VERSITA

Int. J. Appl. Math. Comput. Sci., 2012, Vol. 22, No. 2, 313–326
DOI: 10.2478/v10006-012-0023-8

amcs

A MODIFIED FILTER SQP METHOD AS A TOOL FOR OPTIMAL CONTROL OF NONLINEAR SYSTEMS WITH SPATIO–TEMPORAL DYNAMICS

EWARYST RAFAJŁOWICZ, KRYSTYN STYCZEŃ, WOJCIECH RAFAJŁOWICZ

Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology, Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland
e-mail: ewaryst.rafajlowicz@pwr.wroc.pl

Our aim is to adapt Fletcher's filter approach to solve optimal control problems for systems described by nonlinear Partial Differential Equations (PDEs) with state constraints. To this end, we propose a number of modifications of the filter approach, which are well suited for our purposes. Then, we discuss possible ways of cooperation between the filter method and a PDE solver, and one of them is selected and tested.

Keywords: filter approach, nonlinear programming, optimal control, partial differential equations.

1. Introduction

In a series of papers (Chin and Fletcher, 2003; Chin, 2007; Fletcher and Leyffer, 2002; Fletcher *et al.*, 2002a; 2002b; Audet and Dennis, 2004; Fletcher, 2010; Ulbrich, 2004; Su and Yu, 2009), Fletcher and coworkers developed a filter approach to nonlinear programming problems. This approach is a breakthrough in constructing optimization solvers, because it clearly puts an emphasis on the fact that coping with nonlinear constraints is as difficult and as important as finding the minimum of an objective function. A filter contains pairs: penalty for constraints violation and a value of the objective function, which are not dominated by each other in the same sense as used in multi-objective optimization. It determines a taboo region for searching. This way of reasoning is of importance for solving optimal control problems with pointwise state constraints, because they have to be approximated by a large number of inequalities. Their number is magnified when we consider state constrained Distributed-Parameter Systems (DPSs), i.e., systems with spatial and temporal dynamics described by nonlinear PDEs.

The history of research on DPS control started about 50 years ago (see the work of Butkovskiy (1969) and the bibliography cited therein) by investigating problems for linear, time-invariant and spatially homogeneous systems. The fundamental theoretical results, obtained by the end of the twentieth century, can be found in the works of Fattorini (1999), Nettaanmaki and Tiba (1994), and Troltzsch (2010). At present, optimal control problems for nonlinear DPSs with state constraints are an area of intensive research (see the works of Burger and Pogu (1991), Aschemanna *et al.* (2010), Christofides (2001), Demetriou and Kazantzis (2004), El-Farra and Armaou (2003) as well as Skowron and Styczeń (2009) for an excerpt of different approaches to these problems). We are able to test whether a system is (non-)homogeneous or (non-)linear (Rafajłowicz, 2008; Rafajłowicz and Rafajłowicz, 2010) and to solve difficult problems of identification of systems described by PDEs (see Schittkowski, 2002; Uciński, 2005). Additionally, a substantial progress has been attained in the theory and practice of solving nonlinear optimization problems with a huge (about 200,000,000) number of nonlinear constraints (see Schittkowski, 2002). As proposed by Schittkowski (2009), this technique can be used for optimal control problems with discretized PDEs as constraints.

In this paper we discuss the filter approach as a tool for solving optimal control problems with nonlinear PDEs and state constraints. The idea of using the filter approach together with Sequential Quadratic Programming (SQP) for solving optimal control problems may seem to be obvious. For example, in the work of Betts (2010) the filter SQP is mentioned in this context as one of the approaches for globalization. We can add one more argument for using filter SQP in addition to that stated by Betts (2010), namely, constraints in optimal control problems are in practice not always hard in the sense that we can slightly move their boundaries when this leads to an essential de-

crease in the value of the objective function. In such cases the analysis of the filter contents provides necessary information on the trade off between the constraints violation and the goal function increase.

In our opinion, it is worth elaborating on the details of using the filter approach for solving optimal control problems for DPSs, because this general idea can be realized in many ways, as discussed in Section 3. Additionally, it is useful to consider some modifications of the filter approach (see Section 2) dedicated to solving such problems. We refer the reader to the work of Turco (2010) and to the bibliography cited therein for recent attempts to improve the filter approach. In Section 2 we also provide a general outline of the filter approach, which indicates a vast field of possible modifications, while a discussion on its convergence is deferred to Appendix. In Section 4 we summarize the results of testing the modified filter method when specialized to control problems for DPSs.

2. Modified filter method

In this section we describe the filter approach to Non-Linear Programming (NLP) tasks in a skeletal form, trying to avoid as many details as possible, in order to point out our modifications. We shall keep the notation typical to NLP.

2.1. Filter approach in a skeletal form.

2.1.1. Preliminary assumptions. Let $f(\mathbf{x})$ be a real valued function of d-dimensional vector $\mathbf{x}$, which is defined in a compact set $X \subset \mathbb{R}^d$. Later, we shall not invoke the set X explicitly, treating it as a large set, inside which we are searching for a minimum of f. It can be even as large as the largest hypercube which can be represented in the floating point arithmetic of our computer. In this section, f is our objective function, which is continuous in X.

Remark 1. In fact, the continuity of f is a minimal requirement, because for approximating f we need it to be once or twice continuously differentiable.

Let $\mathbf{c}(\mathbf{x})$ be an m-dimensional vector of functions, $\mathbf{c} : \mathbb{R}^d \to \mathbb{R}^m$. They impose inequality constraints of the form $\mathbf{c}(\mathbf{x}) \leq 0$, which is a shorthand notation for the following set:

$$\mathcal{C} \stackrel{\text{def}}{=} \{\mathbf{x} : c^{(1)}(\mathbf{x}) \leq 0,\, c^{(2)}(\mathbf{x}) \leq 0, \ldots, c^{(m)}(\mathbf{x}) \leq 0\},$$

where $c^{(j)}(\mathbf{x})$ is the j-th component of vector $\mathbf{c}(\mathbf{x})$. We assume that $\mathcal{C} \subset X$ is nonempty. At least the continuity of $c^{(j)}(\mathbf{x})$'s is required (Remark 1 applies correspondingly).

We consider the following optimization problem:

$$\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{subject to} \quad \mathbf{c}(\mathbf{x}) \leq 0. \tag{1}$$

Our aim is to discuss variants of filter based methods, iteratively generating sequences $\mathbf{x}_k \in X$, $k = 1, 2, \ldots$, which are expected to be convergent to a minimizer $\mathbf{x}^*$ of f over $\mathcal{C}$.

We confine to $\mathbf{c}(\mathbf{x}) \leq 0$ for simplicity of the exposition, because equality constraints can be handled by adding two inequality constraints.

2.1.2. Filter. Before describing the notion of a filter, as introduced by Fletcher and his co-workers (see Fletcher and Leyffer, 2002; Fletcher *et al.*, 2002a; 2002b), we have to define a special penalty function, denoted further by h,

$$h(\mathbf{c}(\mathbf{x})) = \sum_{j=1}^{m} \max\Big(0,\, c^{(j)}(\mathbf{x})\Big). \tag{2}$$

Clearly, for $\mathbf{x} \in \mathcal{C}$ we have $h(\mathbf{c}(\mathbf{x})) = 0$, while for $\mathbf{x} \notin \mathcal{C}$ $h(\mathbf{c}(\mathbf{x})) > 0$ is a measure of the constraint violation.

For given $\mathbf{x}_k$ we shall denote by $(h_k,\, f_k)$ a pair of the form $(h(\mathbf{c}(\mathbf{x}_k)),\, f(\mathbf{x}_k))$. In the k-th iteration a filter $\mathcal{F}_k$ is a list of pairs $(h_k,\, f_k)$, which were generated according to the rules described below.

We say that a pair $(h_k,\, f_k)$ dominates $(h_l,\, f_l)$ iff

$$f_k \leq f_l \quad \text{AND} \quad h_k \leq h_l \tag{3}$$

and at least one of these inequalities is strict. This definition differs slightly from the one given by Fletcher and Leyffer (2002), by adding the last requirement, but—in fact—we shall need even a more demanding notion of a dominance between such pairs. The following rules are applied to filter $\mathcal{F}_k$:

R1: Filter $\mathcal{F}_k$ contains only pairs $(h_l,\, f_l)$, which were generated up to the k-th iteration and such that no pair dominates any other.

R2: A pair $(h_l,\, f_l)$ is allowed to be included to $\mathcal{F}_k$, if it is not dominated by any point already contained in $\mathcal{F}_k$.

R3: If a pair $(h_l,\, f_l)$ is acceptable for inclusion in the filter $\mathcal{F}_k$, i.e., R2 holds, then all entries dominated by this pair are removed from $\mathcal{F}_k$ (in agreement with R1).

An example of a filter which fulfills the above rules is shown in Fig. 1 by dots. The lines and the shaded area indicate taboo regions, which are defined by the union of north-east orthants of the points included in the filter list at the k-th iteration. Desirable points for inclusion to $\mathcal{F}_k$ are as close as possible to $(0,\, f_{\min})$, where $f_{\min} = f(x^*)$. The role of the filter is to decide whether or not a new point, generated by a search algorithm should be accepted. Note that the rules R1–R3 allow a new point to be included into a filter, even if the corresponding value of the objective function is worse than that found so far. However, this may happen only if the penalty for constraints violation decreases.

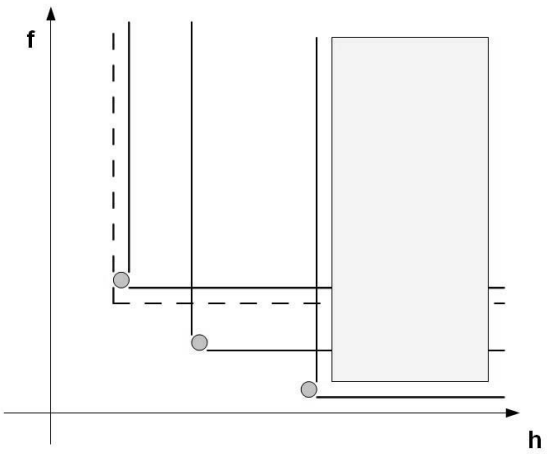


Fig. 1. Example of a filter. Dots indicate points included in the filter.

Remark 2. An important additional role of the filter content, especially after stopping the search, is that it provides valuable information on the trade off between attainable values of the objective function and the degree of constraint violation (see Fig. 6 in Section 4).

The rule R2 is frequently replaced by a stronger condition.

R2a: A pair $(h,\ f)$ is allowed to be included to $\mathcal{F}_k$ if for every $h_l,\ f_l$ from $\mathcal{F}_k$ the following conditions hold:

$$h \leq \beta\, h_l \quad \text{OR} \quad f \leq f_l - \gamma\, h, \tag{4}$$

where constants $\beta,\ \gamma \in (0, 1)$.

In practice, one can use $\beta = 0.9$ and $\gamma = 0.01$ or similar values. The conditions (4) extend a taboo region coded in $\mathcal{F}_k$ by adding thin strips (an envelope), as illustrated in Fig. 1 for one filter entry by the dashed lines. Note that the second condition $f \leq f_l - \gamma\, h$ contains the same h for each filter entry (in contrast to earlier papers on filter methods, in which $f \leq f_l - \gamma\, h_l$).

We refer the reader to the works of Fletcher and Leyffer (2002) as well as Fletcher (2010) for many details concerning more technical aspects of filter handling, such as bounds in the north-west and south-east corners of the taboo region and removing blocking entries. We also refer the reader to variants of the SQP-filter method (Li, 2006; Shen *et al.*, 2010; 2009; Biegler, 2010; Su and Che, 2007; Nie and Ma, 2006; Byrd *et al.*, 2010).

2.1.3. Skeletal algorithm based on the filter approach. Having $\mathbf{x}_k$, we approximate $f(\mathbf{x}_k + \mathbf{d})$ in its vicinity by a function which is denoted as $ap(f,\ \mathbf{x}_k,\ \mathbf{d})$, where $\mathbf{d} \in \mathbb{R}^d$ is a direction of the next search. Thus,

$$ap(f,\ \mathbf{x}_k,\ \mathbf{d}) \approx f(\mathbf{x}_k + \mathbf{d}), \tag{5}$$

without explicitly specifying the way of approximation. Chin and Fletcher (2003) used a linear approximation, while in the work of Fletcher and Leyffer (2002) and subsequent papers the following quadratic approximation was applied:

$$ap(f,\ \mathbf{x}_k,\ \mathbf{d}) = \frac{1}{2}\,\mathbf{d}^T\, W_k\, \mathbf{d} + \mathbf{d}^T\, \mathbf{g}_k, \tag{6}$$

where $\mathbf{g}_k$ is the gradient of f, calculated at $\mathbf{x}_k$, while W_k is a $d \times d$ matrix, which approximates the Hessian matrix, e.g., by the BFGS formula (see Broyden, 1970). In fact, for filter-like methods, one may use any other reasonable approximation instead of linear or quadratic (6), e.g., a mixture of the linear one, if we are far from optimum and the quadratic one when we are closer to it.

Similarly, by $\mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d})$ we denote an approximation of $\mathbf{c}(\mathbf{x})$ in a vicinity of $\mathbf{x}_k$, i.e.,

$$\mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d}) \approx \mathbf{c}(\mathbf{x}_k + \mathbf{d}), \tag{7}$$

again, without specifying approximating functions. A typical approximation is the linear one:

$$\mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d}) = \mathbf{c}(\mathbf{x}_k) + A_k^T\, \mathbf{d}, \tag{8}$$

where A_k is the Jacobian matrix of $\mathbf{c}(\mathbf{x})$, calculated at $\mathbf{x}_k$. However, one may also consider locally quadratic approximations of constraints or switching between the linear and the quadratic one.

Filter methods for the problem (1) are based on solving the following subproblems:

$$\min_{\mathbf{d}}\ ap(f,\ \mathbf{x}_k,\ \mathbf{d}),\ \text{subject to}\ \mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d}) \leq 0. \tag{9}$$

The selection of $\mathbf{d}$ is usually additionally constrained. The most popular approach is the trust-region method, in which $||\mathbf{d}||_\infty \leq \rho$, where $||\mathbf{d}||_\infty = \max_i |d^{(i)}|$ and $\rho > 0$ is the radius of the trust-region (see, e.g., the work of Nocedal and Wright (2006) for a discussion on trust regions). As usual, the inequalities in $\mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d}) \leq 0$ are understood component-wise. Clearly, one can select any reasonable method for solving (9), but in practice different variants (see, e.g., Fletcher *et al.*, 2002a; 2002b; Chin, 2007) of Quadratic Programming (QP) are used when $ap(f,\ \mathbf{x}_k,\ \mathbf{d})$ is a quadratic approximation and $\mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d})$ is a linear one. If also $ap(f,\ \mathbf{x}_k,\ \mathbf{d})$ is linear, then Linear Programming (LP) solvers are applied for (9). However, we emphasize that—in principle—any reasonable optimization method which is compatible with approximations used for $f(\mathbf{x})$ and $\mathbf{c}(\mathbf{x})$ can be used for solving the subproblems (9) within the filter methodology framework.

The next important ingredient of this methodology is the so-called *Feasibility Restoration Phase* (FRP). Its utilization is necessary when it happens that

$$\{\mathbf{d}:\ \mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d}) \leq 0\} \cap \{\mathbf{d}:\ ||\mathbf{d}||_\infty \leq \rho\} = \emptyset. \tag{10}$$

The typical reason for (10) to occur is that ρ is too small. It can also happen that approximations (e.g., linearizations) of $\mathbf{c}(\mathbf{x})$ near $\mathbf{x}_k$ may lead to inconsistent constraints.

As proposed by Fletcher and Leyffer (2002), FRP is accomplished by solving the following subproblem:

$$\min_{\mathbf{x}} \sum_{j \in J_k +} \max\left(0,\, c^{(j)}(\mathbf{x})\right) \tag{11}$$

subject to the constraints

$$c^{(j)}(\mathbf{x}) \le 0, \quad j \in J_k -, \tag{12}$$

starting from $\mathbf{x}_k$, where J_k+ is the set of those indices, for which constraints at $\mathbf{x}_k$ are violated, while J_k- contains indices for which $c^{(j)}(\mathbf{x}_k) \le 0$ holds. In practice, (11) is minimized under approximated constraints. We shall not describe a way of solving (11) and (12), because one can use any sufficiently efficient optimization method, taking into account that the objective function in (11) is formally non-differentiable.

The main modifications that are proposed in this paper are directed to improvements in the restoration phase, retaining its main goal. In order to point out our modifications, we provide an outline of the filter approach in a way, which is close in spirit to the presentation in (Fletcher and Leyffer, 2002).

Skeletal filter method. Select a starting point $\mathbf{x}_0$ and $\rho > 0$ as well as $\beta, \gamma \in (0,\, 1)$. Select $\epsilon > 0$ and $\eta > 0$ for stopping conditions imposed on $||\mathbf{d}||$ and $h(\mathbf{c}(\mathbf{x}))$, respectively, where $||\cdot||$ is the Euclidean norm. Select a sequence of numbers $\gamma_k > 0$, $k = 0,\, 1, \ldots$, used in the second stopping condition, which are such that

$$\lim_{k \to \infty} \gamma_k = 0, \quad \sum_{k=0}^{\infty} \gamma_k = \infty. \tag{13}$$

This sequence can be as simple as $\gamma_k = c/(k+1)$, where $c > 0$ is not too large a constant.

Step 0. Set $k = 0$ and calculate $f_0 = f(\mathbf{x}_0)$, $h_0 = h(\mathbf{c}(\mathbf{x}_0))$. Initialize the filter $\mathcal{F}_0$ by introducing a pair $(H,\, -\infty)$ to it, where $H > 0$ is a (large) constant, preventing too large values of $h(\mathbf{c}(\mathbf{x}))$. If $\mathbf{x}_0 \in \mathcal{C}$, then add to the filter $\mathcal{F}_0$ a pair $0,\, f_0)$. Otherwise, add $(0,\, F_{\max})$ to $\mathcal{F}_0$, where $F_{\max}$ is a crude upper bound to f. Alternatively, one can solve (11), (12) and treat the result as new x_0.

Step 1. Verify the feasibility of the subproblem (9). If (9) is feasible, then go to Step 4. If it is not feasible, solve the FRP subproblem (11), (12), starting from $\mathbf{x}_k$. Denote its solution by $\mathbf{x}_{k+1}$.

Step 2. Calculate $f_{k+1} = f(\mathbf{x}_{k+1})$ and then $h_{k+1} = h(\mathbf{c}(\mathbf{x}_{k+1}))$. If for $(h_{k+1},\, f_{k+1})$ the condition (4) holds, then enter $(h_{k+1},\, f_{k+1})$ to $\mathcal{F}_k$, remove all pairs dominated by the new one from $\mathcal{F}_k$ and go to Step 7. Otherwise, go to Step 3.

Step 3: Emergency step. Try to solve the FRP subproblem (11), (12), starting from a point different from $\mathbf{x}_k$. If, after several trials, the condition (4) does not hold, then STOP, suspecting that the original problem does not have a solution.

Step 4. Solve (9) and denote its solution by $\mathbf{d}_k$. Calculate $f_{k+1} = f(\mathbf{x}_k + \mathbf{d}_k)$ and $h_{k+1} = h(\mathbf{c}(\mathbf{x}_k + \mathbf{d}_k))$. If $h_{k+1} < \eta$ and at least one of the following conditions holds:

4a. $||\mathbf{d}_k|| < \epsilon$ OR

4b. $f_{k+1} \ge f_k - \gamma_k$,

then STOP—$\mathbf{x}_k + \mathbf{d}_k$ is a solution close to optimum.

Step 5. If $(h_{k+1},\, f_{k+1})$ is acceptable for the filter $\mathcal{F}_k$, then include this pair and clean the filter from dominated entries. Accept also $\mathbf{x}_{k+1} = \mathbf{x}_k + d_k$, consider an increase in ρ and go to Step 7.

Step 6. If $(h_{k+1},\, f_{k+1})$ is not acceptable for $\mathcal{F}_k$, then reduce ρ, keep $\mathbf{x}_{k+1} = \mathbf{x}_k$ as a starting point for a new search.

Step 7. Set $k = k + 1$ and go to Step 1.

Several remarks are in order, concerning the above skeletal algorithm.

Remark 3. In Step 0 we put bounds on the search region in $(h,\, f)$ coordinates. They preclude that sequences $(h_k,\, f_k)$ with h_k decreasing to zero and f_k increasing to infinity are added to the filter. They also do not allow sequences with decreasing f_k and h_k growing to infinity to be entered into the filter.

Remark 4. After Step 6 we cannot directly solve the subproblem (9), because a decrease in ρ may cause that (10) holds. Hence, we are forced to check feasibility again.

Remark 5. We do not describe the so-called Second Order Correction (SOC) steps, which are used by Fletcher and Leyffer (2002), because they rely on the second order approximation of f and the linear approximation of $\mathbf{c}$. They are useful when $(h_{k+1},\, f_{k+1})$ is not acceptable to $\mathcal{F}_k$, because they bend a search direction to $\mathcal{C}$ and try to retain a fast local convergence in a vicinity of optimum, stemming to avoid the Maratos effect.

Remark 6. The stopping condition $||\mathbf{d}_k|| < \epsilon$ is a standard one. Note, however, that $\mathbf{x}_k$ can be outside $\mathcal{C}$ and it is necessary to check also whether $h(\mathbf{c}(\mathbf{x}_k))$ is sufficiently small. If the stopping condition 4b does not hold, then we

can be sure that the convergence is not very slow, because in this case

$$f_{k+1} < f_k - \gamma_k, \quad k = 0, 1, \ldots. \tag{14}$$

On the other hand, this condition is not too restrictive, because it means that we require less than a Q-linear convergence rate (see the work of Nocedal and Wright (2006), Appendix A2) for a discussion on convergence rates). In practice, the condition 4b can even be simplified to $f_{k+1} \geq f_k - \zeta$, where $\zeta > 0$ is a small constant. It is added in a more complicated form merely for the discussion of the convergence of the skeletal algorithm.

Remark 7. It may happen that a sequence $\mathbf{x}_k$, $k = 0, 1, 2\ldots$, converges to $\mathbf{x}_c$, say, which is infeasible $h(\mathbf{c}(x_c)) > 0$. For fast detection of an infeasible stationary point for the FRP subproblem, the exact penalty approach is applicable (Byrd *et al.*, 2010).

Remark 8. We allow entries $(0,\ f_k)$ in the filter. There are variants of the filter algorithms that preclude such cases (Fletcher, 2010).

2.2. Convergence of the skeletal filter algorithm. Our aim in this subsection is to discuss sufficient conditions for the convergence of the skeletal algorithm. The framework provided by the skeletal algorithm is very wide. In our discussion on its covergence we use many arguments that appeared earlier in many papers cited in Introduction for more specific versions of filter type SQP or SLP. We shall try to catch their most important features, which are important for their convergence.

2.2.1. Additional assumptions. This cannot be done without imposing additional, but hopefully still general and nonvoid conditions on $f(\mathbf{x})$, $\mathbf{c}(\mathbf{x})$, their approximations, and on the efficiency of solving the subproblems (9) and (11), (12).

A1. The admissible set $\mathcal{C}$ is nonempty and compact. Thus, by continuity, $f(\mathbf{x})$ attains its minimum, denoted as $f_{\min}$. It is bounded from above by (attainable) $f_{\max}$. $f(\mathbf{x})$ is also bounded from below in X, by (attainable) $f_{\mathrm{MIN}} \leq f_{\min}$.

A2. If $\mathbf{x}_k \in \mathcal{C}$ and $f(\mathbf{x}_k) > f_{\min}$, then there exists a direction $\hat{\mathbf{d}}_k \neq 0$ such that

$$f(\mathbf{x}_k + \hat{\mathbf{d}}_k) < f(\mathbf{x}_k) - \gamma_k. \tag{15}$$

It is allowed that $(\mathbf{x}_k + \hat{\mathbf{d}}_k)$ is not in $\mathcal{C}$.

A3. Furthermore, we assume that approximations that are used in $ap(f,\ \mathbf{x}_k,\ \mathbf{d})$ and $\mathbf{ap}(\mathbf{c},\ \mathbf{x}_k,\ \mathbf{d})$ are sufficiently accurate so that we can get $\hat{\mathbf{d}}_k$ mentioned in A2 as a result of solving the subproblem (9).

A4. The solver that is used for solving the FRP (11) and (12) is sufficiently efficient so as to ensure that FRP is always successful when needed. Thus, the emergency step (Step 3) is not necessary and we skip it in the algorithm (for the discussions on convergence), retaining the numeration of other steps unchanged.

The requirements imposed by the assumptions A2 and A3 on the approximations of $f(\mathbf{x})$, $\mathbf{c}(\mathbf{x})$ and implicitly on the solver of (9) are fulfilled, e.g., if these functions are twice continuously differentiable in X, a quadratic approximation of $f(\mathbf{x})$ and linear approximations of $\mathbf{c}(\mathbf{x})$ are used in conjunction with a QP solver. The only difficulty can arise when $f(\mathbf{x})$ is very flat (or constant) in a vicinity of optimum. Therefore, we additionally assume that

A5. $\mathcal{C}$ is a convex set and $f(\mathbf{x})$ is strictly convex. Denote by $\mathbf{x}_* \in \mathcal{C}$ the point where a minimum of $f(\mathbf{x})$ is attained.

The assumption A4 describes an ideal solver. Its use simplifies the discussion of convergence. If we allow that A4 does not hold, then we have to discuss separately the case described in Remark 7.

The outline of the proof of the following result is deferred to Appendix.

Corollary 1. *Under the additional assumptions A1–A5 the skeletal algorithm either generates*

1. *a finite sequence* $\mathbf{x}_k$, $k = 1, 2, \ldots, K$ *and* $f(\mathbf{x}_K) = f_{\min}$, *or*
2. *an infinite sequence, which contains a subsequence convergent to* $\mathbf{x}_*$ *and* $f(\mathbf{x}_*) = f_{\min}$.

If A4 does not hold, then we have to take into account that the algorithm stops outside $\mathcal{C}$, because FRP fails (see the works of Fletcher and Leyffer (2002) as well as Fletcher (2010) for a discussion).

We cannot establish the rate of convergence of the skeletal algorithm, because that would require imposing smoothness assumptions on f. Note, however, that the goal of constructing algorithms that are based on a filter is mainly to assure that they are convergent when a starting point is far from an optimum. On the contrary, the Newton method provides the second order rate of convergence, but only when the starting point is located sufficiently close to $\mathbf{x}_*$.

2.3. Our modifications. Our experience gathered so far regarding using the filter algorithm indicates that the following modifications are useful:

Modification 1. If $\mathbf{x}_0 \notin \mathcal{C}$, solve the problem

$$\min_{\mathbf{x}} h(\mathbf{c}(\mathbf{x})) \tag{16}$$

without constraints (or imposing only reasonable constraints preventing overflows) and plug the resulting $(h,\ f)$ into the filter.

Modification 2. One of the reasons for deriving filter-like algorithms was to avoid using penalty function methods, because they require a subtle choice of penalty increase (see Fletcher and Leyffer, 2002). Nevertheless, our experiments indicate that in FRP it is expedient to minimize

$$f(\mathbf{x}) + \nu\, h(\mathbf{c}(\mathbf{x})) \tag{17}$$

instead of $h(\mathbf{c}(\mathbf{x}))$ only. This modification reduces sudden jumps of $\mathbf{x}_{k+1}$'s to the regions where $f(\mathbf{x}_{k+1})$ is large (see the second example in Section 4). The motivation for minimizing (17) in FRP is similar to the one which motivates the minimization of $h(\mathbf{c}(\mathbf{x}))$ only (see the work of Conn *et al.* (2000) for a discussion).

Here $\nu > 0$ is selected according to a simple rule of thumb: choose ν so that $f(\mathbf{x})$ and $\nu\, h(\mathbf{c}(\mathbf{x}))$ are of comparable magnitudes. Then ν can be updated in subsequent iterations, but it is not necessarily growing to infinity. Thus, difficulties typical for penalty function methods are to some extent reduced. On the other hand, for sufficiently large ν the minimization of (17) improves the infeasibility of the QP subproblem for the same reasons as the minimization of $h(\mathbf{c}(\mathbf{x}))$ does.

Modification 3. Formally, $h(\mathbf{c}(\mathbf{x}))$ is not differentiable, even if $\mathbf{c}(\mathbf{x})$ is. Note, however, that the lack of differentiability occurs on the boundary of $\mathcal{C}$ only. On the other hand, FRP is invoked only when $\mathbf{x}_k \notin \mathcal{C}$ and in its vicinity $h(\mathbf{c}(\mathbf{x}))$ is as smooth as $\mathbf{c}(\mathbf{x})$ itself. Thus, in FRP one can safely use a conjugate gradient algorithm or a quasi-Newton method, depending on whether $\mathbf{c}(\mathbf{x})$ is once or twice differentiable, respectively.

The lack of differentiability can also be avoided by reinterpreting the FRP subproblem as a bound-constrained least-squares one or a bound-constrained system of nonlinear equations. Then, one can use a projected conjugate gradient algorithm or an affine scaling trust region algorithm (Bellavia *et al.*, 2004).

Modification 4. The standard way of solving (9) is a variant of quadratic programming with the Hessian update, which can be inaccurate for large problems. For this reason it is expedient to modify Step 4 as follows: Select the number of internal iterations $I > 1$ (in our tests I was selected between 6 and 12).

Step 4.0. Set $i = 0$ and $\mathbf{x}_{k(i)} = \mathbf{x}_k$.

Step 4.1. Solve (9) for $\mathbf{x}_{k(i)}$ with $||\mathbf{d}||_\infty \leq \rho$ and denote this intermediate solution solution as $\mathbf{d}_{\rm int}$. Set $\mathbf{x}_{k(i+1)} = \mathbf{x}_{k(i)} + \mathbf{d}_{\rm int}$.

Step 4.2. Verify the feasibility of (9) for $\mathbf{x}_{k(i+1)}$. If it is feasible and $i < I$, set $i = i + 1$ and go to Step 4.1.

Step 4.3. If it is feasible, but $i = I$, then go to Step 5 of the main algorithm, setting $\mathbf{x}_{k+1} = \mathbf{x}_{k(i+1)}$.

Step 4.4. If it is not feasible, go to Step 5 of the main algorithm, setting $\mathbf{x}_{k+1} = \mathbf{x}_{k(i)}$, i.e., with the previous feasible solution.

Note that we solve (9) several times, if FRP is not necessary, without confronting intermediate results with the filter contents. However, we stress that the final result of such sub-iterations is then compared with the filter content. It seems that the above modification does not spoil the convergence of the filter algorithm, because it is determined by the rules governing the filter, independently of the method of generating trial points, providing that this method is able to produce points with smaller values of f or h.

It follows from our experience, partly reported at the end of this paper, that they allow an increase in the computational efficiency.

3. Interactions of a filter method with a PDE solver in optimal control problems

3.1. Framework for optimal control problems for DPSs. To fix ideas, we shall use the following simple model:

$$\frac{\partial q(\chi,\, t)}{\partial t} = \frac{\partial}{\partial \chi}\left(a(q)\,\frac{\partial q(\chi,\, t)}{\partial \chi}\right) + F(q, u(\chi,\, t)),$$
$$t \in (0,\, T_c),\ \chi \in (0,\, 1), \tag{18}$$

where $q(\chi,\, t)$ is the system state at a spatial point $\chi \in (0,\, 1)$ at time $t > 0$. $T_c > 0$ is a time horizon. Equation (18) is accompanied by initial and boundary conditions. $u(\chi,\, t)$ is a control, while $a(q)$ and $F(q,\, u)$ are known functions, which may depend on $q(\chi,\, t)$ as well as on $q_\chi(\chi,\, t)$, where q_χ denotes the partial derivative with respect to χ.

In order to formulate an optimal control problem, we also have to specify the set U of admissible control actions $u(\cdot)$, which is usually a subset of the space $L_2((0,\, 1) \times (0,\, T_c))$ of square integrable functions.

C1. We assume that for any control $u(\cdot) \in U$ and appropriate initial and boundary conditions the solution of (18) exists and is unique.

Particular sets of sufficient conditions for C1 depend on $F(\cdot)$, $a(\cdot)$, boundary conditions and the required smoothness of $q(\cdot)$.

Additional constraints can be imposed on the system state $q()$. The set of admissible states will be denoted by $C_q \subset L_2((0,\, 1) \times (0,\, T_c))$. The last ingredient is the performance index (criterion), which serves as an indicator to what extent the objective of a control action is achieved.

A standard example of a performance criterion, further denoted by $J(u,\, q)$, is the following one:

$$J(u,\, q) = \int_0^1 \left(q^*(\chi) - q(\chi,\, T_c)\right)^2 \,\mathrm{d}\chi, \qquad (19)$$

where $q^*(\chi)$ is a desired system state at time T_c.

Our aim is to discuss possible approaches to find an approximate solution to the following problem:

$$J(u^*,\, q^*) = \min_u J(u,\, q), \quad (u,\, q) \in U \times C_q, \qquad (20)$$

where q and u are additionally constrained by (18) and its initial and boundary conditions. In (20), U is the set of admissible controls, e.g.,

$$U = \{u \in L_2((0,1) \times (0,T_c)) : \int_0^1 \int_0^{T_c} u^2(\chi,t)\,\mathrm{d}t\,\mathrm{d}\chi \le 1\}.$$

Alternatively, one can add the term

$$\int_0^1 \int_0^{T_c} u^2(\chi,t)\,\mathrm{d}t\,\mathrm{d}\chi$$

penalizing excessive use of energy to the performance index.

C2. We assume that the solution of problem (20) exists and is unique.

C2 holds for a wide class of control problems (see the works of Lasiecka and Triggiani (2000), Lasiecka and Chueshow (2010), Lasiecka and Chueshow (2008), Nettaanmaki and Tiba (1994) as well as Troltzsch (2010) for a wide range of results).

3.2. Intermediate decisions. It is usually impossible to solve the problem (20) in closed form. Before deciding how to solve (20) numerically, it is expedient to list intermediate choices that we have to make. Our discussion on these topics extends interesting deliberations of Hinze *et al.* (2009) and Betts (2010). Our decisions will be partially biased by our actual goal, i.e., testing the filter algorithm on a moderate size, but already difficult, DPS control problem. For this reason we decide to consider an open loop control problem (20). In practice, optimizing a closed loop control law can be beneficial, if we know the structure of a controller, which is a rare case when a system is nonlinear.

Abstract vs. $\mathbb{R}^d$ formulation. It is well known that optimal control problems can be formulated as optimization problems in Banach spaces. Also SQP has its abstract version (see Hinze *et al.*, 2009). Alternatively, one can use finite dimensional, in $\mathbb{R}^d$, say, approximations of $u(\cdot)$, $q(\cdot)$ and the PDE from the beginning. In this paper we select the latter approach.

Direct search vs. solving optimality conditions. A direct search for u^* that solves (20) will be discussed in detail below. The second approach means that we have to derive necessary optimality conditions and then search for their solution. Such conditions are usually variants of the maximum principle, accompanied by a pair of PDEs: the original one (18) and the adjoint one.

Remark 9. Although searching for a maximum of the Hamiltonian is, in principle, easier than a variational problem, solving the above mentioned pair of PDEs is a source of severe numerical instabilities, because one of these equations must be solved forward and the second one backward in time. Thus, one of them must be unstable the one which bears information about a future that is hidden in the model. As mentioned by Betts (2010), these difficulties have been known since the 1960s and they have been met even for systems described by ODEs.

Thus, later on we concentrate on direct methods for (20).

Solving the PDE vs. treating it as a constraint. The next step—within direct minimization approaches—is to decide how to handle (18). The first choice is to solve PDEs in each iteration. The second is to treat the PDE as a constraint, which leads—after its discretization—to a large number of constraints (usually hundreds or thousands). Recent advances in large scale optimization techniques and available computational power moved this approach from dreams to accessible reality (see the work of Hinze *et al.* (2009) and the bibliography cited therein). However, in this paper we shall follow the first choice, which is more traditional, but also still more reliable, because the number of additionally introduced decision variables and constraints is much smaller. Thus, we can attack more difficult nonlinear problems with a larger number of constraints on system states that arise in practice.

An additional advantage of solving the PDE instead of treating it as a constraint arises when the PDE is a nonlinear one. Namely, its approximate solution is close to the exact one, even if it was iteratively linearized in the mean time. On the other hand, a nonlinear PDE as a constraint must be linearized directly, increasing the danger of obtaining an empty set of linearized constraints.

Finite dimensional representation of the PDE and control. The number of numerical methods for PDEs is so large (finite differences, finite elements, boundary elements, etc.) that we do not even try to list them all. Our aim is only to point out that there are relationships between the way of solving the PDE and a finite dimensional approximation of $u(\chi,\, t)$. When the PDE is solved by finite differences on a grid $(\chi_i,\, t_j)$, say, then it seems "nat-

ural" to select $u(\chi_i,\, t_j)$ as decision variables. Similarly, if $q(\chi,\, t)$ is approximated by finite elements, then the obvious choice is to represent $u(\chi,\, t)$ in the same basis. Note, however, that the number of grid points or the number of triangles in a finite element basis is very large—usually more than several thousands. One can reduce the number of decision variables approximating $u(\chi,\, t)$ as follows:

$$u_d(\chi,\, t,\, \mathbf{x}) = \sum_{l=1}^{d} x^{(l)}\, \phi_l(\chi,\, t), \tag{21}$$

where $\mathbf{x} = [x^{(1)},\, x^{(2)}, \ldots,\, x^{(d)}]^T$, $\phi_l(\chi,\, t)$'s are selected basis functions, e.g, splines, trigonometric functions, etc. Now, when finite differences are used, then we insert $u_d(\chi_i,\, t_j,\, \mathbf{x})$ at nodal points. Similarly, scalar products of (21) with finite elements can be used for solving PDEs.

Optimization method. Derivative free optimization methods are very time consuming when applied to optimal control for PDEs. In our simulations it occurred that the Nelder–Mead method is useful when the problem is very irregular (steep valleys, large gradients, etc.). However, in general, we prefer SQP-like methods. It remains to discuss the way of evaluating the gradient of $J(u_d(\cdot, \mathbf{x}),\, q)$, taking into account that q also depends on $\mathbf{x}$ indirectly through (18). We shall denote by $q(\chi,\, t,\, \mathbf{x})$ the solution of (18) with u replaced by u_d. Let us denote by $f(\mathbf{x}) = J(u_d(\cdot,\, \mathbf{x}), q(\cdot,\, \mathbf{x}))$. One can select one of the following approaches in order to calculate $\mathrm{grad}_{\mathbf{x}}\, f(\mathbf{x}) = \mathrm{grad}_{\mathbf{x}}\, J(u_d(\cdot,\, \mathbf{x}), q(\cdot,\, \mathbf{x}))$:

(i) Solve sensitivity equations for each $\partial q(\chi,\, t,\, \mathbf{x})/\partial x^{(l)}$, $l = 1,\, 2, \ldots,\, d$. The sensitivity equations are derived by taking partial derivatives of (18).

(ii) Use adjoint equations.

(iii) Evaluate $\mathrm{grad}_{\mathbf{x}}\, q(\chi,\, t,\, \mathbf{x})$ by finite differences.

The disadvantages of (2) are the same as mentioned in Remark 9. The computational burdens of (1) and (3) are comparable with the case when we have to solve the PDE $(d+1)$-times. We have selected the approach (3) in our computational experiments, because it is easier to implement.

Constraints. When u_d is represented as (21), it is easy to impose constraints on it. In particular, constraints

$$c^{(i)}(\mathbf{x}) = x^{(i)} - \upsilon \le 0, \quad i = 1,\, 2, \ldots,\, d \tag{22}$$

are interpretable, where $\upsilon > 0$ is an upper bound. Also constraints on the amplitude or energy of u_d can be easily formulated.

Checking constraints imposed on q requires solving the PDE, but this done simultaneously with evaluating the objective function. For example, one would like to avoid too large tensions invoked by sudden changes in the temperature along a spatial variable. The corresponding constraints can be formulated as follows: For $l = 1,\, 2, \ldots,$

$$c^{(d+l)}(\mathbf{x}) = \left(q(\chi_{l+1}, \tau_j, \mathbf{x}) - q(\chi_l, \tau_j, \mathbf{x})\right)^2 - \omega \le 0, \tag{23}$$

where $\omega > 0$ is a bound for admissible changes, while χ_l's and τ_j's are grid points.

3.3. Outline of a filter based algorithm searching for optimal control. Below we provide an outline of a filter method dedicated for searching optimal control. We put emphasis on the most time consuming step, i.e., on solving the PDE. Define $\mathbf{e}_j$'s as the unit vectors of orthogonal basis in $\mathbb{R}^d$.

Algorithm B

Repeat

Step A. For $u_d(\chi,\, t,\, \mathbf{x}_k)$ and $u_d(\chi,\, t,\, \mathbf{x}_k + \delta_j\, \mathbf{e}_j)$, $j = 1,\, 2, \ldots,\, d$ solve the PDE. Use the results for calculating:
(a) $J_k \stackrel{\mathrm{def}}{=} J(u_d(\cdot, \mathbf{x}_k),\, q(\cdot,\, \mathbf{x}_k))$ and $\mathbf{c}_k \stackrel{\mathrm{def}}{=} \mathbf{c}(\mathbf{x}_k)$,
(b) $\nabla_k\, J$, which approximates

$$\mathrm{grad}_{\mathbf{x}} J(u_d(\cdot, \mathbf{x}),\, q(\cdot,\, \mathbf{x}))|_{\mathbf{x}=\mathbf{x}_k},$$

(c) $\nabla_k\, \mathbf{c}$, which approximates $\mathrm{grad}_{\mathbf{x}} \mathbf{c}(\mathbf{x})|_{\mathbf{x}=\mathbf{x}_k}$,
(d) update the Hessian approximation by the BFGS formula (denote it by H_k).

Step B. Solve the following QP problem:

$$\min_{\mathbf{d}} \left[\mathbf{d}^T\, \nabla_k\, J + \frac{1}{2}\, \mathbf{d}^T\, H_k\, \mathbf{d}\right], \quad \mathbf{c}_k + \mathbf{d}^T\, \nabla_k\, \mathbf{c} \le 0. \tag{24}$$

Step C. If (24) is infeasible, enter the FRP (again the PDE must be solved several times). Otherwise, denote its solution by $\mathbf{d}_k$ and consider repeating (24) (according to our modification 4.). Perform Steps 4–6 of the skeletal algorithm.

until convergence.

The results of testing the above algorithm are provided in the next section.

4. Simulation studies

The aim of our simulations is to test Algorithm B.

4.1. DPS under study. Consider the system (18) specialized as follows:

Control $F(q,\, u(\chi,\, t)) = u(\chi,\, t)$, where

$$u(\chi,\, t) = U_1(t)\, s_1(\chi) + U_2(t)\, s_2(\chi) + U_3(t)\, s_3(\chi).$$

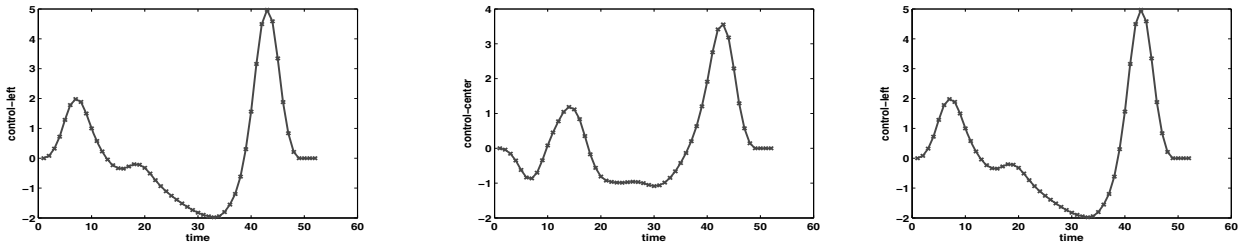


Fig. 2. Control signals in the left, central and right segments: final (optimal) values.

$U_j(t)$, $j = 1, 2, 3$ are control actions (manipulated input signals) to be selected. They act along the spatial variable in a piecewise constant way, i.e., $s_1(\chi) = 1$ for $\chi \in (0, 1/3)$ and zero otherwise, $s_2(\chi) = s_1(\chi - 1/3)$, $s_3(\chi) = s_1(\chi - 2/3)$. Each $U_j(t)$ was finitely parametrized as follows:

$$U_j(t) = \sum_{l=1}^{L} \kappa_{lj}\, B_l(t),$$

where $B_l(t)$ is a third-order B-spline. We additionally require $|\kappa_{lj}| \leq \Upsilon$, where $\Upsilon > 0$ is a given constant. These constraints impose bounds on the amplitudes of $U_j(t)$'s. Rearranged coefficients κ_{lj} form vector $\mathbf{x}$ of our decisions.

Initial and boundary conditions $q(\chi, 0) = q_0(\chi)$ are given, $q(0, t) = q(1, t) \equiv 0$.

Diffusion coefficient has the form

$$a(q) = \left[1 + \left(\frac{1}{\varsigma}\,\frac{\partial\, q(\chi, t)}{\partial \chi}\right)^2\right]^{-1}, \qquad (25)$$

where $\varsigma > 0$ is a given constant. In our experiments $\varsigma = 30$ was selected.

As can be noticed, $a(q)$ given by (25) has the form which is not typical for systems that can be met in nature but rather in Perona–Malik filters (see Perona and Malik, 1990). Let us note that the diffusion is highly reduced at spatial points, in which steep changes in q occur. Our idea is to make our testing problem more interesting by imposing also the constraints

$$\left(\frac{\partial\, q(\chi, t)}{\partial \chi}\right)^2 \leq \omega, \quad \forall t \in (0, T_c),$$

where $\omega > 0$ is a given constant. They were implemented as (23).

Summarizing, our aim is to select $\mathbf{x}$, which determines $U_j(t)$, $j = 1, 2, 3$, in such a way that our objective function (19) is minimized subject the constraints (22), (23) and (18) with $a(q)$ given by (25). The initial condition $q_0(\chi) = 0.5 \exp(-\chi^2/12.5)$ is given, while our desired final state $q^*(\chi) = 3$, $\chi \in (0, 1)$. Note that we have $q(0, t) = q(1, t) \equiv 0$ and we allow large changes in q near the end points, but not inside $(0, 1)$.

The PDE (18) with $a(q)$ given by (25) was solved by the explicit finite difference scheme on the equidistant grid with 52 time steps and 201 points in space. Thus, we also have 200 constraints (23) and 30 constraints (22).

In our numerical experiments all the modifications, described in Section 2.3, were implemented. Details of our implementation of Algorithm B are the following: According to Modification 3, (16) and (17) (with $\nu = 4$) were minimized using the quasi-Newton method with a cubic line search procedure and the BFGS formula for updating the approximation of the Hessian matrix (see Broyden, 1970).

4.2. Discussion of the results. An excerpt of our numerical experiments is shown in Table 1. The results of three runs are reported. They differ only in that the starting point was different. Run I can be named an "easy start", while Run III is an example of a "wild start", because it was artificially selected very far from an optimum in order to test the ability of the algorithm to cope with such cases. If (24) was feasible, then the quadratic programming problem was solved at most 12 times in each iteration. When it led to large constraints violation, then (17) was minimized. For these reasons the number of calculations of the objective function is large in comparison with the number of global iterations. An additional reason is that gradients were evaluated by finite differences.

The simulations were run using an Intel i7 2.67 GHz processor, using only one of its cores. The code was written in the Matlab language.

The results obtained after stopping Algorithm B are shown in Fig. 3, which describes the system state evolution in space and time under control signals plotted in Fig. 2. The initial and the final state along χ axes are shown in Fig. 5.

The behavior of the objective function in subsequent iterations was the following: In the first two (sometimes

Table 1. Summary of testing.

	Run I	Run II	Run III
Start crit.	49.7	$6.9\,10^3$	$6.7\,10^5$
Final crit.	9.1	9.0	8.7
Start penalty	0.0	3.3	$3.7\,10^3$
Final penalty	0.05	0.07	0.09
CPU (sec.)	77	170	245
Fun. eval.	1372	2237	3008
Iterations	2	4	8

three or four) iterations, the values of the goal function were reduced from 10^5 to 10, say. Typical behavior in the second phase is shown in Fig. 4. The algorithm tries to find a compromise between the values of the objective function (top plots) and the penalty $h(\mathbf{c}(\mathbf{x}))$ (bottom plots). As expected, lower values of the criterion correspond exactly to larger penalties for constraint violation. It is also expedient to analyze the content of the filter in the last iteration (see Fig. 6; note that the last point $(0.09,\ 8.7)$ is not shown). The content of the filter indicates that we can achieve 8.4 as the value of the objective function, if we can afford the constraint violation at the level 0.36. On the other hand, the achievable criterion value equals 8.7, if we require $h(\mathbf{c}(\mathbf{x}))$ less than 0.1 as in Run III. Clearly, we are able to make use of the filter content only if the corresponding $\mathbf{x}_k$'s are stored, but this is not a problem with modern computers.

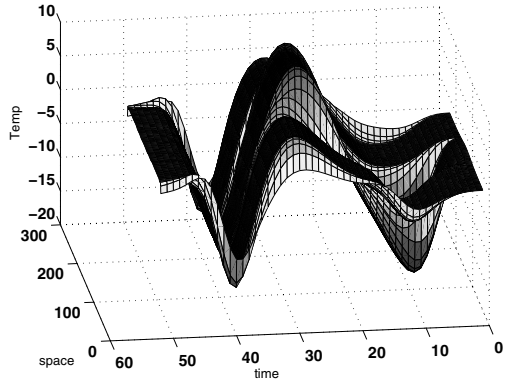


Fig. 3. Space-time evolution of the system state when optimal control is used.

4.3. How our modification works. Our main modification of the filter method is in minimizing (17) instead of minimizing (11) only. In order to illustrate its perfor-

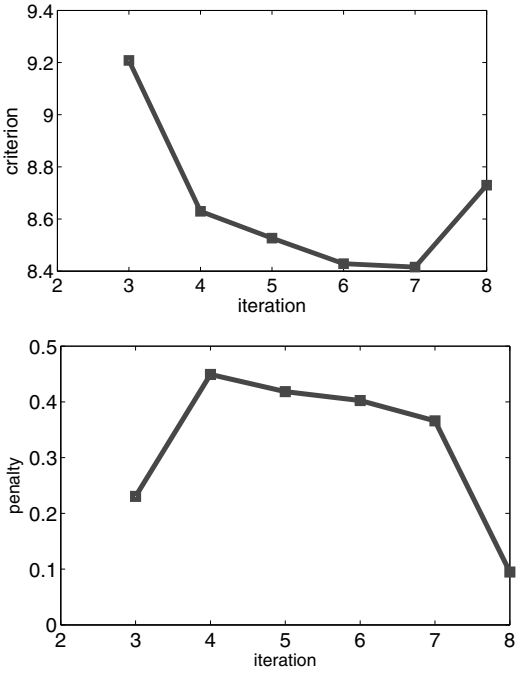


Fig. 4. Optimality criterion (top panel) and the penalty $h(\mathbf{c}(\mathbf{x}_k))$ in subsequent iterations, the first and the second iteration omitted.

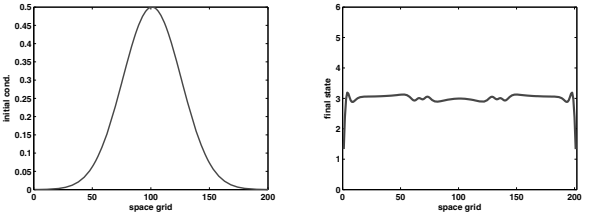


Fig. 5. Initial condition (left panel) and the final state (along χ) under optimal control (right panel).

mance, we have repeated the simulations described above with the following changes:

(i) a starting point was selected far away from the optimal one;

(ii) the calculations were conducted twice: minimizing (11) and (17), keeping the rest conditions unchanged;

(iii) in both cases the algorithms were stopped when the PDE (18) was solved about 8000 times (independently of the number of iterations in which they were used) for evaluating the objective function, its derivatives and in FRP.

The results (for $\nu = 25$) are shown in Fig. 7 for the case without our modifications and with them, the left and the right column, respectively. As one can notice, the objective function exhibits larger variability when our modifications are not used, while in the right column of Fig. 7 it

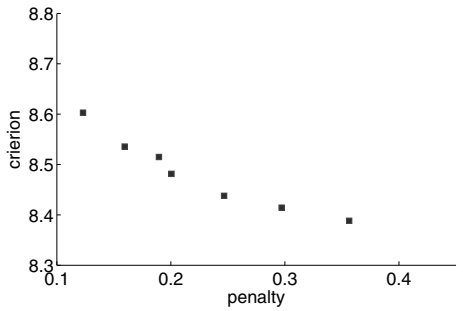


Fig. 6. Example of the filter content.

remains monotone. We have observed a similar behavior in many other runs, for different starting points and tuning parameters.

5. Concluding remarks

As documented in the present paper, optimization algorithms which are based on the idea of a filter can be developed to provide efficient methods for solving optimal control problems for nonlinear systems with spatio-temporal dynamics of at least moderate size. Our modifications of the filter approach make it even more robust to failures when a starting point is very far from optimum and constraints are highly violated. One may hope that the proposed approach will also be useful for solving optimal control problems with more difficult optimality criteria, such as arising in searching for optimal input signals for identification of DPSs (see Uciński, 2005).

Acknowledgment

The authors express their thanks to the anonymous referees for helpful comments and suggestions.

References

Armaou, A. and Christofides P.D. (2002). Dynamic optimization of dissipative PDE systems using nonlinear order reduction, *Chemical Engineering Science* **57**: 5083–5114.

Aschemanna, H., Kostinb, G.V., Rauha, A. and Saurinb, V.V. (2010). Approaches to control design and optimization in heat transfer problems, *International Journal of Computer and Systems Sciences* **49**(3): 380–391.

Audet, C. and Dennis, J.E. (2004). A pattern search filter method for nonlinear programming without derivatives, *SIAM Journal on Optimization* **14**(4): 980-1010.

Bellavia, S., Macconi, M. and Morini, B. (2004). STRSCNE: A scaled trust-region solver for constrained nonlinear equations, *Computational Optimization and Applications* **28**(1): 31–50.

Betts, J.T. (2010). *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, 2nd Edn.,* Society for Industrial and Applied Mathematics, Philadelphia, PA.

Biegler, L.T. (2010). *Nonlinear Programming. Concepts, Algorithms, and Applications to Chemical Processes*, SIAM, Philadelphia, PA.

Burger, J. and Pogu, M. (1991). Functional and numerical solution of a control problem originating from heat transfer, *Journal of Optimization Theory and Applications* **68**(1): 49–73.

Broyden, C.G. (1970). The convergence of a class of double-rank minimization algorithms, *Journal of the Institute of Mathematics and Its Applications* **6**: 76–90.

Butkovskiy, A.G. (1969). *Distributed Control Systems,* 1st Edn., Elsevier, New York, NY.

Byrd, R.H., Curtis, F.E. and Nocedal,J. (2010). Infeasibility detection and SQP methods for nonlinear optimization, *SIAM Journal on Optimization* **20**(5): 2281–2299.

Chin, C.M. and Fletcher, R. (2003). On the global convergence of an SLP-filter algorithm that takes EQP steps, *Mathematical Programming* **96**(1): 161–177.

Chin, C.M., Rashid, A.H.A. and Nor, K.M. (2007). Global and local convergence of a filter line search method for nonlinear programming, *Optimization Methods and Software* **22**(3): 365–390.

Christofides, P.D. (2001). *Nonlinear and Robust Control of PDE Systems: Methods and Applications to Transport-Reaction Processes*, Birkhauser, Boston, MA.

Conn, A.R., Gould, N. I. and Toint, P.L. (2000). *Trust-Region Methods*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, PA.

Demetriou, M.A. and Kazantzis, N. (2004). A new actuator activation policy for performance enhancement of controlled diffusion processes, *Automatica* **40**(3): 415–421.

El-Farra, N.E., Armaou, A. and Christofides, P.D. (2003). Analysis and control of parabolic PDE systems with input constraints, *Automatica* **39**(3): 715–725.

Fattorini, H.O. (1999). *Infinite Dimensional Optimization and Control Theory*, Cambridge University Press, Cambridge.

Fletcher R. and Leyffer, S. (2002). Nonlinear programming without a penalty function, *Mathematical Programming, Series A* **91**(2): 239–269.

Fletcher, R., Leyffer, and Toint, P.L. (2002a). On the global convergence of a filter-SQP algorithm, *SIAM Journal on Optimization* **13**(1): 44–59.

Fletcher R., Gould, N.I.M., Leyffer, S., Toint, Ph.L. and Wachter, A. (2002b). Global convergence of trust-region SQP-filter algorithms for general nonlinear programming, *SIAM Journal on Optimization* **13**(3): 635–659.

Fletcher, R. (2010). The sequential quadratic programming method, *in* G. Di Pillo and F. Schoen (Eds.), *Nonlinear Optimization*, Lecture Notes in Mathematics, Vol. 1989, Springer-Verlag, Berlin/Heidelberg.

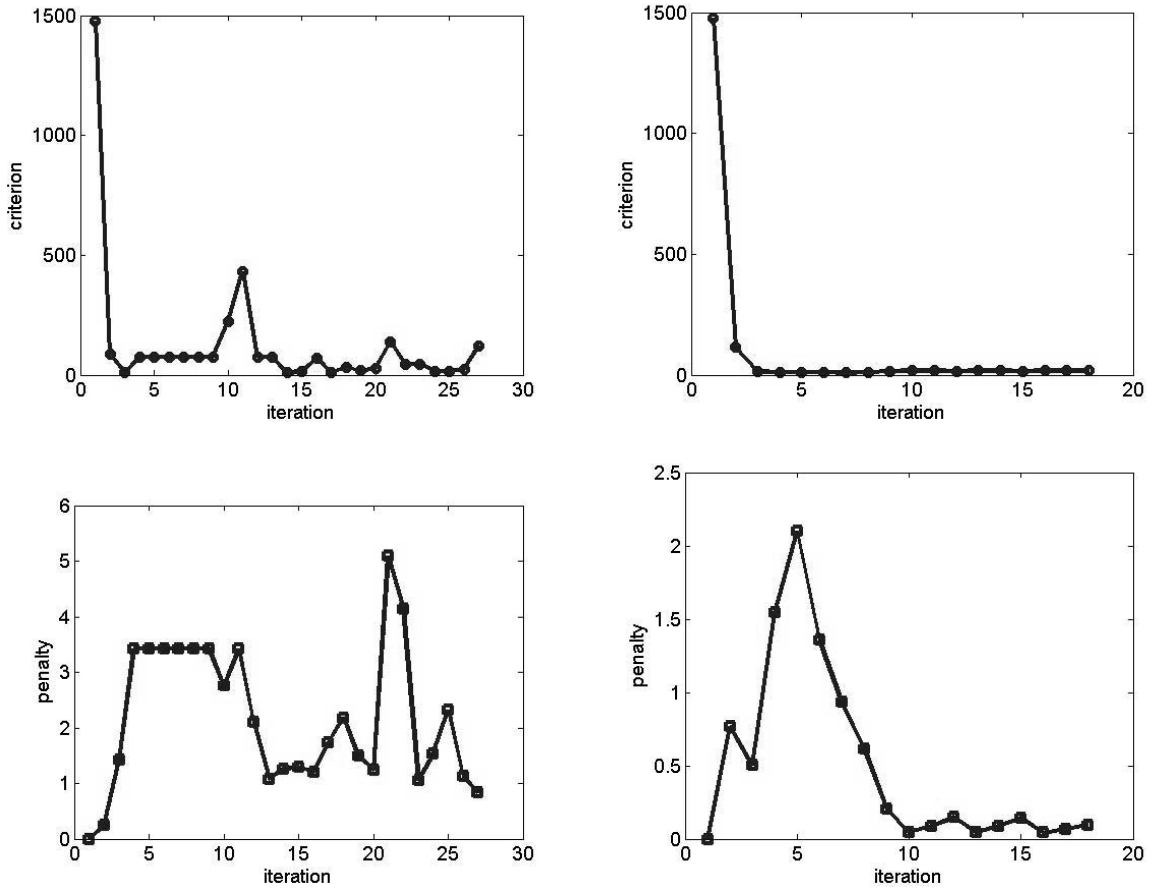


Fig. 7. Optimality criterion (top panels) and the penalty $h(\mathbf{c}(\mathbf{x}_k))$ (bottom panels) in subsequent iterations without our modification (left panels) and when it is applied (right panels).

Han, J. and Papalambros, P.Y. (2010). An SLP Filter algorithm for probabilistic analytical target cascading, *Structural and Multidisciplinary Optimization* **41**(5): 935–945.

Hinze, M., Pinnau, R., Ulbrich, M. and Ulbrich S. (2009). *Optimization with PDE Constraints*, Springer, Berlin/Heidelberg.

Lasiecka, I. and Triggiani, R. (2000). *Control Theory for Partial Differential Equations: Continuous and Approximation Theories, Vol. I: Abstract Parabolic Systems, Vol. II: Abstract Hyperbolic-Like Systems over a Finite Time Horizon,* Encyclopedia of Mathematics and Its Applications, Vol. 74, Cambridge University Press, Cambridge.

Lasiecka, I. and Chueshow, I. (2010). *Von Karman Evolution Equations: Well-posedness and Long Time Dynamics*, Springer, Berlin/Heidelberg.

Lasiecka, I. and Chueshow I. (2008). *Long-time Behavior of Second Order Evolution Equations with Nonlinear Damping*, Memoirs of the American Mathematical Society, Philadephia, PA.

Li, D. (2006). A new SQP-filter method for solving nonlinear programming problems, *Journal of Computational Mathematics* **24**(5): 609–634.

Nie, P. and Ma, C. (2006). A trust-region filter method for general non-linear programming, *Applied Mathematics and Computation* **172**(2): 1000–1017.

Nettaanmaki P. and Tiba, D. (1994). *Optimal Control of Nonlinear Parabolic Systems*, Marcel Dekker, New York, NY.

Nocedal J. and Wright, S.J. (2006). *Numerical Optimization*, Springer, Berlin/Heidelberg.

Perona, P. and Malik, J. (1990). Scale-space and edge detection using anisotropic diffusion, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(7): 629–639.

Rafajłowicz, E. (2008). Testing homogeneity of coefficients in distributed systems with application to quality monitoring, *IEEE Transactions on Control Systems Technology* **16**: 314–321.

Rafajłowicz, E. and Rafajłowicz, W. (2010). Testing (non-)linearity of distributed-parameter systems from a video sequence, *Asian Journal of Control* **12**(2): 453–461.

Schittkowski, K. (2002). *Numerical Data Fitting in Dynamical Systems; A Practical Introduction with Applications and Software*, Applied Optimization, Vol. 77, Kluwer Academic Publishers, Dordrecht.

Schittkowski, K. (2009). An active set strategy for solving optimization problems with up to 200,000,000 nonlinear constraints, *Applied Numerical Mathematics* **59**(12): 2999–3007.

Shen, C., Xue, W. and Pu, D. (2009). Global convergence of a tridimensional filter SQP algorithm based on the line search method, *Applied Numerical Mathematics* **59**(2): 235–250.

Shen, C., Xue, W. and Chen, X. (2010). Global convergence of a robust filter SQP algorithm, *European Journal of Operational Research* **206**(1): 34–45.

Skowron, M. and Styczeń, K., (2009). Evolutionary search for globally optimal stable multicycles in complex systems with inventory couplings, *International Journal of Chemical Engineering,* Article ID 137483, DOI:10.1155/2009/137483.

Su, K. and Che, J. (2007). A modified SQP-filter method and its global convergence, *Applied Mathematics and Computation* **194**(1): 92–101.

Su, K. and Yu, Z. (2009). A modified SQP method with nonmonotone technique and its global convergence, *Computers and Mathematics with Applications* **57**(2): 240–247.

Troltzsch, F. (2010). *Optimal Control of Partial Differential Equations. Theory, Methods and Applications*, American Mathematical Society Press, Providence, RI.

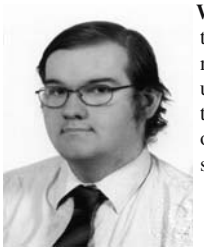
Turco, A. (2010). Adaptive filter SQP, *in* C. Blum and R. Battiti (Eds.), *Learning and Intelligent Optimization*, Lecture Notes in Computer Science, Vol. 6073, Springer-Verlag, Berlin/Heidelberg, pp. 68–81.

Uciński, D. (2005). *Optimal Measurement Methods for Distributed Parameter System Identification*, CRC Press, London/New York, NY.

Ulbrich, S. (2004). On the superlinear local convergence of a filter-SQP method, *Mathematical Programming, Series B* **100**(1): 217–245.

Ewaryst Rafajłowicz received his M.Sc. in control systems in 1977, Ph.D. in 1979, and D.Sc. in 1986 from the Wrocław University of Technology. He has held the full professorial position at the same university since 2001. His area of research includes optimization of input signals for system identification, control charts, pattern recognition and image processing. He is a senior member of the IEEE.

Krystyn Styczeń received the diploma degree in electronics from Kiev Polytechnical Institute in 1971. He is a professor of control theory at the Institute of Computer Engineering, Control and Robotics, Wrocław University of Technology, Poland. His main research interests are in the areas of optimal periodic control, evolutionary optimization, nonlinear optimization, nonsmooth and smoothing optimization methods, as well as DAE optimization for lumped and distributed parameter systems.



Wojciech Rafajłowicz received his M.Sc. in control systems at the Wrocław University of Technology, Faculty of Electronics, as the best graduate in 2011, and started work as an assistant at the same university. His research interests are in optimization and control of systems with time-space dynamics and image processing.

Appendix

In order to outline the proof of Corollary 1, it is expedient to consider the following cases.

For our discussion on convergence, we also set $\epsilon = 0$ and $\eta = 0$ in Step 4.

Case 1: Finite number of steps. Let us suppose that the skeletal algorithm stopped after a finite number of steps, say $0 < K < \infty$. We claim that x_K is the optimal solution.

The condition 4b cannot hold at $\mathbf{x}_K$. Indeed, if $\mathbf{d}_K \neq 0$ was found by solving (9), then by A2 and A3 we have $f(\mathbf{x}_K + \mathbf{d}_K) < f(\mathbf{x}_K) - \gamma_K$, which is in contradiction with 4b.

If 4a (with $\epsilon = 0$ and $\eta = 0$) holds, then $x_K \in \mathcal{C}$. If $f(\mathbf{x}_K) > f_{\min}$ then, by virtue of A2, we can find a descent direction, but we stopped with $\mathbf{d}_K = 0$; this is a contradiction, which implies $f(\mathbf{x}_K) = f_{\min}$.

Let us verify whether we can point out scenarios of stopping after a finite number of steps, which are not in contradiction with the rules imposed on the filter operation.

1. Suppose that $\mathbf{x}_0 \in \mathcal{C}$ and also all $\mathbf{x}_k \in \mathcal{C}$, $k = 1, 2, \ldots, K$. In this case all pairs that are entered to the filter have the form $(0,\ f_k)$. Furthermore, we must have $f_{k+1} < f_k - \gamma_k$, $k = 0, 1, \ldots, K$ and the radius of the trust region is never reduced. Such a simple scenario is possible, e.g., when f can be accurately approximated by a quadratic form and its unconstrained minimum is located in $\mathcal{C}$.
2. Also the case when $\mathbf{x}_0 \notin \mathcal{C}$ and the unconstrained minimum of f is located in $\mathcal{C}$ usually leads to stopping in a finite number of steps, because then $h(\mathbf{c}(\mathbf{x}_k))$ and $f(\mathbf{x}_k)$ are reduced in each iteration and they can be included into the filter. When $\mathbf{x}_k$ enters into $\mathcal{C}$, then the rest of the scenario is the same as in Case 1.

The case when the unconstrained minimum is of f is located outside $\mathcal{C}$ usually leads to an infinite sequence of iterates.

Case 2: Infinite number of steps. Now, let us consider the case that the skeletal algorithm generates an infinite sequence $\mathbf{x}_k$, $k = 0, 1, 2, \ldots$, which is contained in X and such that the corresponding pairs $(h_k,\ f_k)$ were admissible to the filter $\mathcal{F}_k$. By the compactness of X, the

sequence $\mathbf{x}_k$'s contain a convergent subsequence, $\mathbf{x}_{k(i)}$, $i = 1, 2, \ldots$, say. Denote its limit by $\mathbf{x}_\infty \in X$.

We shall use a terminology which is similar to the one introduced by Fletcher, classifying iterations as

(i) f-improving, if condition (14) holds;

(ii) h-improving, if the FRP was invoked and the resulting pair $(h_k,\ f_k)$ $\mathcal{F}_k$ is allowed to be included to $\mathcal{F}_k$ according to R2a.

We refrain from using the terms f-type and h-type iterations, because they have similar meaning, but details are slightly different (see Fletcher, 2010).

As in the above cited papers, it is useful to distinguish two cases, namely, the number of h-improving steps is infinite or finite.

Case 2a: Infinite number of steps which are only h-improving. Let us assume that the sequence $k(i)$, $i = 1, 2, \ldots$ contains an infinite subsequence of steps in which only h was improved and f was not. Abusing the notation, we denote this subsubsequence by $k'(i)$, $i = 1, 2, \ldots$. Clearly, $\mathbf{x}_{k'(i)} \to \mathbf{x}_\infty$.

We claim that

$$\lim_{i\to\infty} h(\mathbf{c}(\mathbf{x}_{k(i)})) = h(\mathbf{c}(\mathbf{x}_\infty)) = 0. \tag{26}$$

For the proof we can use directly arguments from Fletcher and Leyffer (2002, Lemma 3.3), because the proof is based solely on the rules which are used for handling the filter.

Our next step is to infer that $f(\mathbf{x}_\infty) = f_{\min}$. For deriving a contradiction, assume that $f(\mathbf{x}_\infty) > f_{\min}$.

From (26) we have $\mathbf{x}_\infty \in \mathcal{C}$. Thus, according to A2, for a certain $\mathbf{x}_{k'(i')}$, which is selected sufficiently close to $\mathbf{x}_\infty$, there exists $\hat{\mathbf{d}}_{k'(i')}$, say, such that $f(\mathbf{x}_{k'(i')} + \hat{\mathbf{d}}_{k'(i')}) < f(\mathbf{x}_{k'(i')}) - \gamma_{k'(i')}$. Thus, this step is f-improving, which is in contradiction to the fact our subsubsequence contains only steps that improve only h.

Case 2b: Finite number of only h-improving steps. We return to the subsequence selected at the beginning of our deliberations. Now, we assume that the sequence $k(i)$, $i = 1, 2, \ldots$ contains a finite subsequence of steps in which only h was improved and f was not. Let $I = k(i^*)$ be the first step starting from which h is no longer improved. It is convenient to renumber this subsequence in such a way that steps $k(i^* + 1)$, $k(i^* + 2), \ldots$ are now numbered as $I + 1$, $I + 2, \ldots$.

Firstly, we have to prove that also in this case $\mathbf{x}_\infty \in \mathcal{C}$. If it is not true, h-improving iterations would start when $\mathbf{x}_{I+m}$ is sufficiently close to $\mathbf{x}_\infty$ and ρ is reduced to a sufficiently small number, but this is in contradiction with our assumption that after I-iteration there are no h-improving iterations.

To prove that $\lim_{I\to\infty} f(\mathbf{x}_I) = f_{\min}$, let us note that all the steps I, $I + 1$, $\ldots$ must be f-improving and $f(\mathbf{x}_{I+1}) < f(\mathbf{x}_I) - \gamma_I$. Iterating this inequality we obtain

$$f_{I+m} < f_I - \sum_{m=I}^{I+m} \gamma_m. \tag{27}$$

Hence, we have a strictly monotone sequence f_{I+m}, $m = 1, 2, \ldots$, which is bounded from below by $f_{\min}$. Thus, this sequence is convergent and its limit equals $f_{\min}$. If it had a limit f_G, say, larger than $f_{\min}$, then—due to (27)—we obtain a contradiction, because for m sufficiently large the left hand side of (27) is smaller than f_G by (13).

Received: 28 March 2011
Revised: 19 August 2011