On Minimization of Nonlinear Energies Using FEM in MATLAB

Alexej Moskovka[1(✉)], Jan Valdman[2,3], and Marta Vohnoutová[2]

[1] Department of Mathematics, Faculty of Applied Sciences, University of West Bohemia, Technická 8, 30100 Plzeň, Czech Republic
alexmos@kma.zcu.cz

[2] Department of Computer Science, Faculty of Science, University of South Bohemia, Branišovská 31, 37005 České Budějovice, Czech Republic

[3] The Czech Academy of Sciences, Institute of Information Theory and Automation, Pod vodárenskou věží 4, 18208 Prague, Czech Republic

Abstract. Two minimization problems are added to the Moskovka and Valdman MATLAB package (2022): a Ginzburg-Landau (scalar) problem and a topology optimization (both scalar and vector) problem in linear elasticity. Both problems are described as nonlinear energy minimizations that contain the first gradient of the unknown field. Their energy functionals are discretized by finite elements, and the corresponding minima are searched using the trust-region method with a known Hessian sparsity or the Quasi-Newton method.

Keywords: minimization · nonlinear energy · finite elements · Ginzburg-Landau model · topology optimization

1 Introduction

For solving problems given by (a system of) partial differential equations, the variational approach is based on finding a minimum of the corresponding energy functional

$$J(u) = \min_{v \in V} J(v)\,, \tag{1}$$

where V is a space of test functions defined in a domain Ω and includes Dirichlet boundary conditions on $\partial\Omega$. Problems of this type appear in various applications of physics and are mathematically studied in the calculus of variations. The energy functionals are then described by integrals over domains in two- or three-dimensional space. The finite element method [9] can be applied as an approximation of (1) and results in a minimization problem

$$J(u_h) = \min_{v \in V_h} J(v) \tag{2}$$

formulated in the finite-dimensional subspace V_h of V.

A. Moskovka was supported by the MSMT CR 8J21AT001 Model Reduction and Optimal Control in Thermomechanical Systems project. J. Valdman announces the support of the Czech Science Foundation (GACR) through the GF21-06569K grant Scales and shapes in continuum thermomechanics.

R. Wyrzykowski et al. (Eds.): PPAM 2022, LNCS 13827, pp. 331–342, 2023.
https://doi.org/10.1007/978-3-031-30445-3_28

A recent MATLAB implementation of [5,6] using the simplest linear nodal basis functions allows us to solve (2) efficiently. The energy formulations of the studied problems, including p-Laplace and hyperelasticity, contain the first gradient parts of searched functions discretized by the finite element method (FEM) and formulated as the sum of energy contributions from local elements. The key ingredient is the vectorization of exact or approximate energy gradients in the nodal patches (sets of elements adjacent to particular nodes). This leads to a time-efficient implementation with a higher memory cost. New attempts to apply available techniques to problems of elastoplastic deformations of layered structures and shape memory alloys are reported in [11,12].

In this contribution, we comment on the implementation of the Ginzburg-Landau model in superconductivity [1,3,4] and the topology optimization problem of the elastic medium [2,8]. The resulting MATLAB codes are provided for download and testing at the following link:

https://www.mathworks.com/matlabcentral/fileexchange/97889

Assembly times were obtained on Lenovo ThinkPad T14 Gen 1 (Intel Core i7 processor, 2021) with 16 GB memory running MATLAB R2018a.

2 Finite Element Method and Minimization

The subspace V_h is spanned by a set of n_b basis functions $\varphi_i(x) \in V_h, i = 1, \ldots, n_b$, and a trial function $v \in V_h$ is expressed by a linear combination

$$v(x) = \sum_{i=1}^{n_b} v_i \, \varphi_i(x), \qquad x \in \Omega, \tag{3}$$

where $\bar{v} = (v_1, \ldots, v_{n_b}) \in \mathbb{R}^{n_b}$ is a vector of coefficients. We consider only the case $V_h = P^1(\mathcal{T})$, where $P^1(\mathcal{T})$ is the space of piecewise linear nodal basis functions defined on a triangulation $\mathcal{T}$ of the domain Ω with a Lipschitz boundary. Note that the number of nodes corresponds to the number of all the basis functions of V_h, therefore, $n_b = |\mathcal{N}|$. Consequently, the minimizer $u_h \in V_h$ of (2) is represented by a vector of coefficients $\bar{u} = (u_1, \ldots, u_{|\mathcal{N}|}) \in \mathbb{R}^{|\mathcal{N}|}$ and some coefficients of $\bar{u}, \bar{v}$ related to the Dirichlet boundary conditions are prescribed.

An appropriate minimization method is needed to solve (2). We use the MATLAB Optimization Toolbox [10] which provides minimization techniques based on two methods. The first, the Quasi-Newton method, computes a descent direction and the corresponding optimal step length to compute a new iteration. This method does not need to know the gradient vector of $J(v)$ from (2) explicitly but instead computes the numerical gradient and the corresponding Hessian matrix using the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update formula. The second, the trust-region method, is based on approximating the objective function using the quadratic model function with the appropriate trust-region radius. Contrary to the Quasi-Newton method, the trust region also requires knowledge

of the discrete gradient of $J(v)$. The gradient can be explicitly derived or evaluated numerically using the central difference scheme. Additionally, the Hessian sparsity can be specified and follows directly from the FEM discretization (see Fig. 1).

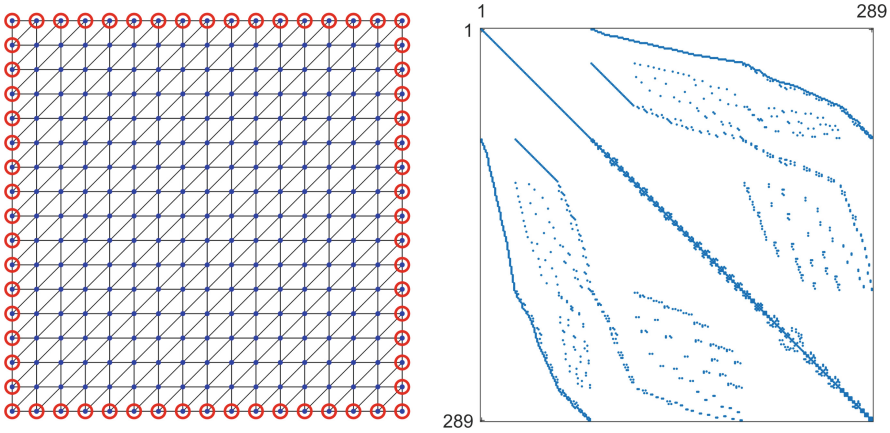

Fig. 1. Discretization of a rectangular domain (left) including Dirichlet boundary nodes (red) and the corresponding Hessian sparsity (right). (Color figure online)

If the Hessian matrix is sparse (i.e. for the Ginzburg-Landau problem), the trust-region method is much more time-efficient than the Quasi-Newton method. In contrast, if the Hessian matrix has many non-zero elements (i.e. for the topology optimization), the trust-region method can be significantly slower.

2.1 Solution Algorithm

consists for $d = 2$ of several typical steps:

- triangulation of the domain Ω into triangles and assembly of structures '**mesh**' and '**patches**' [6].
- defining the corresponding discrete energy functional $J(v)$ from (2) as a sum of the energy contributions of every element.
- if the trust region method is chosen, the '**patches**' structure is used to define a function that represents the gradient of the discrete energy functional. This gradient can be evaluated either exactly (in the case that the partial derivatives of $J(v)$ from (2) can be derived explicitly) or numerically using the central difference scheme. The Hessian sparsity follows automatically from the FEM discretization.
- the choice of a stopping criterion of the minimization process.

3 Ginzburg-Landau Problem

Superconductors are certain metals and alloys that, when cooled below a critical (typically very low) temperature, lose their resistivity, allowing permanent currents to circulate without loss of energy. Superconductivity was discovered by Ohnes in 1911. As a phenomenological description of this phenomenon, Ginzburg and Landau introduced in 1950 the Ginzburg-Landau model, which has been proven to effectively predict the behavior of superconductors and that was subsequently justified as a limit of the Bardeen-Cooper-Schrieffer (BCS) quantum theory. It is a model of great importance in physics, and Nobel prizes have been awarded for it to Abrikosov, Ginzburg, and Landau in 2003. For more details on the physical and mathematical description of the models studied, see [3,4].

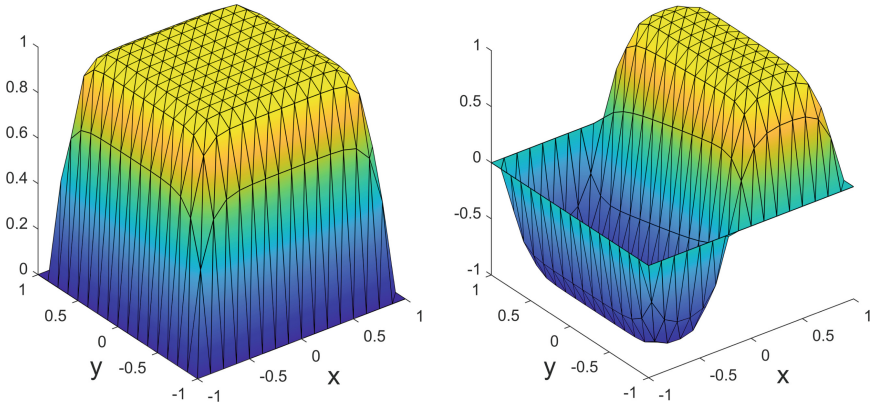

Fig. 2. Two numerical solutions of G-L problem on a rectangular domain Ω for $\varepsilon = 10^{-2}$ and zero Dirichlet boundary conditions on the boundary $\partial\Omega$. We can identify flat regions, where the solutions satisfy $u = 1$ or $u = -1$. The computational mesh consists of 512 elements and 289 nodes including 64 Dirichlet boundary nodes. The mesh is shown independently in Fig. 1.

Leaving out the dependence on the magnetic field, we consider the simpler Ginzburg-Landau minimization problem [1] for a scalar test function $v \in V$, and the minimizer $u \in V$ means the order parameter that indicates the local state of the material (normal or superconducting). The energy functional reads

$$J(v) = \int_{\Omega} \left(\frac{\varepsilon}{2} \|\nabla v\|^2 + \frac{1}{4}(v^2 - 1)^2 \right) \mathrm{d}x, \tag{4}$$

where $\Omega \subset \mathbb{R}^d$ is a given domain, ε a given small positive parameter and

$$\nabla v = \left(\frac{\partial v}{\partial x_1}, \ldots, \frac{\partial v}{\partial x_d} \right)$$

denotes the vector gradient in the dimension d and $\|\cdot\|$ its euclidean norm. The space V above contains testing functions $v : \Omega \to \mathbb{R}$ having the first (generalized) derivatives and satisfying the Dirichlet boundary condition

$$v = 0 \qquad \text{on} \quad \partial\Omega\,. \tag{5}$$

It is possible to show that the structure of (4) allows for more minimizers that satisfy the corresponding Euler-Lagrange equation formulated as the boundary value problem for the nonlinear partial differential equation

$$\begin{aligned} \varepsilon\Delta u = u^3 - u \qquad & \text{in} \quad \Omega\,, \\ u = 0 \qquad & \text{on} \quad \partial\Omega \end{aligned} \tag{6}$$

or its weak form

$$\int_\Omega \frac{\varepsilon}{2}\nabla u \cdot \nabla v \,\mathrm{d}x - \int_\Omega (u - u^3)v \,\mathrm{d}x = 0 \qquad \text{for all } v \in V. \tag{7}$$

Figure 2 shows two different solutions generated by two different initial approximations, and Table 1 the performance of the trust-region method with the specified Hessian sparsity pattern for different levels of uniform refinements. The stopping criteria related to the first-order optimality, tolerance on the argument, and tolerance on the function equal to 10^{-6} are considered.

Table 1. Performance of G-L minimizations for $\varepsilon = 10^{-2}$.

			exact gradient			numerical gradient		
level	$\lvert\mathcal{T}\rvert$	dofs	time [s]	iters	$J(\mathbf{u})$	time [s]	iters	$J(\mathbf{u})$
2	128	49	0.39	8	0.3867	0.08	8	0.3867
3	512	225	0.06	6	0.3547	0.05	6	0.3547
4	2048	961	0.13	7	0.3480	0.12	7	0.3480
5	8192	3969	0.28	6	0.3462	0.34	6	0.3462
6	32768	16129	1.11	7	0.3458	1.26	7	0.3458
7	131072	65025	6.63	8	0.3457	7.17	8	0.3457
8	524288	261121	56.97	8	0.3456	64.98	10	0.3456

Note that the original nonvectorized implementation [1] of the Newton-Ralphson solver based on the weak form (7) requires, for example:

level 6 - 4.33s and 6 iterations,
level 7 - 54.46s and 6 iterations,
level 8 - 936.47s and 6 iterations.

Our implementation only needs a slightly higher number of iterations. The underlying MATLAB code is heavily vectorized and therefore faster. The part most computationally consuming is the function '**energy**' that evaluates the corresponding energy $J(v)$ for a given vector $\mathbf{v} \in \mathbb{R}^{|\mathcal{N}|}$ together with the numerical gradient $\nabla J(v)$. The MATLAB profiler shown in Fig. 3 outputs the number of calls and the total evaluation time of every code line related to the function evaluating (4) and its gradient.

Time	Calls	Line	
		77	function e = energy(v,eps)
< 0.001	340	78	if nargin==1 % global energy only
4.181	170	79	v_elems = v(mesh.elems2nodes); % values on elements
2.609	170	80	F_elems = evaluate_F_2D_scalar(mesh,v_elems); % all gradients
2.176	170	81	densities_elems = densities(v_elems,F_elems);
0.361	170	82	e = sum(mesh.areas.*densities_elems);
< 0.001	170	83	else % local gradient energies using the epsilon perturbation vector
0.006	170	84	e = zeros(numel(dofsMinim),size(eps,2));
11.220	170	85	v_patches = v(patches.elems2nodes);
< 0.001	170	86	for comp=1:size(eps,2)
2.772	340	87	v_patches_eps = v_patches + eps(comp)*patches.logical;
15.213	340	88	F_patches = evaluate_F_2D_scalar(patches,v_patches_eps);
12.610	340	89	densities_patches = densities(v_patches_eps,F_patches);
0.995	340	90	e_patches = patches.areas.*densities_patches;
2.489	340	91	cumsum_all_e = cumsum(e_patches);
2.629	340	92	e(:,comp) = [cumsum_all_e(indx(1)); diff(cumsum_all_e(indx))];
0.001	340	93	end
< 0.001	340	94	end
3.326	340	95	end

Fig. 3. MATLAB profiler for level 8 refinement.

The energy evaluation consists of the following steps:

- (line 79) assembly of the matrix '**v_elems**' of nodal values of **v** on the elements.
- (line 80) evaluation of the cell '**F_elems**' of gradients of **v** on elements stored.
- (line 81) evaluation of the vector '**densities_elems**' of energy densities in the elements. This is done by the function '**densities**' processing both the gradient and the reaction terms of (4). Gaussian quadrature is applied for the evaluation of the reaction term.
- (line 82) the total energy '**e**' is given by the sum of the energy contributions of every element multiplied by their areas.

The energy gradient evaluation procedure is similar, but includes the for loop over two components of the input vector '**eps**', which are $-\epsilon$ and $+\epsilon$. Therefore, the numbers of calls of lines 87–92 are twice as high.

- (line 85) assembly of the matrix '**v_patches**' of the nodal values of **v** on patches.

– (line 87) perturbation of the nodal values by the corresponding component of '**eps**' resulting in a vector '**v_patches_eps**'.
– (line 88) evaluation of the cell '**F_patches**' of gradients of '**v_patches_eps**' on patches.
– (lines 89–90) evaluation of the vector '**e_patches**' of energy densities on patches.
– (line 91) assembly of vector '**cumsum_all_e**' containing the cumulative sums of vector '**e_patches**'.
– (line 92) evaluating a vector '**e**' of differences of cumulative sums using the '**indx**' vector (described in detail in [6]).

This implementation facilitates an easy extension to higher-order difference schemes.

4 Topology Optimization in 2D

Structural topology optimization (TO) is a numerical method that aims, through a density function, to optimally distribute a limited amount of material within a volume, representing the initial geometry of a body that undergoes specific loads and displacement boundary conditions. Among the approaches to solving TO problems ([2]), we focus on the so-called phase field approach. We consider a domain $\Omega \in \mathbb{R}^d$ where the material is distributed using a scalar phase field variable ϕ, representing a density fraction of the material, hence $\phi \in [0, 1]$ with

$\phi \equiv 0$ corresponding to the void (no material),
$\phi \equiv 1$ to the bulk material.

Adopting a linear elastic model, the state equations are of the form

$$\begin{aligned} \operatorname{div}(\sigma) &= 0 \quad \text{in} \quad \Omega\,, \\ u &= 0 \quad \text{on} \quad \Gamma_D\,, \\ \sigma \cdot n &= g \quad \text{on} \quad \Gamma_N. \end{aligned} \tag{8}$$

Here, we have the stress tensor $\sigma = \sigma(\phi)$, the displacement vector u and with zero value (in sense of traces) at the Dirichlet boundary Γ_D, the external load g vector at the Neumann boundary Γ_N with the normal unit vector n.

The stress tensor reads

$$\sigma(\phi) = \mathbb{C}(\phi) : \varepsilon(u)$$

with the fourth-order linear material tensor $\mathbb{C} = \mathbb{C}(\phi)$ and the symmetric strain $\varepsilon(u)$ is defined as

$$\varepsilon(u) = (\nabla u + \nabla u^T)/2.$$

The symbol ':' denotes the contraction of two tensors in the form that yields $\sigma_{ij} = \mathbb{C}_{ijkl}\varepsilon_{kl}$, where the Einstein summation is applied. We consider the void as a very soft material, adopting the following equation for $\mathbb{C}$:

$$\mathbb{C}(\phi) = \mathbb{C}_{bulk}\phi^p + \mathbb{C}_{void}(1-\phi)^p.$$

In practical calculations, we set $p = 3$, and $\mathbb{C}_{void} = 10^{-2}\,\mathbb{C}_{bulk}$ and the matrix $\mathbb{C}_{bulk}$ is specified by two material parameters (the first Lamé parameter λ and the shear modulus μ). The weak form of the linear elastic problem (8) can be written as

$$\int_{\Omega} \sigma(\phi) : \varepsilon(v)\,\mathrm{d}x = \int_{\Gamma_N} g \cdot v\,\mathrm{d}x \tag{9}$$

for any test displacement field v and $\sigma(\phi) = \mathbb{C}(\phi) : \varepsilon(v)$. The goal is to minimize the compliance of a given structure by optimally distributing a limited amount of material. For this purpose, we introduce an objective functional $J(\phi, u(\phi))$ defined as:

$$J\big(\phi, u(\phi)\big) = \int_{\Gamma_N} g \cdot u(\phi)\,\mathrm{d}x + \kappa \int_{\Omega} \Big[\frac{\gamma}{2}\|\nabla\phi\|^2 + \frac{1}{\gamma}\psi_0(\phi)\Big]\,\mathrm{d}x\,, \tag{10}$$

where for a given ϕ the corresponding displacement $u(\phi)$ is given as the solution of (9). The first integral represents a measure of the compliance of the global system, the term $\frac{\gamma}{2}\|\phi\|^2$ penalizes nonconstant values of ϕ, while $\frac{1}{\gamma}\psi_0(\phi)$, where

$$\psi_0(\phi) = (\phi - \phi^2)^2\,,$$

represents the double-well potential function penalizing values of ϕ different from 0 and 1. The parameter γ is usually set between 10^{-4} and 10^{-2} (for a finer mesh, the lower value provides better results). Minimization of the functional (10) is imposed under the assumption of distributing a limited constant quantity of material within the domain; therefore, we introduce the constraint

$$\int_{\Omega} \phi\,\mathrm{d}x = m|\Omega|$$

with $0 < m \leq 1$ representing a volume fraction of the target domain.

Figures 4, 5 and 6 illustrate topology optimization solutions for different domains and the corresponding Dirichlet and Neumann boundary conditions. For the sake of clarity, the computational meshes on the left side are depicted for lower levels of refinement. Red circles indicate the nodes corresponding to Γ_D and green circles indicate the nodes that belong to Γ_N.

Three models are given by the following parameters.

The first model:

- a rectangular domain $(0, 0.02) \times (0, 0.01)$,
- $\gamma = 10^{-4}$,
- the left side of the boundary is fixed,
- a constant traction force $g = 5 \cdot 10^6$ acts on the bottom side of the boundary from $x = 0.016$ to $x = 0.02$ downwards.

The second model:

- an L-shaped domain given by the union of rectangles $(0, 0.06) \times (0, 0.06)$, $(0, 0.06) \times (0.06, 0.2)$ and $(0.06, 0.2) \times (0, 0.06)$,

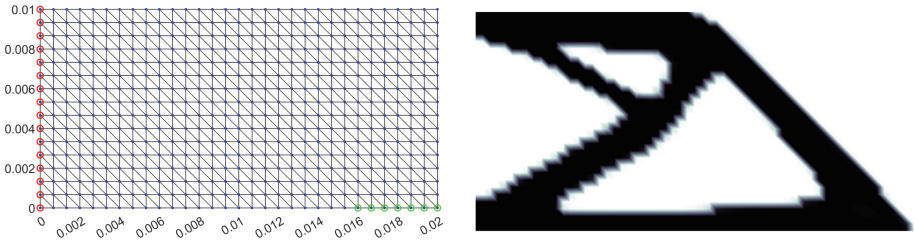

Fig. 4. The first model: triangulation of the rectangle domain (left) with 3600 elements and the solution (right).

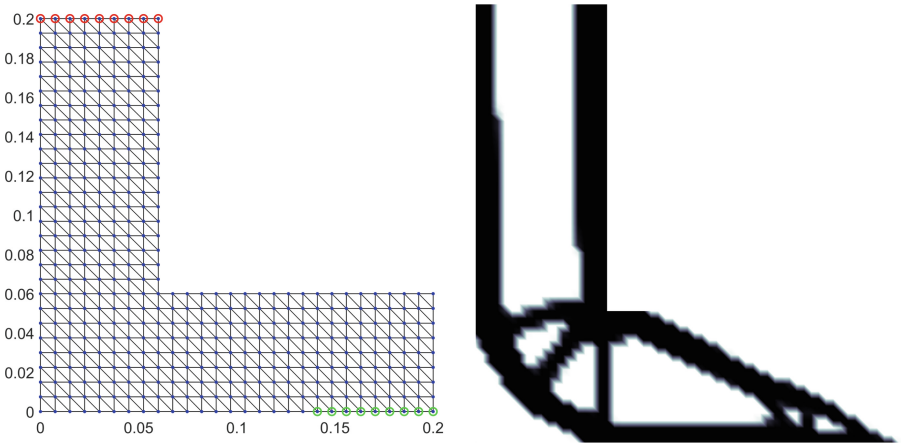

Fig. 5. The second model: triangulation of the L-shaped domain (left) with 3672 elements and the solution (right).

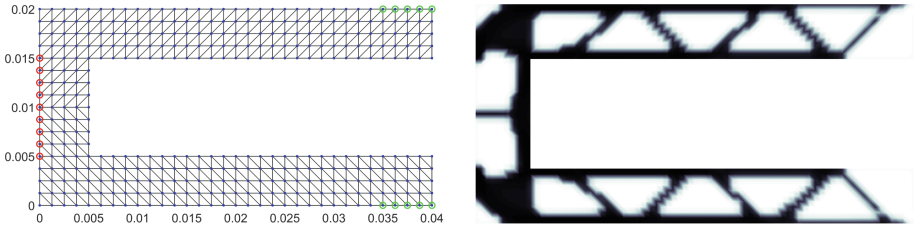

Fig. 6. The third model: triangulation of the pincer domain (left) with 3600 elements and the solution (right).

- $\gamma = 10^{-3}$,
- the top side of the boundary is fixed,
- a constant traction force $g = 10^6$ acts on the bottom side of the boundary from $x = 0.14$ to $x = 0.2$ downwards.

The third model:

- a pincer domain given by the union of rectangles $(0, 0.005) \times (0, 0.02)$, $(0.005, 0.04) \times (0, 0.005)$ and $(0.005, 0.04) \times (0.015, 0.02)$,
- $\gamma = 10^{-3}$,
- the left side of the boundary from $y = 0.005$ to $y = 0.015$ is fixed,
- a constant traction force $g = 2 \cdot 10^5$ acts on the top (upwards) and the bottom (downwards) sides of the boundary from $x = 0.035$ to $x = 0.04$.

The following parameters are the same for all models:

- $E = 12.5 \cdot 10^8$ (Young modulus), $\nu = 0.25$ (Poisson ratio),
- $\kappa = 100$, $m = 0.4$.

Contrary to the Ginzburg-Landau problem, a small change of ϕ in a single node affects the corresponding displacement $u(\phi)$ given by (9) throughout the domain. In this case, the corresponding Hessian matrix is full, and therefore the trust-region method is ineffective, and the quasi-Newton method is used instead. Table 2 shows the performance of the quasi-Newton method for different levels of uniform mesh refinements of the rectangular domain corresponding to the first model. The same stopping criteria as for the GL-problem equal to 10^{-4} are considered.

Table 2. Performance of TopOpt minimizations with domain and parameters from the first model.

			quasi-Newton		
level	$\lvert\mathcal{T}\rvert$	dofs	time [s]	iters	$J(\mathbf{u})$
1.0	100	120	2.11	50	28.0815
1.5	240	270	9.52	44	24.2585
2.0	400	440	25.88	52	23.5845
2.5	900	960	194.13	82	21.4105
3.0	1600	1680	1079.45	125	21.0503
3.5	3600	3720	15737.81	323	20.3108

Similarly to 3, the MATLAB profiler shown in Fig. 7 outputs the number of calls and the total evaluation time of code lines related to the function evaluating (10). The energy evaluation consists of the following steps:

- (line 48) assembly of a vector '**z_elems**' containing averaged values of ϕ on the elements.
- (lines 49–50) evaluating vectors '**mu_elems**' and '**bulk_elems**' that store the values of shear and bulk modulus, respectively, on the elements.
- (lines 52–55) an update of the elastic stiffness matrix [7].
- (line 57) restriction of the stiffness matrix on free degrees.
- (line 58) a new displacement field in free degrees is evaluated based on (9).

Time	Calls	Line	
		46	function e = energy(z)
		47	% averaged values of 'z' on elements
7.444	71549	48	z_elems = evaluate_average(mesh.elems2nodes,z);
15.264	71549	49	mu_elems = mu*Cz(z_elems); % values of 'mu' on elements
12.685	71549	50	bulk_elems = bulk*Cz(z_elems); % values of 'bulk' on elements
		51	
14.926	71549	52	Elast = 2*Dev(:)*mu_elems' + Vol(:)*bulk_elems';
5.114	71549	53	vD = Elast.*(ones(9,1)*WEIGHT);
51.392	71549	54	D = sparse(iD,jD,vD);
112.659	71549	55	K = B'*D*B; % stiffness matrix
		56	
11.882	71549	57	K = K(dofsMinim,dofsMinim); % stiffness matrix restricted on free degrees
1118.380	71549	58	u(dofsMinim) = K\f; % the final displacement in free degrees
		59	
0.949	71549	60	e1 = u'*b/2; % elastic part
1.466	71549	61	e2 = kappa*eps/2*(z'*Kstiff*z); % gradient part
4.751	71549	62	e3 = kappa/eps*sum(psi_0(z).*nodes2areas); % double-well potential part
		63	
0.012	71549	64	e = e1 +e2 +e3; % the final energy
2.932	71549	65	end

Fig. 7. MATLAB profiler for level 3 refinement.

– (lines 60–62) evaluating the first (elastic), second (gradient) and third (double-well potential) part of (10).
– (line 64) the final energy given as a sum of its three components.

The profiler shows that the main cost of the evaluation lies in the reassembling of the elastic stiffness matrix and the solutions of the linear systems of equations in each energy evaluation.

5 Conclusions and outlooks

We introduced a Ginzburg-Landau and topology optimization problem that appears in physics and implemented them using the concept of our codes from [5] based on a minimization of energy functionals.

A comparison with the original implementation of Ginzburg-Landau [1] based on the Newton-Raphson method demonstrates the effectiveness of our vectorization concepts, leading to significantly better evaluation times, but higher memory cost. It shows that the trust region method requires only a slightly higher number of iterations. It would be interesting to apply our vectorization concepts to the assembly of the Hessian matrix in the Newton-Raphson method.

A simple implementation of topology optimization of an elastic medium using the Quasi-Newton method has proved feasible. The elasticity stiffness matrix needs to be assembled and the resulting linear system of equations solved in every energy iteration. The original assembly code of [7] is effectively split using precomputed structures that do not change during the minimization process. Practically, it still takes the majority of the evaluation time. Although this approach is highly inefficient from an optimization point of view, it should allow for

a simple extension to more complicated problems, such as topology optimization of elastoplastic materials, where the elastoplasticity solver of [7] replaces the original elasticity solver. To reduce the number of evaluations, we plan to implement schemes of gradient flow type [8].

Acknowledgment. We thank Prof. Ulisse Stefanelli and Dr. Stefano Almi (University of Vienna) for inspiring discussions on topology optimization models and their numerical implementation.

References

1. Alberty, J., Carstensen, C., Funken, S.A.: Remarks around 50 lines in matlab: short finite element implementation. Numer. Algorithms **20**, 117–137 (1999)
2. Bendsoe, M.P., Sigmund, O.: Topology Optimization, Springer, Berlin (2004). https://doi.org/10.1007/978-3-662-05086-6
3. Romá, C.: Analysis of singularities in elliptic equations: the Ginzburg-Landau model of superconductivity, the Lin-Ni-Takagi problem, the Keller-Segel model of chemotaxis, and conformal geometry. Université Pierre et Marie Curie - Paris VI, Mathematical Physics (2017)
4. Bethuel, F., Brezis, H., Hélein, F.: Ginzburg-Landau vortices, Birkhäuser (2017)
5. Matonoha, C., Moskovka, A., Valdman, J.: Minimization of p-Laplacian via the finite element method in MATLAB. In: Lirkov, I., Margenov, S. (eds.) LSSC 2021. LNCS, vol. 13127, pp. 533–540. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-97549-4_61
6. Moskovka, A., Valdman, J.: Fast MATLAB evaluation of nonlinear energies using FEM in 2D and 3D: nodal elements. Appl. Math. Comput. **424**, 127048 (2022)
7. Čermák, M., Sysala, S., Valdman, J.: Efficient and flexible MATLAB implementation of 2D and 3D elastoplastic problems. Appl. Math. Comput. **355**, 595–614 (2019)
8. Carraturo, M., Rocca, E., Bonetti, E., Hömberg, D., Reali, A., Auricchio, F.: Graded-material design based on phase-field and topology optimization. Comput. Mech. **2019**(64), 1589–1600 (2019)
9. Ciarlet, P.G.: The Finite Element Method for Elliptic Problems. SIAM, Philadelphia (2002)
10. MATLAB documentation on minimization with gradient and Hessian sparsity pattern. https://www.mathworks.com/help/optim/ug/minimization-with-gradient-and-hessian-sparsity-pattern.html
11. Drozdenko, D., Knapek, M., Kružík, M., Máthis, K., Švadlenka, K., Valdman, J.: Elastoplastic deformations of layered structures. Milan J. Math. **90**, 691–706 (2022)
12. Frost, M., Valdman, J.: Vectorized MATLAB implementation of the incremental minimization principle for rate-independent dissipative solids using FEM: a constitutive model of shape memory alloys. Mathematics **10**(23), 4412 (2022)