Solution of Ambrosio-Tortorelli Model for Image Segmentation by Generalized Relaxation Method[☆]

Pasqua D'Ambra*, Gaetano Tartaglione

Institute for High-Performance Computing and Networking (ICAR), CNR, Via P. Castellino, 111, 80131 Naples, Italy.

Abstract

Image segmentation addresses the problem to partition a given image into its constituent objects and then to identify the boundaries of the objects. This problem can be formulated in terms of a variational model aimed to find optimal approximations of a bounded function by piecewise-smooth functions, minimizing a given functional. The corresponding Euler-Lagrange equations are a set of two coupled elliptic partial differential equations with varying coefficients. Numerical solution of the above system often relies on alternating minimization techniques involving descent methods coupled with explicit or semi-implicit finite-difference discretization schemes, which are slowly convergent and poorly scalable with respect to image size. In this work we focus on generalized relaxation methods also coupled with multigrid linear solvers, when a finite-difference discretization is applied to the Euler-Lagrange equations of Ambrosio-Tortorelli model. We show that non-linear Gauss-Seidel, accelerated by inner linear iterations, is an effective method for large-scale image analysis as those arising from high-throughput screening platforms for stem cells targeted differentiation, where one of the main goal is segmentation of thousand of images to analyze cell colonies morphology.

Keywords:
2000 MSC: 65H10, 65K10, 65N55, 68U10
image segmentation, variational models, non-linear iterative methods,

[☆]This work has been partially supported by Italian Flagship Project *InterOmics* and INdAM-GNCS Project on *Numerical Methods and Software for Large-Scale Optimization with Applications to Image Processing.*

*Corresponding Author: pasqua.dambra@cnr.it

1. Introduction

Image segmentation is one of the central tasks in image processing and has the aim of partitioning an image into its constituent regions or objects, leading to also identify the boundaries of these objects. Usually, two types of techniques are used for image segmentation: the *discontinuity-based approach*, where main aim is to identify the set of pixels in which intensity function has high gradients or jumps (*the edges*) or the *similarity-based approach*, where the aim is to identify regions with similar characteristics in intensity function. The problem can be formulated in terms of a variational model, i.e., in terms of an energy minimization criterion, which in some sense merges features of both the above approaches. The original variational model for segmentation is a *free-discontinuity problem* formulated by Mumford and Shah [17] that proposed to look for a piecewise smooth approximation u of the original image function f, with u discontinuous across a closed set K included in the image domain Ω. In more details, let $\Omega \subset \Re^2$ be a bounded open set and $f \in L^\infty(\Omega)$ the observed gray-level image, the problem consists in the minimization of the following functional:

$$E(u, K) = \int_\Omega (u - f)^2 dxdy + \beta \int_{\Omega \setminus K} |\nabla u|^2 dxdy + \alpha |K| \tag{1}$$

where $u \in C^1(\Omega \setminus K)$, $K \subset \Omega$ is a closed set whose $|K|$ is the lenght of K, and α and β are positive coefficients.

This problem is not simple to deal with due to the presence of the last term. However, an extensive theory has been developed since its introduction and many efforts have been devoted to approximate the problem with a relaxed one which can be solved by classical approach of Calculus of Variations. On the other hand, since the functional is not convex, large efforts have been also devoted to relax the model in order to get convexity. It is beyond the focus of this work to discuss theoretical aspects or advantages and drawbacks of the Mumford-Shah (MS) model, however, we can observe that it is the most general way to formulate a segmentation problem and the well-based theory developed during the last 20 years motivates its use to obtain robust and accurate software modules for large-scale analysis. For classical books and recent reviews on theory, numerical approximations and applications of the MS model we refer to [1, 2, 6, 16, 22].

One of the most general approximations of the MS model was proposed by Ambrosio and Tortorelli who showed that the model can be approximated, in the sense of Γ-convergence, by a sequence of elliptic functionals [3, 4]. For details on Γ-convergence and its role in the study of asymptotic variational problems we refer to [9]. Ambrosio and Tortorelli introduced a new variable $0 \leq z \leq 1$, which controls $|\nabla u|$ and gives an approximate representation of the set K in a tubular neighbourhood of radius ϵ of the minimizer K. The possibility to approximate the measure of K by an elliptic functional depending on z leads to the following sequence of functionals depending on ϵ:

$$E_\epsilon(u,z) = \int_\Omega (u-f)^2 dxdy + \beta \int_\Omega z^2 |\nabla u|^2 dxdy + \alpha \int_\Omega \left(\epsilon |\nabla z|^2 + \frac{(z-1)^2}{4\epsilon} \right) dxdy, \tag{2}$$

which Γ-converges to $E(u,K)$ for $\epsilon \to 0$. Note that $E_\epsilon(u,z)$ remains not convex and a drawback of this approximation is its dependence on the choice of a good ϵ parameter in numerical solution. In [3, 21] the reader can find a clear discussion, which we do not report here for sake of brevity, on the behaviour of the function z introduced by Ambrosio and Tortorelli in their work. However, we notice that this function is forced to tend to 1 as ϵ tends to 0 because the term $(z-1)^2$ is positive and vanishes only for $z=1$; furthermore, the term z^2 must go to zero near the discontinuities of u, where $|\nabla u|$ is large, to keep bounded the second term in (2). Therefore, z makes a sharp transition to zero inside a neighbourhood of the set K of thickness of radius ϵ and an accurate numerical solution should require a suitable spatial discretization of the model to well resolve the above tubular region.

Some efforts have been devoted to minimize the functional in (2) by finite-element approximations, see for example [7, 8]. On the other hand, the form of the functional in (2) allows us to apply the classical approach of Calculus of Variations, i.e. writing Euler-Lagrange equations to obtain stationary solutions. Numerical solutions of the Euler-Lagrange equations associated to (2) are usually based on finite-difference discretizations of the equations and alternating minimization schemes are obtained by applying gradient descent iterations [6, 19, 22]. In [21] the authors proposed to use the non-linear Gauss-Seidel method for solving discrete equations arising from a finite-difference approximation of (2). In this paper we show that non-linear Gauss-Seidel relaxation, coupled with a fixed-point approach producing inner linear systems

at each application of a basic step of the non-linear method to a discrete form of the equations, is very effective for large-scale image segmentation. Inner iterations largely reduce the number of non-linear iterations already for very low accuracy requests on inner solutions but also improves robustness when those accuracy requests are increased, leading to reliable, flexible and efficient solver. We discuss convergence results and computational cost of the method in the analysis of 2D gray-scale images of embryonic stem cells colonies, since our final aim is to develop an efficient segmentation module for automatic screening of colonies morphology useful to discriminate different phenotypic transitions.

The paper is organized as follows. In Section 2 we introduce a finite-difference discretization of the Euler-Lagrange equations for model (2). In Section 3 we briefly describe non-linear Gauss-Seidel method when applied to a discrete form of (2), then we describe our fixed-point approach coupled with the non-linear relaxation and outline the linear multigrid solver which can be used to accelerate inner convergence. In Section 4 we present results obtained on real images of cultures of mouse embryonic stem cells. An extensive discussion on the performance of our method, in terms of robustness, accuracy, computational costs and comparison with standard gradient descent methods, varying model parameters, is reported. Concluding remarks and future work are included in Section 5.

2. Finite-Difference Discretization of Euler-Lagrange Equations

Euler-Lagrange equations for Ambrosio-Tortorelli model are the following system of two coupled elliptic PDEs, associated with Neumann boundary conditions:

$$\begin{cases} 2(u-f) - 2\beta\nabla\cdot(z^2\nabla u) = 0 \\ 2\beta z|\nabla u|^2 - 2\alpha\epsilon\nabla^2 z + \frac{\alpha}{2\epsilon}(z-1) = 0 \end{cases} \quad (x,y)\in\Omega, \tag{3}$$

$$\begin{cases} \nabla u\cdot\vec{n} = 0 \\ \nabla z\cdot\vec{n} = 0 \end{cases} \quad (x,y)\in\partial\Omega, \tag{4}$$

where $\vec{n}$ is the exterior normal to $\partial\Omega$.

We note that the first equation in (3) is a form of anisotropic diffusion where function z^2 represents anisotropy and observe that each one of the equations is linear in one of the unknowns if the other is fixed. Main approaches for solving these equations are usually based on iterative methods,

where a minimization method, such as gradient descent obtained from (3) by artificial time evolution, is applied with respect to one of the two variables at each one of the equations, in an alternate way [6, 19, 22]. Gradient descent method, despite its simplicity, is slowly convergent and poorly scalable with respect to the problem dimension, indeed some recent efforts are devoted to apply Newton-type methods for efficient solution of (3)-(4) and more generally of similar models for image processing [5]. Netwon-type methods are very attractive for their quadratic convergence, however building the Hessian matrix and solving the related linear systems to get the Newton step at each iteration leads to high computational costs.

In this paper, following an approach already proposed in [21], we focus on a generalized linear relaxation method applied to the non-linear algebraic system of equations arising from a spatial discretization of (3)-(4). We propose to combine a fixed-point approach at each non-linear relaxation step, leading to the solution of inner linear systems for each one of the involved unknowns. We show that inner linear iterations improve convergence of non-linear relaxation already for low accuracy requests on linear systems solution and allows us to obtain a very efficient and reliable method.

We start from a second-order finite-difference discretization of the equations. Let the domain be a rectangle $\Omega = (0, x_f) \times (0, y_f)$; the computational grid can be obtained by fixing the step size $h > 0$, typically related to image size, and therefore considering the set of points $\Omega_h = (x_i, y_j) = (ih, jh)$ for $i = 0, \ldots, n$ and $j = 0, \ldots, m$. Let $u_{i,j} \approx u(x_i, y_j)$, $z_{i,j} \approx z(x_i, y_j)$ and $f_{i,j} = f(x_i, y_j)$.

Finite-difference approximations of partial derivatives of function $z(x, y)$ can be written at each internal point of the computational grid using second-order accurate centered finite-difference formulas, as follows:

$$\left(\frac{\partial z}{\partial x}\right)_{i,j} = \frac{z_{i+1,j} - z_{i-1,j}}{2h}, \quad \left(\frac{\partial z}{\partial y}\right)_{i,j} = \frac{z_{i,j+1} - z_{i,j-1}}{2h},$$

then the Laplace operator is approximated by the classical five-points scheme:

$$(\nabla^2 z)_{i,j} = \frac{z_{i,j-1} + z_{i-1,j} - 4z_{i,j} + z_{i+1,j} + z_{i,j+1}}{h^2}.$$

Similar formulas can be used for the function $u(x, y)$, where half step size is used for second-order accurate centered formulas of partial derivatives and symmetrization is applied to avoid values of $z(x, y)$ in the mid-points of the

grid. Therefore, the equations in (3) can be written at each internal point of the computational grid as follows:

$$\begin{cases} -\frac{2\beta}{h^2} z_{i,j-1}^2 u_{i,j-1} - \frac{2\beta}{h^2} z_{i-1,j}^2 u_{i-1,j} + \left(2 + \frac{2\beta}{h^2} \hat{z}_{i,j}\right) u_{i,j} \\ -\frac{2\beta}{h^2} z_{i+1,j}^2 u_{i+1,j} - \frac{2\beta}{h^2} z_{i,j+1}^2 u_{i,j+1} = 2 f_{i,j} \\ \\ -\frac{2\alpha\epsilon}{h^2} z_{i,j-1} - \frac{2\alpha\epsilon}{h^2} z_{i-1,j} + \left(2\beta |\nabla u|_{i,j}^2 + \frac{\alpha}{2\epsilon} + \frac{8\alpha\epsilon}{h^2}\right) z_{i,j} \\ -\frac{2\alpha\epsilon}{h^2} z_{i+1,j} - \frac{2\alpha\epsilon}{h^2} z_{i,j+1} = \frac{\alpha}{2\epsilon} \end{cases}$$

where $\hat{z}_{i,j} = z_{i,j-1}^2 + z_{i-1,j}^2 + z_{i+1,j}^2 + z_{i,j+1}^2$.

When Neumann boundary conditions are approximated by second-order finite-difference formulas and a standard row-wise ordering of the unknowns (also known as lexicographical order) is considered [14], the discrete problem to be solved has the form of the following system of non-linear algebraic equations:

$$F_h(\mathbf{u}, \mathbf{z}) = f_h(x, y), \tag{5}$$

where $\mathbf{u}$ and $\mathbf{z}$ are the vectors obtained from the discretization of u and z, respectively, F_h is the finite-difference operator associated to equations (3)-(4) and f_h is the related right-hand side (r.h.s.). We observe that the system can be written in the following compact form:

$$\begin{bmatrix} A(\mathbf{z}) & \mathbf{0} \\ \mathbf{0} & B(\mathbf{u}) \end{bmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{z} \end{pmatrix} = \begin{pmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{pmatrix}. \tag{6}$$

Matrices $A(\mathbf{z})$ and $B(\mathbf{u})$ have the same symmetric sparsity pattern but are unsymmetric in values due to the application of Neumann boundary conditions. Furthermore, they are both diagonally-dominant M-matrices, per each finite vector $(\mathbf{u}, \mathbf{z})$, therefore a well-established convergence theory for iterative solution of related linear systems is available [20].

3. Numerical Algorithms

The non-linear system (5) can be solved by a generalized linear method such as that arising from basic point-wise iterative methods, which have a simple interpretation if we consider the system in the form (6). For example, the basic step of non-linear Gauss-Seidel, starting from a vector $(\mathbf{u}^k, \mathbf{z}^k)$, requires the application of a basic step of the linear Gauss-Seidel relaxation to the system:

$$A(\mathbf{z}^k)\mathbf{u} = \mathbf{f}_1, \tag{7}$$

in order to obtain a new approximation $\mathbf{u}^{k+1}$, and then a basic step of the linear Gauss-Seidel relaxation to the system:

$$B(\mathbf{u}^{k+1})\mathbf{z} = \mathbf{f}_2 \tag{8}$$

to get the new value $\mathbf{z}^{k+1}$. Properties of the matrices $A(\mathbf{z})$ and $B(\mathbf{u})$ ensures applicability and asymptotic convergence of the non-linear Gauss-Seidel (GS) method to the solution of the original system (see [18]).

Observe that, each iteration of GS requires the updating of the matrices A and B with the new values of the unknown vector that, especially for increasing image size, is the most computational demanding task of the algorithm (see Section 4). With the final aim to accelerate the convergence of GS, also reducing frequency of operators updating, we followed an approach similar to that proposed in [23] for image denoising, combining the GS method with fixed-point (or Picard) iterations to obtain better accuracy in the solution of systems (7)-(8) at each non-linear iteration. The corresponding algorithm, named GS-FP, is described in Algorithm 1.

Algorithm 1: Gauss-Seidel method coupled with Fixed-Point iterations (GS-FP)

$k = 0$, $\mathbf{z}^0 = \mathbf{1}$ (unitary vector) and $\mathbf{u}^0 = f_h$ (original image);
build r.h.s. $\mathbf{f}_1$ and $\mathbf{f}_2$;
repeat
  build matrix $A(\mathbf{z}^k)$;
  compute $\mathbf{u}^{k+1}$ by *iterative solution of system (7)*, starting from $\mathbf{u}^k$;
  build matrix $B(\mathbf{u}^{k+1})$;
  compute $\mathbf{z}^{k+1}$ by *iterative solution of system (8)*, starting from $\mathbf{z}^k$;
  $k = k + 1$;
until *convergence*;

Classical point-wise iterative solvers are very effective for the solution of systems with diagonally dominant or M-matrices, therefore, in Algorithm 1, we can still use simple Jacobi or Gauss-Seidel methods as inner linear solvers. In our experiments we choose the Gauss-Seidel method and observed a more rapid convergence of GS-FP with respect to GS already when very low accuracy was required for solution of systems (7) and (8). On the other hand, we verified that increasing accuracy requests in the inner solvers also

improve robustness of non-linear relaxation thus allowing convergence to high accuracy global solution for a larger set of model parameters.

As already said, point-wise relaxation methods are good linear solvers for our systems, however it is well-known that they suffer of convergence rate degradation, especially in case of problem anisotropy, for increasing problem dimensions [11]. To overcome this limitation we also used multigrid linear solvers for efficient and scalable solution of the linear systems involved in Algorithm 1.

In the following we briefly recall the main elements of a classical geometric multigrid method; for more details on theory and algorithms we refer to [11, 13].

3.1. Key Components of Linear Multigrid Solvers

Multigrid solvers are the most efficient methods for linear systems arising from boundary-value problems for elliptic partial differential equations with constant or slowly-varying coefficients. They have the capability to solve the systems with a linear computational complexity, by combination of two key components: *smoothing* and *coarse-grid correction*. Smoothing is the process based on the application of a relaxation method, such as the basic Jacobi or Gauss-Seidel methods, to obtain a fine-grid solution (corresponding to the original grid), while coarse-grid correction is the process of transferring information (the system residual) to a coarser grid through *restriction*, solving a coarse-grid system (the error system), and then transferring the solution back to the fine grid through *interpolation* with the final aim to improve the previous fine-grid solution. In this work, in order to exploit the regular (rectangular) grid implicitly defined by the image pixels, we focus on classical geometric multigrid, where a pre-defined hierarchy of L ever more coarser grids $\Omega_1 = \Omega_h \supset \Omega_2 = \Omega_{2h} \supset \ldots \Omega_L = \Omega_{Lh}$ is considered and transferring operators are built using geometric information of two consecutive grids. Prolongation operators P_{k+1}^k, which transfer a grid vector $\mathbf{v}$ from level $k+1$ to level k, are defined in terms of linear interpolation as in the following:

$$
\begin{aligned}
&v_{2i,2j}^k = v_{i,j}^{k+1},\\
&v_{2i+1,2j}^k = \tfrac{1}{2}(v_{i,j}^{k+1} + v_{i+1,j}^{k+1}),\\
&v_{2i,2j+1}^k = \tfrac{1}{2}(v_{i,j}^{k+1} + v_{i,j+1}^{k+1}),\\
&v_{2i+1,2j+1}^k = \tfrac{1}{4}(v_{i,j}^{k+1} + v_{i+1,j}^{k+1} + v_{i,j+1}^{k+1} + v_{i+1,j+1}^{k+1})
\end{aligned}
$$

for $0 \leq i \leq n/2 - 1$ and $0 \leq j \leq m/2 - 1$; while restriction operator R_k^{k+1} can be built using injection, i.e. the components of $\mathbf{v}^{k+1}$ are simply obtained

from the corresponding fine grid point:

$$v_{i,j}^{k+1} = v_{2i,2j}^{k}, \quad 0 \leq i \leq n/2 - 1, \quad 0 \leq j \leq m/2 - 1.$$

Once all the transferring operators for a given hierarchy of grids are built, a hierarchy of coarse grid matrices is obtained by $A^{k+1} = R_k^{k+1} A^k P_{k+1}^k$, for $k = 1, \ldots L-1$ (where $A^1 = A$ is the original system matrix, also called fine matrix) and a multigrid cycle for the solution of a linear system of the type $A\mathbf{v} = \mathbf{f}$ can be defined in terms of the recursive scheme described in Algorithm (2), also known as L-level V(ν_1, ν_2)-cycle with ν_1 pre-smoothing and ν_2 post-smoothing steps.

Algorithm 2: $\mathbf{v}^k = MG(\mathbf{v}_0^k, \mathbf{f}^k)$

if $l \neq L$ **then**
- apply ν_1 steps of a smoother to $A^k \mathbf{v}_1^k = \mathbf{f}^k$, starting from $\mathbf{v}_0^k$;
- compute the residual $\mathbf{r}^k = \mathbf{f}^k - A^k \mathbf{v}_1^k$;
- restrict the residual $\mathbf{f}^{k+1} = R_k^{k+1} \mathbf{r}^k$;
- $\mathbf{v}^{k+1} = MG(\mathbf{0}, \mathbf{f}^{k+1})$;
- correct the solution $\mathbf{v}_2^k = \mathbf{v}_1^k + P_{k+1}^k \mathbf{v}^{k+1}$;
- apply ν_2 steps of a smoother to $A^k \mathbf{v}^k = \mathbf{f}^k$, starting from $\mathbf{v}_2^k$;

else
- $\mathbf{v}^k = (A^k)^{-1} \mathbf{f}^k$ (or apply ν_1 steps of a smoother to $A^k \mathbf{v}^k = \mathbf{f}^k$, starting from $\mathbf{v}_0^k$);

We recall here that classical multigrid methods as that described in this section are most efficient in case of linear systems arising from elliptic partial differential equations with constant or smoothly varying coefficients, while for highly-varying or discontinuous coefficients problems more specific operators and coarse grids have to be employed [10]. In the following we show that, for our problem, a geometric multigrid method with "standard" operators and grids is able to obtain efficient solution for a large set of model parameters.

4. Numerical Results

In the following we discuss results related to the application of the algorithms described in the above section to gray-scale real images of plates of cultured mouse embryonic stem cells, whose original size is 1040×1392.

First of all we present simulation results on an image with size $nsize^2 = 1024^2$, for varying model coefficients, in order to show their impact on the results' quality. Since in case of convergence, as expected, there is no significative impact of the chosen numerical method on the solutions, in the following we show only results obtained by our prototypal implementation of the GS method. In Figures 1-2 we show the segmented images and the corresponding edge sets, respectively, for different coefficients $(\alpha, \beta) \in \{1, 5, 10\}$. We notice that all the experiments discussed in this work have been run with $\epsilon = 10^{-4}$, which violates the condition $h/\epsilon < 1$ coming from the need of a good resolution of the tubular region including the edge set (see [15]), but still allows us to obtain a good representation of the discontinuity function z, as already noticed in [22], without relating the choice of the ϵ parameter to the different image sizes used in our experiments. We observe that, as

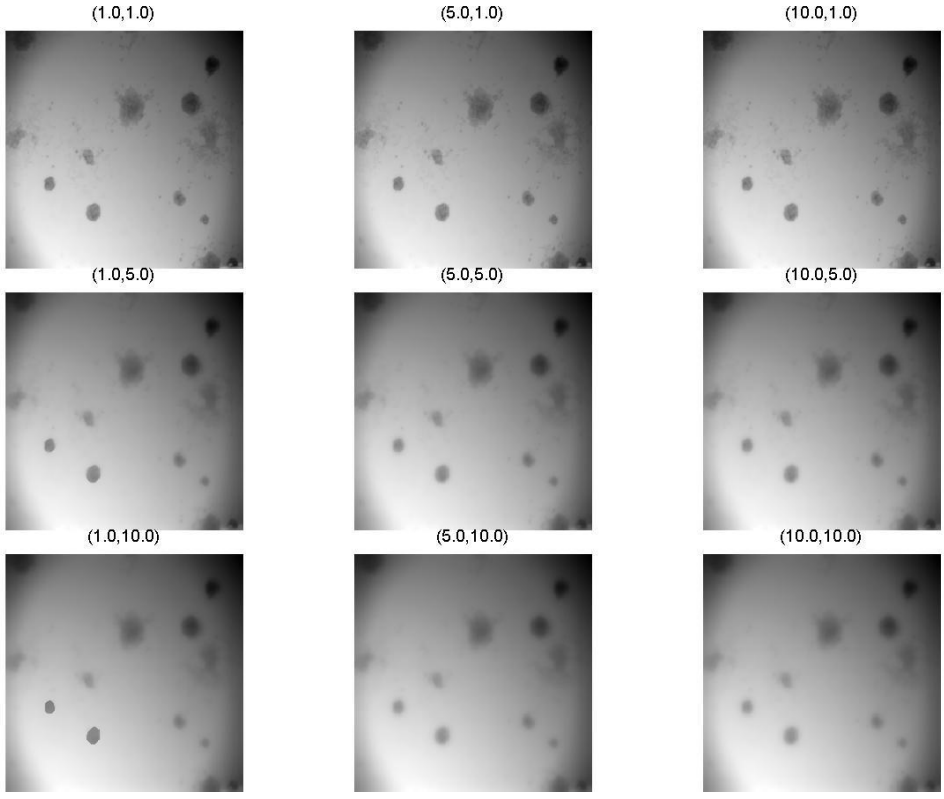

Figure 1: Piecewise-smooth images u obtained by Ambrosio-Tortorelli model for $\epsilon = 10^{-4}$ and different values of (α,β).

expected, decreasing values of both the parameters give more edges (observe pictures in Fig. 2 along the main diagonal from bottom to top); on the other hand, by keeping α fixed and by increasing β we get smoother effects in internal regions, with a more clear representation of the edges (observe pictures in Figs. 1-2 along rows), while an inverse effect can be observed when β is kept fixed and we increase α (observe pictures in Figs. 1-2 along columns).

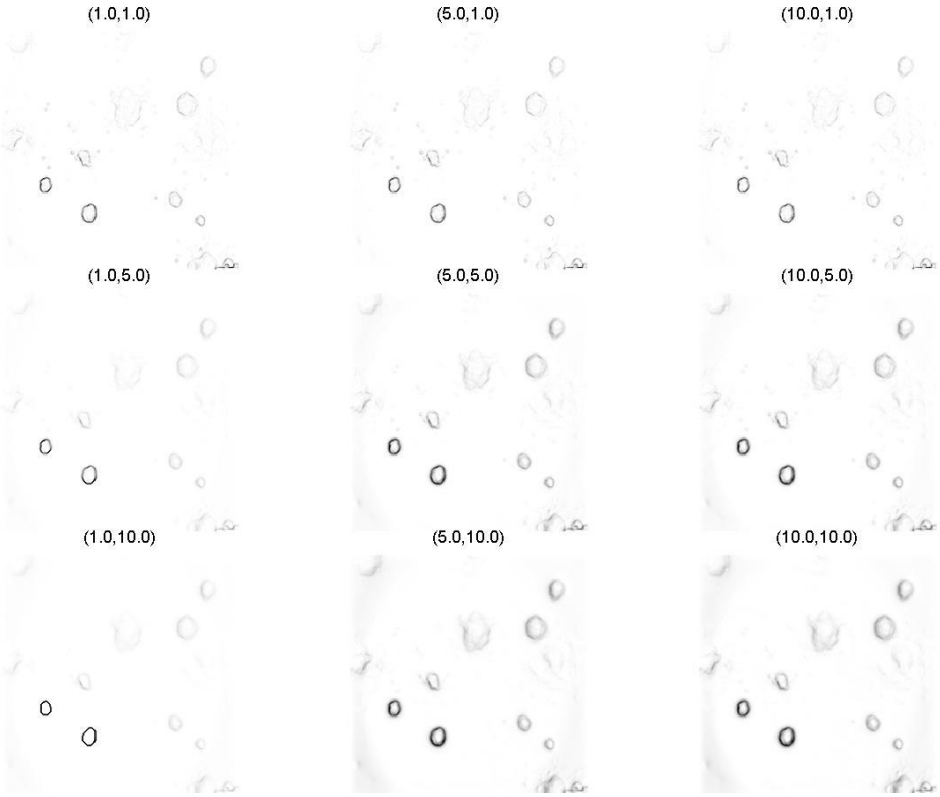

Figure 2: Edge sets z obtained by Ambrosio-Tortorelli model for $\epsilon = 10^{-4}$ and different values of (α,β).

In Fig. 3 we show the original image and the estimate of functional (2) versus the number of iterations, obtained by applying a two-dimensional Cavalieri-Simpson rule when discrete functions u and z are obtained by solving problem (5) with the GS method. The stopping criterion for the iterations was based on the maximum norm of the system residual and the above results were obtained when the requested accuracy was $ATOL = 10^{-10}$.

In the following section we report an extensive analysis of the performance

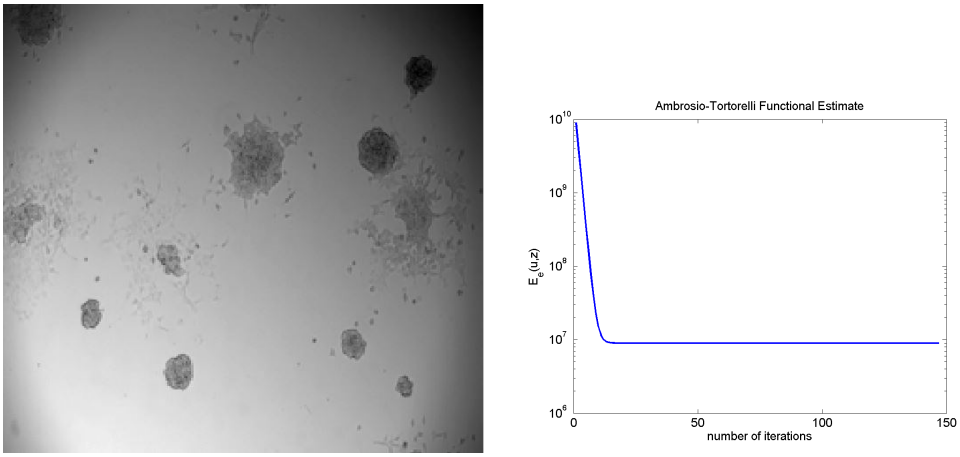

Figure 3: Original Image (left). Estimate of the Ambrosio-Tortorelli functional for $\epsilon = 10^{-4}$ and $(\alpha, \beta) = (1, 1)$.

results of our simulations obtained by prototypal Matlab implementations of all the methods discussed in Section 3, on the gray-level image of Figure 3, by varying both image size and model coefficients α and β, while $\epsilon = 10^{-4}$ was fixed. For all the methods, we stopped (outer) iterations when the maximum norm of the system residual was less than a fixed tolerance $ATOL$. In the case of Algorithm 1, where the linear systems (7)-(8) have to be solved at each outer iteration, we first used (one-level) linear Gauss-Seidel relaxation and stopped linear iterations when the relative residual of the involved variable was less than a fixed tolerance $RTOL$. A maximum number ($maxiter$) of iterations was also fixed in the stopping criteria. In the following we show results related to $ATOL = 10^{-10}$ and $maxiter = 2000$ for outer iterations, while $maxiter = 100$ and various values of $RTOL$ have been considered for inner iterations.

4.1. Performance Results

In Tables 1 and 2 we show iterations and execution times, respectively, related to the application of the GS method. We can observe that, an increase of β produces an increase in the number of iterations needed to obtain the requested accuracy. On the other hand, increasing the α parameter has generally an opposite effect on global convergence. In the case of $(\alpha, \beta) = (1, 10)$ the algorithm does not converge (NC) to the requested accuracy for the

largest image size; in particular we observed a system residual stalling around values of the order of 10^3. We also observe that there is no a regular impact of the image size on the convergence behaviour which essentially depends on the choice of the regularity model parameters. On the other hand, execution times increase with image size, due to the increasing cost for updating the operators $A(\mathbf{z}^k)$ and $B(\mathbf{u}^k)$ at each non-linear iteration. Indeed, we observed that updating the operators, although based on an efficient algorithm which reuses sparsity structure and exploits Matlab vectorial capacity, requires an increasing percentage of the total execution time, going from about 88% in the case of the smallest image size till to 90% in the case of the largest size.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	74	74	74	74	74	74	147	74	74
5	471	312	312	584	312	312	979	710	312
10	639	608	608	1612	608	608	NC	1930	608

Table 1: *GS*: Non-linear iterations.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	11.0	11.0	11.0	48.5	48.0	48.1	405.1	204.4	204.5
5	69.6	46.3	46.4	384.9	202.3	202.6	2726.8	1961.2	907.7
10	94.6	90.3	93.7	1041.3	405.5	406.7	NC	5370.4	1717.9

Table 2: *GS*: Execution Times in seconds.

In the following, results obtained with Algorithm 1 are shown where (one-level) linear Gauss-Seidel was used for inner iterations. Table 3 reports the outer iterations counts and, in the brackets, the average number of inner iterations for the linear systems (7), when $RTOL = 10^{-2}$. In Table 4 the corresponding execution times are shown. We notice that in the following tables we do not report inner iterations needed for the linear systems (8), since for all our tests only 1 iteration is needed to obtain all the requested

accuracies. This is essentially due to the features of the system matrix which is very diagonally dominant for our choices of the model parameters.

We can see that, as expected, the convergence behaviour with respect to the model parameters is the same as in the case of GS, i.e., the number of non-linear iterations increases when β increases for fixed α, while increasing α has the opposite effect. However inner iterations generally produce a more rapid convergence of the non-linear Gauss-Seidel method leading to a large reduction in execution times. On the other hand, we observe that Algorithm 1, in the case of the largest image size, does not converge for both the cases $(\alpha, \beta) = (1, 10)$, as the GS method, and $(\alpha, \beta) = (1, 5)$; the system residual stalls around values of the order 10^0 and 10^{-1}, respectively.

We also note that by increasing β an increasing number of iterations is required to get the requested accuracy in the solution of systems (7). This is essentially due to a moderate increase in the condition number of the matrix $A(\mathbf{z}^k)$ for increasing values of β. The condition number of the matrix $B(\mathbf{u}^k)$ is generally very small (of order $10^1 \div 10^2$) and decreases for increasing values of α. Note that, for fixed values of β, increasing α reduces the number of non-linear iterations, as for GS, but the average number of inner linear iterations needed for solving systems (7) shows a small increase. Perhaps this is due to different levels of spatial anisotropy introduced by z values into the equation of u.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	20	10	8	48	16	12	123	33	19
	[7]	[8]	[10]	[4]	[6]	[7]	[4]	[6]	[7]
5	153	14	10	203	17	12	NC	192	29
	[8]	[23]	[32]	[9]	[19]	[26]		[13]	[15]
10	131	15	11	289	18	12	NC	329	24
	[8]	[41]	[56]	[12]	[34]	[51]		[16]	[26]

Table 3: *GS-FP* - $RTOL = 10^{-2}$: Non-linear and [average number of linear] iterations.

To analyze the impact of the accuracy in the solution of inner linear systems on the convergence of the outer non-linear iterations, in Tables 5-6 and 7-8, we report results related to $RTOL = 10^{-3}$ and $RTOL = 10^{-1}$,

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	3.9	2.1	1.8	35.9	12.8	10.1	376.6	110.0	65.4
5	30.9	4.6	4.1	182.5	20.8	17.4	NC	798.3	128.6
10	27.7	7.6	6.8	292.3	30.4	26.9	NC	1504.1	138.2

Table 4: *GS-FP* - $RTOL = 10^{-2}$: Execution Times in seconds.

respectively, while the other numerical parameters remained fixed.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	18 [11]	9 [12]	7 [13]	45 [8]	14 [10]	11 [10]	107 [7]	30 [10]	17 [10]
5	136 [17]	12 [35]	9 [38]	174 [17]	15 [32]	11 [34]	NC	169 [26]	26 [27]
10	115 [20]	14 [56]	10 [66]	257 [32]	16 [52]	11 [59]	NC	279 [40]	21 [45]

Table 5: *GS-FP* - $RTOL = 10^{-3}$: Non-linear and [average number of linear] iterations.

We observe that increasing the accuracy requests for the inner linear iterations produces a small reduction in the number of outer non-linear iterations. This reduction is about 15% in the best case, related to the choice of parameters $(\alpha, \beta) = (5, 10)$, for the largest image size. However, increasing accuracy for inner iterations seems do not produce convergence in the cases $(\alpha, \beta) = (1, 5)$ and $(\alpha, \beta) = (5, 10)$ on the largest image. On the other hand, the average number of linear iterations for the systems involving the variable u increases; therefore, the execution time for the simulations generally increase, except in a few cases.

For not convergent cases, we analyzed the impact of further decreasing $RTOL$ and verified that convergence is obtained for both the cases when $RTOL \leq 10^{-12}$. In particular for $RTOL = 10^{-12}$, 18 non-linear iterations are needed for the case $(\alpha, \beta) = (1, 5)$, while an average number of 206

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	4.2	2.1	1.7	38.8	12.7	10.1	367.9	114.6	65.7
5	38.7	5.1	4.1	197.8	24.3	18.6	NC	969.8	152.4
10	34.9	8.4	7.0	415.4	36.2	27.3	NC	2096.2	170.8

Table 6: *GS-FP* - $RTOL = 10^{-3}$: Execution Times in seconds.

and 2 iterations are needed for systems (7) and (8), respectively, leading to an execution time of about 499 seconds and a reduction of more than 50% with respect to the non-linear Gauss-Seidel. For the case $(\alpha, \beta) = (5, 10)$, where non-linear Gauss-Seidel did not converge, convergence of GS-FP is obtained within 21 non-linear iterations, an average number of 403 and 2 inner iterations for (7) and (8), respectively, and an execution time of about 1109 seconds.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	24 [4]	13 [6]	13 [6]	59 [2]	17 [5]	13 [6]	139 [2]	38 [6]	21 [4]
5	219 [3]	15 [21]	13 [24]	266 [4]	18 [18]	13 [24]	NC	250 [6]	32 [10]
10	188 [4]	17 [36]	13 [47]	486 [4]	21 [29]	14 [44]	NC	587 [5]	26 [24]

Table 7: *GS-FP* - $RTOL = 10^{-1}$: Non-linear and [average number of linear] iterations.

From Table 7, we can observe that a very low request of accuracy in the solution of inner linear systems is able to reduce in a meaningful way the number of non-linear iterations, generally producing a large reduction in execution times for Algorithm 1 with respect to GS, without affecting the convergence behaviour. Indeed, not converging cases for $RTOL = 10^{-1}$ are the same as in the other two choices of $RTOL$. On the other hand, a good choice for $RTOL$, producing the best tradeoff between reduction of non-linear

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	4.2	2.9	2.5	40.0	15.5	10.5	388.8	136.2	65.0
5	35.6	5.6	4.4	196.8	25.9	18.0	NC	984.6	123.5
10	32.8	9.1	6.9	360.8	40.2	28.1	NC	2050.0	143.0

Table 8: *GS-FP* - $RTOL = 10^{-1}$: Execution Times in seconds.

iterations and increasing of linear iterations, has to be made in order to have the best execution time with the problem at hand. We can observe that, for our test case, $RTOL = 10^{-2}$ is generally the best choice. Possible adaptive strategies to choose the inner solver tolerance could be applied; they could be based on a reduction of the inner solver tolerance in case of slowly decreasing or stagnant system residuals. However, we notice that this approach was not experimented in this work.

Finally, we remark that both algorithms (GS and GS-FP) converge to the same minimum value of the functional (2), with accuracy depending on the value of $ATOL$. However, we point out that Algorithm 1 is able to obtain a more rapid convergence to the minimum value already from the first non-linear iterations, showing that improving accuracy in the solution of inner linear systems is able to improve in a relevant way the reduction of the non-linear error in a few non-linear iterations. In Figure 4 we show the behaviour of the estimate of Ambrosio-Tortorelli functional in the first 15 iterations obtained by both the algorithms in the case $(\alpha, \beta) = (1, 1)$ and $\epsilon = 10^{-4}$ for the smallest image. Note that ATOL was fixed as in the previous analysis and $RTOL = 10^{-2}$ for Algorithm 1. We remark here that the same initial conditions were considered for both the algorithms, corresponding to an initial approximation of the functional of about 1.0×10^6, therefore, for both the algorithms we observe an increasing of the functional value at the first iteration, that in the case of GS is of two orders of magnitude, before starting the progressive descent to the minimum value of 5.9853×10^5. In the case of Algorithm 1 the increasing observed at the first iteration is negligible, indeed the descent already starts at the second iteration. This behaviour suggests that inner iterations are able to largely improve initial values of the unknowns leading to a faster convergence of the non-linear solver.

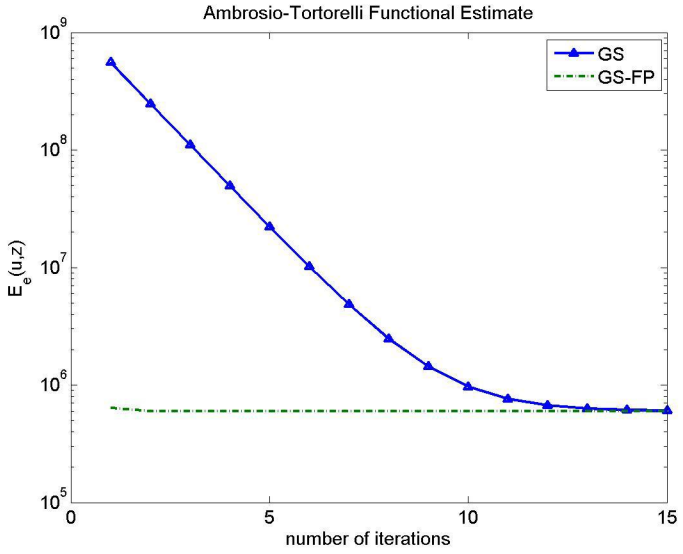

Figure 4: Ambrosio-Tortorelli functional estimate. Image size 256×256, $(\alpha, \beta) = (1, 1)$, $\epsilon = 10^{-4}$.

4.1.1. Results with Multigrid Solvers

In this Section we discuss performance results obtained when Algorithm 1 was used in conjunction with geometric multigrid solver as described in Section 3.1. Note that, since for previous analysis we observed that the most computational demanding solver is represented by the linear solver for systems (7), while solver for systems (8) converges in a few iterations also for increasing accuracy requests, we applied multigrid only for systems (7).

We first analyze results related to the use of a two-level V-cycle with 1 step of Gauss-Seidel as pre/post smoothing (V(1,1)-cycle), 1 step of the Gauss-Seidel method was also employed on the coarsest grid. For sake of brevity we discuss here only results obtained in the case of $RTOL = 10^{-2}$. In Table 9 we show the number of outer iterations and, in brackets, the average number of inner iterations for the variable u. In Table 10 the execution times in seconds are reported. Notice that the number of inner iterations for the variable z was always 1, as it was for the experiments described in the previous section, since the same (one-level Gauss-Seidel) algorithm was used for the related systems.

We observe that the multigrid solver generally produces, as expected, a

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	19	9	8	49	15	11	113	33	19
	[4]	[4]	[3]	[2]	[3]	[3]	[3]	[3]	(3]
5	148	14	10	200	17	12	NC	184	28
	[3]	[6]	[8]	[4]	[5]	[7]		[5]	[5]
10	121	15	11	281	18	12	NC	327	24
	[3]	[11]	[14]	[3]	[9]	[13]		[5]	[7]

Table 9: *GS-FP with $V(1,1)$-cycle - $RTOL = 10^{-2}$*: Non-linear and [average number of linear] iterations.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	4.0	1.9	1.7	39.9	12.5	9.3	392.5	120.2	69.3
5	30.8	3.9	3.1	184.8	17.6	14.1	NC	752.1	116.3
10	25.3	5.4	4.7	260.0	23.8	19.4	NC	1335.6	113.6

Table 10: *GS-FP with $V(1,1)$-cycle - $RTOL = 10^{-2}$*: Execution Times in seconds.

reduction of inner iterations for variable u, that in the best case (that is when $(\alpha, \beta) = (10, 10)$ for the smallest and the medium image size) is of about 75% with respect to the corresponding result in Table 3. In many cases, also a reduction of outer non-linear iterations was observed, with a general reduction of the overall execution times.

Although the use of a simple $V(1,1)$-cycle already reduces in a significative way the number of inner linear iterations in the solution of the systems (7), in the following we also analyze the impact of increasing both the number of pre/post-smoothing steps and the number of levels. In particular, in Tables 11-12 we show results obtained when a two-level V-cycle was applied with 2 sweeps of Gauss-Seidel as pre/post smoother ($V(2,2)$-cycle), while in Tables 13-14 we present results obtained when a three-level V-cycle with 1 pre/post smoothing steps is employed. For both the cases one step of Gauss-Seidel was chosen as coarsest solver. The same numerical parameters of the

experiments carried out with the two-level $V(1,1)$-cycle were used for both the set of experiments.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	18	9	7	45	15	11	117	33	19
	[3]	[3]	[3]	[2]	[2]	[2]	[2]	[2]	[2]
5	150	14	10	195	17	12	NC	191	27
	[2]	[4]	[6]	[2]	[4]	[5]		[3]	[4]
10	128	15	11	284	18	12	NC	317	24
	[2]	[7]	[9]	[3]	[6]	[9]		[4]	[5]

Table 11: *GS-FP with V(2,2)-cycle - RTOL* $= 10^{-2}$: Non-linear and [average number of linear] iterations.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	3.7	1.9	1.4	35.0	12.0	8.8	383.8	114.2	67.4
5	29.5	3.6	2.8	183.0	17.4	13.4	NC	713.6	111.4
10	24.5	4.9	4.3	256.2	22.7	18.6	NC	1305.9	111.7

Table 12: *GS-FP with V(2,2)-cycle - RTOL* $= 10^{-2}$: Execution Times in seconds.

We can see that by increasing both the number of pre/post smoothing steps and the number of levels produces a further improve in the number of inner iterations, sometimes producing also a reduction in the number of non-linear iterations and in the final execution times.

4.2. Comparison with Gradient Descent

Finally, in order to demonstrate the effectiveness of our generalized Gauss-Seidel method coupled with inner linear iterations, in Tables 15-16 we show number of iterations and execution times, respectively, obtained when a standard semi-implicit gradient-descent method was used on our test cases. The

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	19	9	8	50	15	11	113	33	19
	[3]	[3]	[2]	[2]	[2]	[2]	[2]	[2]	[2]
5	146	13	10	193	16	11	NC	190	28
	[3]	[3]	[4]	[3]	[3]	[3]		[3]	[3]
10	120	15	11	271	17	12	NC	327	24
	[2]	[5]	[6]	[3]	[4]	[5]		[3]	[4]

Table 13: GS-FP with three-level V(1,1)-cycle - $RTOL = 10^{-2}$: Non-linear and [average number of linear] iterations.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	4.3	2.0	1.8	42.2	12.8	9.6	409.0	120.2	69.3
5	31.1	3.0	2.4	189.1	15.3	10.8	NC	753.2	115.8
10	25.0	4.0	3.2	245.4	18.0	13.6	NC	1286.7	103.9

Table 14: GS-FP with three-level V(1,1)-cycle - $RTOL = 10^{-2}$: Execution Times in seconds.

method was obtained by introducing a fictitious time evolution for both variables u and z and solving parabolic equations associated to the equations in (3)-(4) to get steady-state solution by a finite-difference semi-implicit scheme. For details refer to [6]. In our experiments with the test cases discussed in the above sections a time step of $\Delta t = 0.01$ was needed for convergence. The stopping criterion was based on the maximum norm of the time variations of variable u, i.e., if the backward approximation of the time derivative was less than a fixed tolerance $ATOL = 10^{-10}$, the algorithm was stopped. We also fixed a maximum number of iterations equal to 10000 in case of slow convergence, reported in Tables 15-16 as NC cases.

We observe that the gradient descent method shows, as expected, a convergence rate largely dependent on image size. On the other hand, if we compare Table 16 with Tables 4-10, we notice that the non-linear Gauss-

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	3963	2283	2277	4773	2452	2262	6886	5062	3554
5	6676	2244	2244	7633	2438	2276	NC	NC	NC
10	6471	2266	2266	NC	2629	2244	NC	NC	NC

Table 15: *Semi-implicit Gradient Descent*: Number of iterations.

	256 ×256			512 ×512			1024 ×1024		
	α			α			α		
β	1	5	10	1	5	10	1	5	10
1	59.4	33.7	32.4	460.3	237.7	220.0	3351.3	2455.4	1724.9
5	94.2	31.3	30.8	737.6	236.6	238.4	NC	NC	NC
10	90.0	32.1	32.2	NC	255.6	223.3	NC	NC	NC

Table 16: *Semi-implicit Gradient Descent*: Execution Times in seconds.

Seidel method coupled with inner linear iterations largely outperforms the gradient descent method. Similar performance results were obtained also when the second-order finite-difference discretization scheme described in Section 2, coupled with an implicit time discretization, was applied to obtain an unconditionally stable scheme. Indeed, also in this case, when a time step $\Delta t = 0.1$ is used and the same accuracy is required on inner systems, non-linear Gauss-Seidel outperforms gradient descent. Detailed results were not included here for sake of brevity.

5. Concluding Remarks

In this work we focus on efficient and reliable numerical solution of the Ambrosio-Tortorelli model for image segmentation. The main motivation was the need to develop an effective software tool for large-scale analysis of images coming from high-throughput screening platforms for stem cells targeted differentiation. Due to its generality and well-based theory, we choose the phase-field approximation of the original Mumford-Shah variational model, developed by Ambrosio and Tortorelli, and we used a generalized relaxation

method, such as the Gauss-Seidel method, as non-linear solver for the related Eulero-Lagrange equations. We proposed to couple Gauss-Seidel non-linear relaxation with a fixed-point scheme which leads to solution of inner linear systems at each non-linear iteration for which efficient methods, such as classical linear multigrid, can be employed. We discuss simulation results obtained on a real image for varying model parameters and image size and we show that our solver largely outperforms the usual gradient descent schemes proposed in the literature. Future works will include both investigation of gradient descent methods with suitable line search algorithms as well as Newton-type methods coupled with effective strategies for calculation of descent directions and parallel implementation of the proposed solver exploiting parallel linear algebra kernels and data management routines of PSBLAS-MLD2P4 software framework [12].

6. Acknowledgements

We would like to thank Dr. Laura Casalino of the Stem Cell Fate Laboratory at the Institute of Genetics and Biophysics (IGB) of the National Research Council (CNR) of Italy for images used in our analysis. We also thank Dr. Riccardo March for useful discussions on the theoretical aspects of the Ambrosio-Tortorelli model and the practical choice of the model parameters. Finally, we would like to thank the anonymous reviewers for their useful comments, which helped us to improve the paper.

References

[1] L. Ambrosio, N. Fusco, D. Pallara (2000), *Functions of Bounded Variations and Free Discontinuity Problems*. Oxford University Press, Oxford.

[2] G. Aubert, P. Kornprobst (2002), *Mathematical Problems in Image Processing. Partial Differential Equations and the Calculus of Variations*. Springer-Verlag, New York.

[3] L. Ambrosio, V. M. Tortorelli (1990), *Approximation of Functional Depending on Jumps by Elliptic Functionals via* Γ*-convergence*, Comm. Pure Appl. Math., 43, pp. 999-1036.

[4] L. Ambrosio, V. M. Tortorelli (1992), *On the Approximation of Free Discontinuity Problem by Elliptic Functionals*, Boll. Un. Mat. Ital. B, vol. 6, pp. 105-123.

[5] L. Bar, G. Sapiro (2009), *Generalized Newton-Type Methods for Energy Formulations in Image Processing*. SIAM Journal on Imaging Sciences, 2, pp. 508-531.

[6] L. Bar et al. (2011), *Mumford and Shah Model and its Applications to Image Segmentation and Image Restoration*. In *Handbook of Mathematical Methods in Imaging, Vol. I* (ed. by O. Scherzer), Springer, pp. 1095-1157.

[7] G. Bellettini, A. Coscia (1994), *Discrete Approximation of a Free Discontinuity Problem*, Numer. Funct. Anal. Optim., 15, pp. 105-123.

[8] B. Bourdin (1999), *Image Segmentation with a Finite Element Method*, Math. Model. Numer. Anal. 33, pp. 229-244.

[9] A. Braides (2002), Γ*-convergence for Beginners*, Oxford University Press, Oxford.

[10] A. Brandt, S. F. McCormick, J. Ruge (1984), *Algebraic Multigrid (AMG) for Sparse Matrix Equations*, in Sparsity and Its Applications, D. J. Evans, ed., Cambridge University Press, Cambridge, pp. 257284.

[11] W. L. Briggs, V. E. Henson, S. F. McCormick (2000), *A Multigrid Tutorial, Second Edition*, SIAM, Philadelphia.

[12] P. D'Ambra, D. di Serafino, S. Filippone (2010), *MLD2P4: a Package of Parallel Algebraic Multilevel Domain Decomposition Preconditioners in Fortran 95*, ACM Transactions on Mathematical Software, 37, pp. 30:1-30:23.

[13] W. Hackbusch (2003), *Multi-Grid Methods and Applications*, Springer-Verlag, Berlin.

[14] R. J. LeVeque (2007), *Finite-Difference Methods for Ordinary and Partial Differential Equations, Steady-State and Time-Dependent Problems*, SIAM, Philadelphia.

[15] R. March, M. Dozio (1997), *A Variational Method for the Recovery of Smooth Boundaries*, Image and Vision Computing, 15, pp. 705-712.

[16] J. M. Morel, S. Solimini (1995), *Variational Methods in Image Segmentation*, Birkhäuser Boston Inc., Boston.

[17] D. Mumford and J. Shah (1989), *Optimal Approximations by Piecewise Smooth Functions and Associated Variational Problems*, Comm. Pure Appl. Math. 42, pp. 577-685.

[18] J. M. Ortega, W. C. Rheinboldt (1970), *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, Inc., London.

[19] T. J. Richardson, S. K. Mitter (1997), *A Variational Formulation-based Edge Focussing Algorithm*, Sādhanā, 22, Part 4, pp. 553-574.

[20] Saad, Y. (2003), *Iterative Methods for Sparse Linear Systems*, II edition, SIAM, Philadelphia.

[21] R. M. Spitaleri, R. March, D. Arena (1999), *Finite Difference Solution of Euler Equations arising in Variational Image Segmentation*, Numerical Algorithms, 21, pp. 353-365.

[22] A. Vitti (2012), *The Mumford-Shah Variational Model for Image Segmentation: An Overview of the Theory, Implementation and Use*, ISPRS J. of Photogrammetry and Remote Sensing, 69, pp. 50-64.

[23] C. R. Vogel, M. E. Oman (1996), *Iterative Methods for Total Variation Denoising*, SIAM J. Sci. Stat. Comput., 17, pp. 227-238.