Journal of Computational and Applied Mathematics 309 (2017) 122–131

Contents lists available at ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Quasi-Newton minimization for the $p(x)$-Laplacian problem

M. Caliari [*], S. Zucher

Department of Computer Science, University of Verona, Strada Le Grazie 15, 37134 Verona, Italy

ARTICLE INFO

Article history:
Received 1 December 2015
Received in revised form 4 May 2016

Keywords:
$p(x)$-Laplacian
Degenerate quasi-linear elliptic problem
Quasi-Newton minimization

ABSTRACT

We propose a quasi-Newton minimization approach for the solution of the $p(x)$-Laplacian elliptic problem, $x \in \Omega \subset \mathbb{R}^m$. This method outperforms those existing for the $p(x)$-variable case, which are based on general purpose minimizers such as BFGS. Moreover, when compared to *ad hoc* techniques available in literature for the p-constant case, and usually referred to as "mesh independent", the present method turns out to be generally superior thanks to better descent directions given by the quadratic model.

1. Introduction

We consider the $p(x)$-Laplacian elliptic problem

$$\begin{cases} -\mathrm{div}(|\nabla u(x)|^{p(x)-2}\,\nabla u(x)) = f(x) & x \in \Omega \subset \mathbb{R}^m, \\ u(x) = 0 & x \in \partial\Omega \end{cases} \tag{1}$$

where Ω is an open bounded subset of $\mathbb{R}^m$ with $\partial\Omega$ Lipschitz continuous, $p \in \mathcal{P}^{\log}$, that is p is a measurable function, $p\colon \Omega \to [1, +\infty]$ and $1/p$ is globally log-Hölder continuous. Moreover, we assume $1 < p_{\min} \le p(x) \le p_{\max} < \infty$, $f \in L^{p'(x)}(\Omega)$ (where $p'(x)$ denotes the dual variable exponent of $p(x)$) and $u \in V = W_0^{1,p(x)}(\Omega)$. Since $p(x)$ is bounded, we may see the space $W_0^{1,p(x)}(\Omega)$ as the space of functions in $W^{1,p(x)}(\Omega)$ with null trace on $\partial\Omega$. The trace operator can be defined on $W^{1,p(x)}(\Omega)$ in such a way that, as usual, if $u \in W^{1,p(x)}(\Omega) \cap \mathcal{C}(\overline{\Omega})$, then its trace coincides with $u|_{\partial\Omega}$. We refer to [1] for a general introduction to variable exponent Sobolev spaces. This model occurs in many applications, such as image processing [2,3] and electrorheological fluids [4–6], in which $p(x)$ may assume values close to the extreme ones [7–9]. Hereafter we leave the explicit dependence on $x \in \Omega \subset \mathbb{R}^m$ only for the exponent $p(x)$ and all integrals are intended over the domain Ω. The $p(x)$-Laplacian problem (1) admits a unique [10] weak solution $\underline{u}$ satisfying

$$\underline{u} = \arg\min_{v \in V} J(v)$$

where

$$J(u) = \int \frac{|\nabla u|^{p(x)}}{p(x)} - \int fu \tag{2}$$

or, equivalently,

$$J'(\underline{u})v = 0, \quad \forall v \in V \tag{3}$$

[*] Corresponding author.
E-mail addresses: marco.caliari@univr.it (M. Caliari), simone.zuccher@univr.it (S. Zuccher).

http://dx.doi.org/10.1016/j.cam.2016.06.026
0377-0427/© 2016 Elsevier B.V. All rights reserved.

where

$$J'(\underline{u})v = \int |\nabla \underline{u}|^{p(x)-2}\nabla \underline{u} \cdot \nabla v - \int f v. \tag{4}$$

A common way [11–14] to tackle the problem is the direct minimization, in a suitable finite dimensional subspace of V, of the functional J in Eq. (2), rather than solving the nonlinear equation (3) [15]. However, to our knowledge, *ad hoc* minimization algorithms were developed only for the p-constant case [13–15], whereas only general purpose methods such as the quasi-Newton method BFGS (Broyden–Fletcher–Goldfarb–Shanno) have been used for the $p(x)$-variable case [12].

In this work we minimize $J(u)$ employing a new quadratic model which makes use of the exact second differential $J''(u)$, only slightly regularized in order to handle possible analytic or numerical degeneracy when $|\nabla u|$ is small and $p(x)$ is close to the extreme values $p_{\min}$ or $p_{\max}$. The result is an efficient and robust algorithm converging faster than those available in literature, both for the p-constant case and the $p(x)$-variable one.

2. Minimization problem

We minimize $J(u)$ in a suitable finite element subspace of V and we call $\underline{u}^h$ the solution

$$\underline{u}^h = \arg\min_{v^h \in V_0^h} J(v^h) \Leftrightarrow J'(\underline{u}^h)v^h = 0 \quad \forall v^h \in V_0^h.$$

Given a regular triangulation of a polygonal approximation Ω_h of the domain, we select the subspace $V_0^h \subset V$ of continuous piecewise linear functions which are zero at the boundaries of Ω_h. Since for $p \neq 2$ problem (1) is degenerate quasi-linear elliptic, its solution has a limited regularity (see, for instance, [16]) and therefore higher-order finite element approximations do not worth (see Ref. [17]). For the variable exponent case, $p(x)$ is approximated by continuous piecewise linear functions as well, even if a local approximation by constant functions is possible (see Ref. [10,18]). Given the approximation $u^n \in V_0^h$ of the solution $\underline{u}^h$ at iteration n, we look for a direction $d^n \in V_0^h$ such that

$$J(u^n + \alpha_n d^n) < J(u^n).$$

The descent direction d^n is called *steepest descent* direction if

$$J'(u^n)d^n = -\left\|J'(u^n)\right\|_* \left\|d^n\right\|$$

where $\|\cdot\|$ is a suitable norm in V_0^h and $\|\cdot\|_*$ its dual norm. The idea (see Ref. [13,14]) is to find d^n as the solution of

$$d^n : b_n(d^n, v) = -J'(u^n)v, \quad \forall v \in V_0^h$$

where $b_n(\cdot,\cdot)$ is a suitable bilinear form depending on iteration n. The choice of b_n characterizes the minimization method.

The extension to non-homogeneous Dirichlet boundary conditions is straightforward. The solution $\underline{u}$ belongs to the variable exponent Sobolev space $W_g^{1,p(x)} = \{v \in W^{1,p(x)} : v = g \text{ on } \partial\Omega\}$ and its piecewise approximation must be in the space $V_{g_h}^h$, that is the space of continuous piecewise linear functions whose value of $\partial\Omega_h$ is g_h, where g_h is chosen to approximate the Dirichlet boundary data. The search directions are still in the space V_0^h.

2.1. Gradient-based directions

The choice in Ref. [13], for the p-constant case, is $d^n = w^n$, where

$$b_n(w^n, v) = \begin{cases} \displaystyle\int (\varepsilon + \left|\nabla u^n\right|^{p-2})\nabla w^n \cdot \nabla v, & p > 2 \\ \displaystyle\int (\varepsilon + \left|\nabla u^n\right|)^{p-2}\nabla w^n \cdot \nabla v, & p < 2. \end{cases} \tag{5}$$

The bilinear form $b_n(\cdot,\cdot)$ corresponds to a simple linearization of $J'(u^n)v$. The parameter ε is introduced in order to handle possible analytic or numerical degeneracy where $|\nabla u^n|$ is small. In fact, for $p \gg 2$ the term $|\nabla u^n|^{p-2}$ may underflow even if $|\nabla u^n| > 0$. On the other hand, for $p < 2$ the same term may overflow. We notice that the parameter ε is introduced only for finding the descent direction and not for regularizing the original $p(x)$-Laplacian functional J. With the above choice, the authors in Ref. [13] proved a convergence result ($J(u^n) \to J(u)$) only for the case $p > 2$. Their complicated proof is hardly extendible to the case $p < 2$ or to the general case with variable $p(x)$. The direction w^n is called in Ref. [13] *preconditioned steepest descent*. The scalar value α_n is chosen by exact line search

$$\alpha_n = \arg\min_{\alpha} J(u^n + \alpha d^n). \tag{6}$$

In Ref. [14] w^n is computed for all $1 < p < +\infty$ using the first definition in (5). The descent direction is then computed by

$$d^n = w^n + \beta_n d^{n-1}$$

where

$$\beta_n = \max\left\{0, \min\left\{\frac{{w^n}^T w^n}{{w^{n-1}}^T w^{n-1}}, \frac{(w^n - w^{n-1})^T w^n}{{w^{n-1}}^T w^{n-1}}\right\}\right\}.$$

The definition of β_n corresponds to an hybridization of the popular Fletcher–Reeves and Polak–Ribière–Polyak parameters for the nonlinear conjugate gradient method (see also Ref. [19]). Direction d^n is called in Ref. [14] *hybrid conjugate gradient*. The scalar α_n is chosen as in Eq. (6).

2.2. Quasi-Newton direction

Our proposal for the direction d^n in the general case $p(x)$ is the following. We start with the second differential of J, which is well defined for $p(x) \geq 2$

$$J''(u)(v, w) = \int (p(x) - 2)|\nabla u|^{p(x)-4}(\nabla u \cdot \nabla w)(\nabla u \cdot \nabla v) + \int |\nabla u|^{p(x)-2} \nabla w \cdot \nabla v \tag{7a}$$

$$= \int |\nabla u|^{p(x)-2}\left((p(x) - 2)\frac{\nabla u}{|\nabla u|} \cdot \nabla w \frac{\nabla u}{|\nabla u|} \cdot \nabla v + \nabla w \cdot \nabla v\right) \tag{7b}$$

$$= \int |\nabla u|^{p(x)-2}\big((p(x) - 2)\,(\mathrm{Sign}(\nabla u) \cdot \nabla w)\,(\mathrm{Sign}(\nabla u) \cdot \nabla v) + \nabla w \cdot \nabla v\big) \tag{7c}$$

where we denoted

$$\mathrm{Sign}(\nabla u(x)) = \frac{\nabla u(x)}{\sqrt{|\nabla u(x)|^2 + \big(1 - \mathrm{sign}(|\nabla u(x)|^2)\big)}} = \begin{cases} \dfrac{\nabla u(x)}{|\nabla u(x)|} & \text{if } \nabla u(x) \neq 0 \\ 0 & \text{if } \nabla u(x) = 0. \end{cases}$$

Formula (7c) is well defined and numerically computable for $p(x) \geq 2$ even if $\nabla u(x)$ is zero somewhere. On the other hand it is in general still not positive definite and not defined if $p(x) < 2$ and $\nabla u(x) = 0$ somewhere. Therefore we modify $|\nabla u|^{p(x)-2}$ in formula (7c) into

$$|\nabla u|_\varepsilon^{p(x)-2} = \varepsilon + \big(\varepsilon^2 \cdot \big(1 - \mathrm{sign}(|\nabla u|^2)\big) + |\nabla u|^2\big)^{\frac{p(x)-2}{2}}. \tag{8}$$

In this way, we accomplish both the regularizations in Eq. (5), since $p(x)$ can be simultaneously very large in some regions and small in some other regions. Hence, the regularized second differential is

$$J''_\varepsilon(u)(v, w) = \int |\nabla u|_\varepsilon^{p(x)-2}\big((p(x) - 2)\,(\mathrm{Sign}(\nabla u) \cdot \nabla w)\,(\mathrm{Sign}(\nabla u) \cdot \nabla v) + \nabla w \cdot \nabla v\big). \tag{9}$$

Therefore our descent direction is $d^n = w^n$ defined by

$$w^n : b_n(w^n, v) = -J'(u^n)v, \quad \forall v \in V_0^h \tag{10}$$

where $b_n(w^n, v) = J''_\varepsilon(u^n)(w^n, v)$. In this way, we are in practice approximating $J(u)$ by a quadratic positive definite model

$$J(u) \approx J(u^n) + J'(u^n)(u - u^n) + \frac{1}{2}J''_\varepsilon(u^n)((u - u^n), (u - u^n))$$

from which the name *quasi-Newton*. Other regularizations would be possible, by replacing $|\nabla u|^{p(x)-2}$ with

$$\varepsilon + (\varepsilon + |\nabla u|)^{p(x)-2}$$

(see, for instance, [13]) or with

$$\varepsilon + \big(\varepsilon^2 + |\nabla u|^2\big)^{\frac{p(x)-2}{2}}$$

(see, for instance, [20]), which is similar to the idea used in [21], where the problem itself, and not only the second differential, is regularized in the same way. Our choice (8) turned out to be the most effective in the numerical experiments. We notice that the choice of the gradient-based directions [13,14] can be generalized for the $p(x)$-variable case as

$$w^n : P_\varepsilon(u^n)(w^n, v) = \int \left|\nabla u^n\right|_\varepsilon^{p(x)-2} \nabla w^n \cdot \nabla v = -J'(u^n)v, \quad \forall v \in V_0^h \tag{11}$$

and

$$d^n = w^n, \qquad \text{preconditioned steepest descent [13]} \tag{12a}$$

$$d^n = w^n + \beta_n d^{n-1}, \qquad \text{hybrid conjugate gradient [14].} \tag{12b}$$

The scaling length α_n in $u^n + \alpha_n d^n$ is found by a backtracking line search method based on sufficient decrease condition (Armijo's rule). Together with reasonable assumptions on $J''_\varepsilon(u^n)$, this is enough to guarantee convergence to a stationary point of $J(u)$ (see, for instance, [22, Th. 3.2.4]).

3. Numerical examples

We implemented the quasi-Newton and, for comparison, the gradient-based minimization method [14] in FreeFem++ 3.31 [23] for the solution of two-dimensional problems, being the extension to three dimensions straightforward. The numerical solution at iteration n is denoted by

$$u^n(x,y)=\sum_j u_j^n\phi_j(x,y)$$

where $\phi_j(x,y)$ is the jth nodal finite element basis function. In the following numerical examples, the initial guess $u^0(x,y)$ is always the solution of Poisson's problem corresponding to $p = 2$. The descent directions w^n in (10) and (11) are approximated by the linear conjugate gradient method.

The exit criterion (see Ref. [24, p. 160]) is

$$\max_j\left|\frac{J'(u^n)\phi_j\circ u_j^n}{J(u^n)}\right|\le 10^{-6}$$

where $J'(u^n)\phi_j$ is defined in (4) and $\circ$ denotes Hadamard's product. For comparison, in Ref. [13,14] the initial guess is $u^0(x,y)=0$, w^n is computed by a multigrid solver, the bisection method and the golden section method are used in the line search, respectively and the exit criterion is

$$\frac{\sqrt{b_n(d^n,d^n)}}{\sqrt{b_0(d^0,d^0)}}\le 10^{-6}.$$

The solution of Eq. (10) is obtained by the default linear conjugate gradient method provided by FreeFem++, which employs the diagonal preconditioner. We also tried the matrix of entries $P_\varepsilon(u^n)(\phi_j,\phi_i)$ (see (11)) as preconditioner, since it is an approximation of the matrix $J_\varepsilon''(u^n)(\phi_j,\phi_i)$ used for the quasi-Newton direction. In this way, in general, we observed a smaller number of iterations needed for the convergence of the linear conjugate gradient method. However this approach never paid in terms of total CPU time due to the cost of the factorization of the preconditioner.

In the next tables, we report the total number of minimization iterations, the CPU time, the relative error of J and of the approximated solution in the $W^{1,p}$ norm

$$\|u\|_{W^{1,p}}=\left(\int|u|^p\right)^{\frac{1}{p}}+\left(\int|\nabla u|^p\right)^{\frac{1}{p}}$$

whenever the exact solution is known. In case of variable $p(x)$, we report either the Luxemburg norm in $L^{p(x)}$, that is

$$\|u\|_{L^{p(x)}}=\inf_{\gamma>0}\left\{\gamma:\int_\Omega\left|\frac{u(x)}{\gamma}\right|^{p(x)}\mathrm{d}x\le 1\right\}$$

or in $W^{1,p(x)}$, that is

$$\|u\|_{W^{1,p(x)}}=\|u\|_{L^{p(x)}}+\|\nabla u\|_{L^{p(x)}}.$$

We have to say that the CPU time here shown, taken on an Intel Quad Core i7-4600U 2.10 GHz, is not a reliable measure of the computational effort, since in our experiments we sometimes found significant variations in different instances of the same experiment.[1]

3.1. p-constant case

Example 1. This case is taken from Ref. [13,14], with $\Omega = B(0,1)$ and $f = 1$. The exact solution is

$$\underline{u}(x,y)=\frac{p-1}{p}\left(\frac{1}{2}\right)^{\frac{1}{p-1}}\left(1-(x^2+y^2)^{\frac{p}{2p-2}}\right)$$

and the corresponding value of $J(\underline{u})$ is

$$J(\underline{u})=\pi\left(\frac{1}{2}\right)^{\frac{1}{p-1}}\frac{(p-1)^2}{p(2-3p)}.$$

The disk $B(0,1)$ is discretized with four different meshes called D1, D2, D3 and D4 with number of vertices (dof) 1600, 6221, 24 444 and 97 451 respectively. The choice of the number of dof is almost the same as in Refs. [13,14], being not possible in

[1] All the next numerical experiments are reproducible with the code available at the web page http://profs.scienze.univr.it/caliari/software.htm.

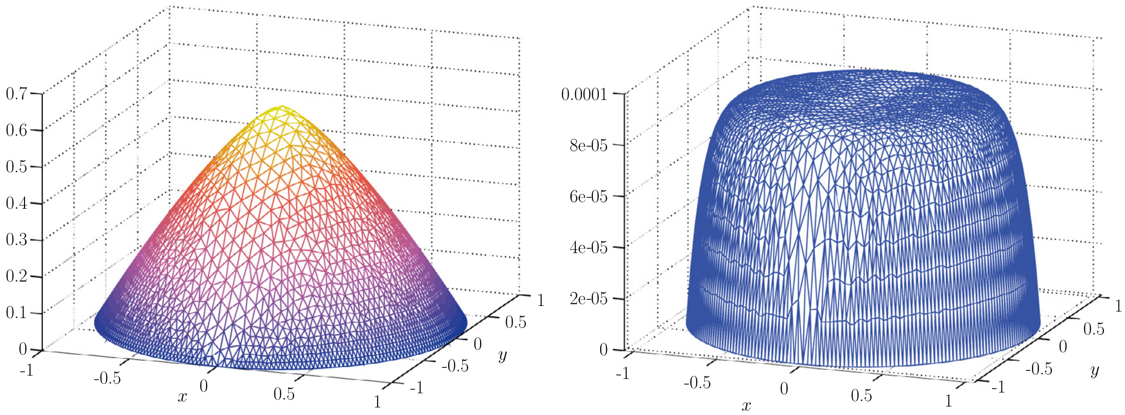

Fig. 1. Solutions of Example 1 with $p = 4$ (left) and, $p = 1.1$ (right).

Table 1
Results for Example 1, $p = 1.05$.

	D1		D2		D3		D4	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	142	264	109	256	98	180	91	257
CPU [s]	11.22	12.9	40.44	53.58	185	171.8	1072	1263
J err.	9.49e−02	9.49e−02	2.85e−02	2.85e−02	7.20e−03	7.20e−03	1.87e−03	1.87e−03
$W^{1,p}$ err.	2.09e−01	2.09e−01	9.78e−02	9.78e−02	4.55e−02	4.48e−02	2.22e−02	2.20e−02

Table 2
Results for Example 1, $p = 1.1$.

	D1		D2		D3		D4	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	75	128	75	128	75	128	75	128
CPU [s]	5.794	6.455	25.36	28.28	120.2	129.8	620.1	686.3
J err.	2.36e−02	2.36e−02	6.34e−03	6.34e−03	1.53e−03	1.53e−03	3.93e−04	3.93e−04
$W^{1,p}$ err.	9.74e−02	9.74e−02	4.68e−02	4.68e−02	2.24e−02	2.24e−02	1.11e−02	1.11e−02

FreeFem++ to match them exactly. Fig. 1 shows the numerical solutions on mesh D1 for the cases $p = 4$ and $p = 1.1$. We notice that for the relatively small value $p = 4$ the shape is very close to the cone $1 - \sqrt{x^2 + y^2}$ corresponding to the limit $p \to \infty$. On the other hand, in the limit $p \to 1^+$ the solution tends to zero with a cake like shape.

We compared the quasi-Newton method (qN) with our own implementation of the preconditioned hybrid Conjugate Gradient method (phCG) [14] with w^n given by condition (11) and the descent direction given by Eq. (12b). The latter is proven in Ref. [14] to be superior to the preconditioned descent algorithm described in Ref. [13], especially for large values of p. We selected a range of constant p values from 1.05 to 1000. The value 1.05 was chosen because in Ref. [13] the smallest successfully tried value was 1.06, whereas the value 1000 was the maximum tested in both Refs. [13,14]. We also considered the value 1.1 because used in Ref. [13] and claimed to overflow in Ref. [14].

In Tables 1–6 we collect our results. Method qN clearly outperforms phCG both in terms of number of iterations and CPU time. The only case in which qN takes few more iterations (365 versus 355) is for $p = 1000$ with mesh D4 (see Table 6). In order to reduce the number of iterations for large values of p, we tried a very simple continuation strategy (see Ref. [18]) in which we solved the minimization problem for an intermediate value $p_i = 2 + i \cdot (p - 2)/50$, $i = 1, 2, \dots, 50$ with initial condition given by the solution at the $(i - 1)$th step. For instance, for the case $p = 1000$ with mesh D4, we obtained convergence with the same errors reported in Table 6 in 244 iterations for qN and in 344 iterations for phCG. However, the description of an optimal and general continuation strategy suitable for p not necessarily constant, large and/or close to one, is beyond the scopes of the present paper. We finally notice that the iteration number weakly depends on the mesh size, especially for not too large values of p. This property was already observed in Refs. [13,14] for the gradient-based methods and therein named “mesh independence”.

Example 2. This case is taken from Ref. [13], with $\Omega = (0, 1)^2$, $f = 0$ and with non-homogeneous Dirichlet boundary conditions such that the exact solution is

$$\underline{u}(x, y) = (x^2 + y^2)^{\frac{p-2}{2p-2}}.$$

Table 3
Results for Example 1, $p = 4$.

	D1		D2		D3		D4	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	8	17	9	20	8	23	8	28
CPU [s]	0.52	0.8819	2.396	4.233	9.402	21.61	44.84	128
J err.	9.92e−04	9.92e−04	2.49e−04	2.49e−04	6.37e−05	6.37e−05	1.59e−05	1.59e−05
$W^{1,p}$ err.	2.73e−02	2.73e−02	1.47e−02	1.47e−02	9.76e−03	9.75e−03	5.06e−03	5.06e−03

Table 4
Results for Example 1, $p = 10$.

	D1		D2		D3		D4	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	10	18	10	19	12	16	13	14
CPU [s]	0.795	1.23	3.339	5.354	16.98	19.98	86.34	86.64
J err.	1.16e−03	1.16e−03	2.88e−04	2.88e−04	7.64e−05	7.64e−05	1.89e−05	1.89e−05
$W^{1,p}$ err.	1.19e−01	1.19e−01	8.17e−02	8.17e−02	1.17e−01	1.17e−01	6.20e−02	6.20e−02

Table 5
Results for Example 1, $p = 100$.

	D1		D2		D3		D4	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	31	63	37	73	43	74	50	83
CPU [s]	3.187	6.439	15.86	31.24	92.83	142.1	514.2	714.7
J err.	3.78e−03	3.78e−03	1.00e−03	1.00e−03	3.11e−04	3.11e−04	7.83e−05	7.83e−05
$W^{1,p}$ err.	3.15e−01	3.15e−01	2.24e−01	2.24e−01	5.17e−01	5.17e−01	2.67e−01	2.67e−01

Table 6
Results for Example 1, $p = 1000$.

	D1		D2		D3		D4	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	146	156	193	234	264	306	365	355
CPU [s]	18.43	31.83	105.3	188.4	651.3	1026	4629	5363
J err.	7.38e−03	7.38e−03	2.84e−03	2.84e−03	1.25e−03	1.25e−03	4.22e−04	4.22e−04
$W^{1,p}$ err.	3.28e−01	3.28e−01	2.41e−01	2.41e−01	5.89e−01	5.89e−01	2.98e−01	2.98e−01

Table 7
Results for Example 2, $p = 20$.

	$N = 27$		$N = 54$		$N = 108$		$N = 216$	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	9	29	9	34	10	29	10	29
CPU [s]	0.2985	0.7421	1.234	3.598	5.535	17.88	44.82	104.4
J err.	1.48e−01	1.48e−01	7.62e−02	7.62e−02	3.94e−02	3.94e−02	2.04e−02	2.04e−02
$W^{1,p}$ err.	1.89e−01	1.89e−01	1.83e−01	1.83e−01	1.77e−01	1.77e−01	1.71e−01	1.71e−01

The square is discretized with a uniform grid with $N+1$ points in each direction, giving a similar number of dof as in Ref. [13], where this problem was solved for $p = 20$ without the hybrid strategy for the minimization direction. This causes a number of iterations much larger then those reported in Table 7 for the phCG method. However, the newly introduced qN method is by far faster than both gradient-based methods.

Example 3. This case is taken from Ref. [17]. It is the same problem of Example 1 extended to $\Omega = (-1, 1)^2$ with the corresponding non-homogeneous Dirichlet boundary conditions. The square is discretized with a uniform grid with $N + 1$ points in each direction. In Ref. [17] the authors show the correct order of convergence as N increases, which is linear in $W^{1,p}$ norm if $p < 2$ and linear in $W^{1,1}$ norm if $p > 2$, since $\underline{u}$ is regular enough. Here we reproduce the convergence behavior for two values of p taken from Example 1 ($p = 1.1$ and $p = 4$), see Tables 8 and 9 and Fig. 2.

Example 4. In the last example for the p-constant case we consider the problem on the disk $\Omega = B(0, 1)$ with discontinuous, namely

$$f = \begin{cases} 2 & \text{if } x > 0 \\ 1 & \text{if } x \leq 0. \end{cases}$$

Table 8
Results for Example 3, $p = 1.1$.

	$N = 10$		$N = 20$		$N = 40$		$N = 80$	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	54	72	56	68	57	69	57	74
CPU [s]	0.4273	0.3793	1.762	1.44	7.342	5.996	38.59	32.73
J err.	1.94e−01	1.94e−01	5.05e−02	5.05e−02	1.28e−02	1.28e−02	3.21e−03	3.21e−03
$W^{1,p}$ err.	5.30e−01	5.30e−01	2.43e−01	2.44e−01	1.16e−01	1.16e−01	5.64e−02	5.66e−02

Table 9
Results for Example 3, $p = 4$.

	$N = 10$		$N = 20$		$N = 40$		$N = 80$	
	qN	phCG	qN	phCG	qN	phCG	qN	phCG
Iter	6	18	7	20	8	22	9	16
CPU [s]	0.05741	0.1058	0.1978	0.4484	0.8779	2.097	4.972	8.054
J err.	4.20e−02	4.20e−02	1.06e−02	1.06e−02	2.66e−03	2.66e−03	6.66e−04	6.66e−04
$W^{1,1}$ err.	7.19e−02	7.19e−02	3.48e−02	3.48e−02	1.71e−02	1.71e−02	8.47e−03	8.47e−03

Table 10
Results for Example 4.

	$p = 1.1$		$p = 4$	
	qN	phCG	qN	phCG
Iter	51	121	9	18
CPU [s]	21.0	33.2	2.9	4.7
Residual	7.33e−1	7.93e−2	3.59e−9	1.83e−6

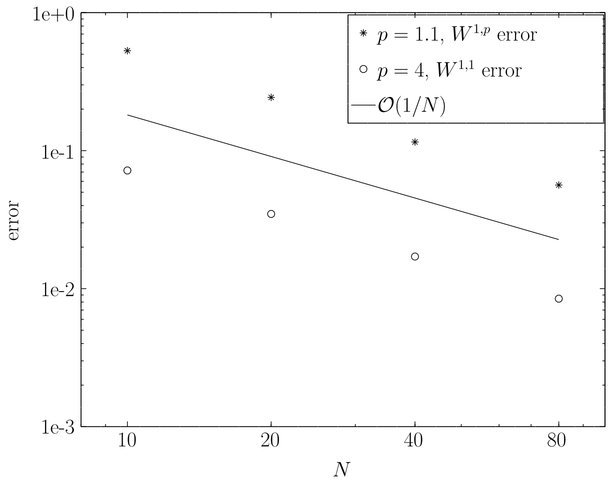

Fig. 2. Convergence order for Example 3.

The disk $B(0, 1)$ is discretized with a mesh with 6039 vertices. Since the exact solution is not available, we measured the goodness of the numerical solutions by computing the relative residual

$$\frac{\max_j \left|J'(u^n)\phi_j\right|}{\max_j \left|u_j^n\right|}. \tag{13}$$

In Table 10 we report the results corresponding to $p = 1.1$ and $p = 4$. The number of iterations and the CPU time is always smaller for the quasi-Newton method. For the case $p = 1.1$ the relative residual is smaller for the preconditioned hybrid Conjugate Gradient method. Compared with all the previous results, this could be due to the residual (13) not being a good indicator of the error for solutions with low regularity and in the case $p < 2$.

Table 11
Results for Example 5, $\epsilon = 0.04$.

	qN	phCG	BFGS
Iter	3	13	50
CPU [s]	2.9007	5.56235	121.529
$L^{p(x)}$ error	9.66e−02	9.67e−02	1.05e−01

Table 12
Results for Example 5, $\epsilon = 0.02$.

	qN	phCG	BFGS
Iter	3	5	50
CPU [s]	3.9337	4.72026	128.144
$L^{p(x)}$ error	3.92e−01	3.92e−01	4.03e−01

3.2. $p(x)$-variable case

Example 5. This case is the two-dimensional extension of the one-dimensional example reported in Ref. [12], with $\Omega = (-1, 1)^2, f = 0$ and

$$p(x,y) = \begin{cases} \dfrac{1-\epsilon}{\epsilon}|x| + 1 + \epsilon & \text{if } |x| \le \epsilon \\ 2 & \text{if } \epsilon < |x| \le 1 \end{cases}$$

where ϵ is a small parameter and $p(0, y) \to 1^+$ when $\epsilon \to 0^+$. The non-homogeneous Dirichlet boundary conditions are such that the exact solution is

$$\underline{u}(x,y) = \begin{cases} (U(|x|) - U(0)) \cdot \operatorname{sign}(x) & \text{if } |x| \le \epsilon \\ (C(|x|-1) + B) \cdot \operatorname{sign}(x) & \text{if } \epsilon < |x| \le 1 \end{cases}$$

where C is set to 1.3, and, for $0 \le x \le \epsilon$,

$$U(x) = \frac{\left(\frac{1-\epsilon}{\epsilon}x + \epsilon\right) \exp\left(\frac{\ln C}{\frac{1-\epsilon}{\epsilon}x+\epsilon}\right) - \ln C \cdot \operatorname{Ei}\left(\frac{\ln C}{\frac{1-\epsilon}{\epsilon}x+\epsilon}\right)}{\frac{1-\epsilon}{\epsilon}}$$

and $B = U(\epsilon) - U(0) + C(1-\epsilon)$. The function $\operatorname{Ei}(x)$ is the exponential integral defined as

$$\operatorname{Ei}(x) = -\int_{-x}^{\infty} \frac{e^{-t}}{t} dt.$$

For small values of ϵ the solution has a steep gradient along $x = 0$. For instance, for $\epsilon = 0.02$, $\partial_x \underline{u}(0, y) = C^{\frac{1}{\epsilon}} = 1.3^{50} \approx 5 \cdot 10^5$. As correctly observed in Ref. [12], a more efficient and accurate finite element approximation would require a discontinuous Galerkin approach. For this reason, in Tables 11 and 12, we report the Luxemburg norm in $L^{p(x)}$ space of the relative error. In fact, even if the solution is in $W^{1,p(x)}$ space, due to the steep gradient along $x = 0$, we had no reliable numerical approximation of $\|\nabla \underline{u}\|_{L^{p(x)}}$ on the uniform grid we used ($N = 101$ points in each direction).

Tables 11 and 12 show that for relatively small values of ϵ the quasi-Newton method takes only three iterations. On the other hand, if we use the BFGS method implemented in FreeFem++ (the same method was chosen by the authors in Ref. [12] for the one-dimensional example), then the maximum number of allowed iterations is reached and the CPU time is much larger. The hybrid preconditioned Conjugate Gradient method, never applied before to the $p(x)$-Laplacian, is better than BFGS but in any case worse than our quasi-Newton method.

Example 6. In this case we consider an approximation of a discontinuous piecewise constant exponent $p(x, y)$, namely

$$p(x,y) = \begin{cases} p^+ & \text{if } x < -0.01 \\ p_- + (p_- - p^+)\dfrac{x - 0.01}{0.02} & \text{if } |x| \le 0.01 \\ p_- & \text{if } x > 0.01 \end{cases}$$

in the domain $\Omega = B(0, 1)$ with right hand side $f = 1$.

The solution corresponding to $p^+ = 4$ and $p_- = 1.1$, computed on the mesh D1, is shown in Fig. 3. It resembles a merge of the two plots reported in Fig. 1 for the constant cases $p = 4$ and $p = 1.1$. We see in Table 13 that once again the quasi-Newton method clearly outperforms the others both in terms of iteration number and value of the residual. In fact, since

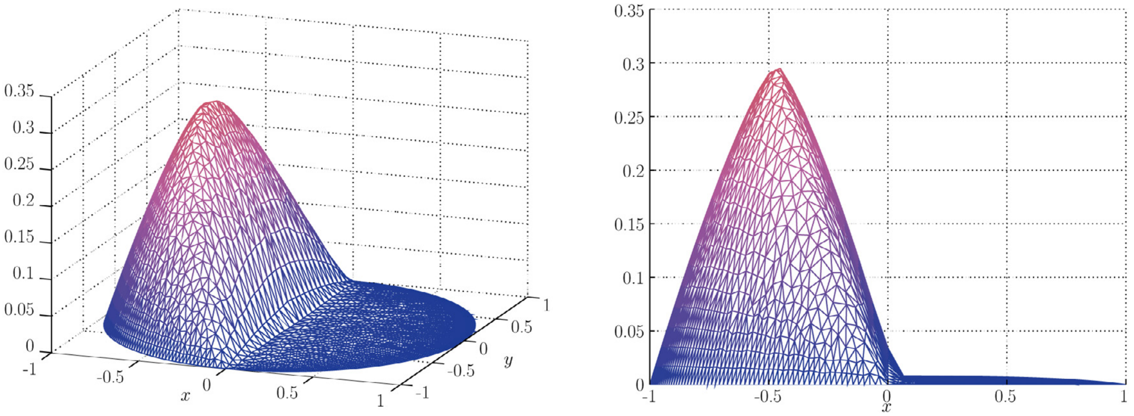

Fig. 3. Solutions of Example 6 with $p^+ = 4$ and $p_- = 1.1$ on mesh D1 in two different views. See Fig. 1 for comparison with the constant case $p(x) = p$.

Table 13
Results for Example 6, $p^+ = 4$, $p_- = 1.1$.

	qN	phCG	BFGS
Iter	18	44	50
CPU [s]	1.891	3.247	8.758
Residual	1.39e−06	2.00e−05	2.49e−01

Table 14
Results for Example 7.

	$N = 20$	$N = 40$	$N = 60$	$N = 80$	$N = 100$	$N = 120$
Iter qN	3	3	3	3	2	2
CPU [s]	0.07415	0.2949	1.382	2.563	1.686	2.87
J err.	1.11e−03	2.77e−04	1.23e−04	6.93e−05	4.44e−05	3.08e−05
$W^{1,p(x)}$ err.	2.19e−02	1.08e−02	7.16e−03	5.36e−03	4.30e−03	3.58e−03

the exact solution is not available, we measured the goodness of the numerical solutions by computing the relative residual

$$\frac{\max_j \left|J'(u^n)\phi_j\right|}{\max_j \left|u_j^n\right|}.$$

Example 7. This case is taken from Ref. [11], with $\Omega = (-1, 1)^2, f = 0$ and

$$p(x, y) = 1 + \left(\frac{1}{2}(x + y) + 2\right)^{-1}.$$

The corresponding exact solution is

$$\underline{u}(x, y) = \sqrt{2}e^2\left(e^{\frac{1}{2}(x+y)} - 1\right).$$

This $p(x)$-variable case is quite simple from the minimization point of view, even if the BFGS method took more than 50 iterations (not reported in Table 14). As shown in Ref. [11], the correct linear order in N of the error in $W^{1,p(x)}$ norm is achieved (see Fig. 4), where $N + 1$ is the number of points for each direction of the uniform grid on the square Ω.

4. Conclusions

We developed a minimization approach for the $p(x)$-Laplacian problem based on a quadratic model of the objective functional with a regularized second differential (quasi-Newton minimization). We have carried out several numerical examples in two space dimensions with constant p or variable $p(x)$, verified the results against existing analytic solutions, and found that our method outperforms those available in literature, both in number of iterations and CPU time. In particular, the quasi-Newton approach proved to be robust and efficient for values of p very small (up to 1.05) or very large (up to 1000) and for examples of $p(x)$ varying on the domain in a range between p_1 and p_2 with $1.02 \leq p_1 < 2$ and $2 \leq p_2 \leq 4$.

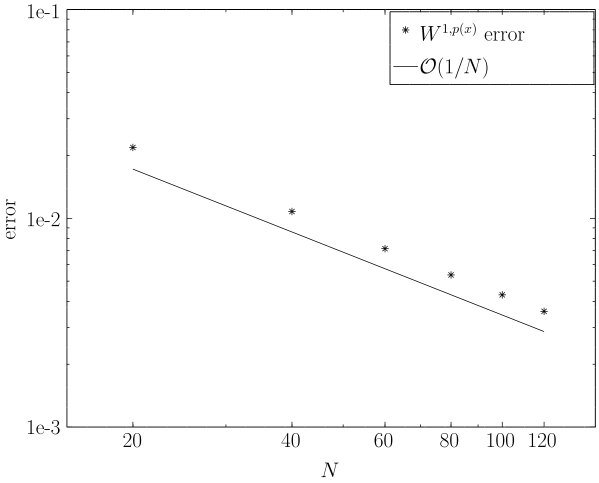

Fig. 4. Convergence order for Example 7.

References

[1] L. Diening, P. Harjulehto, P. Hästö, M. Růžička, Lebesgue and Sobolev Spaces with Variable Exponents, in: Lect. Notes Math., vol. 2017, Springer, 2011.

[2] Y. Chen, S. Levine, M. Rao, Variable exponent, linear growth functionals in image restoragion, SIAM J. Appl. Math. 66 (4) (2006) 1383–1406.

[3] P. Harjulehto, P.P. Hästö, U.V. Lê, M. Nuortio, Overview of differential equations with non-standard growth, Nonlinear Anal. 72 (2010) 4551–4574.

[4] K. Rajagopal, M. Růžička, On the modeling of electrorheological materials, Mech. Res. Commun. 23 (4) (1996) 401–407.

[5] M. Růžička, Electrorheological Fluids: Modeling and Mathematical Theory, in: Lect. Notes Math., vol. 1748, Springer-Verlag, Berlin, 2000.

[6] L.C. Berselli, D. Breit, L. Diening, Convergence analysis for a finite element approximation of a steady model for electrorheological fluids, Numer. Math. 132 (2016) 657–689.

[7] J.W. Barrett, L. Prigozhin, Bean's critical-state model as $p \to \infty$ limit of an evolutionary p-Laplacian equation, Nonlinear Anal. 42 (2000) 977–993.

[8] E.M. Bollt, R. Chartrand, S. Esedoglu, P. Schultz, K.R. Vixie, Graduated adaptive image denoising: local compromise between total variation and isotropic diffusion, Adv. Comput. Math. 31 (2009) 61–85.

[9] G. Bouchitte, G. Buttazzo, L. De Pascale, A p-Laplacian approximation for some mass optimization problems, J. Optim. Theory Appl. 118 (2003) 1–25.

[10] D. Breit, L. Diening, S. Schwarzacher, Finite element approximation of the $p(x)$-Laplacian, SIAM J. Numer. Anal. 53 (1) (2015) 551–572.

[11] L.M. Del Pezzo, S. Martínez, Order of convergence of the finite element method for the $p(x)$-Laplacian, IMA J. Numer. Anal. 35 (4) (2015) 1864–1887.

[12] L.M. Del Pezzo, A.L. Lombardi, S. Martínez, Interior penalty discontinuous Galerkin FEM for the $p(x)$-Laplacian, SIAM J. Numer. Anal. 50 (5) (2012) 2497–2521.

[13] Y.Q. Huang, R. Li, W. Liu, Preconditioned descent algorithms for p-Laplacian, J. Sci. Comput. 32 (2) (2007) 343–371.

[14] G. Zhou, Y. Huang, C. Feng, Preconditioned hybrid conjugate gradient algorithm for p-Laplacian, Int. J. Numer. Anal. Model. 2 (Suppl.) (2005) 123–130.

[15] R. Bermejo, J.-A. Infante, A multigrid algorithm for the p-Laplacian, SIAM J. Sci. Comput. 21 (5) (2000) 1774–1789.

[16] T. Iwaniec, J.J. Manfredi, Regularity of p-harmonic functions on the plane, Rev. Mat. Iberoamericana 5 (1–2) (1989) 1–19.

[17] J.W. Barrett, W.B. Liu, Finite element approximation of the p-Laplacian, Math. Comp. 61 (204) (1993) 523–537.

[18] M. Caliari, S. Zuccher, The inverse power method for the $p(x)$-Laplacian problem, J. Sci. Comput. 65 (2) (2015) 698–714.

[19] S. Babaie-Kafaki, R. Ghanbari, A hybridization of the polak–ribière–polyak and fletcher–reeves conjugate gradient methods, Numer. Algorithms 68 (2015) 481–495.

[20] R.J. Biezuner, J. Brown, G. Ercole, E.M. Martins, Computing the first Eigenpair of the p-Laplacian via inverse iteration of sublinear supersolutions, J. Sci. Comput. 52 (1) (2012) 180–201.

[21] A. Hirn, Finite element approximation of singular power-law systems, Math. Comp. 82 (283) (2013) 1247–1268.

[22] C.T. Kelley, Iterative Methods for Optimization, in: Frontiers in Applied Mathematics, vol. 18, SIAM, Philadelphia, 1999.

[23] F. Hecht, New development in FreeFem++, J. Numer. Math. 20 (3–4) (2012) 251–265.

[24] J.E. Dennis, R.B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, in: Classics in Applied Mathematics, vol. 16, SIAM, Philadelphia, PA, USA, 1996.