A gradient-type algorithm for constrained optimization with applications
to multi-objective optimization of auxetic materials

C. Barbarosie, S. Lopes, A.M. Toader

November 6, 2018

Abstract

An algorithm is devised for solving minimization problems with equality constraints. The algorithm uses first-order derivatives of both the objective function and the constraints. The step is computed as a sum between a steepest-descent step (which minimizes the objective functional) and a correction step related to the Newton method (which aims to solve the equality constraints). The linear combination between these two steps involves coefficients similar to Lagrange multipliers which are computed in a natural way based on the Newton method. The algorithm uses no projection and thus the iterates are not feasible; the constraints are satisfied only in the limit (after convergence). This algorithm was proposed by one of the authors in a previous paper. In the present paper, a local convergence result is proven for a general non-linear setting, where both the objective functional and the constraints are not necessarily convex functions. The algorithm is extended, by means of an active set strategy, to account also for inequality constraints and to address minimax problems. The method is then applied to the optimization of periodic microstructures for obtaining homogenized elastic tensors having negative Poisson ratio (so-called auxetic materials) using shape and/or topology variations in the model hole. In previous works of the same authors, anisotropic homogenized tensors have been obtained which exhibit negative Poisson ratio in a prescribed direction of the plane. In the present work, a new approach is proposed, that employs multi-objective optimization in order
to minimize the Poisson ratio of the (possibly anisotropic) homogenized elastic tensor in several prescribed directions of the plane. Numerical examples are presented.

Keywords: non-linear programming, constrained minimization, multi-objective optimization, optimization of microstructures, porous materials, microstructure, auxetic materials

1 Introduction

We propose a numerical method for the minimization (or maximization) of a functional, subject to constraints. We also present a numerical application of this method.

Section 2 is devoted to the description of the algorithm and to a convergence result, along with extensions to accommodate inequality constraints and to deal with minimax problems. The method seeks for local solutions and works in the general case, for non-linear and non-convex objective functional and constraints. Smoothness of these functions is however required, since their gradients are used. The method is two-fold, involving a sum between a steepest-descent step (which minimizes the objective functional) and a correction step related to the Newton method (which aims to solve the equality constraints). The linear combination between these two steps uses certain coefficients similar to Lagrange multipliers which are computed in a natural way based on the Newton method. The algorithm uses no projection and thus the iterates are not feasible; the constraints are satisfied only in the limit (after convergence). Convergence is proven under the hypothesis that a certain Hessian-like matrix is positive definite at the solution. Inequality constraints are dealt with by using an active set methodology. Special attention is given to activation and deactivation strategies; the deactivation criterion is entirely based on the sign of the associated Lagrange multiplier. The case of an infinite (continuous) family of inequality constraints is discussed, as well as the extension of the method for treating minimax problems.

Section 3 shows an application on a large-scale example, involving the minimization of the Poisson ratio(s) of a composite material, in the context of homogenization theory. To achieve this goal, we perform shape and/or topology variations in the model hole that characterizes the microstructure. We use the minimax algorithm in order to minimize in simultaneous the Poisson coefficient of the (possibly anisotropic) homogenized material along many directions of the plane.

Some closing comments are made in Section 4.

2 The minimization algorithm

In this Section, we propose an algorithm for the minimization of a functional subject to constraints. The algorithm seeks for local solutions (as usual for gradient-based methods). It deals with non-essential constraints, that is, with
constraints whose violation does not render the problem ill-posed. In the method
here proposed, the constraints are usually violated during the optimization pro-
cess, and become satisfied only in the limit (after convergence); see, however,
subsection 2.4 for an exception.

We begin by describing the case of equality constraints (subsection 2.1). The
method here described has already been used in [1] by one of the authors to
solve large-scale optimization problems; in the present paper, the convergence of
the algorithm is proven under the hypothesis that a certain Hessian-like matrix
is positive definite at the solution (Theorem 3 in subsection 2.2). Our method-
ology can be regarded as a gradient method applied in the direction tangent to
the manifold determined by the constraints, together with a Newton method
applied in the orthogonal direction. This method, although not very fast (it has
linear convergence) is quite natural, easy to implement, and has the advantage
of requiring solely the first derivatives of the objective and of the constraint
functions.

A generalization is proposed in subsections 2.3 and 2.5 which deals with in-
equality constraints, based on an active-set strategy. During the optimization
process, an inequality constraint is activated as soon as it is violated. Its de-
activation depends on the sign of the associated Lagrange multiplier. To some
extent, this procedure can be seen as a generalization of the simplex method to
nonlinear functions. Based on these ideas, in subsection 2.7 we further extend
the algorithm with minimax problems in sight.

The following notation will be used: \( x \in \mathbb{R}^n \) is the vector of variables (also
called unknowns or parameters); \( x_i \) will denote the components of \( x \) while \( x^{(k)} \)
will denote a sequence of vectors; \( f \) is the objective function, a scalar function
of \( x \) that we want to minimize or maximize; the constraints will be modelled
by a vector function \( g : \mathbb{R}^n \to \mathbb{R}^m \). The Jacobian matrix of a vector function \( g \)
will be denoted by \( Dg \) while its transpose will be denoted by \( \nabla g \). In particular,
for a scalar function \( f \), \( \nabla f \) will be the usual gradient. The Hessian matrix of \( f \)
will be denoted by \( D^2 f \). Partial derivatives will be denoted by a comma, e.g.
\( g_{i,j} = \frac{\partial g_i}{\partial x_j} \).

We have collected in Appendix A some well-known results on unconstrained
optimization, while Appendix B gives the theoretical background on (equality)
constrained optimization.

### 2.1 A gradient algorithm for equality constrained problems

Consider the minimization problem

\[
\min_{x \in \mathcal{C}} f(x), \quad \mathcal{C} = \{ x \in \mathbb{R}^n : g(x) = 0 \}. \quad (\mathcal{P})
\]

A typical case in structural design arises when engineers adjust the parameters
(variables) to optimize the performance of a structure while keeping a prescribed
cost. In such a framework, the constraint function \( g \) appearing in \( (\mathcal{P}) \) is thought
of as a cost function, a scalar function that (in a broad sense) stands for the
structure’s “price” (or more precisely, the difference between the cost function and a prescribed “price”). For presentation purposes, the discussion will be initially restricted to this case of only one constraint \((m = 1)\) and subsequently extended to account for multiple constraints.

For the treatment of \((P)\) we will try to combine the ideas from Appendix \(A\) with those from Appendix \(C\). The algorithm should pursue two goals simultaneously: decrease the value of \(f\) while solving the equation \(g = 0\). Our approach sets up a direction that targets both goals at once, as described below.

Given an iterate \(x(\mathcal{X})\), the next iterate will be defined by an increment \(\delta(\mathcal{X})\), that is, \(x(\mathcal{X} + 1) = x(\mathcal{X}) + \delta(\mathcal{X})\). The increment \(\delta(\mathcal{X})\) will be the sum of two components: one of them is the vector \(-\eta \nabla f(x(\mathcal{X}))\) (with \(\eta > 0\) fixed) corresponding to the steepest descent algorithm; the other one aims at fulfilling the constraint equation \(g = 0\) and has the form \(-\lambda(\mathcal{X}) \nabla g(x(\mathcal{X}))\), where \(\lambda(\mathcal{X}) \in \mathbb{R}\) is a sort of Lagrange multiplier:

\[
\delta(\mathcal{X}) = -\eta \nabla f(x(\mathcal{X})) - \lambda(\mathcal{X}) \nabla g(x(\mathcal{X}))
\]

The multiplier \(\lambda(\mathcal{X})\) is defined adaptively in a natural way, inspired in Proposition \(19\) Appendix \(C\). It suffices to impose the Newton-type condition, relative to the equation \(g = 0\),

\[
\langle \nabla g(x(\mathcal{X})), \delta(\mathcal{X}) \rangle = -g(x(\mathcal{X}))
\]

which is immediately solvable:

\[
\lambda(\mathcal{X}) = \frac{g(x(\mathcal{X})) - \eta \langle \nabla g(x(\mathcal{X})), \nabla f(x(\mathcal{X})) \rangle}{\|\nabla g(x(\mathcal{X}))\|^2} \quad (1)
\]

With this choice of the multiplier, the whole procedure amounts to performing a “tangential gradient method” to minimize \(f\), together with a unidimensional Newton method to solve the constraint equation \(g = 0\).

To better understand the last assertion, consider the following reasoning. In the neighborhood of a solution \(x^*\) there are two main directions to consider from \(x(k)\): the direction \(\nabla g(x(k))\), orthogonal to the level set \(C_k = \{y \in \mathbb{R}^n : g(y) = g(x(k))\}\), and the subspace orthogonal to it (whose vectors are tangent to \(C_k\) at \(x(k)\)). In this latter subspace we have to minimize \(f\) (note that, since the solution \(x^*\) should minimize \(f\) in a level set of \(g\), \(C\), there is no point in decreasing \(f\) along directions other than tangent ones); in the direction of \(\nabla g(x(k))\) we want to solve the equation \(g = 0\), moving the next iterate closer to \(C\). A very simple method is obtained which, somewhat surprisingly, is not mentioned in the literature.

**Remark 1.** The algorithm here proposed is somewhat similar to the Newton method described in [6, Section 12.1] with the major difference that we use information related to the first derivatives only (of both the objective function and the constraints). Note that there are many practical problems in which second derivatives are impossible (or very expensive) to compute, see Section 3 of the present paper for an example. Note also that we do not use any projection.
matrices (see e.g. [3]); we don’t need to project since our iterates do not have to satisfy the constraints. Also, we have no need of introducing penalty functions (see e.g. [2]).

The algorithm generalizes naturally to vector-valued constraint functions $g : \mathbb{R}^n \to \mathbb{R}^m$ (with $m < n$). In this case $\lambda^{(k)} \in \mathbb{R}^m$ but the iterates are defined in a similar fashion by

$$x^{(k+1)} = x^{(k)} - \eta \nabla f(x^{(k)}) - \nabla g(x^{(k)}) \lambda^{(k)}.$$  

By imposing the Newton-type condition (again, inspired in Proposition 19, Appendix C)

$$Dg(x^{(k)}) \delta^{(k)} = -g(x^{(k)}),$$

we obtain

$$Dg(x^{(k)}) \nabla g(x^{(k)}) \lambda^{(k)} = g(x^{(k)}) - \eta Dg(x^{(k)}) \nabla f(x^{(k)}).$$  \hspace{1cm} (2)

In coordinate notation:

$$x^{(k+1)}_j = x^{(k)}_j - \eta f_{j}(x^{(k)}) - \sum_{i=1}^{m} \lambda^{(k)}_i g_{i,j}(x^{(k)}), \quad 1 \leq j \leq n,$$

where

$$\sum_{i=1}^{m} \sum_{j=1}^{n} g_{i,j}(x^{(k)}) g_{i,j}(x^{(k)}) \lambda^{(k)}_i = g_l(x^{(k)}) - \sum_{j=1}^{n} \eta g_{i,j}(x^{(k)}) f_{j}(x^{(k)}) , \quad 1 \leq l \leq m.$$

This linear system of equations uniquely determines $\lambda^{(k)}$ if $Dg(x^{(k)})$ has full rank (equal to $m$); see Definition 13 in Appendix B and the comments following it. Even in the case of vector-valued constraints, the method can be interpreted geometrically as a steepest descent method in the directions tangent to $C_k$ combined with a Newton method in the directions normal to $C_k$.

**Algorithm 2.**

**INPUT:** initial guess $x^{(0)}$, step size $\eta > 0$, tolerance $\varepsilon > 0$, maximum number of iterations $N$.

**OUTPUT:** approximate solution $x$ or message of failure.

**Step 1** With $k$ from 1 to $N$, do Steps 2–5.

**Step 2** Compute $\lambda$ by solving $Dg(x^{(0)}) \nabla g(x^{(0)}) \lambda = g(x^{(0)}) - \eta Dg(x^{(0)}) \nabla f(x^{(0)})$.

**Step 3** Set $x = x^{(0)} - \eta \nabla f(x^{(0)}) - \nabla g(x^{(0)}) \lambda$.

**Step 4** If $\|x - x^{(0)}\| < \varepsilon$ then OUTPUT($x$);

**STOP.**

**Step 5** Set $x^{(0)} = x$.

**Step 6** OUTPUT('The method failed after $N$ iterations.');

**STOP.**
2.2 Convergence results

We now state and prove the main theorem regarding the method proposed in subsection 2.1 above. This result has been presented in the preprint [4].

**Theorem 3.** Let \( f : \mathbb{R}^n \to \mathbb{R} \) and \( g : \mathbb{R}^n \to \mathbb{R}^m \) \((m < n)\) be twice continuously differentiable functions. Let \( x^* \in C \) be a regular point (see Definition 13 in Appendix B) satisfying the KKT conditions \((17)\) and such that the matrix \( H^* = D^2 f(x^*) + \sum_{i=1}^m \lambda_i^* D^2 g_i(x^*) \) is positive definite on \( T_{x^*} \) (see Theorem 15 in Appendix B). Then there exists \( r > 0 \) such that, given \( x(0) \in \overline{B}_r(x^*) \), the sequence of iterates defined by
\[
x^{(k+1)} = x^{(k)} - \eta \nabla f(x^{(k)}) - \nabla g(x^{(k)}) \lambda^{(k)}, \quad k \in \mathbb{N}_0,
\]
with \( \lambda^{(k)} \) determined by \((2)\), converges linearly to \( x^* \) for sufficiently small step lengths \( \eta > 0 \).

The reasoning follows the same pattern of the proof of Theorem 12 in Appendix A, but a bit more care will have to be exercised in this case. First of all, an auxiliary result is established.

**Lemma 4.** Let \( P \not\equiv 0 \) be an orthogonal projection on \( \mathbb{R}^n \). If \( A \not\equiv 0 \) is a self-adjoint linear operator on \( \mathbb{R}^n \), then \( v \not= 0 \) is an eigenvector of \( PA \), associated with the eigenvalue \( \mu \not= 0 \), if and only if

\[(i) \ v \in \text{Ran}(P),
\]
\[(ii) \ (A - \mu I)v \in \text{Ker}(P).
\]

Hence, the following estimate of the spectral radius holds: \( \rho(PA) \leq \rho(A|_{\text{Ran}(P)}) \).

**Proof.** The “if” part of the assertion is trivial. The “only if” part follows basically from the fact that, \( P \) being an orthogonal projection, one has the direct sum decomposition \( \mathbb{R}^n = \text{Ker}(P) \oplus \text{Ran}(P) \). Hence, given an eigenpair \( u \not= 0 \) and \( \mu \not= 0 \) of \( PA \), there are unique \( v \in \text{Ker}(P) \) and \( w \in \text{Ran}(P) \) such that \( Au = v + w \); but then, \( PAu = \mu w \) reads \( w = \mu u \). Therefore, it must be \( u \in \text{Ran}(P) \) and \( Au - \mu u = v \in \text{Ker}(P) \).

The last estimate is now obvious, since \( \rho(PA) = \rho(PA|_{\text{Ran}(P)}) \) and the spectral radius of an operator is dominated by the \( \ell^2 \) norm of that same operator (recall also that \( \|P\|_2 = 1 \) and that the spectral radius of a self-adjoint operator equals its \( \ell^2 \) norm).

**Remark 5.** Another useful result regarding spectral radii and matrix norms (whose proof can be found in [5] Section 1.4]), is that for any square matrix \( A \) and \( \varepsilon > 0 \), there exists a natural norm with the property that \( \|A\| < \rho(A) + \varepsilon \). Adding to this fact the considerations made in Corollary 11 Appendix A, one concludes that contractivity properties of differentiable maps \( S : \mathbb{R}^n \to \mathbb{R}^n \) are essentially governed by the spectral radius of their Jacobian matrices: if \( \rho(DS(x)) < 1 \), it always exists a vector norm for which \( S \) is locally contractive around \( x \).
Proof of Theorem 3. We begin by rewriting the algorithm to display its fixed point nature. Because $x^*$ is a regular point, $Dg(x^*)$ has full rank and the same is true for $Dg(x)$ with $x$ nearby $x^*$. Thus, equation (2) has a unique solution

$$
\lambda^{(k)} = [Dg(x^{(k)}) \nabla g(x^{(k)})]^{-1} [g(x^{(k)}) - \eta Dg(x^{(k)}) \nabla f(x^{(k)})].
$$

(4)

Putting this expression into (3) yields

$$
x^{(k+1)} = x^{(k)} - \eta \left[ I - \nabla g(x^{(k)}) [Dg(x^{(k)}) \nabla g(x^{(k)})]^{-1} Dg(x^{(k)}) \right] \nabla f(x^{(k)})
$$

$$
- \nabla g(x^{(k)}) [Dg(x^{(k)}) \nabla g(x^{(k)})]^{-1} g(x^{(k)});
$$

so $x^{(k+1)} = S(x^{(k)})$, upon defining $S(x) = x - P(x) \nabla f(x) - K(x)g(x)$. Because $x^*$ is a regular point, the operator $S$ is well defined locally around $x^*$. Because of Remark 5, one is left to establish that $Dg(x)$ is true for Remark 6.

Proof of Theorem 3. We begin by rewriting the algorithm to display its fixed point nature. Because $x^*$ is a regular point, the operator $S$ is well defined locally around $x^*$. Because of Remark 5, one is left to establish that $\rho(DS(x^*)) < 1$.

$K(x)$ is clearly a right inverse of $Dg(x)$ and it is not difficult to prove that $P(x)$ is the matrix of the orthogonal projection onto the tangent subspace $T_x$ to $C_x = \{ y \in \mathbb{R}^n : g(y) = g(x) \}$ at $x$. There are some trivial relations involving $P(x)$, $K(x)$ and $Dg(x)$, namely: $K(x)Dg(x) = I - P(x)$, $P(x)K(x) = 0$ and $P(x)\nabla g(x) = 0$; in view of this last equality, one can write

$$
S(x) = x - \eta P(x) [\nabla f(x) + \nabla g(x) \lambda^*] - K(x)g(x),
$$

and it is now easy to see, due to the KKT conditions (17) in Appendix B that the Jacobian matrix of $S$ at $x^*$ is given by

$$
DS(x^*) = I - \eta P(x^*) H^* - K(x^*) Dg(x^*)
$$

$$
= I - \eta P(x^*) H^* - [I - P(x^*)] = P(x^*) (I - \eta H^*).
$$

Since $I - \eta H^*$ is a symmetric matrix and $P(x^*)$ is the orthogonal projection’s matrix onto $T_{x^*}$, precisely the subspace where $H^*$ is positive definite, recalling Lemma 4 and the proof of Theorem 12 Appendix A the conclusion is now at hand.

Remark 6. The “true” Lagrange multiplier $\lambda^*$ can be easily approximated because the functional expression defining $\lambda^{(k)}$, using either (1) or (4) depending on the number of constraints, evaluates to $\eta \lambda^*$ at $x^*$. More precisely, the function

$$
\Lambda(x) = [Dg(x) \nabla g(x)]^{-1} [g(x) - \eta Dg(x) \nabla f(x)]
$$

is well defined around $x^*$ and $\Lambda(x^*) = \eta \lambda^*$ in view of the KKT conditions (17), Appendix B since $\Lambda$ is continuous (because $g$, $Dg$ and $\nabla f$ all are), for $x^{(k)}$ near $x^*$ we have $\lambda^{(k)} = \Lambda(x^{(k)}) \approx \Lambda(x^*)$, that is $\lambda^* = \eta^{-1} \lambda^{(k)}$.

Note that the constraints $g(x^{(k)})$ converge to zero faster than the iterates, see Remark 20 in Appendix C. This, together with Remark 21 implies that the
distance between $x^{(k)}$ and the manifold $C$ defined by the constraints converges to zero faster than the distance $\|x^{(k)} - x^*\|$. It is interesting to observe that the proposed algorithm converges even on certain minimization problems which do not satisfy the hypotheses of Theorem 3. This is the case of Example 12.1 in [6], brought by the authors as an evidence that a good minimization algorithm should take into account the curvature of the level set defined by the constraints, that is, information from the second-order derivatives of the constraints. The algorithm described in the present paper shows good convergence on this example, although it uses information from the first derivatives only.

2.3 Extension to inequality constraints

We now consider the problem

$$\min_{x \in C} f(x), \quad C = \{x \in \mathbb{R}^n : g(x) \leq 0\},$$

where the inequality is to be understood componentwise:

$$C = \{x \in \mathbb{R}^n : g_i(x) \leq 0, 1 \leq i \leq m\}.$$

The necessary optimality conditions for this sort of problem are better expressed in terms of the active constraints at a solution $x^* \in C$, that is, those constraints which attain equality:

$$\mathcal{A}^* = \{i \in \mathbb{N} : 1 \leq i \leq m, g_i(x^*) = 0\}.$$

The KKT conditions can then be written as follows:

$$\begin{cases}
\nabla f(x^*) + \sum_{i \in \mathcal{A}^*} \lambda_i^* \nabla g_i(x^*) = 0, \\
g_i(x^*) = 0, \quad i \in \mathcal{A}^*, \\
g_i(x^*) < 0, \quad i \notin \mathcal{A}^*, \\
\lambda_i^* \geq 0, \quad i \in \mathcal{A}^*, \\
\lambda_i^* = 0, \quad i \notin \mathcal{A}^*.
\end{cases}$$

See, for instance, [6, page 160]. The first two equations are simply the optimality conditions for the equality constrained problem obtained by requiring the active constraints to be zero. The third condition ensures that inactive constraints are satisfied. The last condition specifies that inactive constraints have null Lagrange multipliers attached; however, this condition is usually imposed in the KKT conditions for mere convenience; the values of those multipliers have no relevance whatsoever. The fourth condition is most important for practical purposes: Lagrange multipliers associated with active constraints must be non-negative. This will be useful in order to decide when to deactivate constraints along the iterations.

We propose a generalization of Algorithm 2 which can handle inequality constraints. As in Algorithm 2 the iterates are not necessarily feasible; see,
however, subsection 2.3 for an exception. The strategy is based on the concept of active set; this means that, at each iteration, the constraints are partitioned in two separate groups. Those inequalities considered active will be treated much in the same manner as the equality constraints are treated in Algorithm 2. The inequalities considered inactive are essentially ignored. Obviously, the set of active indices is not constant along the optimization process. Activating and deactivating inequality constraints is the central (and difficult) point of Algorithm 7.

In the proposed algorithm, an inequality is activated as soon as it is violated (step 3 in Algorithm 7). The deactivation criterion is not as straightforward. It is certainly not a good idea to deactivate a constraint as soon as it is fulfilled again (i.e., when the value of $g_i$ becomes negative again). Recall that an active inequality constraint is treated essentially as an equality constraint. Recall also that in our approach the constraints are not fulfilled along the optimization process (they are satisfied only in the limit). So, activating and deactivating an inequality constraint on the sole criterion of it being fulfilled or violated would often produce a zigzagging phenomenon (the same constraint being activated and deactivated repeatedly).

We propose that a constraint should be kept active as long as the process of minimization of $f$ has the tendency of violating that particular constraint. In order to measure this tendency, we use the sign of the respective Lagrange multiplier as a criterion. Lagrange multipliers associated to active constraints should be positive (see the above KKT conditions). Thus, we choose to deactivate a constraint when the associated Lagrange multiplier becomes negative (step 6 of Algorithm 7). To some extent, this procedure can be seen as a generalization of the simplex method to nonlinear functions. See [7, Section 16.5]) for a somewhat similar strategy; note that in [7] a distinction is made between active constraints and a working set of constraints, a terminology that we do not use. See also the discussion in [2, Section 3] where the term “basis” is used for the set of current active constraints.

The question arises as to what to do when more than one Lagrange multiplier becomes negative at the same iteration. Should we deactivate all the constraints corresponding to negative multipliers? Note that, if we deactivate one constraint, the remaining Lagrange multipliers should be computed again, and they may change signs. Should we deactivate only the constraint corresponding to the most negative multiplier? Does it make sense to compare the value of one Lagrange multiplier to another? In order to fix ideas, in Algorithm 7 we choose to deactivate the constraint corresponding to the most negative Lagrange multiplier, then compute again the remaining multipliers (steps 5 and 6). A more detailed discussion of the deactivation criterion is postponed to subsection 2.5.

Algorithm 7.

INPUT: initial guess $x^{(0)}$, step size $\eta > 0$, tolerance $\varepsilon > 0$, maximum number of iterations $N$.
OUTPUT: approximate solution $x$ or message of failure.

Step 1 Set $A = \emptyset$. (no active constraints)
Step 2 With \( k \) from 1 to \( N \), do Steps 3–9.

Step 3 With \( i \) from 1 to \( m \), do

If \( g_i(x(0)) > 0 \) then set \( \mathcal{A} = \mathcal{A} \cup \{i\} \); (constraint \( g_i \leq 0 \) is being violated, thus we set it active)

Step 4 Compute \( \lambda_j \) (\( j \in \mathcal{A} \)) by solving

\[
\sum_{j \in \mathcal{A}} \lambda_j \langle \nabla g_i(x(0)), \nabla g_j(x(0)) \rangle = g_i(x(0)) - \eta \langle \nabla g_i(x(0)), \nabla f(x(0)) \rangle,
\]

\( i \in \mathcal{A} \).

Step 5 Set \( i = \arg \min_{j \in \mathcal{A}} \lambda_j \).

Step 6 If \( \lambda_i < 0 \) then set \( \mathcal{A} = \mathcal{A} \setminus \{i\} \); (constraint \( g_i \leq 0 \) is set inactive)

GOTO Step 4.

Step 7 Set \( x = x(0) - \eta \nabla f(x(0)) - \sum_{j \in \mathcal{A}} \lambda_i \nabla g_i(x(0)) \).

Step 8 If \( \|x - x(0)\| < \varepsilon \) then OUTPUT(x);

STOP.

Step 9 Set \( x(0) = x \).

Step 10 OUTPUT('The method failed after \( N \) iterations.');

STOP.

Convergence proofs for such methods usually assume some idealized procedure that is hardly employed in practice. We prefer not to state any kind of convergence result. In general, convergence cannot be guaranteed and zigzagging\(^1\) can sometimes occur, although experience shows it to be a rare phenomenon.

Step 4 of Algorithm 7 can become very heavy if many constraints are active (and thus many Lagrange multipliers must be computed). Note that the number of active constraints cannot exceed the number of variables, so this can only happen for a large number of variables. In subsection 2.4 below, we describe how the computational burden associated with many active constraints can be significantly alleviated in a specific particular case.

Finally, let us note that it is not difficult to combine Algorithms 2 and 7 in order to treat the case when equality constraints are present together with inequality constraints. Simply, the equality constraints should be kept always active.

2.4 The case of box-like constraints

We now turn our attention to constraints of the simple form \( g_i(x) = a_i - x_i \) or \( g_i(x) = x_i - b_i \); they confine the vector variable \( x \) to a rectangular box in \( \mathbb{R}^n \).

Due to their particular form, constraints of this type deserve a special treatment. First, it is very easy to make a projection for such inequalities. So, they can be treated as essential constraints by performing a projection as soon as they are violated (in step 3), that is, by setting \( x_i = a_i \) or \( x_i = b_i \) respectively.

Second, their gradient has only one non-zero component (in the variable \( x_i \)). Because of this, the corresponding Lagrange multipliers can be eliminated from the linear system in step 4 of Algorithm 7. Thus, it suffices to compute the other Lagrange multipliers (if any) by solving a reduced system of linear equations. Then, the Lagrange multipliers associated to box-like constraints

\(^1\)The set of active constraints changes many times.
can be recovered one by one without computational effort. We shall explain this process in some detail.

Suppose there are \( m_1 + m_2 \) active constraints; suppose that the first \( m_1 \) of them are of box-type, either of the form \( x_i \geq a_i \) or \( x_i \leq b_i \). Note that this implies that each of these box-type constraints corresponds to a certain variable; of course it is impossible for both \( x_i \geq a_i \) and \( x_i \leq b_i \) to become active simultaneously for the same variable \( x_i \).

The system of linear equations defining the Lagrange multipliers \( \lambda_j \) in step 4 of Algorithm 7 writes

\[
\sum_{j=1}^{m_1+m_2} \lambda_j \langle \nabla g_i, \nabla g_j \rangle = g_i - \eta \langle \nabla g_i, \nabla f \rangle, \quad 1 \leq i \leq m_1 + m_2
\]

We treat differently the first \( m_1 \) unknowns and the first \( m_1 \) equations:

\[
\sum_{j=1}^{m_1} \lambda_j \langle \nabla g_i, \nabla g_j \rangle + \sum_{k=m_1+1}^{m_1+m_2} \lambda_k \langle \nabla g_i, \nabla g_k \rangle = g_i - \eta \langle \nabla g_i, \nabla f \rangle, \quad 1 \leq i \leq m_1
\]

\[
\sum_{j=1}^{m_1} \lambda_j \langle \nabla g_i, \nabla g_j \rangle + \sum_{k=m_1+1}^{m_1+m_2} \lambda_k \langle \nabla g_i, \nabla g_k \rangle = g_i - \eta \langle \nabla g_i, \nabla f \rangle, \quad m_1 + 1 \leq l \leq m_1 + m_2
\]

Since \( 1 \leq i \leq m_1 \), we have that \( \nabla g_i = \pm e_i \), \( e_i \) being the canonical basis in \( \mathbb{R}^n \).

To fix ideas, we suppose that \( \nabla g_i = e_i \); since \( 1 \leq j \leq m_1 \), we have \( \nabla g_j = e_j \).

Also, note that \( g_i = 0 \) (when the box-like constraints are activated, a projection operation is performed, thus they are satisfied exactly) Thus, the linear system writes

\[
\lambda_i + \sum_{k=m_1+1}^{m_1+m_2} \lambda_k g_{k,i} = -\eta f_{i,i}, \quad 1 \leq i \leq m_1
\]

\[
\sum_{j=1}^{m_1} \lambda_j g_{i,j} + \sum_{k=m_1+1}^{m_1+m_2} \lambda_k \langle \nabla g_i, \nabla g_k \rangle = g_i - \eta \langle \nabla g_i, \nabla f \rangle, \quad m_1 + 1 \leq l \leq m_1 + m_2
\]

By using the first \( m_1 \) equations, we easily express each \( \lambda_i \) in terms of \( \lambda_k \) \((m_1 + 1 \leq k \leq m_1 + m_2)\). By plugging these expressions into the second part of the system, we get

\[
- \sum_{j=1}^{m_1+m_2} \left( \sum_{k=m_1+1}^{m_1+m_2} \lambda_k g_{k,j} + \eta f_{j,j} \right) g_{i,j} + \sum_{k=m_1+1}^{m_1+m_2} \lambda_k \langle \nabla g_i, \nabla g_k \rangle = g_i - \eta \langle \nabla g_i, \nabla f \rangle, \quad m_1 + 1 \leq l \leq m_1 + m_2
\]

and thus

\[
\sum_{k=m_1+1}^{m_1+m_2} \lambda_k \langle \nabla g_i, \nabla g_k \rangle - \sum_{k=m_1+1}^{m_1+m_2} \lambda_k g_{k,j} g_{i,j} = g_i - \eta \langle \nabla g_i, \nabla f \rangle + \sum_{j=1}^{m_1} \eta f_{j,j} g_{i,j}, \quad m_1 + 1 \leq l \leq m_1 + m_2
\]
By expanding the inner products between gradients, we rewrite the above as
\[
\sum_{k=m_1+1}^{m_1+m_2} \sum_{j=m_1+1}^{n} \lambda_k g_{k,j} g_{i,j} = g_l - \eta \sum_{j=m_1+1}^{n} f_j g_{l,j}, \quad m_1 + 1 \leq l \leq m_1 + m_2
\]
or, equivalently,
\[
\sum_{k=m_1+1}^{m_1+m_2} \lambda_k \langle \nabla^* g_l, \nabla^* g_k \rangle = g_l - \eta \langle \nabla^* g_l, \nabla^* f \rangle, \quad m_1 + 1 \leq l \leq m_1 + m_2
\]
where the symbol \( \nabla^* \) denotes the gradient of the respective function \textit{with respect to the last} \( n - m_1 \) variables only. Also, the symbol \( \langle \cdot, \cdot \rangle \) in the above system of linear equations represents the inner product in \( \mathbb{R}^{n-m_1} \) and not in \( \mathbb{R}^n \) as in the previous formulae.

Based on the above considerations, Algorithm 7 can be reformulated as follows (we denote by \( B \) the set of box-like constraints).

**Algorithm 8.**

**INPUT:** initial guess \( x^{(0)} \), step size \( \eta > 0 \), tolerance \( \varepsilon > 0 \), maximum number of iterations \( N \).

**OUTPUT:** approximate solution \( x \) or message of failure.

**Step 1** Set \( A = \varnothing \). (\textit{no active constraints})

**Step 2** With \( k \) from 1 to \( N \), do Steps 3–9.

**Step 3** With \( i \) from 1 to \( m \), do

If \( g_i(x^{(0)}) > 0 \) then set \( A = A \cup \{i\} \); (constraint \( g_i \leq 0 \) is being violated, thus we set it active)

If \( i \in B \) then set \( x_i^{(0)} = a_i \) or \( b_i \); (we project)

**Step 4** Compute \( \lambda_j \) \( (j \in A \setminus B) \) by solving
\[
\sum_{j \in A \setminus B} \lambda_j \langle \nabla^* g_i(x^{(0)}), \nabla^* g_j(x^{(0)}) \rangle = g_i(x^{(0)}) - \eta \langle \nabla^* g_i(x^{(0)}), \nabla^* f(x^{(0)}) \rangle,
\]
i \( \in A \setminus B \).

**Step 5** Compute \( \lambda_j \) \( (j \in A \cap B) \) as \( \lambda_j = -\eta f_j(x^{(0)}) - \sum_{k \in A \setminus B} \lambda_k g_{k,j}(x^{(0)}) \)

**Step 6** Set \( i = \arg \min_{j \in A} \lambda_j \).

**Step 7** If \( \lambda_i < 0 \) then set \( A = A \setminus \{i\} \); (constraint \( g_i \leq 0 \) is set inactive)

GOTO Step 4.

**Step 8** Set \( x = x^{(0)} - \eta \nabla f(x^{(0)}) - \sum_{i \in A} \lambda_i \nabla g_i(x^{(0)}) \).

**Step 9** If \( \|x - x^{(0)}\| < \varepsilon \) then OUTPUT(\( x \));

STOP.

**Step 10** Set \( x^{(0)} = x \).

**Step 11** OUTPUT(‘The method failed after \( N \) iterations.’);

STOP.

Recall the tricky detail that “blocked” variables \( x_i \) with \( i \in A \cap B \) should be ignored when computing the scalar products between gradients is step 4 of Algorithm 8; they should also be left unchanged in step 8 since a projection has been performed previously (in step 3). In a word, those “blocked” variables
must be treated as if they were no longer variables but mere parameters, equal
to $a_i$ or to $b_i$. However, Lagrange multipliers associated to constraints in $\mathcal{A} \cap \mathcal{B}$
are meaningful (and are used in steps 6 and 7 to decide deactivation).

The technique above described has been successfully used in [8].

2.5 Activation and deactivation strategies

Different criteria for deactivating constraints (steps 5 and 6 of Algorithm [7]
steps 6 and 7 of Algorithm [8]) may be considered when more than one Lagrange
multiplier becomes negative at a certain step of the algorithm. For instance, one
could deactivate at once all the constraints with negative multipliers instead of
deactivating only the most negative one.

We suggest that two different situations should be distinguished. In the first
one, which we shall describe as discrete constraints, there is a relatively small
number of inequality constraints. These constraints may be very different of each
other. They may have different physical nature, perhaps different physical units
and different orders of magnitude. It makes no sense to compare their values,
and it makes no sense to compare their associated Lagrange multipliers. Thus,
there is no point in choosing the “most negative” multiplier, as done in step 5 of
Algorithm [7] (step 6 of Algorithm [8]). We consider that in this case at most one
Lagrange multiplier should become negative at each step of the algorithm. The
event of more than one Lagrange multiplier becoming negative at a certain step
should be interpreted as a warning that the optimization process is going too
fast. Perhaps the value of the parameter $\eta$ should be decreased. Thus, in this
case, steps 5 and 6 of the Algorithm [7] (steps 6 and 7 of Algorithm [8]) should be
reformulated in order to test whether more than one multiplier is negative, and
to take appropriate measures if this happens.

Actually, the above considerations apply also to the activation of constraints.
If more than one constraint is violated at one step of the algorithm, this again
should be viewed as a warning that the optimization process is going too fast.

In the second situation, there are many inequality constraints, very similar
to each other. They share the same physical units and have the same order
of magnitude. In a word, they are comparable. We shall call such a set of
constraints an almost-continuum of constraints because this may appear, for
instance, as a discretization of the continuum case described in subsection 2.6.
Note that this is not the same as the situation described in subsection 2.4,
where there are many constraints but they are very different from each other
(they constrain different variables $x_i$).

In the case of an almost-continuum of constraints, even before discussing
the deactivation strategy, we should take a look at how activation is done, that
is, at how constraints are violated. Since there are many constraints, close to
each other, if one of them is violated we expect the “neighbour” ones, that is,
one which are similar to it, to be violated, too. So, “group violations” are
to be expected in this second situation. If this happens, there is no point in
activating the whole group of constraints. We propose that only the “worst”
one should be activated, that is, the one which becomes more positive (recall
that the constraints are of the form \( g_i \leq 0 \). In subsequent steps, the algorithm will push the value of that “worse” \( g_i \) towards zero, and this will have the side effect of pushing also its “neighbour” constraints towards zero.

It is not an easy task to implement the above described ideas into a computer program. Clusters of violated constraints must be identified and monitored along the optimization process. Each cluster should have a “leader” (the most violated constraint in that cluster) which is active. By pushing the “leader” towards zero one hopes to control the behaviour of the whole cluster. Along the optimization process, clusters may merge or split, which makes this programming task really challenging.

The implementation of the above ideas is only possible if the (almost-continuous) set of constraints has some internal organization which allows us to identify the closest neighbours of a given constraint, like the one described in Remark 9 for instance. Using this notion of vicinity, we suggest the following activation strategy. Among the set of all violated constraints, one should only activate those which are “more violated” than all of its neighbours (these will be the “leaders”). Along the optimization process, one should keep checking if the neighbours of the “leaders” become “more violated” than the “leader” and switch the activation flag towards the “most violated” one.

Going back to the deactivation issue, we see that the case of an almost-continuum of constraints is actually not very different from the case of discrete constraints, since only the “leaders” of the clusters of constraints have been activated. Thus, the active constraints continue to be few and “far” from each other (that is, different), and the same deactivation strategy should be employed as in the first situation.

Finally, note that a combination of the two situations (discrete constraints and almost-continuum constraints) may appear in some examples. It is not difficult to adapt the algorithm in order to deal with such problems. Even several almost-continua of constraints can be treated in the same manner.

2.6 A continuum of constraints

A true continuum of constraints may also be of interest for certain problems. In the sequel, we shall use the notation \( g_\xi \) instead of \( g_i \), \( \xi \) being a parameter indexing the family of constraints.

For instance, in the example studied in section 3, it would be interesting to allow for any angle between 0 and 180 degrees (this would give rise to a one-dimensional continuum of constraints). Another example is the optimization of a structure subject to an incoming wave; the wave may come from any direction of the plane and may have any frequency within a certain range (this would be a two-dimensional continuum of constraints). Often, the optimization of a structure subject to multiple loads also falls into this category.

This case of a continuum of constraints can be dealt with, more or less in the same manner as described in subsections 2.3 and 2.5. The main novelty is that we should now actively seek for the worst case (the most violated constraint) within a continuum, and this should be done by means of a maximization al-
algorithm. This may be viewed as a dual approach: on one hand, we minimize (in $x$) a function $f(x)$ while on the other hand we look for the worst case by maximizing (in $\xi$) $g_\xi(x)$.

Note that we have not implemented these ideas yet; the example given in Section 3 has been solved using the discrete approach described in subsections 2.3 and 2.5.

2.7 Extension to minimax problems

We now turn to the problem of minimizing simultaneously a family of functionals. More precisely, the goal is to minimize the maximum of several different functionals:

$$\min_x \max \{f_1(x), f_2(x), \ldots, f_m(x)\},$$

where all of the $f_i$ are smooth functions (making it tempting to try and bypass the non-smoothness of the inner max-function in some way). It is easy to rewrite this problem into a form appropriate to be treated by the method described in subsections 2.3 and 2.5. It suffices to introduce a new (fake) variable $z$ and minimize (in $x$ and $z$) the function $F(x,z) = z$ subject to the constraints $f_i(x) \leq z$, $i = 1, 2, \ldots, m$.

The case of a continuous minimax problem can be dealt with in the same way. The problem

$$\min_x \max_{\xi} f_\xi(x)$$

can be reformulated as the minimization of $F(x,z) = z$ subject to $f_\xi(x) \leq z$, $\forall \xi$. Thus, the method described in subsection 2.6 can be applied.

3 Application of the algorithm to the optimization of auxetic materials

We now show an application of the minimization algorithm described in subsection 2.7 to the optimization of macroscopic properties of periodic microstructures, in the context of linearized elasticity. With the goal of designing a composite material having negative Poisson ratio along all directions, we use the algorithm for minimizing in simultaneous the Poisson ratio of the composite along many directions of the plane (10 directions in the first example, 18 directions in the second example). These results have been presented in the preprint [9].

A composite material will be described as a periodic microstructure, that is, a linearly elastic body whose material coefficients vary at a microscopic scale, according to a periodic pattern. Homogenization theory allows one to accurately describe the macroscopic behaviour of such a microstructure by means of so-called cellular problems, which are elliptic PDEs subject to periodicity conditions. Porous materials, that is, bodies with periodic infinitesimal perforations, can be described in a similar manner.
3.1 The cellular problem

We shall consider a model hole, which is a compact set $T \subset Y$ (see Figure 1), where $Y$ is the periodicity cell. Usually, $Y$ is the unit cube in $\mathbb{R}^n$; see, however, [10] for a general notion of periodicity.

The perforated body is obtained by removing from $\mathbb{R}^n$ translations of the model hole. For a cubic cell $Y$, one has (see Figure 2)

$$\mathbb{R}^n_{\text{perf}} = \mathbb{R}^n \setminus \bigcup_{\vec{k} \in \mathbb{Z}^n} (T + \vec{k})$$

Figure 1: Periodicity cell with model hole (zoomed)

Figure 2: Periodically perforated plane $\mathbb{R}^2_{\text{perf}}$

The cellular problem describing the behaviour of such a porous material is:

$$\begin{cases}
\text{find } u_A(\vec{x}) = A\vec{x} + \phi_A(\vec{x}), \\
\text{where } \phi_A \text{ is a periodic function, and} \\
-\text{div}(C\varepsilon(\vec{u}_A)) = \vec{0} \text{ in } \mathbb{R}^n_{\text{perf}} \\
C\varepsilon(\vec{u}_A)\vec{n} = \vec{0} \text{ on } \partial T
\end{cases}$$

The above problem models the microscopic behaviour of a microstructure with elastic tensor $C$, occupying the domain $\mathbb{R}^2_{\text{perf}}$ and subject to the macroscopic strain $A$. The homogenized elastic tensor $C^H$, describing the effective (macroscopic) behaviour of this microstructure, is given by

$$C^H A = \frac{1}{|Y|} \int_{Y \setminus T} C\varepsilon(\vec{u}_A)$$

16
\[ \langle C^H A, B \rangle = \frac{1}{|Y|} \int_{Y \setminus T} \langle C \varepsilon(\vec{u}_A), \varepsilon(\vec{u}_B) \rangle, \]  

(8)

where \( A \) and \( B \) are given macroscopic strains.

The cellular problem (6) can be reformulated in stress, as follows (see \[10\]):

\[
\begin{cases}
\vec{w}_\sigma \in L^p_{\text{perf}}, \\
-\text{div}(C \varepsilon(\vec{w}_\sigma)) = \vec{0} \text{ in } \mathbb{R}^n_{\text{perf}} \\
C \varepsilon(\vec{w}_\sigma) \vec{n} = \vec{0} \text{ on } \partial T \\
\frac{1}{|Y|} \int_{Y \setminus T} C \varepsilon(\vec{w}_\sigma) = \sigma,
\end{cases}
\]

(9)

where \( \sigma \) represents an applied macroscopic stress.

We shall denote by \( \mathbf{D}^H \) the homogenized compliance tensor, that is, the inverse of \( C^H \).

### 3.2 Shape and topology derivatives

The effective elastic properties of the above described porous body can be optimized by varying the size and shape of existing holes in the periodicity cell \( Y \), and also by creating new, infinitesimal, holes.

The first approach is called shape optimization (here applied at the cellular level). The shape derivative describes the variation of a certain objective functional when an infinitesimal deformation is applied to a given geometry. Consider \( \vec{\theta} : \mathbb{R}^n \rightarrow \mathbb{R}^n \) a vector field defining the deformation; note that \( \vec{\theta} \) itself should be periodic in order to preserve the periodic character of the microstructure under study. Then the variation induced by this deformation in the quantity \( \langle C^H A, B \rangle \) is (see \[1\] and \[10\], Section 6)

\[
D_S \langle C^H A, B \rangle = \frac{1}{|Y|} \int_{\partial T} \left[ 2\mu \varepsilon(\vec{u}_A), \varepsilon(\vec{u}_B) \right] \vec{\theta} \cdot \vec{n}
\]

where \( \vec{n} \) is the unit vector normal to the boundary of the hole \( T \) and pointing inside \( T \). Assuming that \( C \) is a linear isotropic elastic tensor, \( C \xi = 2\mu \xi + \lambda (\text{tr} \xi) I \), the above formula becomes

\[
D_S \langle C^H A, B \rangle = \frac{1}{|Y|} \int_{\partial T} \left[ 2\mu \varepsilon(\vec{u}_A), \varepsilon(\vec{u}_B) + \lambda \text{tr} \varepsilon(\vec{u}_A) \text{tr} \varepsilon(\vec{u}_B) \right] \vec{\theta} \cdot \vec{n}
\]

In particular, this gives the shape derivative of the homogenized coefficients:

\[
D_S C^H_{ij} = D_S \langle C^H f_i, f_j \rangle = \frac{1}{|Y|} \int_{\partial T} \left[ 2\mu \varepsilon(\vec{u}_f), \varepsilon(\vec{u}_f) + \lambda \text{tr} \varepsilon(\vec{u}_f) \text{tr} \varepsilon(\vec{u}_f) \right] \vec{\theta} \cdot \vec{n}
\]

(10)

where \( (f_i)_{i=1,2,3} \) is the following basis in the space of symmetric matrices

\[
\begin{align*}
\mathbf{f}_1 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\
\mathbf{f}_2 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \\
\mathbf{f}_3 &= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}
\end{align*}
\]
and $\tilde{u}_i$ are the corresponding solutions of the cellular problem (6) with effective strain $f_i$.

A second approach for the optimization of a structure is topology variation (here applied at the cellular level). It consists in drilling an infinitesimal circular hole and imposing zero Neumann condition on the newly created boundary. The topological derivative describes the infinitesimal variation thus induced in the functional $\langle C^H A, B \rangle$, and depends on the location $x$ of the new hole. It can be proven (see [10, Section 5] and [11]) that the topological derivative is given by

$$
D_T \langle C^H A, B \rangle (x) = -\frac{\pi \lambda + 2\mu}{\lambda + \mu} \left[ 4\mu \varepsilon(\tilde{u}_A) \varepsilon(\tilde{u}_B) + \lambda^2 + 2\lambda\mu - \mu^2 \frac{\varepsilon(\tilde{u}_A) \varepsilon(\tilde{u}_B)}{\mu} \right](x)
$$

(11)

In [12], an algorithm was proposed for optimizing the microgeometry of the hole(s) in the cellular problem, with the goal of improving certain macroscopic properties of the porous microstructure (which is a body with periodically distributed infinitesimal perforations). The algorithm alternates shape variations with topology variations until a certain convergence criterion is fulfilled. The properties to be optimized include the effective bulk modulus, the effective response to shear and the effective Poisson coefficient (see [12, Section 6]).

Both shape and topology derivatives of the homogenized compliance tensor $D^H$ are obtained from the derivatives of the homogenized tensor $C^H$ by:

$$
D_S D^H_{ijkl} = -D_S^H D_C^H_{ij} \delta \gamma \delta D^H_{ijkl}, \quad D_T D^H_{ijkl} = -D_T^H D_T C^H_{ij} \delta \gamma \delta D^H_{ijkl}.
$$

(12)

Note that formula (12) uses the coordinate notation for the fourth-order tensors $C^H$ and $D^H$, while in (11) the indices $i$ and $j$ are relative to the basis $(f_i)_{i=1,2,3}$ in the space of symmetric matrices.

### 3.3 Poisson ratios and the minimax technique

This work focuses on the search of two-dimensional periodic microstructures exhibiting negative Poisson ratio at the macroscopic level (so-called auxetic materials). In previous works of the same authors [10], [12], anisotropic effective elastic tensors have been obtained which exhibit negative Poisson ratio in a prescribed direction of the plane (the horizontal direction), see Figure 3. In the present work, we look for periodic microstructures with the same negative Poisson ratio among all directions in the plane. This is done by combining the techniques described in the above subsection 3.2 (for shape optimization at the cellular level) with the minimax algorithm described in subsection 2.7 which ensures that the largest Poisson ratio among many directions in the plane is being minimized.

Note that the effective elastic tensor resulting from the homogenization technique, defined by (7) or (8), is not isotropic in general. Thus, the notion of Poisson coefficient must be defined with care: it is minus the ratio between the
Figure 3: Optimized microstructures with respect to one direction only

transverse strain and the axial strain when the material is stretched or compressed along the axial direction, see [13]. In the two dimensional case under consideration, a Poisson ratio can be associated to each unit vector $\vec{v} = (v_1, v_2)$, arbitrarily chosen in the plane. Consider a stretching stress applied along the direction of $\vec{v}$; in the frame $\{\vec{v}, \vec{v}^\perp\}$ the stress writes $\sigma = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$. Then the Poisson ratio $\nu_{\vec{v}}$ in the direction $\vec{v}$ is defined as $\nu_{\vec{v}} = -\frac{\varepsilon_{\perp\perp}}{\varepsilon_{\vec{v}\vec{v}}}$, where $\varepsilon_{\perp\perp}$ is the strain in the direction $\vec{v}^\perp$ and $\varepsilon_{\vec{v}\vec{v}}$ is the strain in the direction $\vec{v}$. The above defined stress $\sigma$ expressed in cartesian coordinates has the form

$$\sigma = \begin{pmatrix} \frac{v_1^2}{v_1 v_2} & \frac{v_1 v_2}{v_2^2} \\ \frac{v_1 v_2}{v_1 v_2} & \frac{v_2^2}{v_1} \end{pmatrix}$$

and the associated strain matrix is $\varepsilon = D^H \sigma$ (recall that $D^H$ is the homogenized compliance tensor). Then the axial strain is $\varepsilon_{\vec{v}\vec{v}} = D^H \sigma \cdot \vec{v}$ and the transverse strain is $\varepsilon_{\perp\perp} = D^H \sigma \vec{v}^\perp \cdot \vec{v}^\perp$. The Poisson ratio writes as $\nu_{\vec{v}} = -\frac{D^H \sigma \vec{v}^\perp \cdot \vec{v}^\perp}{D^H \sigma \vec{v} \cdot \vec{v}}$ and introducing the stress corresponding to a stretch in the direction $\vec{v}^\perp$, denoted by

$$\sigma^{\perp} = \begin{pmatrix} \frac{v_2^2}{v_1 v_2} & -\frac{v_1 v_2}{v_2^2} \\ -\frac{v_1 v_2}{v_1 v_2} & \frac{v_1^2}{v_1} \end{pmatrix},$$

it becomes :

$$\nu_{\vec{v}} = -\frac{\langle D^H \sigma, \sigma^{\perp} \rangle}{\langle D^H \sigma, \sigma \rangle}.$$  

The derivative of $\nu_{\vec{v}}$ with respect to $D^H$ is given by :

$$\frac{\partial \nu_{\vec{v}}}{\partial D^H_{ijkl}} = -\frac{\sigma_{ij} \sigma_{kl}^{\perp}}{\langle D^H \sigma, \sigma \rangle} + \frac{\langle D^H \sigma, \sigma^{\perp} \rangle}{\langle D^H \sigma, \sigma \rangle^2} \sigma_{ij} \sigma_{kl}$$

19
The above formulae (15) and (16) are suitable for implementation since they can actually be seen as depending on the homogenized tensor $C^H$, see (12).

The minimax algorithm proposed in subsection 2.7 will be applied to this problem. The family of functionals to minimize simultaneously consists of all Poisson ratios associated to the effective elasticity tensor $C^H$ in many different directions of the plane. Specifically, we consider a (large but finite) set of directions $\vec{v}$ in the plane and for each direction we associate the respective Poisson ratio $\nu_v$ defined by (15), (13) and 14. Each Poisson ratio $\nu_v$ is a functional of $D^H$ which is the inverse tensor of $C^H$ which in turn is function of the shape and topology of the perforations denoted by $T$ in subsection 3.1. The respective derivatives of these dependencies are given by formulas (12) and (16).

The algorithm will minimize the largest Poisson ratios; this means that, after convergence, the effective elastic tensor thus obtained will have roughly the same Poisson ratio in all directions of the plane; also, if the process is successful, this Poisson ratio will be negative. Note, however, that this does not ensure that $C^H$ is isotropic.

**Remark 9.** We use a (finite) family of directions $\vec{v}$ indexed by an angle varying from $0^\circ$ to $180^\circ$. Using the terminology from subsection 2.5, this is an almost-continuous family of functionals organized as a ring (0 degrees actually gives the same direction as 180 degrees). Within this organization, each constraint has two closest neighbours.

### 3.4 Numerical implementation and numerical results

The algorithm used in this work is an improved version of our home-made code, presented in [12, Section 4]. The improvement consists in the addition of a minimax routine which handles the optimization of the worst case among several functionals, as described subsection 2.7.

As explained in [12, Section 4], in order to discretize problem (6), the microstructure is meshed with triangular finite elements of Lagrange type of degree two. Some of the triangles are marked as “full”, corresponding to the elastic material, while other triangles are marked as “empty”, corresponding to the hole $T$. The interface between material and hole is marked in red (see Figure 3). Note that, although for graphical purposes several contiguous cells are represented, the mesh covers only one cell $Y$ and is “closed” in itself, having no boundary. It can be described as a mesh on the two-dimensional torus; the graphical representation in Figure 3 refers to an unfolded mesh where vertices, segments and triangles are drawn more than once.

In order to implement the periodicity condition in (6), linear+periodic functions are considered on this mesh (they can be identified with multi-functions on the torus). This is done by keeping track of segments crossing the boundary of the cell $Y$ and by taking into account the jump of the function along those segments.

Along the optimization process, the mesh deforms in order for the holes to change their shape. The deformation of the mesh is accomplished by simply
moving the vertices. However, this implies a gradual loss in the quality of the mesh: sharp angles appear eventually, as well as too long or too short segments. At some point, certain triangles may even become flat or be reversed. Of course this must be prevented, since it turns the process of solving problem (6) by the finite element method ill posed and consequently unstable. With this end in view, the program improves frequently the quality of the mesh, either by moving the vertices (equilibrating the mesh) or by changing the elements of the mesh (flipping segments, adding/eliminating vertices). See [12, Section 5] for details.

For shape optimization, the integrands in (10) are computed. These are scalar functions defined on the boundary of the holes and depending on the solutions $u_j$ of three cellular problems ($j = 1, 2, 3$). A functional $J$ is chosen which depends on the homogenized coefficients $C_{ij}^H$ (as explained in subsection 3.3 here we actually consider several functionals to be minimized simultaneously, the Poisson ratios). The shape derivative of $J$ is computed as

$$D_SJ = \sum_{ij} \frac{\partial J}{\partial C_{ij}^H} D_SC_{ij}^H = \int_{\partial T} \gamma \bar{\theta} \cdot \vec{n}$$

where $\gamma$ is a scalar function computed as a linear combination of the integrands in (10). If a steepest descent method were used, one should choose a deformation of the interface $\partial T$ equal to $\bar{\theta} = -\gamma \vec{n}$ (multiplied by some positive constant $\eta$ which controls the speed of the process) in order to decrease the value of $J$. Here, we use instead the method described in subsection 2.7 in order to minimize simultaneously several functionals, and the shape derivative of each of the Poisson ratios is used in Algorithm 8 accordingly. The algorithm provides a desired deformation of the interface $\partial T$, which is then propagated into the whole mesh by means of an averaging process (see [12, Section 9]).

Topology optimization can be performed by simply changing the triangles neighbour to a certain vertex from material to void. However, in the present paper we focus only on shape optimization, which means that we begin the optimization process with a certain number of holes in the periodicity cell, and this number is going to remain constant up to the end of the optimization process.

In the first example we optimize the Poisson ratios along ten directions in the plane, at angles uniformly distributed between 0 and 180 degrees. The largest Poisson ratio among those ten directions is minimized. The initial microstructure (initial guess), see Figure 4, has a square periodicity, property that does not vary during the optimization process. It presents two model holes that repeat periodically. The algorithm, after 54 iterations, produced the microstructure on the right in Figure 4, still with square periodicity and still presenting two model holes. At iteration 47 the Poisson ratios became negative and the final design presents Poisson ratios less than -0.7. In the history of convergence, see Figure 5, one can observe that in the first iterations the algorithm makes the ten Poisson ratios as close as possible and afterwards it decreases them all together.

In the second example eighteen directions in the plane are chosen, at angles uniformly distributed between 0 and 180 degrees. The largest Poisson ratio among those directions is minimized. The initial guess has a hexagonal periodicity and presents one model hole repeated periodically (with respect to the
hexagonal periodicity), see Figure 6. After 60 iterations the algorithm produces the microstructure presented in Figure 6, on the right. The history of convergence is shown in Figure 7, with two zoom-in views. The final design has Poisson ratios close to -0.9 in all eighteen directions.

4 Conclusions and future development

An algorithm for optimization with equality constraints is presented and proven to be convergent. It can be regarded as a gradient method applied in the direction tangent to the manifold determined by the constraints, together with a Newton method applied in the orthogonal direction. This method, although not very fast (it has linear convergence) is quite natural, easy to implement, and has the advantage of requiring solely the first derivatives of the objective and of the constraint functions.

Generalizations of this algorithm are described in order to address inequality constrained problems also, as well as minimax problems. Criteria for activation and deactivation of constraints are discussed in some detail, as well as the special case of box-like constraints. An approach to solve problems involving a (continuous) infinite family of inequality constraints is also discussed.

An application to optimization of periodic microstructures, for obtaining homogenized elastic tensors with negative Poisson ratio, is presented. It uses shape and/or topology variations in the model hole that characterizes the microstructure. Multi-objective optimization is employed in order to minimize the Poisson ratio of the homogenized elastic tensor in several prescribed directions.
of the plane, in order to obtain a material having roughly the same negative Poisson ratio in all directions.

It is possible to obtain periodic 2D microstructures with Poisson ratio close to −1 and roughly the same in all directions of the plane. The examples show that the algorithm tends to “cut” the structure, so the bulk modulus approaches zero.

Future work includes the study of three-dimensional microstructures, the treatment of a continuum of constraints and its extension to continuous minimax problems. When put in the context of optimization of microstructures, the capacity to deal with a continuum of constraints would allow one to impose as a constraint that the composite material be truly isotropic.

Acknowledgement

This work was supported by Fundação para a Ciência e a Tecnologia, UID/MAT/04561/2013.

A Unconstrained optimization

Unconstrained optimization is the minimization or maximization of a scalar function $f$ defined on the whole $\mathbb{R}^n$.

Minimization algorithms require the user to supply a starting point, which will be denoted by $x^{(0)}$. Then, at each iteration, the algorithm chooses a direction $\delta^{(k)}$ and searches along this direction, from the current iterate $x^{(k)}$, for a new iterate with a lower function value. The distance to move along $\delta^{(k)}$, the step length, is commonly chosen after a finite number of trial step lengths; this strategy is known as line search. This kind of procedure is useful for obtaining convergence from “remote” initial approximations $x^{(0)}$, which is not our main concern. Besides, once a locally convergent algorithm has been devised, with the step length taken to be constant throughout, one can always modify it to encompass line search in order to enhance its convergence properties (this is usually the order things are done anyway).

It is quite natural to look for a descent direction, that is, a direction $\delta^{(k)}$ such that $\langle \nabla f(x^{(k)}), \delta^{(k)} \rangle < 0$, where $\langle \cdot, \cdot \rangle$ denotes the usual dot product in $\mathbb{R}^n$. The steepest descent direction $\delta^{(k)} = -\nabla f(x^{(k)})$ is the most obvious choice. This steepest descent method has the advantage of requiring the calculation of first derivatives only, but it can be quite slow.

The following standard results will be used (they can be easily found in textbooks on Functional Analysis):

Theorem 10 (Banach fixed-point theorem). Assume that $K$ is a nonempty closed set in a Banach space $E$ (with norm $\| \|$), and further, that $S : K \to K$ is a contractive mapping (i.e., a Lipschitzian mapping with Lipschitz constant $L$ strictly lower than one). Then there exists a unique $x^* \in K$ such that $x^* = S(x^*)$ and, for any $x^{(0)} \in K$, the sequence $(x^{(k)})$ defined by $x^{(k+1)} = S(x^{(k)})$, $k \in \mathbb{N}_0$,
stays in \( K \) and converges to \( x^* \). Furthermore, the following estimate holds: 
\[
\|x^{(k)} - x^*\| \leq L^k \|x^{(0)} - x^*\|, \text{ for all } k \in \mathbb{N}_0.
\]

**Corollary 11.** Let \( S : E \to E \) be a continuously Fréchet differentiable operator and \( x^* \in E \) a point such that \( S(x^*) = x^* \). If the Fréchet derivative of \( S \) at \( x^* \) has operator norm strictly lower than one, then the conclusions of the previous theorem hold with \( K = \{ x \in E : \|x - x^*\| \leq r \} \), for some \( r > 0 \). In the finite dimensional case \( E = \mathbb{R}^n \), this is equivalent to the requirement that \( \|DS(x^*)\| < 1 \) for some natural norm.\(^2\)

The classical local convergence result for the steepest descent method is now presented. The assumptions, as well as the proof, are somewhat different than the usual ones encountered in most of the literature, in the sense that we regard the method as a fixed-point iteration. This choice suits best our reasoning for the convergence proof in subsection 2.2.

**Theorem 12.** Assume that \( f \) is a twice continuously differentiable function whose Hessian matrix \( D^2f(x^*) \) at a local minimizer \( x^* \) is positive definite. Then there exists \( r > 0 \) such that, given \( x^{(0)} \in B_r(x^*) = \{ x \in \mathbb{R}^n : \|x - x^*\|_2 \leq r \} \), the steepest descent method \( x^{(k+1)} = x^{(k)} - \eta Df(x^{(k)}) \), \( k \in \mathbb{N}_0 \), converges linearly to \( x^* \) for sufficiently small step lengths \( \eta > 0 \).

**Proof.** Taking \( S : \mathbb{R}^n \to \mathbb{R}^n \) defined as \( S(x) = x - \eta Df(x) \), the steepest descent method becomes \( x^{(k+1)} = S(x^{(k)}) \), \( k \in \mathbb{N}_0 \). Since we are not interested in proving global convergence, the contractivity property will not be needed in all of \( \mathbb{R}^n \), but only locally near \( x^* \). By Corollary 11, it suffices to check that \( \|DS(x^*)\| < 1 \) for some natural norm.

It is clear that \( DS(x^*) = I - \eta D^2f(x^*) \) is a symmetric matrix; then we know that the \( \ell_2 \) norm of \( DS(x^*) \) coincides with the spectral radius of this same matrix (see \([5\, Section 1.4]\)). The eigenvalues of \( DS(x^*) \) take the form \( 1 - \eta \mu_i^* \) \( (1 \leq i \leq n) \), where \( \mu_1^* \geq \cdots \geq \mu_n^* \) are the eigenvalues of \( D^2f(x^*) \); given that the latter are all positive, we have \( 1 - \eta \mu_i^* \in [1 - \eta \mu_1^*, 1 - \eta \mu_n^*] \) \( (1 \leq i \leq n) \) and the choice \( 0 < \eta < \frac{\mu_1^*}{\mu_n^*} \) implies that \( [1 - \eta \mu_1^*, 1 - \eta \mu_n^*] \subset ] - 1, 1[ \). Hence, one gets \( \|DS(x^*)\|_2 = \rho(DS(x^*)) \) strictly lower than one. \( \square \)

### B Optimization under equality constraints

Constrained optimization can be viewed as the superposition of a minimization problem and a non-linear equation. Besides the **objective function** \( f : \mathbb{R}^n \to \mathbb{R} \) that we want to minimize, a **constraint function** \( g : \mathbb{R}^n \to \mathbb{R}^m \) is given defining certain equations that the unknown vector \( x \) must satisfy. Thus, the optimization problem can be written (considering only equality constraints) as:

\[
\min_{x \in \mathcal{C}} f(x), \quad \mathcal{C} = \{ x \in \mathbb{R}^n : g(x) = 0 \}. \quad (P)
\]

\(^2\)A matrix norm that is associated with a vector norm is called a natural norm.
Constrained optimization problems arise from models involving, for instance, budgetary limitations or other specifications on the design. Unconstrained optimization problems arise directly in many practical applications, and also as reformulations of constrained ones, if the constraints are replaced by penalization terms added to the objective function and having the effect of discouraging violations of the constraints, or by other means (e.g. by parametrizing the set $C$).

**Definition 13.** A point $x \in \mathbb{R}^n$ satisfying the constraint $g(x) = 0$ is said to be a **regular point** if the gradient vectors $\nabla g_1(x), \nabla g_2(x), \ldots, \nabla g_m(x)$ are linearly independent. In other words, the Jacobian matrix $Dg(x)$ should have full rank (equal to $m$).

Note that at a regular point $x$ the constraint function $g$ is a submersion, giving $C$ the appropriate geometrical concept, namely that of a submanifold of $\mathbb{R}^n$; the tangent subspace to $C$ is given by $T_x = \{\tau \in \mathbb{R}^n : Dg(x)\tau = 0\}$. Note also that $m < n$; in fact, $m \geq n$ would yield a discrete set of feasible points, a situation which is outside the scope of the present paper.

The optimality conditions for constrained optimization problems are more complicated than for the unconstrained case.

**Theorem 14.** If $x^* \in \mathbb{R}^n$ is a solution of $(P)$ and $x^*$ is a regular point, then there exists a unique $\lambda^* \in \mathbb{R}^m$ (called the Lagrange multiplier) such that the following conditions hold:

\[
\begin{cases}
\nabla f(x^*) + \nabla g(x^*) \lambda^* = 0, \\
g(x^*) = 0.
\end{cases}
\]

In coordinate notation:

\[
\begin{cases}
f_{i,j}(x^*) + \sum_{i=1}^m \lambda^*_i g_{i,j}(x^*) = 0, & 1 \leq j \leq n, \\
g_i(x^*) = 0, & 1 \leq i \leq m.
\end{cases}
\]

These equations are often referred to as **Karush-Kuhn-Tucker conditions** or **KKT conditions** for short. They are necessary for optimality, but not sufficient.

A sufficient optimality condition can be given by the action of the Hessian matrices $D^2f(x^*)$ and $D^2g_i(x^*)$ of $f$ and $g_i (1 \leq i \leq m)$, respectively, over tangent vectors to $C$ at $x^*$. This sufficient optimality condition involves also the values of the Lagrange multipliers $\lambda^*_i (1 \leq i \leq m)$. The following result can be found in many textbooks on optimization (see, for instance, [6, Section 11.4]).

**Theorem 15.** Suppose there are $x^* \in \mathbb{R}^n$ and $\lambda^* \in \mathbb{R}^m$ such that the KKT conditions hold. Suppose also that the matrix $H^* = D^2f(x^*) + \sum_{i=1}^m \lambda^*_i D^2g_i(x^*)$ is positive definite on $T_{x^*}$, that is, for any nonzero vector $\tau$ tangent to $C$, there holds $\langle H^* \tau, \tau \rangle > 0$. Then $x^*$ is a strict local minimizer of $(P)$.  

25
C Newton’s method

Let $g : \mathbb{R} \to \mathbb{R}$ be a (non-linear) function and consider the problem of finding a root of $g$, that is, of solving the equation $g(x) = 0$. One well-known method is Newton’s method (also known under the name of Newton-Raphson). It consists of starting with some initial guess $x^{(0)} \in \mathbb{R}$ and defining the sequence $x^{(k)}$ by iterating

$$x^{(k+1)} = x^{(k)} - g(x^{(k)}) / g'(x^{(k)})$$

(18)

Theorem 16. Suppose $g$ is twice continuously differentiable and let $x^* \in \mathbb{R}$ such that $g(x^*) = 0$ and $g'(x^*) \neq 0$. Then, if $x^{(0)}$ is sufficiently close to $x^*$, the sequence $x^{(k)}$ defined by (18) converges to $x^*$. Moreover, the convergence is quadratic.

The same basic idea can be applied to systems of (non-linear) equations:

Theorem 17. Suppose $g : \mathbb{R}^n \to \mathbb{R}^n$ is twice continuously differentiable and let $x^* \in \mathbb{R}^n$ such that $g(x^*) = 0$ and $Dg(x^*)$ is invertible. Then, if $x^{(0)}$ is sufficiently close to $x^*$, the sequence $x^{(k)}$ defined by

$$x^{(k+1)} = x^{(k)} - (Dg(x^{(k)}))^{-1} g(x^{(k)})$$

(19)

converges to $x^*$. Moreover, the convergence is quadratic.

Newton’s method can be extended to solve under-determined equations. For instance, let $g : \mathbb{R}^n \to \mathbb{R}^m$, with $m \leq n$, be a non-linear function and consider the problem of finding a root of $g$, that is, of solving the equation $g(x) = 0$. If $m < n$, this equation is under-determined; the set of solutions will be a manifold $C$ in $\mathbb{R}^n$ of dimension $n - m$. However, it may be of interest to solve numerically this equation, that is, to start with some $x^{(0)} \in \mathbb{R}^n$ and to build iteratively a sequence $x^{(k)}$ which converges to some $x^*$ such that $g(x^*) = 0$. This can be done, roughly speaking, by building a step orthogonal to the level set of $g$ and obeying, within that orthogonal subspace, to the basic idea of the Newton method. The following result describes the procedure in detail.

Theorem 18. Let $g : \mathbb{R}^n \to \mathbb{R}^m$ ($m \leq n$) be differentiable, $Dg$ be of full rank $m$ in an open convex set $D$ and let the following hold:

(i) there exists $K \geq 0$ and $\alpha \in [0, 1]$ such that $\|Dg(y) - Dg(x)\| \leq K \|y - x\|^\alpha$ for all $x, y \in D$;

(ii) there is a constant $B$ for which $\|Dg(x)^+\| \leq B$ for all $x \in D$, where $Dg(x)^+ = \nabla g(x) [Dg(x) \nabla g(x)]^{-1}$ is the Moore-Penrose inverse of $Dg(x)$.

Furthermore, for $\eta > 0$, let $D_\eta = \{x \in D : \|y - x\| < \eta \Rightarrow y \in D\}$. Then there exists an $\epsilon > 0$ depending only on $K$, $\alpha$, $B$ and $\eta$ such that if $x^{(0)} \in D_\eta$ and $\|Dg(x^{(0)})\| < \epsilon$, then the iterates $x^{(k)}$ determined by

$$x^{(k+1)} = x^{(k)} - Dg(x^{(k)})^+ g(x^{(k)})$$

(20)
are well defined and converge to a point \( x^* \in D \) such that \( g(x^*) = 0 \). Moreover, there is a constant \( \beta \) for which

\[
\left\| x^{(k+1)} - x^* \right\| \leq \beta \left\| x^{(k)} - x^* \right\|^{1+\alpha}, \quad k = 0, 1, 2, \ldots
\]

Note that, if \( m = n \), the iteration (20) above reduces to equation (19) which defines the sequence \( x^{(k)} \) in the determined case; the rate of convergence is still quadratic if \( Dg \) is Lipschitz continuous. Convergence theory on such Newton-like methods, for under-determined systems of equations, can be found in [14] and references therein.

Since the considered system of equations has many solutions (a manifold of them), it is legitimate to question whether the sequence \( x^{(k)} \) must be so rigidly defined. That is, one may ask whether the step \( \delta^{(k)} \) must be necessarily orthogonal to the level set of \( g \). The following result (whose proof is immediate) shows that the philosophy behind Newton’s method can be applied while allowing for uncertainties in the definition of the sequence \( x^{(k)} \).

**Proposition 19.** Let \( g : \mathbb{R}^n \to \mathbb{R}^m \) be continuously differentiable and consider a sequence \( x^{(k)} \) which converges to some \( x^* \in \mathbb{R}^n \) and which satisfies

\[
Dg(x^{(k)}) \delta^{(k)} = -g(x^{(k)}) \tag{21}
\]

where \( \delta^{(k)} = x^{(k+1)} - x^{(k)} \). In coordinate notation,

\[
\sum_{j=1}^{n} g_{i,j}(x^{(k)}) \delta^{(k)}_j = -g_i(x^{(k)}), \quad 1 \leq i \leq m
\]

Then \( x^* \) is a solution, that is, \( g(x^*) = 0 \).

Note that condition (20) implies (21). Note also that, if \( m = n \), then (19), (20) and (24) are all equivalent. However, for \( m < n \), unlike (19) or (20), equation (21) does not define uniquely the sequence \( x^{(k)} \); it simply states a property of the sequence. The user has the freedom to chose \( n - m \) components of \( \delta^{(k)} \) : those orthogonal to \( \nabla g_1(x^{(k)}), \nabla g_2(x^{(k)}), \ldots, \nabla g_m(x^{(k)}) \), that is, components tangent to the level set of \( g \). The user may use this freedom in order to solve other equation(s) or to minimize some functional.

**Remark 20.** The quantity \( g(x^{(k)}) \) converges to zero faster than the convergence of \( x^{(k)} \to x^* \). For instance, suppose \( x^{(k)} \) converges linearly, that is, there is a constant \( L \in [0, 1] \) such that \( \| x^{(k)} - x^* \| \) is of order \( O(L^k) \). Then \( \delta^{(k)} = x^{(k+1)} - x^{(k)} \) is also of order \( O(L^k) \) and a simple Taylor expansion about \( x^{(k)} \) yields

\[
g(x^{(k+1)}) = g(x^{(k)}) + Dg(x^{(k)}) \delta^{(k)} + O(\| \delta^{(k)} \|^2);
\]

\[
= 0, \quad \text{due to equation (21)}
\]

thus, \( g(x^{(k)}) \) is of order \( O(L^{2k}) \). Note however that this is not quadratic convergence, but simply an improved linear one. Nor should quadratic convergence be expected because, unlike in Theorem 18 here the step \( \delta^{(k)} = x^{(k+1)} - x^{(k)} \) is not a full Newton-type step.
Remark 21. It can be proven that, locally around a regular point \( x^* \in \mathcal{C} = \{ x \in \mathbb{R}^n : g(x) = 0 \} \) (see Definition 13 in [3]), the quantity \( \| g(x^{(k)}) \| \) is of the same order of magnitude as the distance \( \text{dist}(x^{(k)}, \mathcal{C}) \) to the manifold \( \mathcal{C} \). This geometric property, taken together with Remark 20 above, implies that the distance \( \text{dist}(x^{(k)}, \mathcal{C}) \) converges to zero faster than the convergence of \( x^{(k)} \to x^* \).

References

[1] C. Barbarosie, Shape optimization of periodic structures, Computational Mechanics, 30, 235–246, 2003.

[2] J.T. Betts, An Accelerated Multiplier Method for Nonlinear Programming, Journal of Optimization Theory and Applications, 21(2), 137–174, 1977.

[3] J. Rosen, The Gradient Projection Method for Nonlinear Programming, II. Nonlinear Constraints, Journal of the Society for Industrial and Applied Mathematics, 9, 514–532, 1961.

[4] C. Barbarosie, S. Lopes, A gradient-type algorithm for optimization with constraints, preprint CMAF Pre-2011-001, available at http://cmaf.fc.ul.pt/preprints.html

[5] P.G. Ciarlet, Introduction à l’Analyse Numérique Matricielle et à l’Optimisation, Masson, 1990.

[6] J. Bonnans, J. Gilbert, C. Lemaréchal, C. Sagastizábal, Numerical Optimization – Theoretical and Practical Aspects, Springer, 2003.

[7] J. Nocedal, S. Wright, Numerical Optimization (Second Edition), Springer, 2006.

[8] C. Barbarosie, S. Lopes, A generalized notion of compliance, Comptes Rendus Mécanique, 339, 641–648, 2011.

[9] C. Barbarosie, A.-M. Toader, Multi-objective optimization of composite materials with negative Poisson ratio along several directions, preprint CMAF Pre-011, 2013, available at http://cmaf.fc.ul.pt/preprints.html

[10] C. Barbarosie, A.-M. Toader, Shape and Topology Optimization for periodic problems, Part I, The shape and the topological derivative, Structural and Multidisciplinary Optimization, 40, 381–391, 2010.

[11] A.-M. Toader, The topological derivative of homogenized elastic coefficients of periodic microstructures, SIAM Journal on Control and Optimization, 49(4), 1607–1628, 2011.

[12] C. Barbarosie, A.-M. Toader, Shape and Topology Optimization for periodic problems, Part II, Optimization algorithm and numerical examples, Structural and Multidisciplinary Optimization, 40, 393–408, 2010.
[13] A. Boresi, R. Schmidt, O. Sidebottom, *Advanced Mechanics of Materials*, Wiley, 1993.

[14] H. Walker, L. Watson, *Least-change secant update methods for underdetermined systems*, SIAM Journal of Numerical Mathematics, 27(5), 1227–1262, 1990.
Figure 5: History of convergence, zoom of the first 40 iterations and zoom of the last 6 iterations
Figure 6: Initial guess and final microstructure for hexagonal periodicity
Figure 7: History of convergence, zoom of the first 40 iterations, zoom of the last 8 iterations