Computer Vision, Graphics, and Pattern Recognition Group
Department of Mathematics and Computer Science
University of Mannheim
D-68131 Mannheim, Germany

Reihe Informatik
5/2000

# Globally–Convergent Iterative Numerical Schemes for Non–Linear Variational Image Smoothing and Segmentation on a Multi–Processor Machine

Josef Heers, Christoph Schnörr, H. Siegfried Stiehl

Technical Report 5/2000
Computer Science Series
February 2000

This report is a revision of the Technical Report 283/99, Department of Computer Science, University of Hamburg, Hamburg, January 1999.

The technical reports of the CVGPR Group are listed at the site
`http://www.ti.uni-mannheim.de/~bmg/Publications-e.html`

**Abstract**

We investigate several iterative numerical schemes for nonlinear variational image smoothing and segmentation implemented in parallel. A general iterative framework subsuming these schemes is suggested for which global convergence irrespective of the starting point can be shown. We characterize various edge–preserving regularization methods from the recent image processing literature involving auxiliary variables as special cases of this general framework. As a by–product, global convergence can be proven under conditions slightly weaker than those stated in the literature. Efficient Krylov subspace solvers for the linear parts of these schemes have been implemented on a multi–processor machine. The performance of these parallel implementations has been assessed and empirical results concerning convergence rates and speed-up factors are reported.

**Keywords**

Adaptive smoothing, variational segmentation, non-linear regularization, images and pde's, auxiliary variables, parallel numerical algorithms

## I. Introduction

### *A. Overview*

Low-level feature extraction and image segmentation are key issues in image processing and computer vision. Variational approaches [1], [2], [3], [4] provide a mathematically sound problem formulation being superior to ad-hoc segmentation schemes. For a survey, we refer to [5].

A common problem with these approaches, however, is the high computational cost involved from an optimization point-of-view. Stochastic optimization [1] is not feasible for typical image sizes, while deterministic annealing procedures [2], [4] cannot guarantee to attain a "good" local minimum. Therefore, the use of *non-quadratic* but *convex* functionals has been advocated to simplify nonlinear variational image processing from the computational viewpoint [6], [7], [8]. Despite being mathematically much simpler, convex functionals provide a reasonable approximation to the prototypical but mathematically sophisticated and computationally expensive variational approach of Mumford and Shah [3] (see Section II-B below and [10]).

### B. Related work

An important feature of nonlinear convex variational approaches, particularly in the context of (semi-)automated image processing tasks, is the existence of algorithms that globally converge to the unique minimizer, irrespective of the starting point. A simple example is iterative gradient descent with sufficiently small steps[1]. Despite convexity, however, typical variational approaches are highly nonlinear, and it is difficult to obtain fast convergence by applying a standard Newton–like second order method, due to very narrow regions around the unknown global minimum where quadratic convergence holds true.

In this context, the work of Geman and Reynolds [11] and Geman and Yang [12] is very interesting. The scheme presented by Geman and Reynolds [11], originally developed to minimize locally sophisticated *non*-convex functionals, has recently be shown to converge globally in the convex case [13]. The scheme presented by Geman and Yang [12], known under the notion *half quadratic regularization*, has recently been extended by Cohen [14] to a large class of two-step algorithms for computer vision problems, and their scheme can be shown to converge globally under mild conditions, too (see section III-C below). The performance of these schemes, using efficient numerical solvers on parallel architectures for the linear systems of equations involved, has not been investigated so far.

### C. Contribution

In this paper, we adopt a general iterative scheme from the current literature on numerical multigrid methods [16] and characterize the schemes discussed in the previous section as its special cases. As a result of this mathematical characterization, we can slightly weaken the conditions derived by Charbonnier et al. [13] and show global convergence for both schemes. In addition, it turns out that for the case of *convex* variational approaches, the linearization technique introduced by Geman and Reynolds using auxiliary variables is identical to the so–called Kačanov method known from mathematical elasticity theory.

To assess the performance of these schemes, efficient Krylov subspace solvers for the resulting linear systems have been implemented under MPI [37] on a multi-processor SGI

[1]The notion "sufficient" depends on an upper bound for the Lipschitz constant of the gradient.

Power-Challenge machine. Empirical results concerning convergence rates and speed-up factors are reported.

### *D. Organization of the paper*

Section II describes the general form of convex functionals considered here along with a generic iterative scheme that can be shown to converge globally to the unique minimizer under certain conditions. Various linearization techniques known from the fields of image processing and mathematical elasticity theory are identified as instances of this scheme in section III. The conditions for global convergence are stated in each case. A discretization of the approaches, being consistent with the underlying continuous formulation, based on the Finite Element Method is sketched in section IV. Section V summarizes basic results from numerical linear algebra concerning the preconditioned Conjugate Gradient method. Experimental results concerning convergence rates and speed-up factors of our implementations on a multi–processor machine are reported and discussed in section VI. We conclude with suggestions for further work in section VII.

## II. Convex functionals and a general iterative minimization scheme

### *A. Problem statement: Minimizing non–quadratic convex functionals*

In the following, we focus on algorithms to efficiently compute functions $u \in \mathcal{H}$ minimizing functionals of the following form:

$$J(v) = \frac{1}{2} \int_{\Omega} (v - g)^2 + \lambda(|\nabla v|)\, dx \tag{1}$$

where $\Omega \subset \mathbf{R}^2$ denotes the image domain, and $\mathcal{H}$ denotes the standard Sobolev space $\mathcal{H} = H^1(\Omega) = W^{1,2}(\Omega)$. These functionals comprise two terms. The first term measures the similarity between admissible functions $v \in \mathcal{H}$ and the given image function $g$, and the second term measures the smoothness of $v$. Functionals of the form (1) are well-known from numerous papers on image segmentation and regularization (see, e.g., [2], [4], [18]).

Here, we restrict ourselves to functions $\lambda(t) \in C^1(\mathbf{R})$ which fulfill the following condi-

tions:

$$|\lambda(t)| \leq c_0\, t^2 \qquad \forall t \in \mathbf{R}^+, \quad c_0 > 0 \tag{2}$$

$$|\lambda'(t) - \lambda'(s)| \leq c_1\, |t - s| \quad \forall t, s \in \mathbf{R}^+,\ c_1 > 0 \tag{3}$$

$$\lambda'(t) \geq c_2\, t \qquad \forall t \in \mathbf{R}^+, \quad c_2 > 0 \tag{4}$$

We use the notation $\lambda'(t) = \frac{d\lambda(t)}{dt}$. These conditions guarantee [7] the convexity of the functional (1) and the existence of an unique function $u \in \mathcal{H}$ minimizing (1). Furthermore, $u$ is the unique solution to the non-linear variational equation [7]:

$$L(u, v) := \int_\Omega (u - g)v + \rho(|\nabla u|)\nabla u \cdot \nabla v\, dx = 0 \quad \forall v \in \mathcal{H}\,. \tag{5}$$

where:

$$\rho(t) := \frac{\lambda'(t)}{2t}\,. \tag{6}$$

*B. Convex functionals as edge–preserving regularizers*

The class of convex functionals obeying (1)–(4) comprises regularizing functions $\lambda(\cdot)$ that grow without bound as a function of the magnitude $|\nabla v|$ of the gradient of an admissible function at loci of image transitions like edges which delineate image regions. Hence, from the point-of-view of applications, one may ask whether such functions are useful for adaptive image processing.

To discuss this point in more detail, let us consider a representative example [7] that has been used for our work:

$$\lambda(t) = \begin{cases} \lambda_h^2 t^2 & ,0 \leq t \leq c_\rho \\ \lambda_l^2 t^2 + c_\psi(2t - c_\rho) & ,0 < c_\rho \leq t \end{cases} \tag{7}$$

with $c_\psi = (\lambda_h^2 - \lambda_l^2)c_\rho$ and $\lambda_h^2 \gg \lambda_l^2$. Figure 1 shows the corresponding function $\rho(\cdot)$ defined by (6) and (7).

Equation (7) illustrates that in general the convex functionals considered here combine standard quadratic regularizers $\lambda \sim t^2$ with regularizers growing at a *sub*–quadratic rate $\lambda \sim t^\alpha$, $1 \leq \alpha < 2$. In the particular case of (7), we have $\alpha = 1$.[2] Accordingly,

[2]The quadratic term corresponding to $\lambda_l \ll 1$ ensures strict convexity of the functional (1) over $\mathcal{H}$ but is dominated by the linear term.

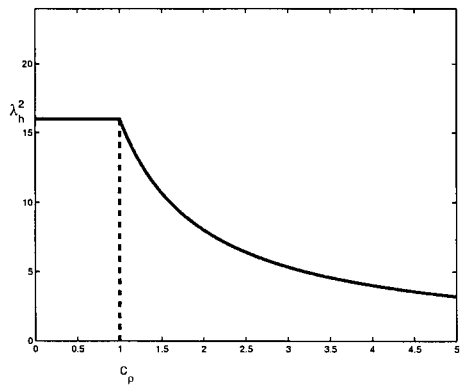


Fig. 1. Function $\rho(t)$ defined by the convex regularizing function $\lambda$ (7).

the parameter $\lambda_h$ in (7) determines the overall degree of smoothing within image regions $\{x \in \Omega : |\nabla v|(x) \leq c_\rho\}$ (cf. the constant part of the graph in Figure 1), whereas parameter $c_\rho$ controls the sensivity to image transitions. At signal transitions, the linear isotropic smoothing process becomes nonlinear and anisotropic, and adapts to the local image structure. This can be clearly seen by observing how the linear Laplacian operator $const \cdot \Delta v = \nabla \cdot (\rho \nabla v)$ corresponding to quadratic regularizers $\rho = const$ (cf. Fig. 1) decomposes at such locations [10]:

$$\nabla \cdot \big(\rho(|\nabla v|)\nabla v\big) = f_{\|}(|\nabla v|)\Big(\frac{d^2}{de_{\|}^2} v\Big) + f_{\perp}(|\nabla v|)\Big(\frac{d^2}{de_{\perp}^2} v\Big) \ ,$$

where

$$e_{\|} = \frac{\nabla v}{|\nabla v|} \ , \qquad e_{\perp} = \frac{(\nabla v)^{\perp}}{|\nabla v|}$$

denote the directions along and perpendicular to the image gradient $\nabla v$, respectively, and:

$$f_{\|}(t) = \begin{cases} \lambda_h^2 \ , & t \leq c_\rho \\ \lambda_l^2 \ , & t > c_\rho \end{cases} \ , \qquad f_{\perp}(t) = \rho(t) \ .$$

This clearly shows that convex regularizers indeed preserve edges: Smoothing almost stops across image transitions and gradually decreases along image transitions.

To illustrate this fact, Figure 2 depicts a numerical example where for reasons of comparison we used the same data as the authors of [9, Fig. 7]. In [9], these data were regularized

with a certain Markov Random Field by computing a (possibly local) minimum with a deterministic annealing approach, thus resulting in a (non–unique) solution of a computationally much more involved non–convex optimization problem. Figure 2 clearly shows that non–linear but convex regularization essentially does the same job, however with having the advantage of an unique and easy–to–compute solution. Figure 3 shows a result of our convex regularization as applied to a real image with a depiction of those image locations $\{x \in \Omega : |\nabla u| > c_\rho\}$ where the smoothing term adapted to image transitions (black pixels in the rightmost image of Fig. 3)

For a more complete discussion of convex regularization in this context and many numerical examples we refer to [10], [25].

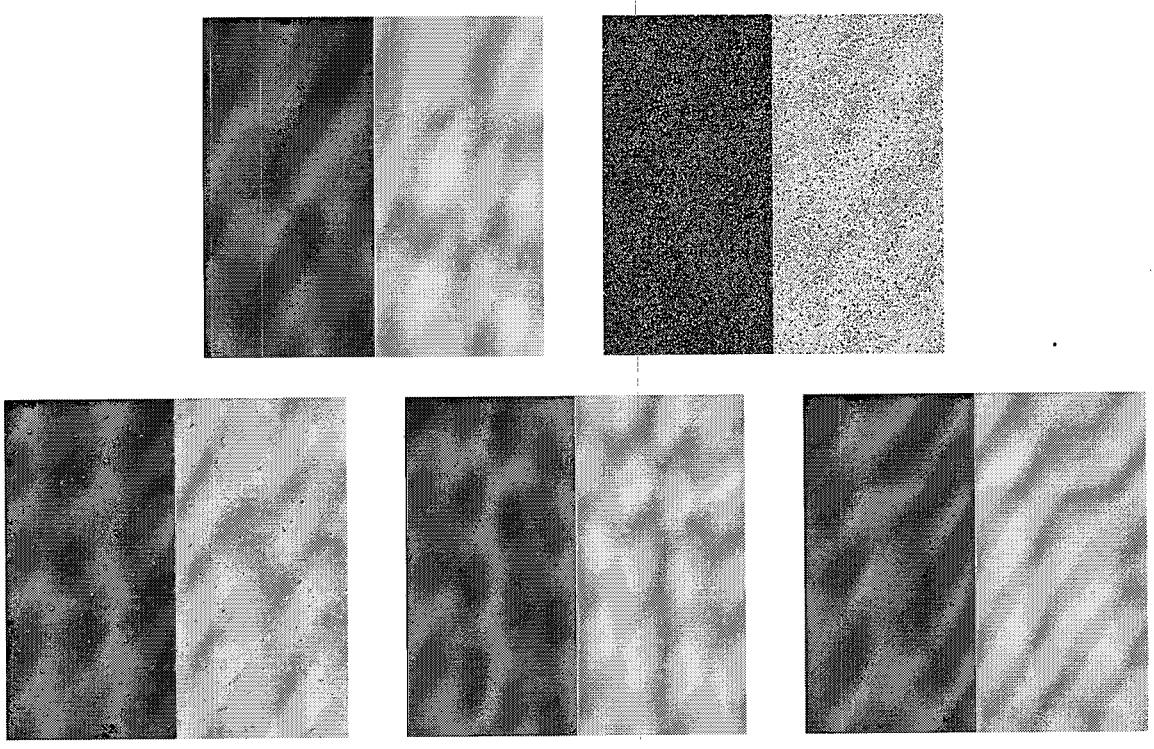

Fig. 2. Top: The same data as depicted in [9, Fig. 7]. Top, left: Step edge with 140 gray value units for the step. Top, right: White noise with standard deviation 30 gray value units has been added. Bottom, from left to right: Unique solution $u$ of nonlinear convex regularization for $\lambda_h = 7$, and $c_\rho = 0.5, 1.0, 2.0$, respectively. Note that (in this case) convex regularization performs as good as non–convex approaches.

For *image restoration* applications, an obvious modification of the class of functionals (1) is to include a linear operator $K$,

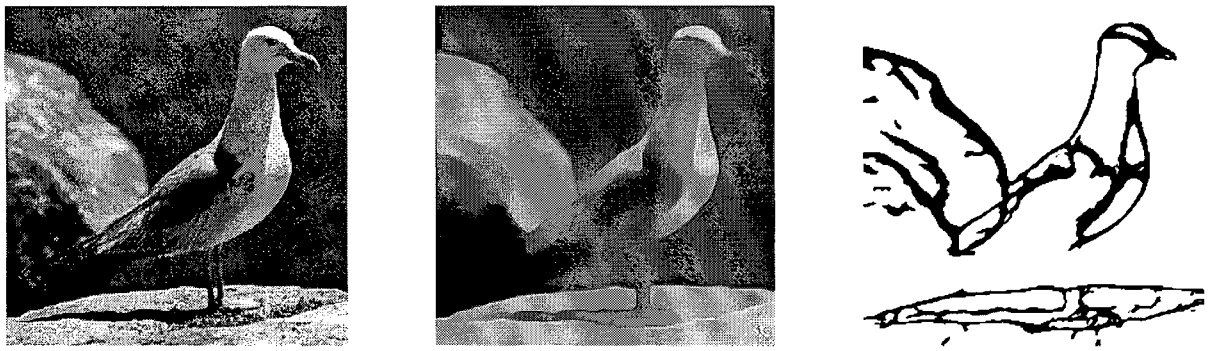

Fig. 3. Non–linear convex processing of a real image. Right: Image locations where the smoothing term adapted are labeled with black.

$$J(v) = \frac{1}{2}\int_{\Omega} (Kv - g)^2 + \lambda(|\nabla v|)\, dx$$

in order to account for blurring effects due to the point spread function of the imaging device. In this case, the approach (1) may be regarded as a generalized Tikhonov–Phillips regularization of a convolution equation. Although we have not yet elaborated such a modification (since we were mainly interested in computer vision applications rather than image restoration), we believe that the reported results can be carried over to variational image restoration applications.

Furthermore, we remark that the so–called total–variation measure $\lambda(|\nabla v|) = |\nabla v|$ has received considerable attention in *image denoising* applications [15], [22], [23]. Choosing a small value for the parameter $c_\rho$ in (7), the approach (1) may be considered as an approximation that can be conveniently evaluated numerically. Figure 4 provides an illustration.

### *C. A general iterative minimization scheme*

Our major objective is to investigate numerical schemes which globally converge to the unique minimizer $u$ of (1), irrespective of the point where the iteration starts. As discussed in section I, the design of such schemes is not straightforward, despite convexity of the functional (1).

In order both to compare and to unify several different linearization approaches in this context (see Section III below), we adopt the following general iterative scheme from

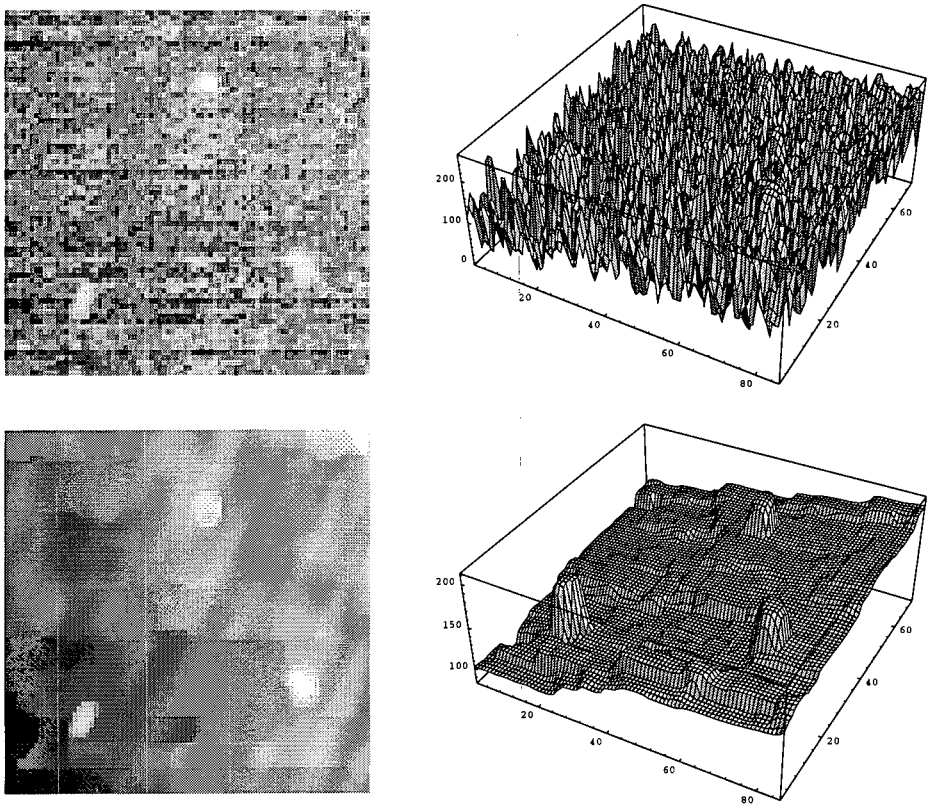


Fig. 4. Denoising a noisy medical image section using nonlinear convex regularization (see text).

Shaidurov [16]:

$$B(u^k; u^{k+1}, v) = B(u^k; u^k, v) - \omega_k L(u^k, v)\,, \tag{8}$$

where $k$ denotes the iteration number and $L$ is defined in (5). Note that for the case of fixed $u^k$, equation (8) takes the form $a(u^{k+1}, v) = f(v)$ which has a unique solution under assumptions stated in the Lax–Milgram Lemma (see, e.g., [17]). Accordingly, in order to compute a unique solution $u^{k+1}$ of this equation at every iteration step, we require the following properties of the operator $B : \mathcal{H}^3 \to \mathbf{R}$ in (8):

- $B(u; \cdot, \cdot)$ is bilinear $\forall u \in \mathcal{H}$
- $B(u; \cdot, \cdot)$ is symmetric, i.e. $B(u; v, w) = B(u; w, v) \quad \forall u, v, w \in \mathcal{H}$
- $B(u; \cdot, \cdot)$ is $\mathcal{H}$-elliptic, i.e. $B(u; v, v) \geq \delta \, ||v||^2_{\mathcal{H}} \quad \forall u, v \in \mathcal{H},\ \delta > 0$ .

The scalar damping factor $\omega_k > 0$ gives an additional degree of freedom for the purpose of controling the convergence behavior of (8). Specific formulations of the general iterative scheme (8) will be considered in the next section.

## III. Linearization techniques and global convergence

In this section, we show that the general scheme (8) subsumes quite diverse linearization schemes from the literature. Hence, this framework sheds some light on the mathematical basis for globally convergent iterative minimization schemes for convex nonlinear regularization approaches.

### *A. Linearization with the Kačanov method*

A basic idea, known as the *Kačanov method* from elasticity theory since more than two decades [19], [20], [21], is to linearize the non-linear equation (5) by "freezing" its non-linear part $\rho(t)$ for one iteration step. To be more specific, we introduce the notation:

$$b(v) = \int_\Omega gv\,dx \tag{9}$$

$$B_1(u;v,w) = \int_\Omega vw + \rho(|\nabla u|)\nabla v \cdot \nabla w\,dx \tag{10}$$

and formulate the Kačanov iteration and the conditions under which global convergence holds true:

*Lemma III.1:* Assume (9), (10) and (2)-(4) to hold. If the function $\rho(t)$ is bounded, continuous and monotonously decreasing, then the sequence $\{u^k\}_{k\geq 1}$ determined by

$$B_1(u^k;u^{k+1},v) = b(v) \quad \forall v \in \mathcal{H} \tag{11}$$

converges to the unique solution $u$ of (5), for any initial $u^0 \in \mathcal{H}$.

A general proof can be found in [21], based on assumptions specified in appendix A. To complete the proof, it remains to be checked whether these assumptions are valid for our case as stated in (1) and (5) (see appendix A).

We conclude this section by recognizing that the iteration (11) is a special case of the general iterative scheme (8) with operator $B = B_1$, $b(v) = B_1(u;u,v) - L(u,v)$, and damping factor $\omega_k = 1$:

$$B_1(u^k;u^{k+1},v) = B_1(u^k;u^k,v) - L(u^k,v)\,. \tag{12}$$

In the next section, we turn to a more general linearization technique comprising the Kačanov method as a special case. However, the conditions for global convergence stated

in Lemma III.1 are then slightly weaker than those stated in the recent literature (see next section).

*B. Adaptive linearization with auxiliary variables*

Geman and Reynolds [11] suggested an edge-preserving smoothing approach for solving non-convex image restoration problems based on a specific use of auxiliary variables. Charbonnier et al. [13] investigated this scheme further and presented a proof of global convergence for the case of convex functionals. An application of the approach of Geman and Reynolds to the restoration of non-smooth functions[3] has been investigated by Dobson [22], who also derived an iterative scheme for this case similar to (11).

In [11], auxiliary variables $w$ are introduced by replacing (1) with the functional:

$$J_A(v,w) = \frac{1}{2}\int_\Omega (v-g)^2 + w|\nabla v|^2 + \psi(w)\,dx\,.$$

For certain functions $\lambda(t)$ in (1) one can find functions $\psi(t)$ [11], [13], for which $J_A$ is convex with respect to $w$ and

$$J(v) = \inf_w J_A(v,w)\,. \tag{13}$$

Thus, $J(v)$ in (1) can be conveniently minimized by the two-step iteration:

$$\begin{aligned} w^k &= \arg\min_w J_A(u^k, w) \\ u^{k+1} &= \arg\min_u J_A(u, w^k)\,. \end{aligned} \tag{14}$$

Furthermore, if $\lambda(t)$ is convex and $\rho(t) \in C^1(\mathbf{R})$ is bounded as well as *strictly* monotonously decreasing, Charbonnier et al. [13] have shown that i) the above two-step minimization procedure (14) converges to the unique minimizer $u$ of the convex functional $J(v)$ in (1), and that ii) the auxiliary variables are computed (i.e. the first step of (14)) as:

$$w^k = \rho(|\nabla u^k|)\,. \tag{15}$$

Variational calculus shows that the second step in (14) explicitly reads:

$$\int_\Omega (u^{k+1} - g)v + w^k \nabla u^{k+1} \cdot \nabla v\,dx = 0\,. \tag{16}$$

[3] "non-smooth" here means that the solution space is $L^2(\Omega)$ rather than some Sobolev space.

After discretization (see section IV), $u^{k+1}$ can be computed as the solution of a linear system of equations which, however, has to be updated at every iteration step. For this reason we call this linearization technique *adaptive*.

Substitution of (15) into (16) shows that the method of Geman and Reynolds reduces in the convex case just to the Kačanov method (11). However, the conditions for global convergence, as stated in lemma III.1, are slightly weaker than those stated in [13]. In particular, global convergence of (11) holds true for functions $\rho(t)$ which are (not necessarily strictly) monotonously decreasing and (merely) continuous. This, for example, includes the function $\rho(t)$ depicted in Figure 1. The constant part of this function up to some value $t = c_\rho$ ensures homogeneous smoothing within image regions $\{x \in \Omega : \ |\nabla u(x)| \leq t\}$, which is only approximately the case for strictly monotonously decreasing functions as discussed in [13].

### *C. Non-adaptive linearization using auxiliary variables*

We turn now to a method proposed by Geman and Yang under the notion *half quadratic regularization* [12]. Cohen [14] has extended this approach to a broad class of two-step algorithms for computer vision problems. In this section, we apply this method to the specific class of minimization problems as denoted by (1).

We introduce auxiliary variables $w$ by replacing (1) with the functional:

$$J_{NA}(v, w) = \frac{1}{2} \int_\Omega (v - g)^2 + \alpha |\nabla v - w|^2 + \psi(w)\, dx\,. \tag{17}$$

In appendix B we show that for $\lambda(t)$ satisfying (2)-(4), we can find functions $\psi(t)$ for which $J_{NA}$ is convex in $w$, and

$$J(v) = \inf_w J_{NA}(v, w)\,. \tag{18}$$

Analogously to the method described in the last section, minimizing the original functional $J(v)$ in (1) can be achieved through the two-step iteration:

$$\begin{aligned} w^k &= \arg\min_w J_{NA}(u^k, w) \\ u^{k+1} &= \arg\min_u J_{NA}(u, w^k) \end{aligned} \tag{19}$$

The conditions for global convergence of this two-step minimization are given in the following lemma:

*Lemma III.2:* Assume (9)-(10) and (2)-(4) to hold. If the function $\rho(t)$ is bounded, continuous and monotonously decreasing for $t \geq 0$, then there exists an $\alpha > 0$, for which the iterates $u^k$ from (19) converge to the unique solution $u$ of (5), for any $u^0 \in \mathcal{H}$. The auxiliary variables are computed by

$$w^k = \left(1 - \frac{1}{\alpha}\rho(|\nabla u^k|)\right) \nabla u^k \,. \tag{20}$$

For a proof, see appendix B. Variational calculus shows that the second step in (19) reads:

$$\int_\Omega (u^{k+1} - g)v + \alpha(\nabla u^{k+1} - w^k) \cdot \nabla v \, dx = 0 \quad \forall v \tag{21}$$

After discretization (see section IV), $u^{k+1}$ can be computed as solution of a linear system of equations. In contrast to the last section, however, this linear system does *not* change during the iteration. Hence, computational steps related to the acceleration of convergence (preconditioning) can be carried out in advance and off–line. For this reason we call this linearization technique *non-adaptive*.

Again, this method can be formulated as a special case of the general scheme (8). Substituting (20) in (21), we obtain:

$$\int_\Omega u^{k+1}v + \alpha\nabla u^{k+1} \cdot \nabla v \, dx = \int_\Omega gv - \rho(|\nabla u^k|)\nabla u^k \cdot \nabla v + \alpha\nabla u^k \cdot \nabla v \, dx \tag{22}$$

Defining the operator

$$B_2(u, v) := \int_\Omega uv + \alpha\nabla u \cdot \nabla v \, dx \,, \tag{23}$$

the two-step minimization (19) thus reads

$$B_2(u^{k+1}, v) = B_2(u^k, v) - L(u^k, v), \quad \forall v \in \mathcal{H} \,, \tag{24}$$

which is a special case of the general scheme (8) for $B = B_2$ and $\omega_k = 1$. Comparing the definitions (10) and (23) reveals[4]:

$$B_2(u, v) \approx B_1(0; u, v) \,.$$

This clearly shows how the updating of the system of equations at every iteration step as described above drops out from this method.

[4] "$\approx$" means that the constant $\rho(0)$ becomes $\alpha$.

## IV. Discretization

In this section, we apply the Finite Element Method (FEM) for the purpose of discretizing the general iterative scheme (8). After a sketch of the basic idea underlying FEM, we explicitly describe the case of 2D gray-value images to facilitate a parallel implementation of the instances (12), (24), or others which might be derived by the reader from (1). Algorithms to solve the resulting systems of linear equations at each iteration step is the objective of the subsequent section.

### *A. The Finite Element Method*

For the discretization of variational problems, FEM is the natural choice. FEM can be applied in a mechanistic way, boundary conditions are incorporated automatically, and the resulting discrete formulation is consistent in the sense that, under certain conditions [7], discrete solutions $u_h$ to (5), for example, converge to the continuous solutions $u$ for vanishing mesh width. In this sense, a discrete formulation of a variational problem attempts to approximate favorable properties of the underlying continuous problem formulation, like rotational invariance of smoothness terms, for example. For a thorough introduction to FEM we refer to, e.g., [24].

An alternative and equally valid way to discretize our problem is i) to apply Finite Differences to the Euler–Lagrange equation which corresponds to the variational equation (5), and ii) to take care of the natural boundary conditions at the boundary $\partial\Omega$ of the underlying domain $\Omega$. Note that the latter is not difficult for well–shaped domains $\Omega$ like rectangular image domains, say, but may become quite cumbersome for irregularly shaped domains $\Omega$ which, for example, an user has specified interactively in some medical image analysis application. By contrast, the FEM relieves one of such particularities, thus it should be preferred over Finite Differences.

The basic idea of the FEM is the restriction of optimization problems to finite-dimensional subspaces. Let $\{\phi_1, ..., \phi_n\}$ denote basis functions of a finite-dimensional subspace $\mathcal{H}_h \subset \mathcal{H}$. Then, the restriction of (5) to $\mathcal{H}_h$ reads:

$$L(u_h, \phi_i) = 0\,, \quad \forall i = 1, ..., n\,, \tag{25}$$

with the unique minimizer $u_h \in \text{span}\{\phi_1, ..., \phi_n\}$. If we define the isomorphism

$$I : \mathbf{R}^n \to \mathcal{H}_h; \quad \mathbf{u} \to u_h = \sum_j u_j \phi_j \,, \tag{26}$$

and the mappings

$$\mathbf{L}_i(\mathbf{u}) := L(I(\mathbf{u}), \phi_i) \,, \quad \mathbf{u} = (u_1, ...u_n)^T \,, \tag{27}$$

then the solution of (25) is equivalent to the solution of the non-linear system:

$$\mathbf{L}(\mathbf{u}) = \mathbf{0} \,, \quad \mathbf{L} = (\mathbf{L_1}, ..., \mathbf{L_n})^\mathbf{T} \,. \tag{28}$$

As mentioned above, we know from FEM theory that the solutions $u_h$ converge to the solution $u$ of (5), if the formal discretization parameter $h$ goes to zero ([7], [24]).
As a result, we obtain from (8) the discrete iteration scheme:

$$\mathbf{B}^k \mathbf{u}^{k+1} \;=\; \mathbf{B}^k \mathbf{u}^k - \omega_k \mathbf{L}(\mathbf{u}^k) \tag{29}$$

The matrices are computed as

$$\mathbf{B}^k_{i,j} = \mathbf{B}(I(\mathbf{u}^k); \phi_i, \phi_j) \,. \tag{30}$$

Since the operators $B(u^k; \cdot, \cdot)$ are bilinear, symmetric and $\mathcal{H}$-elliptic, the matrices $\mathbf{B}^k$ are also bilinear, symmetric and positive definite.

Note that if $\mathbf{B}^k$ equals the Jacobian of $\mathbf{L}(\mathbf{u}^k)$ and $\omega_k = 1$, (29) is nothing but the Newton method. In this sense, the iteration (29) with the matrices $\mathbf{B}^k$ resulting from (30) can be understood as approximations of the Newton method while *preserving global convergence.* Furthermore, the *non-linear Jacobi method*, i.e. parallel gradient descent, is obtained by choosing $\mathbf{B}^k = \mathbf{I}$.

*B. The case of 2D gray-value images*

Now we apply the approach outlined in the previous section to the case of two-dimensional (2D) gray-value images, using piecewise linear basis functions. The first step is to triangulate the underlying image domain, in this case the rectangular area $\Omega = [0, n] \times [0, m]$, as illustrated in Figure 5. Next, we assign to each mesh node $P_{i,j}$ a basis function $\phi_{i,j}$ which

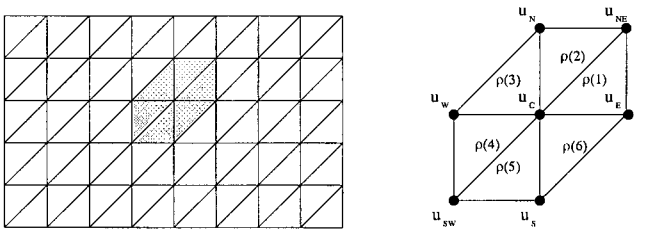


Fig. 5. Left: Triangulation of a rectangular image domain $\Omega$ with mesh width 1. The nodes correspond to pixel positions. Right: An interior node $u_C$ with adjacent triangles $d_i$ and the corresponding $\rho$-values (for piecewise linear basis functions $\rho(k) = \rho(|\nabla u|) = const.$ at each triangle $d_k$).

is uniquely defined by the following conditions:

$$\begin{aligned} &\phi_{i,j}(x) && \text{is linear within each triangle } d_k \ , \\ &\phi_{i,j}(x) = 1 && \text{at node } P_{i,j} \ , \\ &\phi_{i,j}(x) = 0 && \text{at every node } P_{k,l} \neq P_{i,j} \ . \end{aligned}$$

Discrete gray-value images $\mathbf{u}, \mathbf{v}, \mathbf{g}$, etc. are represented as elements of the subspace $\mathcal{H}_h$ by simply interpolating the values of corresponding nodal variables $u_{i,j}$ in a piecewise linear fashion:

$$I : \mathbf{R}^{n \times m} \to \mathcal{H}_h \ , \quad \mathbf{u} \to \sum_{i,j} u_{i,j} \phi_{i,j} \ ,$$

and similarly for $\mathbf{v}, \mathbf{g}$. From (26) and (27) we thus obtain:

$$\mathbf{L}_{i,j}(\mathbf{u}) = \sum_{k,l} (u_{k,l} - g_{k,l}) \int_\Omega \phi_{k,l} \phi_{i,j} \, dx + \sum_{k,l} u_{k,l} \int_\Omega \rho(|\nabla u_h|) \nabla \phi_{k,l} \cdot \nabla \phi_{i,j} \, dx \qquad (31)$$

These integrals vanish for all pairs of nodes $(i,j)$ and $(k,l)$ for which the intersection of the support of the corresponding basis functions $\phi_{ij}$ and $\phi_{kl}$ is empty. The remaining integrals can be computed analytically to obtain a sparse system of non-linear equations in terms of the nodal variables of the solution $\mathbf{u}$. For additional details and applications to different variational problems we refer to [25].

The terms of (31) are weighted sums, the coefficients of which can be conveniently depicted as *stencils*. Figure 6 shows the linear and non-linear stencils computed for interior mesh points. The necessary modifications of these stencils at boundary points are automatically obtained by taking into consideration the correct domain of integration in (31).

We use these stencils to rewrite the non-linear equation (31) in the more suggestive form:

$$\mathbf{L(u)} = \frac{\mathbf{1}}{\mathbf{12}}\mathbf{D_{2D}} * (\mathbf{u} - \mathbf{g}) + \frac{\mathbf{1}}{\mathbf{2}}\mathbf{R_{2D}} * \mathbf{u}\,. \tag{32}$$

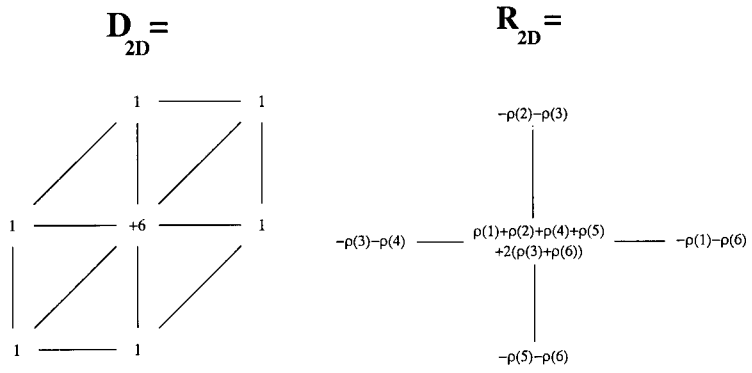


Fig. 6. Stencils for interior nodes resulting from FEM discretization. Left: The stencil for the linear data-fitting term. Right: The stencil for the non-linear smoothing term.

## V. The Inexact Conjugate Gradient Method and Preconditioning

Next we briefly describe some concepts from numerical mathematics which are basic to the solution of the discrete approaches discussed above.

### *A. The Conjugate Gradient (CG) Method*

Discretization of the approaches described in Section III using the FEM leads to the problem of solving (sequences of) numerically sparse systems of linear equations:

$$\mathbf{Ax} = \mathbf{b}\,. \tag{33}$$

Due to the problem size ($\mathbf{A}$ is a $n \times n$ matrix, $n$ being the number of pixel positions) direct methods (such as Gauss elimination or LU decomposition) are not feasible, since i) they produce *fill-in* and ii) the computational cost as well as the demand for memory become prohibitive. Consequently, we have to focus on iterative methods that preserve the sparse problem structure.

If $\mathbf{A}$ in (33) is symmetric and positive definite, the well-known CG (conjugate gradient) method (along with preconditioning; see next section) is a good choice in general. Alter-

natives are discussed in sections V-C and VII.

Defining the residuals

$$\mathbf{r}^k = \mathbf{A}\mathbf{x}^k - \mathbf{b}\,,$$

the CG method reads [26]:

$$\begin{aligned}
\beta_k &= \frac{(\mathbf{r}^k)^T\mathbf{r}^k}{(\mathbf{r}^{k-1})^T\mathbf{r}^{k-1}} \qquad (\beta_0 := 0)\\
\mathbf{p}^k &= \mathbf{r}^k + \beta_k\mathbf{p}^{k-1}\\
\alpha_k &= \frac{(\mathbf{r}^k)^T\mathbf{r}^k}{(\mathbf{p}^k)^T\mathbf{A}\mathbf{p}^k}\\
\mathbf{x}^{k+1} &= \mathbf{x}^k + \alpha_k\mathbf{p}^k\\
\mathbf{r}^{k+1} &= \mathbf{r}^k + \alpha_k\mathbf{A}\mathbf{p}^k
\end{aligned}$$

At every iteration step one only needs one matrix-vector multiplication, two scalar vector multiplications and three vector updates. For details on convergence we refer to, e.g., [27], [28].

The CG method along with preconditioning (see next section) works nearly optimal for the case of 2D images. Results from parallel implementations are reported in the next section. We encountered numerical instabilities, however, in the case of large 3D medical images. The reason is the huge number of variables resulting in a bad condition number, although preconditioning was applied.

At this point it is useful to point out that the CG method is a special case of Krylov subspace methods which have been developed during the last 30 years ([26], [29], [30], [31]). A more general Krylov subspace method that helped us to avoid numerical problems in the above mentioned case of large 3D images, is the GMRES method [29]. However, the investigation of the application of this method to the problem class considered here is beyond the scope of this paper.

*B. Inexact Conjugate Gradient Method*

Solving large linear systems of equations is the primary computational task in solving non-linear equations by the methods described above. To minimize the effort for solving these equations, we adopt the concept of the Inexact Newton Method [32].

The computational cost for solving linear equations with the CG method depends strongly on an error bound as stopping criterion. In general, the CG method is controlled by the threshold $||\mathbf{r}^0||_2 \cdot rtol + atol$, where $\mathbf{r}^0 = \mathbf{A}\mathbf{x}^0 - \mathbf{b}$ is the initial *linear* residual, *rtol* controls its relative reduction, and *atol* controls its absolute reduction. In our investigations, the absolute tolerances are not relevant, i.e. we set $atol = 0.0$. "Inexact" refers to choosing a fairly large value for the parameter *rtol*, 0.1 say.

As a consequence, the inner loop in (29) of the two–step minimization approaches discussed in Section III stops after 2–3 iteration steps of the CG method, hence is very fast. Surprisingly, it turned out through our experiments that convergence of the overall iteration, measured using the *non–linear* residuals $\mathbf{L}(\mathbf{u^k})$ (32) of the discretized equation (5), still holds true for $rtol \leq 0.1$. We have no proof of this fact so far, but presume that for each of the approaches (maybe under additional assumptions) there is some upper bound $rtol \leq rtol_{max}$ ensuring convergence.

### *C. Preconditioning*

To improve the condition number and, in turn, convergence speed, preconditioning has to be applied. To this end, (33) is replaced by:

$$(\mathbf{L}^{-1} \cdot \mathbf{A} \cdot \mathbf{R}^{-1}) \cdot \mathbf{R}\,\mathbf{x} = \mathbf{L}^{-1}\,\mathbf{b}\,. \tag{34}$$

The matrices $\mathbf{R}$ and $\mathbf{L}$ are called preconditioners. Special cases are the *left preconditioning* ($\mathbf{R} = \mathbf{I}$) and the *right preconditioning* ($\mathbf{L} = \mathbf{I}$). If $\mathbf{L} \cdot \mathbf{R} \approx \mathbf{A}$, one can expect a reduction of the condition number. To avoid loss of sparsity, the matrix $\mathbf{L}^{-1}\mathbf{A}\mathbf{R}^{-1}$ should not be computed. Rather, the preconditioners $\mathbf{L}$ and $\mathbf{R}$ are chosen such that the equations $\mathbf{R}\mathbf{u} = \mathbf{w}$ and $\mathbf{L}\mathbf{u} = \mathbf{w}$ can be easily solved (i.e. as triangular matrices). Thus, in every iteration step of the CG method one or two "easy" linear equations have to be solved.

Classical preconditioners are obtained through either an additive matrix splitting $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$ (Jacobi or S(S)OR-preconditioning) or a multiplicative matrix splitting $\mathbf{A} = \mathbf{L} \cdot \mathbf{D} \cdot \mathbf{U}$ (ILU or ICC factorizations) [33], [34]. The approach of Domain Decomposition is an alternative way of preconditioning (Block-Jacobi or Block-Gauss-Seidel, for example), which is more suited for parallel implementations, however, as demonstrated below.

## VI. Experimental results

In this section, we first sketch details of our parallel implementation. Next, we summarize our experimental results through mainly focussing on two aspects: Efficiency of the various iterative schemes discussed so far in terms of computation time, and speed-up caused by using multiple processing units.

### *A. Parallel implementation*

For the implementation of the approaches discussed in Section III, discretized as described in Section IV, we use the software package PETSc (Portable Extensible Toolkit for Scientific Computing; [35], [36]) which is based on the Message Passing Standard (MPI; [37]). PETSc provides special methods and data types for solving equations arising from PDE's in a parallel fashion. PETSc supports in both parallel and sequential working mode for example:

- data types for vectors and matrices
- distributed arrays and index sets
- Krylov subspace solver for linear equations
- preconditioner
- time-stepper

PETSc also supports implementations on different computer architectures (for example: SGI PowerChallenge with multiple processing units or SUN workstation cluster) and enables investigations of different numerical methods.

In our implementation, iteration (29) is carried out in three steps:

$$\mathbf{r}^k = \mathbf{L}(\mathbf{u}^k) \tag{35}$$

$$\mathbf{d}^k = (\mathbf{B}^k)^{-1}\mathbf{r}^k \tag{36}$$

$$\mathbf{u}^{k+1} = \mathbf{u}^k - \omega_k \mathbf{d}^k \tag{37}$$

Distributed arrays, provided by PETSc, are used for partitioning rectangular image domains into rectangular subdomains. These subdomains are mapped on parallel vector data types, which are distributed over a range of processing units. Additional overlapping domains are used to facilitate inter–process communication (see Figure 7) that is necessary for computing the residuals $\mathbf{L}(\mathbf{u}^k)$ and compiling the matrices $\mathbf{B}^k$.

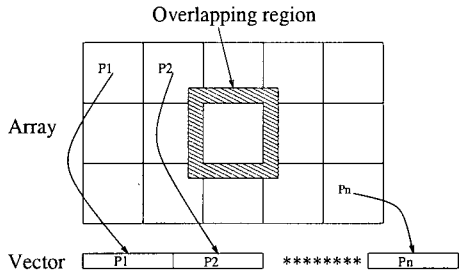


Fig. 7. Partitioning and distributing a rectangular array on multiple processing units Pn using distributed arrays. Overlapping domains are used for inter–process communication.

### *B. Numerical results and discussion*

In this Section we discuss numerical experiments using the linearization approaches presented in Section III:

- adaptive linearization,
- non–adaptive linearization,
- and, as a further reference, the non–linear Jacobi method.

We recall from Section III that adaptive linearization is equivalent to the Kačanov method applied to the convex functional (1), and that the non–linear Jacobi method corresponds to iterative gradient descent.

The performance of these approaches depends on various aspects, each of which are discussed next, but to a negligible amount only on the particular image being processed. Hence it suffices to summarize and discuss our results for some "general" image like that depicted in Figure 3. Apart from the experiments with varying image size in Section B.3, the results reported were computed for images of size $256 \times 256$ pixels.

#### B.1 *Convergence speed*

The three iterative schemes listed above differ considerably in the computational cost for *a single* iteration step: Adaptive linearization requires the inversion of a linear system that has to be re–compiled at each iteration step. The step of solving the linear system corresponding to non–adaptive linearization can be optimized once and for all before the iteration starts. The non–linear Jacobi iteration, finally, just requires the evaluation of (32). An interesting question therefore is whether adaptivity is payed off by faster

convergence.

Figure 8 shows how the implemented numerical schemes reduce the initial non-linear residual $\mathbf{r}(t_0) = ||\mathbf{L}(\mathbf{u}^0)||_2$ as a function of the computation time. It can be seen that adaptive linearization performs best, followed by the non-linear Jacobi method. The non-adaptive linearization performs worst, because the smaller number of iteration steps required cannot compensate for the additional computation cost (w.r.t. non-linear Jacobi) caused by solving the linear system at each iteration step.

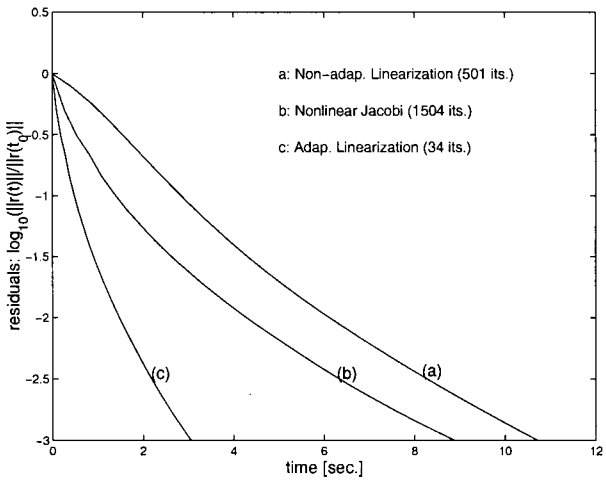


Fig. 8. Decrease of the *non-linear* residuals as a function of computation time. Computed on a SGI PowerChallenge with 16 processing units ($c_\rho = 1.0$, $\lambda_h = 4.0$, CG method with $rtol = 0.1$ and Block-Jacobi preconditioning).

## B.2 *Influence of using an inexact linear solver*

Figure 9 shows the influence of the accuracy parameter $rtol$ of the CG method on convergence. The computational cost decreases considerably for increasing $rtol$, and our experiments have shown that monotone convergence is preserved for $rtol \leq 0.1$ This fact has already been discussed in Section V-B.

## B.3 *Influence of image size and parameters controlling the non–linear regularizer*

The convergence rates of the numerical schemes also depend on the size of the sample images in our experiments as well as on the parameters $c_\rho$ and $\lambda_h$. As expected, there is a typical linear relation between image size and computational cost depicted in Figure 10.

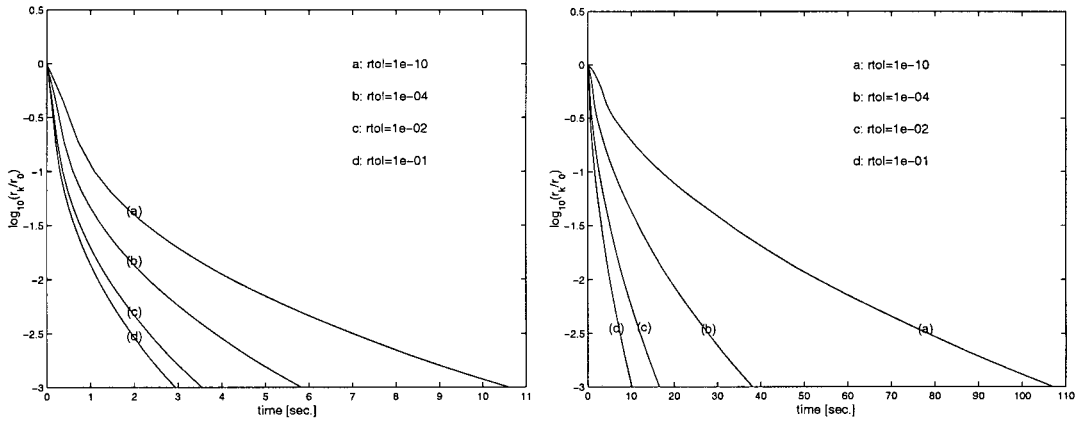


Fig. 9. Convergence of the adaptive (left) and non-adaptive linearization (right) for various accuracies *rtol* of the linear solver. Computed on a SGI PowerChallenge with 16 processing units ($c_\rho = 1.0$, $\lambda_h = 4.0$, CG method with Block-Jacobi preconditioning).

Concerning the parameters controlling regularization, the computational cost increase for decreasing $c_\rho$ and increasing $\lambda_h$, respectively (see Figures 11 and 12). Both relations are reasonable: Larger $\lambda_h$ means more smoothing whereas smaller $c_\rho$ results in higher sensivity against image transitions, i.e. the process becomes "more non–linear". Clearly, for a large enough value of $c_\rho$ the non-linear regularizer ignores image transitions completely, hence becomes linear in fact and the iteration terminates after one step.

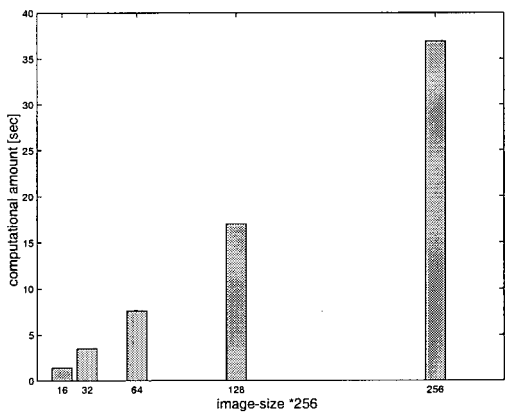


Fig. 10. Computational cost as a function of image size. Computed on a SGI PowerChallenge with one processing unit (adapt. linearization, $c_\rho = 1.0$, $\lambda_h = 4.0$, CG method with $rtol = 0.1$ and Block-Jacobi preconditioning).

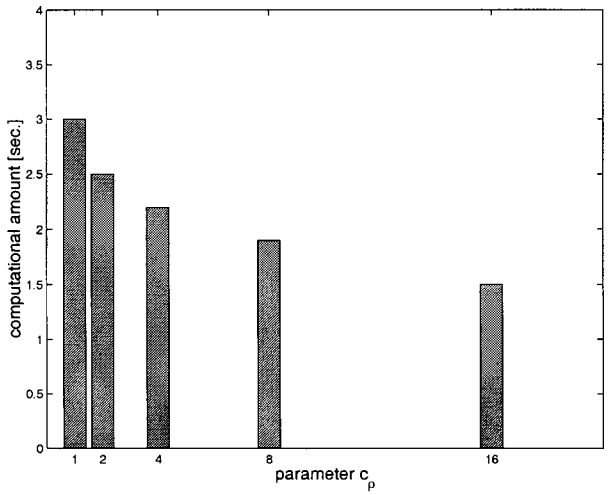


Fig. 11. Computational cost as a function of $c_\rho$. Computed on a SGI PowerChallenge with 16 processing units (adapt. linearization, $\lambda_h = 4.0$, CG method with $rtol = 0.1$ and Block-Jacobi preconditioning).

### B.4 *Inexact linear solver vs. preconditioning*

The well–known effect of classical preconditioning of the CG method is depicted in Figure 13. If we solve the linear equations "exactly", i.e $rtol \leq 10^{-6}$, the use of ILU or Block-Jacobi preconditioning performs best (Fig. 13, left). However, if we solve the linear equations inexactly, i.e. $rtol \approx 10^{-1}$, the strong effect of using an adequate preconditioner more or less disappears (Fig. 13, right). Note that using no preconditioner at all performs nearly as good in this case as using ILU preconditioning. We further remark that, in contrast to the more complicated ILU method, parallelization of the simple Block-Jacobi method is straightforward and performs best in this case (i.e. using an inexact solver), too.

### B.5 *Speed–up by using multiple processing units*

Figure 14 demonstrates a significant speed-up as a function of the number of processing units. This result indicates (i) a nearly optimal implementation of the iterative *linear* solvers involved and (ii), as a consequence, that the main remaining problem concerns the convergence rate of the *outer* iteration loop (29).

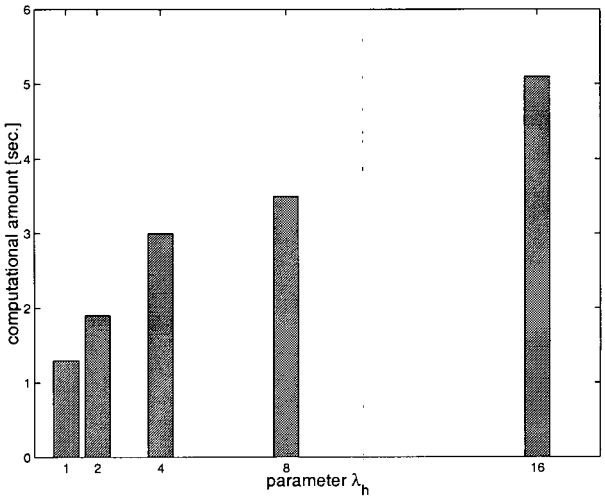


Fig. 12. Computational cost as a function of $\lambda_h$. Computed on a SGI PowerChallenge with 16 processing units (adapt. linearization, $c_\rho = 1.0$, CG method with $rtol = 0.1$ and Block-Jacobi preconditioning).

## VII. Further work

Further work may be conducted in several directions. One concerns the classical preconditioner used to solve the linear systems. As Figure 13 shows, a considerable improvement has been achieved by the (currently used) Block-Jacobi preconditioner. This method can be understood as a primitive form of a domain decomposition method. We are currently working on further improvements in this direction.

A second direction concerns a better theoretical foundation of the *inexact* linearization methods, since a proof of global convergence in these cases is lacking.

Finally, better approximations of Newton-like methods under the condition of global convergence should be sought for because these improved approximations can be expected to reduce further the number of iteration steps required in the outer loop of two–step minimization approaches.

Acknowledgment

J. Heers gratefully acknowledges the support by the German National Science Foundation (DFG) under grant Sti 147/1–2.

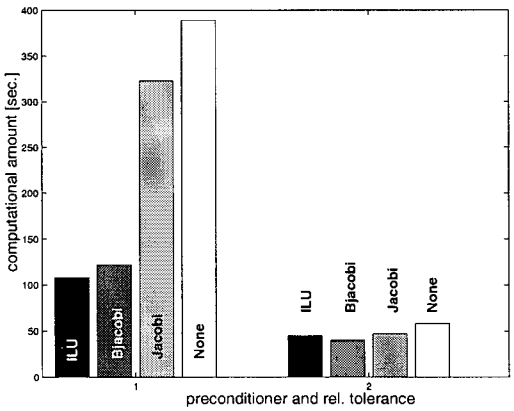


Fig. 13. Computational cost of different preconditioners for exact (left: $rtol = 10^{-6}$) and inexact linear solver (right: $rtol = 10^{-1}$). Computed on a SGI PowerChallenge with one processing unit (adapt. linearization, $c_\rho = 1.0$, $\lambda_h = 4.0$, CG method).

## Appendix A

*Proof of lemma III.1*

We show that the functional (1) fulfills the conditions for convergence of the Kačanov method (theorem 25.L in [21]). These conditions are:

*a)* $B_1(u;\cdot,\cdot)$ is bilinear, symmetric, and $\mathcal{H}$-elliptic for all $u \in \mathcal{H}$.
*b)* $L(u,v) = B_1(u;u,v)$
*c)* $J(v) - J(u) \leq \frac{1}{2}[B_1(u;v,v) - B_1(u;u,u)]$

Item a) holds true because $\rho(t)$ is positive and bounded. Item b) is obviously true. To show that item c) holds true, we prove:

$$\frac{1}{2}[B_1(u;v,v) - B_1(u;u,u)] - (J(v) - J(u))]$$
$$= \rho(|\nabla v|)(|\nabla u|^2 - |\nabla v|^2) - (\lambda(|\nabla v|) - \lambda(|\nabla u|)) \geq 0\,,$$

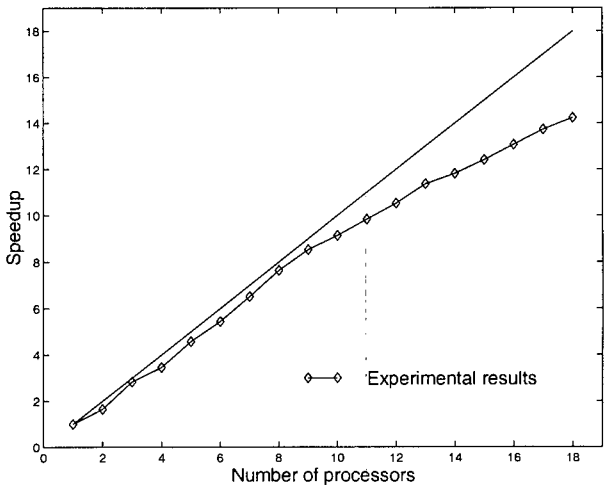


Fig. 14. Speed-up factor as a function of processing units, which results from using up to 18 processing units in parallel (maximal number available for the authors).

for all $u, v$. We compute the last term of the right hand side:

$$\begin{aligned}\lambda(t) - \lambda(s) &= \int_0^1 \frac{\partial}{\partial p}\lambda(s + p(t-s))\, dp \\ &= \int_0^1 \lambda'(s + p(t-s))(t-s)\, dp \\ &= \int_0^1 2\rho(|s + p(t-s)|)(s + p(t-s))(t-s)\, dp\end{aligned}$$

By virtue of (4), $\lambda(t)$ is monotonously increasing for $t > 0$ and we get $\lambda(t) - \lambda(s) \lesseqgtr 0$, if $t - s \lesseqgtr 0$. Furthermore, $0 < \rho(t)$ is monotonously decreasing. Hence we obtain:

$$\lambda(t) - \lambda(s) \leq \int_0^1 2\rho(s)(s(t-s) + p(t-s)^2)\, dp = \rho(s)(t^2 - s^2)$$

This completes the proof of convergence of the Kačanov method. □

## Appendix B

To show the existence of the function $\psi$ in (17), we need the concept of conjugate functions from convex analysis ([14], [38]):

*Definition VII.1:* If $\phi$ is a function from an euclidian space $\mathcal{E}$ to $\mathbf{R}$, the conjugate function of $\phi$ is given by

$$\phi^*(u) = \sup_{v\in\mathcal{E}}\{(u,v) - \phi(v)\} \tag{38}$$

Note that $\phi^*$ is always convex. Furthermore, we have

$$\phi^{**} = (\phi^*)^* = \phi \tag{39}$$

if and only if $\phi$ is convex.

*Existence of $\psi$:*

We put $\psi(w) =: \tilde{\lambda}(|w|)$ and $V = L_2(\Omega) \times L_2(\Omega)$. Referring to (17), we compute:

$$\begin{aligned}
&\inf_w \frac{1}{2}\left\{\int_\Omega \alpha|\nabla v - w|^2 + \tilde{\lambda}(|w|)\, dx\right\} \\
&= \frac{\alpha}{2}||\nabla v||_V^2 + \inf_w\{-\alpha(\nabla v, w)_V + \alpha H(w)\} \\
&\text{with } H(w) := \frac{1}{2}\int_\Omega |w|^2 + \frac{\tilde{\lambda}(|w|)}{\alpha}\, dx \\
&= \frac{\alpha}{2}||\nabla v||_V^2 - \alpha\sup_w\{(\nabla v, w)_V - H(w)\} \\
&= \frac{\alpha}{2}||\nabla v||_V^2 - \alpha H^*(\nabla v)
\end{aligned}$$

To ensure $J(v) = \inf_w J_{NA}(v,w)$, we must have

$$H^*(w^*) = \frac{1}{2}\int_\Omega |w^*|^2 - \frac{\lambda(|w^*|)}{\alpha}\, dx \tag{40}$$

This condition can only be satisfied if $H^*$ is convex which, however, is true by virtue of (2) and $\alpha > c_0$. Thus, we define $H(w)$ to be $H^{**}(w)$ which, in turn, defines $\tilde{\lambda}$ and $\psi$:

$$\left\{\frac{1}{2}\int_\Omega |w^*|^2 - \frac{1}{\alpha}\lambda(|w^*|)\, dx\right\}^*(w) \;=\; H^{**}(w) \;=:\; H(w) \;=\; \frac{1}{2}\int_\Omega |w|^2 + \frac{\tilde{\lambda}(|w|)}{\alpha}\, dx$$

As a consequence, $J_{NA}(v,w)$ is convex w.r.t $w$. □

*Computation of auxiliary variables $w^k$*

We compute $w^k$ by minimizing $J_{NA}(u^k, w)$ with respect to $w$. Variational calculus yields

$$\begin{aligned} 0 &= \int_\Omega \alpha(\nabla u^k - w^k)\cdot z + \frac{\tilde{\lambda}'(|w^k|)}{2|w^k|} w^k \cdot z \; dx \\ &= \int_\Omega \left( \left( -\alpha + \frac{\tilde{\lambda}'(|w^k|)}{2|w^k|} \right) w^k + \alpha \nabla u^k \right) \cdot z \; dx \,, \qquad \forall z \end{aligned}$$

Hence,

$$\left( 1 - \frac{\tilde{\lambda}'(|w^k|)}{2\alpha|w^k|} \right) w^k = \nabla u^k \,.$$

Thus, $w^k$ is just a multiple of $\nabla u^k$. In order to compute $|w^k|$, we put $h(s) = \frac{1}{2}(s^2 + \frac{\tilde{\lambda}(s)}{\alpha})$ and, correspondingly, $h^*(t) = \frac{1}{2}(t^2 - \frac{\lambda(t)}{\alpha})$. Since $\frac{dh(s)}{ds}$ is continuous and strictly monotone, we obtain:

$$h^*(t) = \sup_s (st - h(s)) \quad \Rightarrow \quad t = \frac{dh(s)}{ds} \quad \Rightarrow \quad s = \left( \frac{dh(s)}{ds} \right)^{-1}(t) \,,$$

and from [21]

$$\left( \frac{dh(s)}{ds} \right)^{-1}(r) = \frac{dh^*(t)}{dt}(r)$$

Thus, we have

$$s = \frac{d}{dt}\left[ \frac{1}{2}t^2 - \frac{1}{2\alpha}\lambda(t) \right] = t - \frac{1}{2\alpha}\lambda'(t)$$

and, after substitution

$$|w^k| = |\nabla u^k| - \frac{1}{2\alpha}\lambda'(|\nabla u^k|) \,.$$

As a result, we can compute $w^k$:

$$w^k = |w^k| \frac{\nabla u^k}{|\nabla u^k|} = \nabla u^k - \frac{1}{\alpha}\rho(|\nabla u^k|)\nabla u^k \,.$$

□

*Convergence of the iteration*

We show convergence of the iteration (22) in three steps.

First, we show the sequence $\{J(u^k)\}_{k\geq 1}$ converges. From (18) and (19) we have

$$J(u^{k+1}) = J_{NA}(u^{k+1}, w^k) \leq J_{NA}(u^{k+1}, w^{k-1}) \leq J_{NA}(u^k, w^{k-1}) = J(u^k)\,.$$

Since $J(u^k)$ is bounded below and strictly decreasing, the sequence $\{J(u^k)\}_{k\geq 1}$ converges and we obtain:

$$\lim_{k\to\infty} J(u^k) - J(u^{k+1}) = 0\,.$$

Next, we show that the sequence $\{u^k\}_{k\geq 1}$ converges. Using (2)-(4) and utilizing the fact that the operator $J'$ is strongly monotone [7], we obtain the inequality

$$J(u^k) - J(u^{k+1}) \geq L(u^{k+1}, u^k - u^{k+1}) + \frac{\sigma}{2}||u^k - u^{k+1}||^2_{H^1}\,,$$

with some constant $\sigma > 0$. Now, we have to show that $L(u^{k+1}, u^k - u^{k+1}) \geq 0$. To this end, we use (21) and define:

$$\underbrace{\int_\Omega (u^{k+1} - g)z + \alpha(\nabla u^{k+1} - w^k)\cdot\nabla z\,dx}_{:=L_{NA}(u^{k+1},w^k,z)} = 0, \quad \forall z$$

With $z := u^k - u^{k+1}$, we obtain:

$$\begin{aligned} L(u^{k+1}, z) &= L(u^{k+1}, z) - \underbrace{L_{NA}(u^{k+1}, w^k, z)}_{=0} \\ &= \int_\Omega \alpha|\nabla z|^2 - (\rho(|\nabla u^k|)\nabla u^k - \rho(|\nabla u^{k+1}|)\nabla u^{k+1})\cdot\nabla z dx \end{aligned}$$

Due to (3) and (4), we can find always some constant $c_3 > 0$ (cf. [7]) such that

$$|\,[\rho(|\nabla u|)\nabla u - \rho(|\nabla v|)\nabla v]\cdot(\nabla u - \nabla v)\,| \leq c_3|\nabla u - \nabla v|^2$$

Hence, $L(u^{k+1}, z) = L(u^{k+1}, u^k - u^{k+1})$ is positive for $\alpha \geq \max\{c_0, c_3\}$ and the sequence $\{u^k\}_{k\geq 1}$ converges.

Finally, we show that $\{u^k\}_{k\geq 1}$ converges to the unique minimizer $u$. With $\tilde{u} = \lim_{k\to\infty} u^k$, we obtain from (21):

$$\begin{aligned} 0 &= \lim_{k\to\infty} \int_\Omega \Big( (u^{k+1} - g)v + \alpha(\nabla u^{k+1} - \nabla u^k + \frac{1}{\alpha}\rho(|\nabla u^k|)\nabla u^k) \cdot \nabla v \Big) dx \\ &= \int_\Omega (\tilde{u} - g)v + \rho(|\nabla \tilde{u}|)\nabla \tilde{u} \cdot \nabla v \; dx \end{aligned}$$

Since the solution of eqn. (5) is unique, the last equation proves $\tilde{u} = u$. □

## REFERENCES


[1] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 6(6), pp. 721–741, 1984.

[2] A. Blake and A. Zisserman, *Visual Reconstruction*, MIT Press, 1987.

[3] D. Mumford and J. Shah, "Optimal approximations by piecewise smooth functions and associated variational problems," *Comm. Pure Appl. Math.*, vol. 42, pp. 577–685, 1989.

[4] D. Geiger and A. Yuille, "A common framework for image segmentation," *Int. J. of Comp. Vision*, vol. 6(3), pp. 227–243, 1991.

[5] J.-M. Morel and S. Solimini, *Variational Methods in Image Segmentation*, Birkhäuser, Boston, 1995.

[6] R.L. Stevenson, B.E. Schmitz, and E.J. Delp, "Discontinuity preserving regularization of inverse problems," *IEEE Trans. Systems, Man and Cyb.*, vol. 24(3), pp. 455–469, 1994.

[7] C. Schnörr, "Unique reconstruction of piecewise-smooth images by minimizing strictly convex non-quadratic functionals," *Journal of Mathematical Imaging and Vision*, vol. 4, pp. 189–198, 1994.

[8] S.Z. Li, Y.H. Huang, and J. Fu, "Convex energy functionals in the DA model," in *Proc. IEEE Int. Conf. Image Processing*, 1995.

[9] D. Geiger and F. Girosi. Parallel and deterministic algorithms from mrf's: Surface reconstruction. *IEEE Trans. Patt. Anal. Mach. Intell.*, 13(5):401–412, 1991.

[10] C. Schnörr, "A study of a convex variational approach for image segmentation and feature extraction," *Journal of Mathematical Imaging and Vision*, vol. 8, no. 3, pp. 271–292, 1998.

[11] D. Geman and G.Reynolds, "Constrained restoration and the recovery of discontinuities," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 367–383, 1992.

[12] D. Geman and C. Yang, "Nonlinear image recovery with half–quadratic regularization," *IEEE Trans. on Image Processing*, vol. 4, pp. 932–945, 1995.

[13] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud, "Deterministic edge–preserving regularization in computed imaging," *IEEE Trans. on Image Processing*, vol. 6, no. 2, pp. 298–311, 1997.

[14] L.D. Cohen, "Auxiliary variables and two–step iterative algorithms in computer vision problems," *J. of Math. Imag. Vision*, vol. 6(1), pp. 59–83, 1996.

[15] L.I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, pp. 259–268, 1992

[16] V.V. Shaidurov, *Multigrid Methods for Finite Elements*, Kluwer Academic Publisher, 1995.

[17] J.P. Aubin, *Applied Functional Analysis*, Wiley & Sons, 1979

[18] M.J. Black and A. Rangarajan, "On the unification of line processes, outlier rejection, and robust statistics with applications in early vision," *IJCV*, vol. 19, no. 1, pp. 57–91, 1996.

[19] S. Fučik, A. Kratochvil, and J. Nečas, "Kačanov-Galerkin method," *Comment. Math. Univ. Carolina*, vol. 14, no. 4, pp. 651–659, 1973.

[20] J. Nečas and I. Hlaváček, *Mathematical Theory of Elastic and Elasto-Plastic Bodies*, Elsevier, Amsterdam, 1981.

[21] E. Zeidler, *Nonlinear Functional Analysis and its Applications*, Springer Verlag, 1995.

[22] D.C. Dobson and C.R. Vogel, "Convergence of an iterative method for total variation denoising," *SIAM J. of. Numerical Analysis*, vol. 34, pp. 1779–1791, 1997.

[23] T. Chan, P. Blomgren, P. Mulet, and C.K. Wong, "Total Variation Image Restoration: Numerical Methods and Extensions", *ICIP'97*, 1997, pp.III:384.

[24] P.G. Ciarlet, *The finite element method for elliptic problems*, vol. 4 of *Studies in mathematics and its applications*, North–Holland Publishing, Amsterdam, 1978.

[25] C. Schnörr, "Variational methods for adaptive image smoothing and segmentation," in *Handbook on Computer Vision and Applications: Signal Processing and Pattern Recognition*, B. Jähne, H. Haußecker, and P. Geißler, Eds., San Diego, 1999, vol. 2, Academic Press, pp.451–484

[26] M.R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *J. Res. Nat. Bur. Stand.*, vol. 49, pp. 409–436, 1954.

[27] G.H. Golub and C.F. van Loan, *Matrix Computations*, The John Hopkins University Press, Baltimore, 1989.

[28] A. Bunse-Gerstner and W. Bunse, *Numerische lineare Algebra*, MG Teubner, Stuttgart, 1985.

[29] Y. Saad and M.H. Schultz, "GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems," *SIAM J. Sci. Statist. Comput.*, vol. 7, pp. 856–869, 1986.

[30] H.A. van der Vorst, "BICGSTAB; a fast and smoothly converging variant of BiCG for the solution of non-symmetric systems," *SIAM J. Sci. Statist. Comput.*, vol. 13, pp. 631–644, 1992.

[31] P. Sonneveld, "CGS: A fast Lanczos-type solver for non-symmetric linear systems," *Numer. Math.*, vol. 48, pp. 543–560, 1989.

[32] O. Axelson, "Globally convergent continuation methods for non–linear equations," in *1th Workshop on Large-Scale Scientific Computations*, Bulgaria, 1997, International House of Scientists "F. Curie".

[33] Claude Pommerell, *Solutions of large un-symmetric systems of linear equations*, Ph.D. thesis, Swiss Federal Institute of Technology, Zürich, 1992.

[34] C.L.G Sleijpen and H.A. van der Vorst, "Krylov subspace methods for large linear systems of equations," *Preprint 803, Department of Mathematics, University Utrecht*, 1993.

[35] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith, "PETSc home page," http://www.mcs.anl.gov/petsc, 1998.

[36] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith, "PETSc 2.0 users manual," Tech. Rep. ANL-95/11 - Revision 2.0.22, Argonne National Laboratory, 1998.

[37] Message Passing Interface Forum, *MPI: A Message-Passing Interface Standard*, University of Tennessee, Knoxville, Tennessee, 1995.

[38] I. Ekeland and R. Temam, *Convex analysis and variational problems*, vol. 1 of *Studies in mathematics and its applications*, North–Holland Publishing, Amsterdam, 1976.