conductivities in the central voxel and the corresponding neighbour. For the eastern neighbour γ_j takes the form

$$\gamma_{\text{east}} = \frac{2h_yh_z}{h_x}\cdot\frac{\sigma_l\cdot\sigma_{\text{east}}}{\sigma_l+\sigma_{\text{east}}} \tag{19}$$

and analogously for the other neighbours. Note that since we have chosen $\sigma_{\text{air}}=0$ this derivation already includes the homogeneous boundary conditions of (10). Evaluation of the right hand side of (10) leads to[1]

$$\begin{aligned} -I\int_{V_l}\left(\delta(x_{\text{source}}-x)-\delta(x_{\text{sink}}-x)\right) &= -I\int_{\Omega}\left(\delta(x_{\text{source}}-x)-\delta(x_{\text{sink}}-x)\right)\mathbb{I}_{V_l} \\ &= \begin{cases} -I & \text{for } x_{\text{source}}\in V_l \text{ and } x_{\text{sink}}\notin V_l \\ +I & \text{for } x_{\text{sink}}\in V_l \text{ and } x_{\text{source}}\notin V_l \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{20}$$

We model the electrodes as point sources within the voxel that is nearest to the position of the center of the respective electrode. The resolution of our head models is fine enough that for the given spacing of the electrodes x_{source} and x_{sink} will never lie within the same voxel. Let us denote by k_{source} and k_{sink} the index of the voxel containing the source and the sink. Then we get that the discrete approximation Φ_h of (10) has to satisfy

$$\left(\sum_{k\in\mathcal{N}_l}\gamma_k\right)\Phi_l - \sum_{k\in\mathcal{N}_l}\gamma_k\Phi_k = I(\delta_{k_{\text{source}},l}-\delta_{k_{\text{sink}},l}) \tag{21}$$

for every voxel V_l that belongs to the head. Here $\delta_{i,j}$ denotes the Kronecker symbol.

[1]Strictly speaking the indicator function $\mathbb{I}_{V_l}$ is not in $C^\infty(\Omega)$ and thus no allowed test function in the sense of distribution theory. Taking this view point we have to see (20) as part of the discretisation process.

Properties of the Problem Matrix

If we write the linear system resulting from (21) in algebraic form as $Ax = b$ the system matrix $A = \{a_{ij}\}$ has the following properties:

- From (19) we see, that the coefficient connecting a voxel V_l to a neighbouring voxel V_k is identical to the coefficient connecting V_k to V_l, thus the matrix A will be symmetric.
- The conductivities σ are always non-negative, thus the coefficients γ_k in (19) will also be non-negative. Since the head is modelled as a contiguous volume conductor there are no isolated head voxels, therefore (21) gives us a matrix with positive diagonal and non-positive off-diagonal elements.
- (21) also shows that the sum of all entries in a row/column of A equals zero. Therefore the matrix is weakly diagonally-dominant and singular with the vector $e = (1, \ldots, 1)^T$ being an eigenvector to the eigenvalue zero.
- From the symmetry of A and the theorem of Gershgorin one can instantly see, that the spectrum of A is real and a subset of $[0, 2 \max a_{ii}]$. Hence we have a positive semi-definite matrix.
- Coming from a stencil-based discretisation on a contiguous grid the matrix is naturally irreducible.

From the above properties we see that the system matrix is a singular M-matrix in the classical notation of Berman & Plemmons. Therefore we know that $\text{rank}(A) = n - 1$ and the eigenspace of the eigenvalue 0 is of dimension one [BP94]. However, the problem can still be solved by standard iterative methods as we will see in the following section. Another possibility is to transform the singular linear system into a regular one and solve this instead. The regular problem is chosen such that its unique solution belongs to the set of solutions of the original singular system. The easiest approach is to fix the value of the potential Φ_h to 0 in one voxel. The special structure of the matrix then allows us to cancel the corresponding row and column in A and also the respective entry in the right hand side vector b. This leads to a problem with a regular M-matrix and its solution obviously solves the initial problem with $\Phi_h = 0$ in the respective voxel. We will give results for both approaches later on.

Another important aspect of the matrix A, which strongly influences the choice of possible solvers, is its sparseness. An example of the structure of the problem matrix is given in Fig. 5. Every matrix row can at most contain seven non-zero entries. This leads to a very small ratio of non-zero to overall entries resulting in a very sparse matrix.

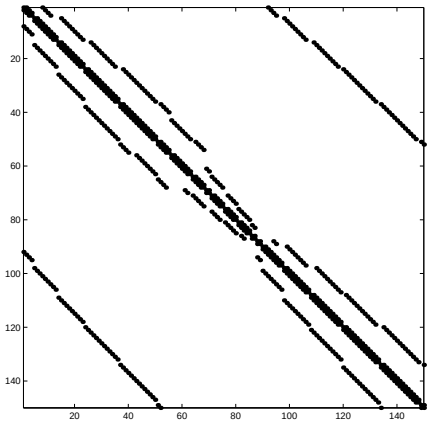


Figure 5: Left upper block of problem matrix for data set A using a lexicographic ordering of the unknowns (black dots represent non-zero matrix entries).

4 Iterative Solvers

The number of algorithms for solving linear systems $Ax = b$ seems nearly infinite. They can roughly be separated into two groups. Direct ones and iterative ones. Extremely sparse matrices are a challenge for direct methods due to fill-in. This term denotes the fact that during the elimination process zero entries of the matrix are replaced by non-zero ones, which have to be stored and eliminated again later on. The prime task of direct methods designed for sparse matrices, besides solving the problem of course, is to avoid fill-in as best as possible. Typically minimal bandwidth or minimum degree re-orderings of the matrix are employed before the start of the elimination process. Matrices coming from the discretisation of PDEs with Finite-Element and Finite-Difference methods are not only very sparse, but also tend to have a very distinct structure, which limits the gain of such strategies. E. g. even after a symmetric reverse Cuthill-McKee reordering of the problem matrices of the following section the amount of storage needed for a Cholesky decomposition is around 780MB for the small and 23.5GB for the large data set, cf. Tab. 1.

Iterative methods on the other hand only require the storage of non-zero matrix entries for an algebraic approach or the stencil coefficients for a grid based implementation, and, in general, the amount of work is dominated by the cost of one or two matrix vector multiplications per iteration. Thus if a stable and quickly converging method for solving the problem exists, it will clearly be the better choice. In the next section we will give numerical experiments to judge the suitability of the following iterative methods for the problem at hand:

- Successive Over-Relaxation (SOR)
- Conjugate Gradients (CG)
- Conjugate Gradients preconditioned by symmetric SOR (PCG)
- Algebraic Multigrid (AMG)

The SOR method is a representative of the classical stationary methods, see e. g. [Var62]. It is known to be not the optimal choice as far as convergence is concerned, but it has a very simple structure. Thus it is a good candidate for an optimized implementation.

The CG method is the typical algorithm from the large class of Krylov subspace methods. It was specially designed for the case of symmetric positive definite matrices. An introduction can be found e. g. in [She] and [Gre97]. The convergence of the CG method depends on the condition number, or more precisely on the spectrum, of the problem matrix. It is therefore seldom used without preconditioning. We have chosen a symmetric SOR preconditioner for this purpose.

The last contestant is an algebraic multigrid method. Multigrid methods in general are known to be very efficient solvers for elliptic boundary value problems. They employ a hierarchy of grid levels to treat individual problem components. This results in the nice property that, if all components are chosen in a proper way, the amount of work scales linearly with the number of unknowns and the convergence rate is independent of the mesh width employed in the discretisation. Originally developed for constant coefficient problems on simple domains, they nowadays are also successfully employed on anisotropic or interface problems and also on complicated domains. Unfortunately finding the proper components can be quite tedious in these cases. Therefore the idea of algebraic multigrid methods, as illustrated e. g. in [RS87], is gaining increased attention. Here a "grid hierarchy" and inter-grid transfer functions are chosen solely on the "strength" of the coupling between unknowns. This information is derived automatically from the problem matrix. This makes multilevel ideas also available for linear systems not stemming from grid based applications and is a step towards black box multilevel methods. For an introduction on algebraic and classical geometric multigrid methods see e.g. [BHM00].

Theoretical Aspects

A crucial question for iterative solvers is, whether they will converge at all for a given problem. In the following we will cite results, that guarantee convergence for the regularized forward problem as well as for the singular one. The question of speed of convergence will then be tackled experimentally in the next section.

Before we consider the convergence issue, note that for the singular problem to have a solution at all the right hand side vector b has to fulfill $b \in \mathrm{Im}(A)$. In this case the problem $Ax = b$ possesses an infinite set of solutions. An iterative method, that from each initial guess converges towards an element of this solution set, is said to be semiconvergent. In our case A is symmetric, thus $\mathrm{Im}(A) = \mathrm{Ker}(A)^\perp$. Since $\mathrm{Ker}(A)$ is spanned by the vector containing only ones as entries, a vector v lies in $\mathrm{Im}(A)$ if and only if

$$\left\langle \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, v \right\rangle = \sum_{k=1}^{n} v_k = 0 \ . \tag{22}$$

From (21) it is easy to see, that the right hand side of our problem satisfies this condition.

In order to prove the convergence of the SOR method for our problem we cite the following theorem from [BP94].

Theorem 1 *Let $A \in \mathbb{C}^{n\times n}$ be a hermitian matrix with a block decomposition $A = D - L - U$ where D is a block diagonal matrix with square nonsingular blocks on the diagonal and $-L$ and $-U$ are the block lower and upper parts of A. Let in addition D be positive definite, then the SOR block iteration matrix $H_\omega = (D - \omega L)^{-1}\left[(1-\omega)D + \omega U\right]$ is semiconvergent for all $0 < \omega < 2$, if and only if A is positive semi-definite.*

As was shown in the previous section, our problem matrix fulfills the requirements of the above theorem with a block size of one. Thus we can conclude that for the singular and for the regular case the SOR method will converge towards an element of the set of solutions.

The convergence of the CG method for the regularized case is well-known, since the problem matrix then is symmetric and positive definite. A proof can e. g. be found in [Gre97]. In order to show the convergence for the singular case, we first proove the following simple

Lemma 1 *Let $A \in \mathbb{R}^{n\times n}$ be symmetric and positive semi-definite with $Ker(A) = \langle\xi\rangle$, $\xi \in \mathbb{R}^n$, and the right hand side $b \in Im(A)$. If we decompose the i-th iterate of the CG method applied to $Ax = b$ into $x^{(i)} = \tilde{x}^{(i)} + \lambda\xi$ with $\tilde{x}^{(i)} \in Ker(A)^\perp$, then the next iterate $x^{(i+1)}$ takes the form $x^{(i+1)} = \tilde{x}^{(i+1)} + \lambda\xi$ with $\tilde{x}^{(i+1)} \in Ker(A)^\perp$.*

Proof: Since A is symmetric, we have $\mathrm{Ker}(A)^\perp = \mathrm{Im}(A)$. Thus for every vector $v \in \mathbb{R}^n$ the residual $r = b - Av$ lies in $\mathrm{Ker}(A)^\perp$. In the CG method the iterate $x^{(i+1)}$ is computed via

$$x^{(i+1)} = x^{(i)} + \alpha^{(i)} d^{(i)} \ ,$$

where $d^{(i)} \in \mathbb{R}^n$ is a search direction and $\alpha^{(i)}$ is a scalar given by

$$\alpha^{(i)} = \frac{(r^{(i)})^T r^{(i)}}{(d^{(i)})^T A d^{(i)}} \ .$$

The factor $\alpha^{(i)}$ exists, if its denominator is non-zero. This is equivalent to $d^{(i)} \in \mathrm{Ker}(A)^\perp \backslash \{0\}$, which also directly leads to the desired result. We will show now that $d^{(i)} \in \mathrm{Ker}(A)^\perp$ and that $d^{(i)} = 0$ can only occur for $r^{(i)} = 0$, in which case $x^{(i)}$ is a solution of the problem and the iteration is stopped anyway.

The first search direction is just the residual of the initial guess $d^{(0)} = r^{(0)}$. Thus $d^{(0)} \in \mathrm{Ker}(A)^\perp$ and we can assume $d^{(0)} \neq 0$ without loss of generality. The i-th search direction is computed from the previous one via

$$d^{(i)} = r^{(i)} + \beta^{(i)} d^{(i-1)} \ ,$$

with

$$\beta^{(i)} = \frac{(r^{(i)})^T r^{(i)}}{(r^{(i-1)})^T r^{(i-1)}} \ .$$

Since $r^{(i)} \in \mathrm{Ker}(A)^\perp$ anyhow, we see by recursion that also for the new search direction we have $d^{(i)} \in \mathrm{Ker}(A)^\perp$. One can easily see that the residual $r^{(i)}$ is orthogonal to the search direction $d^{(i-1)}$ used to compute $x^{(i)}$. Thus $d^{(i)} = 0$ is only possible if the residual $r^{(i)} = 0$. This completes the proof.

□

From the above lemma we can instantly derive the following

Theorem 2 *Let $Ax = b$ be a linear system of equations, where $A \in \mathbb{R}^{n\times n}$ is symmetric and positive semi-definite with $Ker(A) = \langle\xi\rangle$, $\xi \in \mathbb{R}^n$, and the right hand side $b \in Im(A)$. Then the CG method will semiconverge towards an element of the set of solutions. This element x^* is uniquely determined by the initial guess $x^{(0)}$ in the sense that*

$$\xi^T x^* = \xi^T x^{(0)} \ .$$

Proof: It is easily seen that A restricted to $\mathrm{Ker}(A)^\perp$ is a positive definite mapping. Also $Av = A^T v$ still holds for all $v \in \mathrm{Ker}(A)^\perp$. Since according to lemma 1, the CG method applied to $Ax = b$ is, in the semi-definite case, effectively only an iteration on the subspace $\mathrm{Ker}(A)^\perp$, this shows its semiconvergence to an element of the set of solutions. The form of this element is also a direct consequence of lemma 1.

□

Let us turn our attention now to the use of SSOR as a preconditioner for the CG method. The question here is, whether the method is of a form that allows for its application as preconditioner. In general, when preconditioning the CG method, one looks for a linear system of equations

$$\tilde{A}\tilde{x} = \tilde{b} \tag{23}$$

with $\tilde{A} = E^{-1}A(E^{-1})^T$, $\tilde{x} = E^T x$ and $\tilde{b} = E^{-1}b$, that is equivalent to the original one and is more favourable for CG in respect of convergence. In practice the matrix $\tilde{A}$ is never explicitly composed. Instead in the PCG algorithm in each step a system $Mz = r$ must be solved with a matrix $M = EE^T$. For this reason, when constructing a preconditioner, one usually does not look for a matrix E, but instead for a suitable matrix M. If it is chosen to be symmetric and positive definite, then a proper E always exists and $\tilde{A}$ will again be symmetric and positive definite, if the original A was. Before we come to the application of SSOR, we recall that a linear iteration scheme for the solution of a linear system $Ax = b$ is typically given in the form

$$x^{(k+1)} = Sx^{(k)} + B^{-1}b \ ,$$

where its iteration matrix S is given by

$$S = I - B^{-1}A \ .$$

When such a scheme is used as a preconditioner in the CG context, then effectively B is chosen to play the role of M.

We assume now, that the problem matrix A is split as $A = D - L - U$, where D is a diagonal matrix and L and U are strictly lower and upper triangular matrices. In the symmetric case we have $L = U^T$ and the matrix B of the SSOR algorithm is then given by

$$B_{\mathrm{SSOR}} = \frac{\omega}{2-\omega}\left(\frac{1}{\omega}D - L\right)D^{-1}\left(\frac{1}{\omega}D - L\right)^T \ . \tag{24}$$

This can easily be derived, see e. g. [Hac93]. We proceed with the following

Lemma 2 *Let A be a symmetric matrix with positive diagonal entries. Then the matrix B_{SSOR} in the SSOR method is symmetric and positive definite for all $\omega \in (0, 2)$.*

Proof: Since A possesses only positive diagonal entries the matrix $D = \mathrm{diag}(A)$ is invertible and consequently B_{SSOR} from (24) is well-defined. Furthermore D and also D^{-1} are positive definite. Thus there exists a matrix P such that $P^2 = D^{-1}$. Note that P is again symmetric and positive definite.

Let $Q = \left(\frac{1}{\omega}D - L\right)^T$. Since L is a strictly lower triangular matrix, we get $\det(Q) = \det(D)/\omega$, which is positive for all $\omega > 0$. Thus we get that Q is invertible. Using P and Q we can rewrite B_{SSOR} as

$$B_{\mathrm{SSOR}} = \frac{\omega}{2-\omega}(PQ)^T(PQ) \ ,$$

from which one easily sees that B_{SSOR} is symmetric and (since PQ is regular) also positive definite for the specified range of ω.

□

The above lemma directly leads to our central convergence result for the PCG(SSOR) method.

Theorem 3 *Let A be the problem matrix associated with (21) and let $b \in Im(A)$. Then in the regular and in the singular case the PCG(SSOR) method will converge towards a solution of $Ax = b$.*

Proof: Let A be singular or regular, then in both cases the problem matrix A is symmetric and possesses only positive diagonal entries. Thus lemma 2 is applicable and shows that the matrix $M = B_{\text{SSOR}}$ used for preconditioning is symmetric and positive definite. Thus a splitting of $B_{\text{SSOR}} = EE^T$ exists and the matrix E will be regular (it is even symmetric and positive definite).

We now employ the same trick as in the proof of lemma 2 in order to show that the matrix $\tilde{A}$ of the linear system (23) is symmetric and positive (semi-)definite if A is. Since A is symmetric and positive (semi-)definite, it can be split into $A = P^2$, with a symmetric and positive (semi-)definite matrix P. Thus we can write $\tilde{A}$ as

$$\tilde{A} = E^{-1}A\left(E^{-1}\right)^T = \left(E^{-1}P\right)\left(E^{-1}P\right)^T \quad ,$$

from which the desired property of $\tilde{A}$ directly follows. For the regular case the convergence is then well known, see e. g. [She]. For the singular case, we note that $\dim(\text{Ker}(\tilde{A})) = \dim(\text{Ker}(A)) = 1$ and thus theorem 2 is applicable. This completes the proof.

□

Convergence results for AMG are usually hard to derive in general settings. This is mainly due to the fact, that structure and properties of the coarse grid matrices are difficult to predict a priori. However many AMG versions use the Galerkin principle to set up the coarse grid operators. Assume that p is a prolongation operator from grid level $k+1$ to k and r the corresponding restriction operator, then the coarse grid operator A^{k+1} is chosen as

$$A^{k+1} = rA^kp \ .$$

Let now A^k be symmetric, positive definite, then choosing the restriction as the adjoint of interpolation leads to A^{k+1} being also symmetric and positive definite. From this the coarse grid correction can be shown to have the following minimization property in the two-grid case:

Lemma 3 *Let A^k be symmetric, positive definite and $A^{k+1} = rA^kp$ with $r = p^*$ and $Ker(p) = \{0\}$, then we have for the iteration matrix M of the coarse grid correction*

$$\|Me_k\|_{A^k} = \min_{v \in G^{k+1}} \|e_k - pv\|_{A^k}$$

and

$$\|M\|_{A^k} = 1 \ .$$

Here $\|.\|_{A^k}$ denotes the energy norm induced by A^k and G^{k+1} is the vector space A^{k+1} acts on.

For the proof the reader is refered to [TOS01]. There it is also shown, that this so called variational property also holds in the multigrid case, if the approximate solution of the coarse grid equation is precise enough. This alone is clearly not sufficient for convergence. We need also information on the smoothers. We combine two results, that are e. g. shown in [Hac93], and give the following

Lemma 4 *Let A be symmetric and positive definite, than the energy norms of the iteration matrices of the Jacobi and of the Gauß-Seidel method are both smaller than 1.*

The two lemmas combined guarantee the convergence of AMG employing a Galerkin approach and with the respective smoothers under the condition that the solution of the coarse grid problems is exact enough.

For the semi-definite case the authors can unfortunately not provide any theoretical results. Nevertheless we are quite confident, that these results could be proved, since the case of homogeneous von Neumann boundary conditions does not pose any considerable problems in the model case, see e. g. [BHM00] and [Moh01].

Implementation Details

A first decision one has to make, when faced with a linear system coming from a grid based discretisation is, whether the solver should operate on a grid based representation of the problem or in a linear algebra style on a matrix-vector formulation. Both approaches have their pros and cons. The advantage of the matrix based approach is its flexibility. Once we have gone to this abstraction level, there is a rich variety of existing subroutines that expect their input to be matrices and vectors. Although in the sparse case this advantage is somewhat reduced by the large variety of different storage formats. The grid based approach in turn has the advantage that, if the underlying grid is regular, as in our case, the construction of efficient data layouts and data access schemes is easier. It may also appear more "natural".

Where suitable, we have implemented the algorithms in a matrix-vector as well as in grid based version. For the matrix-vector case we have chosen a compressed row storage (CRS) scheme for representing the matrix, [BBC+94]. In each row the diagonal entry is stored in the first position. This is helpful for the SOR case, which includes a division by this entry, and unimportant for the CG variants. Setup of the system matrix is performed in two sweeps over the grid. In the first sweep the number of non-zero matrix entries is determined, to perform memory allocation. In the second sweep the matrix entries are computed and inserted into the CRS structure.

In the grid based approach the solution, stencil coefficients, etc. are stored as 3D arrays. Each array occupies a contiguous memory segment, in which data are stored in row-major ordering. The value at a certain voxel is accessed via a pointer structure, so that in C notation $u_{i,j,k}$ can be manipulated as `u[i][j][k]`. Prior to the solution process the stencil coefficients have to be determined according to (21). This requires one sweep over the grid. Note that due to the symmetry of the stencils we only need to store one coefficient for each coordinate direction. It has also proven to be advantageous not to store the central coefficient, but to compute it from the other six, when it is needed, since this reduces memory traffic, see also [Zet00, KMZ01].

To avoid unnecessary computational work, we do not loop over the whole cube, but try to restrict us, to the area containing head cells. To achieve this, we compute in a preparation step dynamic loop bounds in y- and z-direction. The head can have concave parts. Thus before we update the value of a voxel we still have to check, whether it belongs to the head. This is done by testing, whether the central stencil coefficient equals zero, which marks non-head cells. This procedure is numerically safe due to the range of possible coefficient values, cf. next section.

As far as the individual algorithms are concerned, we have to note that for the pure SOR we have chosen a red-black ordering of the unknowns, while a lexicographic ordering was used for its application as preconditioner. In the symmetric SOR a forward sweep is followed by a backward sweep in reverse direction. The AMG algorithm is the only method we did not program ourselves. Instead we used *BoomerAMG* from the *hypre* package, see [CCF98, HY00, HYP].

5 Numerical Experiments

Data Sets

For our numerical experiments we have employed two data sets, in the following denoted by data set A and B. They have been created from real patient MRI scans at the Epilepsy Monitoring Unit of the University of Gent, Belgium. The data sets have a different resolution and come from two different subjects. The edge length of the voxels is the same in all three dimensions. Thus the computation of the stencil coefficients takes the form

$$\gamma_n = 2h \cdot \frac{\sigma_l \cdot \sigma_n}{\sigma_l + \sigma_n} \ . \tag{25}$$

For the conductivities we use the values given in [OD99]. We scale our linear system by $(\sigma_{\text{skull}} h)^{-1}$ and only use the relative values of the conductivities

$$\left.\begin{array}{rcl} \sigma_{\text{scalp}} &=& 16 \\ \sigma_{\text{skull}} &=& 1 \\ \sigma_{\text{brain}} &=& 16 \end{array}\right\} \ \& \ \sigma_{\text{air}} = 0 \ . \tag{26}$$

Data Set	A	B
number of voxels / cube dimensions	65^3	129^3
edge length of voxels	3.92mm	1.96mm
number of head voxels / unknowns	70,830	541,402
number of non-zeros	482,184	3,738,624
matrix sparsity	0.01%	0.001%
bandwidth (lexicographic ordering)	2,352	9,573
bandwidth (after SRCM re-ordering)	1,456	5,842

Table 1: Details of the two patient data sets and the corresponding linear systems.

Note that the differences in the conductivities are only one order of magnitude. So we are faced with an interface problem with only moderate jumps. From (25) we see that there are only four possible values for the non-central stencil coefficients $\{0, 1, 32/17, 16\}$. An overview of the properties of the two data sets and of the resulting linear system is given in Tab. 1. Note that the given bandwidths are for the regularized problem with a non-singular matrix. The matrix was first set up using a lexicographic ordering of the unknowns and afterwards the symmetric reverse Cuthill-McKee algorithm was employed to find an ordering with a reduced bandwidth. For both patients an EEG was recorded using 27 electrodes. These had been placed according to the international 10–20 standard [Jas58], with three additional electrodes positioned on the temporal lobe areas on both sides. Thus the preparatory step that provides the potential data needed for easy setup of the lead field matrix consists in the solution of 26 forward problems for as many electrode pairs.

In all following tests we have used the same stopping criterion. We accepted an approximate solution as soon as its residual measured in the Euclidean norm became smaller than 10^{-8}.

Parameter dependency

From the four tested methods the CG algorithm is the only one that does not depend on the choice of a special parameter. In case of the SOR and PCG(SSOR) method we have to specify the over-relaxation parameter ω. For AMG a threshold parameter, used to define the "strength" of connections between the unknowns / voxels, must be chosen. We have chosen two of the 26 electrode pairs to test the dependency of the algorithms on these parameters. In both cases the two electrodes of the pair lie on opposing sides of the head. In the first case both have approximately the same vertical coordinate, while in the second, one is lower than the other.

Figure 6 shows the dependency of the number of SOR sweeps needed in order to satisfy the stopping criterion for the case of a singular matrix. We see that this number varies considerable even for the small interval $\omega \in [1.89, 1.99]$. The optimal values seem to be in the vicinity of 1.935 for data set A and 1.970 for data set B. We have employed these values for all subsequent tests.

When we apply SOR to the regular matrix, we get basically the same picture, with two differences. On the one hand the optimal ω is larger (about 1.9935 for dataset A and about 1.9977 for dataset B). On the other hand also the number of iterations needed to reach the desired accuracy is drastically larger. Even for the nearly optimal ω the numbers are on average 10 resp. 14 times larger, as can be seen in Tab. 4.

The choice of ω appears to be not as important for the PCG method, however, as can be seen in Fig. 7. The number of iterations varies over smaller intervals and the valleys around the optimal value are more flat, thus choosing a reasonable ω is easier. The optimal values lie in the vicinity of 1.65 and 1.75 for the singular problem and 1.65 and 1.8 for the regular one. Note that the number of iterations for the singular problem is smaller than for the regular case.

In the AMG algorithm there is quite a number of parameters and algorithmic variants, that have to be fixed, before the method can be applied. These include e. g. cycle type, number of smoothing steps, and so on. We have used a V-cycle with one pre- and one post-smoothing step. The relaxation scheme was a hybrid Gauß-Seidel / Jacobi method. Construction of the grid hierarchy was performed with Ruge-Stüben coarsening. Besides this, we left all other parameters at their default values and investigated the influence of the threshold value α, which allows for the distinction of "weak" and "strong" inter-node connections. A node j is said to be strongly connected to a node

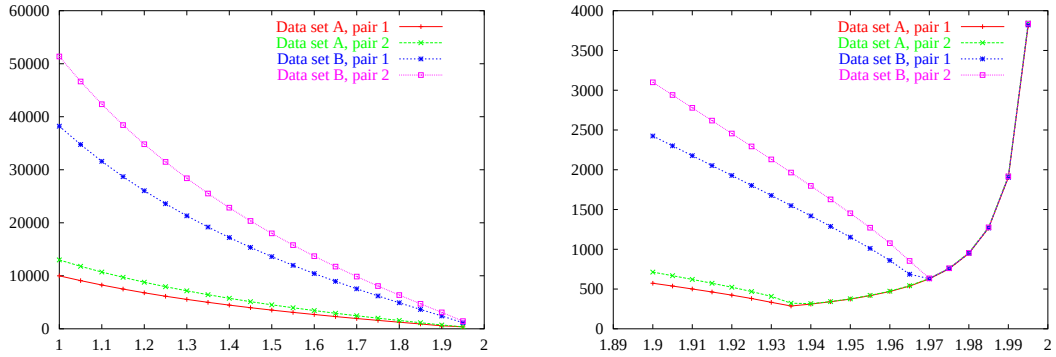


Figure 6: Number of SOR iterations necessary to satisfy stopping criterion for the singular problem depending on choice of relaxation parameter ω.

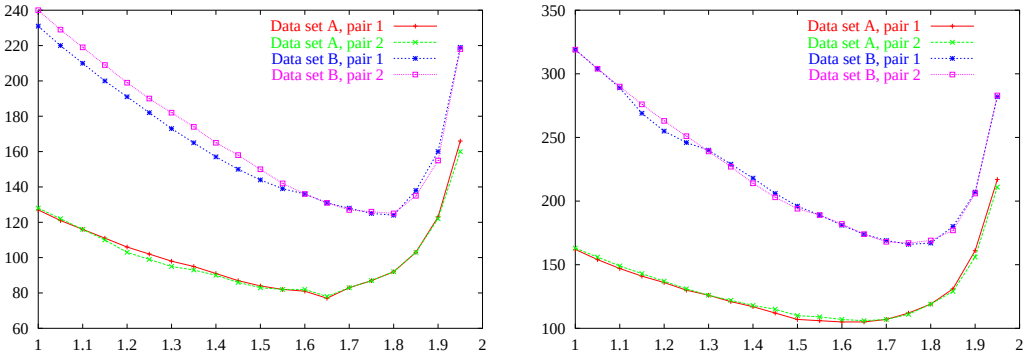


Figure 7: Number of PCG(SSOR) iterations necessary to satisfy stopping criterion depending on choice of relaxation parameter ω; left: singular problem, right: regular problem.

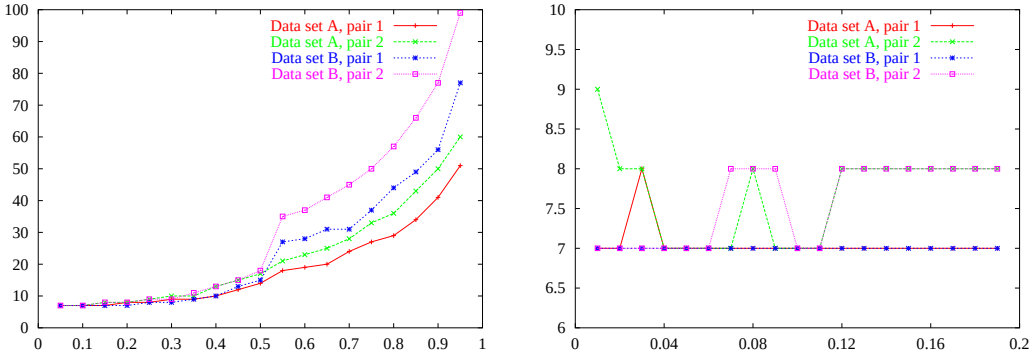


Figure 8: Number of AMG cycles necessary to satisfy stopping criterion depending on choice of threshold parameter α.

i, and thus influences i strongly, if

$$|a_{ij}| \geqslant \alpha \cdot \max_{k \neq i} (|a_{ik}|) \quad . \tag{27}$$

This distinction is critical for the construction of the grid hierarchy. The coarsening process involves determining for every node i its set of dependence, which consists of all other nodes, that strongly influence i. The influence of α on the number of AMG cycles in the singular case is shown in Fig. 8. The regular case is not shown, since the results are very similar, with the only difference being a quicker decay of convergence for large values of α. Details of the method are given in Tab. 2 for data set A and in Tab. 3 for data set B. In these tables we show the number of grid *levels*, the different *complexities*, the average reduction factors *(arf)* for the residual norm, the number of *cycles* and give an estimate of the total amount of *work*. The grid and operator complexities are defined as follows

$$\text{grid complexity} \quad = \quad \frac{\text{\# of nodes on all grids}}{\text{\# of nodes on finest grid}}$$

$$\text{operator complexity} \quad = \quad \frac{\text{\# of non-zero entries in all operators / matrices}}{\text{\# of non-zero entries in fine grid operator / matrix}} \quad .$$

The idea of the cycle complexity is to compare the number of arithmetic operations per cycle on all levels to the cost of a relaxation sweep on the finest grid. The latter is commonly referred to as a *work unit* in multigrid context. The program gives a crude estimate of this by basically counting the work for relaxation on all levels. As can be seen from the tables cycle complexity is twice operator complexity, since we perform two smoothing steps. This of course neglects the costs for intergrid transfers, correction steps and direct solution on the coarsest level. Multiplication of the cycle complexity with the number of cycles gives an estimate for the number of work units needed to solve the problem, up to the desired accuracy.

The average reduction factor of the residual r is defined by

$$\text{arf} = \left(\frac{\|r^{(k)}\|_2}{\|r^{(0)}\|_2} \right)^{1/k} \quad , \tag{28}$$

where k is the total number of cycles needed to satisfy the stopping criterion.

We note two interesting facts about the influence of the threshold parameter α. The first is, that the best convergence is achieved for comparatively small values of α. This also translates to the number of work units. The second fact is that there is a sharp increase in the number of cycles in the interval $\alpha \in [0.50, 0.55]$. This is especially pronounced in the case of data set B. This phenomenon can be explained by taking a closer look at the stencil coefficients.

Consider a voxel that lies deep inside one of the three head compartments and is only surrounded by voxels of the same tissue type. In this case we see from (25) that all non-central stencil coefficients have identical values. Thus, according to (27), all connections of this voxel will be considered strong, independent of the choice of α. Therefore, the only voxels on the finest grid, for which α makes a difference, are those in the vicinity of the compartment boundaries.

Let us now consider the case of a brain voxel, that has neighbours in the brain and in the skull compartment, as is sketched in Fig. 9 on the right. Those legs connecting the central voxel to other brain voxels have a weight of 16, while those leading to skull voxels have a weight of 32/17. Thus for all values $\alpha \geqslant 32/(16 \cdot 17) > 0.11$ the latter connections will be regarded as weak, while the former are always strong. In the case of a skull voxel with brain neighbours the weight of connections to other skull voxels is 1 and that of connections to brain voxels is again 32/17. Thus in this case all connections to skull voxels will be weak for values $\alpha \geqslant 17/32 > 0.53$. Since the relative conductivity of the scalp and brain compartments is the same, we have an identical situation at the scalp-skull interface.

This has the following effect on the interpolation operator. For $\alpha > 0.11$ brain and scalp voxels will no longer have their skull neighbours in their interpolatory sets. If we increase α further, then for $\alpha > 0.53$ the values in skull voxels at the compartment interfaces on the finest grid will only be interpolated from scalp / brain voxels on the next coarser grid. This leads to a bad interpolation

BoomerAMG with Ruge-Stüben coarsening										
treshold	levels	complexities			arf		# cycles		work units	
		grid	operator	cycle	pair 1	pair 2	pair 1	pair 2	pair 1	pair 2
0.01	7	1.62	3.13	6.26	0.07	0.10	7	9	43.8	56.3
0.02	7	1.62	3.15	6.30	0.07	0.08	7	8	44.1	50.4
0.03	7	1.62	3.18	6.36	0.08	0.08	8	8	50.9	50.9
0.04	7	1.64	3.60	7.19	0.06	0.07	7	7	50.4	50.4
0.05	7	1.63	3.47	6.94	0.05	0.07	7	7	48.6	48.6
0.06	8	1.65	3.73	7.45	0.06	0.07	7	7	52.2	52.2
0.07	8	1.66	4.01	8.02	0.07	0.07	7	7	56.1	56.1
0.08	8	1.67	4.14	8.27	0.06	0.07	7	8	57.9	66.2
0.09	8	1.68	4.33	8.66	0.05	0.06	7	7	60.6	60.6
0.10	8	1.68	4.38	8.75	0.06	0.06	7	7	61.3	61.3
0.11	8	1.71	5.03	10.05	0.05	0.06	7	7	70.4	70.4
0.12	9	1.71	5.02	10.04	0.06	0.08	7	8	70.3	80.4
0.13	9	1.71	5.03	10.06	0.06	0.07	7	8	70.4	80.5
0.14	9	1.72	5.29	10.59	0.06	0.08	7	8	74.1	84.7
0.15	9	1.73	5.40	10.80	0.06	0.08	7	8	75.6	86.4
0.16	9	1.73	5.33	10.66	0.06	0.08	7	8	74.6	85.3
0.17	9	1.74	5.54	11.07	0.06	0.08	7	8	77.5	88.6
0.18	9	1.73	5.41	10.81	0.06	0.09	7	8	75.7	86.5
0.19	10	1.74	5.50	11.00	0.06	0.08	7	8	77.0	88.0
0.20	10	1.75	5.83	11.67	0.08	0.09	8	8	93.3	93.3
0.25	10	1.75	5.65	11.29	0.08	0.12	8	9	90.3	101.6
0.30	11	1.75	5.37	10.73	0.10	0.13	9	10	96.6	107.3
0.35	12	1.77	5.49	10.99	0.11	0.15	9	10	98.9	109.9
0.40	12	1.80	6.06	12.12	0.15	0.22	10	13	121.2	157.5
0.45	12	1.79	5.37	10.74	0.20	0.27	12	15	128.9	161.1
0.50	12	1.81	5.54	11.09	0.25	0.32	14	17	155.2	188.5
0.55	13	2.01	6.61	13.21	0.34	0.41	18	21	237.8	277.5
0.60	14	2.01	6.48	12.97	0.37	0.44	19	23	246.4	298.3
0.65	13	2.01	6.23	12.47	0.38	0.47	20	25	249.3	311.7
0.70	13	2.01	5.88	11.76	0.45	0.50	24	28	282.3	329.3
0.75	13	2.02	5.63	11.27	0.50	0.56	27	33	304.3	371.9
0.80	13	2.00	5.36	10.72	0.52	0.59	29	36	310.9	385.9
0.85	13	2.00	5.21	10.42	0.57	0.64	34	43	354.4	448.2
0.90	12	1.99	5.03	10.05	0.63	0.69	41	50	412.1	502.6
0.95	12	2.01	4.91	9.81	0.69	0.73	51	60	500.4	588.7

Table 2: AMG for data set A (singular matrix).

BoomerAMG with Ruge-Stüben coarsening										
treshold	levels	complexities			arf		# cycles		work units	
		grid	operator	cycle	pair 1	pair 2	pair 1	pair 2	pair 1	pair 2
0.01	8	1.61	3.14	6.28	0.07	0.06	7	7	43.9	43.9
0.02	8	1.61	3.14	6.27	0.06	0.06	7	7	43.9	43.9
0.03	8	1.61	3.20	6.40	0.06	0.06	7	7	44.8	44.8
0.04	8	1.61	3.29	6.59	0.05	0.06	7	7	46.1	46.1
0.05	8	1.61	3.32	6.64	0.05	0.06	7	7	46.5	46.5
0.06	9	1.62	3.41	6.83	0.05	0.06	7	7	47.8	47.8
0.07	9	1.64	3.95	7.90	0.06	0.07	7	8	55.3	63.2
0.08	9	1.66	4.30	8.60	0.05	0.08	7	8	60.2	68.8
0.09	9	1.66	4.42	8.85	0.06	0.08	7	8	61.9	70.8
0.10	9	1.67	4.54	9.09	0.06	0.07	7	7	63.6	63.6
0.11	10	1.67	4.81	9.62	0.05	0.07	7	7	67.3	67.3
0.12	10	1.67	4.75	9.50	0.06	0.08	7	8	66.5	76.0
0.13	10	1.67	4.88	9.76	0.06	0.08	7	8	68.3	78.1
0.14	10	1.68	5.16	10.32	0.06	0.09	7	8	72.3	82.6
0.15	10	1.68	5.01	10.02	0.06	0.08	7	8	70.1	80.1
0.16	10	1.67	4.95	9.91	0.06	0.08	7	8	69.3	79.3
0.17	11	1.67	4.97	9.94	0.06	0.09	7	8	69.6	79.6
0.18	11	1.67	5.06	10.12	0.06	0.09	7	8	70.8	81.0
0.19	11	1.68	5.19	10.38	0.06	0.09	7	8	72.6	83.0
0.20	11	1.68	5.34	10.69	0.06	0.09	7	8	74.8	85.5
0.25	12	1.71	6.30	12.59	0.08	0.11	8	9	100.7	113.3
0.30	13	1.70	5.35	10.70	0.09	0.12	8	9	85.6	96.3
0.35	13	1.72	5.79	11.59	0.11	0.17	9	11	104.3	127.5
0.40	14	1.75	6.33	12.65	0.15	0.22	10	13	126.5	164.5
0.45	14	1.77	6.23	12.46	0.23	0.28	13	15	161.9	186.8
0.50	15	1.80	6.23	12.46	0.27	0.34	15	18	186.9	224.2
0.55	15	1.82	6.48	12.95	0.49	0.58	27	35	349.8	453.4
0.60	16	1.82	6.04	12.07	0.51	0.60	28	37	338.1	446.7
0.65	15	1.82	5.85	11.70	0.54	0.63	31	41	362.8	479.9
0.70	16	1.83	5.65	11.29	0.55	0.66	31	45	350.1	508.2
0.75	15	1.83	5.45	10.89	0.60	0.68	37	50	403.0	544.7
0.80	15	1.83	5.23	10.46	0.65	0.72	44	57	460.2	596.2
0.85	15	1.82	5.03	10.05	0.68	0.75	49	66	492.5	663.3
0.90	15	1.82	4.89	9.78	0.71	0.78	56	77	547.7	753.1
0.95	14	1.85	4.90	9.80	0.78	0.83	77	99	754.5	970.0

Table 3: AMG for data set B (singular matrix).

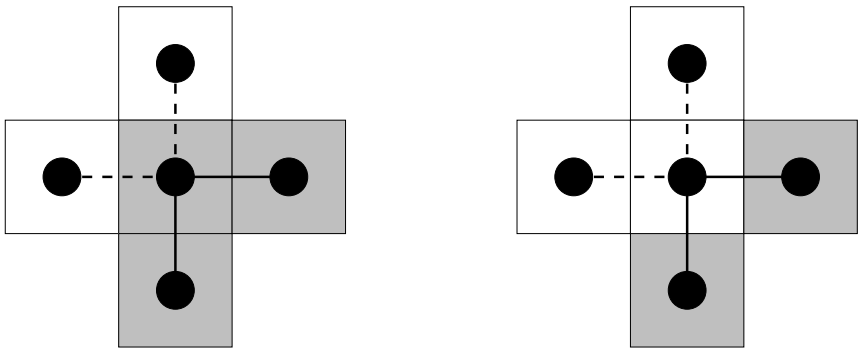


Figure 9: Strong (solid lines) and weak (dashed lines) connections in the case of $\alpha > 0.53$. Grey squares represent brain / scalp voxels and white ones skull voxels. The strength of connections is given for the voxel in the center.

operator, since we will get cases, where the value in a fine grid skull voxel is determined from only two or even only one coarse grid scalp / brain voxel. For data set B this effect is more pronounced, since due the finer resolution, we will get more of these cases.

Our experiments also indicate that performance will decrease again, when α becomes too small. So, for all further experiments we have settled with $\alpha = 0.05$.

Amount of Work

After having determined a set of reasonably optimal parameters for the different methods, we tested the number of iterations needed to satisfy the stopping criterion for all 26 electrode pairs. Table 4 summarizes the mean values of all pairs. To be able to better compare these values we have determined for each method an estimate of the total amount of arithmetic work involved. This estimate is given in Tab. 4 as a percentage of the amount of work for the SOR method in the semi-definite case.

We see that the CG method performs worst, due to a bad convergence rate. This is considerably improved by preconditioning. The number of iterations drops to roughly 18% of the unpreconditioned case. Due to the higher costs per iteration step this is still about two thirds of the reference case in the most favorable situation. The best performance is achieved by the AMG approach, which also shows the typical feature of a multigrid method, namely that the number of cycles remains constant, independent of the fineness of the discretisation.

Number of Iterations						
Method	Data set					
	A			B		
	Mean	% SOR	Std. Dev.	Mean	% SOR	Std. Dev.
SOR (1)	303.5	100	9.6	634.0	100	2.2
SOR (2)	2957.0	974	34.1	8839.0	1400	635.2
CG (1)	407.5	179	9.3	740.7	156	12.5
CG (2)	533.2	234	6.5	986.8	208	9.7
PCG(SSOR) (1)	80.3	88	1.3	126.7	67	1.8
PCG(SSOR) (2)	106.0	116	1.2	165.8	87	1.6
BoomerAMG (1)	7.0	16	0.0	7.0	7	0.0
BoomerAMG (2)	7.0	16	0.0	7.0	7	0.0

Table 4: Iteration counts for different methods, (1) indicates a singular matrix, (2) a regular one.

Run Times

Besides all such things as convergence rates, amount of work and so on, the property that is of primary interest to the user coming from the application side is run time. The latter is determined not only by the numerical characteristics of an algorithm, but also by the way it goes together with modern computer architectures and the way it has been implemented.

We have tested the run times of our four algorithms on three different architectures, details of which are given in Tab. 5. In all cases, except for AMG, we compare the grid-based with the matrix-based implementation, cf. Sec. 4. In Tab. 6 and Fig. 10 we present the user times for the complete problem. In each case we have measured times for 10 program runs and taken the mean value. Standard deviation was always less than 2%. The times consist of the time spent in the setup phase and for the solution of 26 forward problems. The setup phase includes processing of program parameters and geometry information, setup of stencil coefficients or problem matrix and, in the case of AMG, also of the grid hierarchy, which is the same for all of the forward problems. The costs for this setup phase are shown separately in Tab. 7. Times were measured in the same fashion as above. Due to their small sizes standard deviation was larger, but remained smaller than 17% of the mean value. Note that the setup times are always negligible except for the AMG case, where the construction of the grid hierarchy adds considerably to the total costs. These vary between 5 to 11% depending on architecture and problem size.

Concerning the measurements for BoomerAMG we should note that the hypre library, of which it is a part, was developed for solving large, sparse systems of linear equations on massively parallel computers. Thus it is not specially tuned for the sequential environment in which we used it.

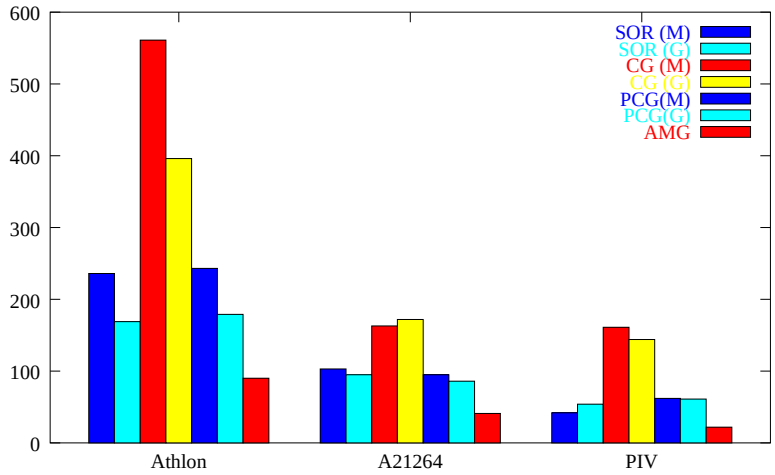
We also want to point out, that, although we have taken into account performance issues in the implementation of the algorithms, there are still numerous possibilities one could test for a further optimization of the code. For some ideas on this, see e. g. [DHK+00, KW01].

Athlon	
CPU	AMD Athlon (cpu family 6, model 2)
Clock rate	700 MHz
Main Memory	768MB
Caches	64KB (L1), 512KB (L2)
OS	Linux, kernel 2.2.16
Compiler/Linker	gcc/g++ (2.95.2)
Tuning Options	-O3
A21264	
CPU	A21264
Clock rate	500MHz
Main Memory	640MB
Caches	64KB (L1), 4MB (L2)
OS	Compaq Tru64 4.0F (Rev. 1229)
Compiler/Linker	cc (DEC C V5.9-005)
Tuning Options	-fast -tune host
PIV	
CPU	Pentium 4 (cpu family 15, model 0)
Clock rate	1500MHz
Main Memory	1024MB
Caches	8KB (L1), 256KB (L2)
OS	Linux, kernel 2.2.18
Compiler/Linker	gcc/g++ (2.95.2)
Tuning Options	-O3

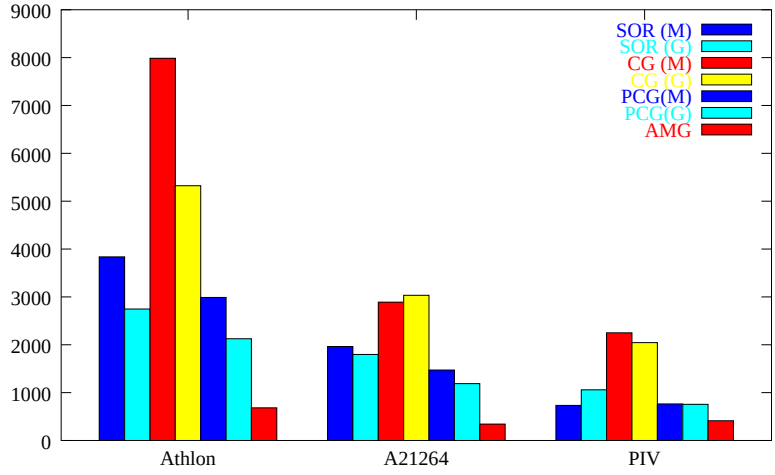
Table 5: Hard- and Software specification of systems used for run time measurements.

User time in seconds							
Method	Approach	Data set A			Data set B		
		Athlon	A21264	PIV	Athlon	A21264	PIV
SOR	matrix	236	103	42	3838	1964	734
	grid	169	95	54	2748	1797	1058
CG	matrix	561	163	161	7986	2890	2250
	grid	396	172	144	5324	3035	2046
PCG	matrix	243	95	62	2990	1471	763
	grid	179	86	61	2127	1187	756
AMG	matrix	90	41	22	682	342	413

Table 6: Run times for the complete problem. This includes the setup phase and the solution of 26 singular forward problems.



Data set A



Data set B

Figure 10: Graphical representation of the run times for the complete problem. See also Tab. 6.

User time in seconds — Setup Phase						
Setup of	Data set A			Data set B		
	Athlon	A21264	PIV	Athlon	A21264	PIV
Stencil Coeff.	0.1	0.1	0.1	1.3	0.8	0.6
CRS matrix	0.2	0.1	0.1	1.5	0.9	0.7
CRS matrix & Grid Hierarchy	4.9	3.8	2.5	42.7	32.3	20.7

Table 7: Run times for setup phase. This includes reading of data, setup of stencil coefficients resp. matrix entries and, in case of AMG, also the creation of a grid hierarchy.

6 Conclusions

Summing up the contents of this paper, we see three major points. The first one is, that the solution of inverse EEG problems can considerably be sped up by the application of multigrid methods for the solution of the forward problems involved. These appear to offer a much better performance than the Krylov subspace methods typically employed in this context.

The second aspect is that it seems pointless to transform the singular forward problem into an equivalent regular one utilizing the approach described in Sect. 3. In our experiments the regularization of the problem could not improve convergence of the tested methods and, in many cases, even led to a worse behaviour. However, this does not rule out the possibility that other regularization approaches, like e. g. demanding that $\sum_l \Phi_l = 0$, may lead to better performance.

The question, finally, whether a matrix- or a grid-based implementation of the methods is preferable in view of runtime behaviour, remains open. Our experiments in this context yielded results that vary from architecture to architecture and are ambiguous in themselves even on the same machine.

7 Acknowledgments

The author wishes to thank his colleague ir. Bart Vanrumste from the Neurology Department of the University of Ghent for introducing him to the inverse EEG problem and for the provision of the data sets. We are also indebted to the staff of the Department of Neurosurgery and the Neurocenter of the University of Erlangen-Nuremberg for their cooperation and the provision of intra-operational photographs.

References

[BBC+94] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, 1994.

[BHM00] W. L. Briggs, Van E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, 2nd edition, 2000.

[BP94] A. Berman and R. Plemmons. *Nonnegative Matrices in the Mathematical Sciences*. Number 9 in Classics in Applied Mathematics. SIAM, 1994.

[CCF98] E. Chow, A. J. Cleary, and R. D. Falgout. Design of the hypre Preconditioner Library. In Mike Henderson, Chris Anderson, and Steve Lyons, editors, *Proc. of the SIAM Workshop on Object Oriented Methods for Inter-operable Scientific and Engineering Computing*. SIAM, October 1998. Also available as Lawrence Livermore National Laboratory technical report UCRL-JC-132025.

[DHK+00] C. C. Douglas, J. Hu, M. Kowarschik, U. Rüde, and C. Weiß. Cache Optimization for Structured and Unstructured Grid Multigrid. *Electronic Transaction on Numerical Analysis*, 10:21–40, February 2000.

[DSB01] F. Darvas, U. Schmitt, and H. Buchner. Time-dependent source reconstruction from EEG-data. In J. Nenonen, R.J. Ilmoniemi, and T. Katila, editors, *Biomag 2000, Proc. of the 12th Internat. Conf. on Biomagnetism*, 2001. http://biomag2000.hut.fi.

[EMBL01] J. Ermer, J. Mosher, S. Baillet, and R. Leahy. Rapidly re-computable EEG forward models for realistic head shapes. In J. Nenonen, R.J. Ilmoniemi, and T. Katila, editors, *Biomag 2000, Proc. of the 12th Internat. Conf. on Biomagnetism*, 2001. http://biomag2000.hut.fi.

[FT99] T. Ferree and D. Tucker. Development of High-resolution of EEG Devices. *International Journal of Bioelectromagnetism*, 1(1), 1999. http://www.tut.fi/ijbem.

[Gre97] A. Greenbaum. *Iterative Methods for Solving Linear Systems.* Frontiers in Applied Mathematics. SIAM, 1997.

[GSK+97] O. Ganslandt, R. Steinmeier, H. Kober, J. Vieth, J. Kassubek, and J. Romstock et al. Magnetic source imaging combined with image–guided frameless stereotaxy: a new method in surgery around the motor strip. *Neurosurgery*, 41(3):621–627, 1997.

[Hac93] W. Hackbusch. *Iterative Lösung großer schwachbesetzter Gleichungssysteme.* Teubner Studienbücher. Teubner, 2 edition, 1993.

[Has99] P. Hastreiter. *Registrierung und Visualisierung medizinischer Bilddaten unterschiedlicher Modalitäten.* PhD thesis, FAU Erlangen-Nürnberg, Institute of Computer Science, 1999.

[HH93] M. Hanke and P. Hansen. Regularization methods for large-scale problems. *Surv. Math. Ind.*, 3:253–315, 1993.

[HY00] Van Emden Henson and Ulrike Meier Yang. BoomerAMG: a Parallel Algebraic Multigrid Solver and Preconditioner. Technical Report UCRL-JC-141495, Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, 2000. submitted to a Special Issue of Applied Numerical Mathematics.

[HYP] Homepage of the hypre project. http://www.llnl.gov/casc/hypre.

[Jas58] H. Jasper. Report of committee on methods of clinical exam in EEG. *Electroencephalography and Clinical Neurophysiology*, 10:370–375, 1958.

[KMZ01] M. Kowarschik, M. Mohr, and M. Zetlmeisl. Performance Optimization of Numerically Intensive Codes: A Case Study from Biomedical Engineering. Technical report, Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich-Alexander-Universität Erlangen-Nürnberg, 2001. In preparation.

[Köh98] T. Köhler. *Lösungen des bioelektrischen inversen Problems.* Dissertation, Fachbereich Physik, Universität Hamburg, 1998.

[KW01] M. Kowarschik and C. Weiß. DiMEPACK — A Cache-Optimized Multigrid Library. In H. R. Arabnia, editor, *Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2001)*, volume I, pages 425–430, Las Vegas, NV, U.S.A., June 2001. CSREA, CSREA Press.

[MG87] A. R. Mitchell and D. F. Griffiths. *The Finite Difference Method in Partial Differential Equations.* John Wiley & Sons, 1987. 3rd ed.

[ML99] J. Mosher and R. Leahy. Source localization using recursively applied and projected (RAP) MUSIC. *IEEE Transactions on Signal Processing*, 47(2):332–340, 1999.

[Moh01] M. Mohr. Model Problem Analysis for Multigrid in the Semidefinit Case. Technical report, Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich-Alexander-Universität Erlangen-Nürnberg, 2001. In preparation.

[NM65] J. Nelder and R. Mead. A simplex method for function minimization. *Computing Journal*, 7:308–313, 1965.

[OD99] T. Oostendorp and J. Delbeke. The Conductivity of the Human Skull in vivo and in vitro. In *Serving Humanity, Advancing Technology, Proceedings of The First Joint BMES/EMBS Conference*, page 456. IEEE, 1999.

[PM99] R. Pascaul-Marqui. Review of Methods for Solving the EEG Inverse Problem. *International Journal of Bioelectromagnetism*, 1(1), 1999. http://www.tut.fi/ijbem.

[RS87] J. W. Ruge and K. Stüben. Algebraic multigrid (AMG). In S. F. McCormick, editor, *Multigrid Methods*, volume 3 of *Frontiers in Applied Mathematics*, pages 73–130. SIAM, 1987.

[She] J. Shewchuk. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Unpublished draft, ed. $1\frac{1}{4}$, School of Computer Science, Carnegie Mellon University, http://www.cs.cmu.edu/~jrs.

[TOS01] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*, chapter An Introduction to Algebraic Multigrid (by K. Stüben). Academic Press, 2001.

[TWD+99] D. Tuch, V. Wedeen, A. Dale, J. George, and T. Belliveau. Conductivity mapping of biological tissue using diffusion MRI. In *Proceedings of The Third International Conference on Occupational Electrical Injury and Safety, Shanghai, P. R. China, October 26 – 28, 1998*, volume 888 of *Annals of the New York Academy of Sciences*, pages 314–316, 1999.

[Van01] B. Vanrumste. *EEG dipole source analysis in a realistic head model*. PhD thesis, Faculteit Toegepaste Wetenschappen, Universiteit Gent, 2001.

[Var62] R. Varga. *Matrix Iterative Analysis*. Series in Automatic Computation. Prentice-Hall, 1962.

[VHB+98] B. Vanrumste, G. Van Hoey, P. Boon, M. D'Havé, and I. Lemahieu. Inverse calculations in EEG source analysis applying the finite difference method, reciprocity and lead fields. In *Proceedings of 20th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 20, part 4/6, pages 2112–2115, Hong Kong, 1998.

[WRB+01] C. Wolters, S. Reitzinger, A. Basermann, S. Burkhardt, U. Hartmann, F. Kruggel, and A. Anwander. Improved tissue modelling and fast solver methods for high-resolution FE-modelling in EEG/MEG-source localization. In J. Nenonen, R.J. Ilmoniemi, and T. Katila, editors, *Biomag 2000, Proc. of the 12th Internat. Conf. on Biomagnetism*, 2001.

[WZJ00] D. Weinstein, L. Zhukov, and C. Johnson. Lead-field bases for electroencephalography source imaging. *Annals of Biomedical Engineering*, 28(9):1059–1065, 2000.

[Zet00] M. Zetlmeisl. Performance Optimization of Numerically Intensive Codes — A Case Study from Biomedical Engineering. Studienarbeit, 2000. Lehrstuhl für Informatik 10 (Systemsimulation), Friedrich-Alexander-Universität Erlangen-Nürnberg.