Oper Res Int J (2015) 15:163–198
DOI 10.1007/s12351-015-0171-3

ORIGINAL PAPER

# A generic column generation principle: derivation and convergence analysis

**Torbjörn Larsson · Athanasios Migdalas · Michael Patriksson**

Received: 5 February 2015 / Accepted: 12 February 2015 / Published online: 8 March 2015


**Abstract** Given a non-empty, compact and convex set, and an a priori defined condition which each element either satisfies or not, we want to find an element belonging to the former category. This is a fundamental problem of mathematical programming which encompasses nonlinear programs, variational inequalities, and saddle-point problems. We present a conceptual column generation scheme, which alternates between solving a restriction of the original problem and a column generation phase which is used to augment the restricted problems. We establish the general applicability of the conceptual method, as well as to the three problem classes mentioned. We also establish a version of the conceptual method in which the restricted and column generation problems are allowed to be solved approximately, and of a version allowing for the dropping of columns. We show that some solution methods (e.g., Dantzig–Wolfe decomposition and simplicial decomposition) are special instances, and present new convergent column generation methods in nonlinear programming, such as a sequential linear programming type method. Along the way, we also relate our quite general scheme in nonlinear programming presented in this paper with several other classic, and more recent, iterative methods in nonlinear optimization.

T. Larsson
Department of Mathematics, Linköping University, 581 83 Linköping, Sweden
e-mail: torbjorn.larsson@liu.se

A. Migdalas (✉)
Department of Business Administration, Technology and Social Sciences,
Luleå University of Technology, 971 87 Luleå, Sweden
e-mail: athmig@ltu.se

M. Patriksson
Department of Mathematical Sciences, Chalmers University of Technology,
412 96 Gothenburg, Sweden
e-mail: mipat@chalmers.se

**Keywords** Convex programming · Variational inequality problems · Saddle-point problems · Column generation · Simplicial decomposition · Dantzig–Wolfe decomposition · Sequential linear programming

**Mathematics Subject Classification** Primary 90C25 · Secondary 65K05 · 49J40

## 1 Introduction

This section formalizes the problem studied, provides sample instances along with the basic assumptions made, and discusses the background to the work behind this paper.

### 1.1 Statement of the problem

Let $X$ be a non-empty, compact and convex subset of $\mathbb{R}^n$. Suppose that each element in the set $X$ either satisfies or violates a well-defined condition, which is denoted by $\mathcal{C}(X)$ since it may in general depend on the set $X$. This condition might, for example, correspond to the requirement that an element in $X$ should be an optimizer of an objective function defined on $X$, solve a variational inequality over $X$, or satisfy a saddle-point defining inequality over $X$.

The *archetype problem* under consideration in this work is to

$$\text{find an } x \in X \text{ that satisfies the Condition } \mathcal{C}(X). \quad [P(X)]$$

The set $X$ is referred to as the *admissible set* of the problem $P(X)$, and the set of elements in $X$ satisfying the Condition $\mathcal{C}$(X), denoted $X^*$, is then referred to as the *solution set*. (Whenever an optimization problem is considered, the admissible set will be referred to as the *feasible set*.) Further, if a set $\widehat{X} \subset X$, that is, an inner approximation of the admissible set, is non-empty, compact and convex, then the problem $P(\widehat{X})$ is said to be a (*proper*) *restriction* of the problem $P(X)$.

The solution sets of the problem $P(X)$ and its proper restrictions $P(\widehat{X})$, $\widehat{X} \subset X$, are required to have the following property.

**Assumption 1** (*solution sets*). Whenever $\widehat{X} \subseteq X$ is non-empty, compact and convex, the solution set $\widehat{X}^*$ is non-empty and compact.

Assumption 1 implies, in particular, that $X^*$ is non-empty and compact. Some instances of the archetype problem $P(X)$ which fulfill Assumption 1 are given in the following examples.

*Example 1* (convex programming). Consider the class of convex programming problems of the form

$$\min_{x \in X} f(x), \quad [CP]$$

where $X \subset \mathbb{R}^n$ is a non-empty, compact and convex set, and $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and convex.[1] This problem can be stated as an instance of the archetype problem $P(X)$ as, for example, to find an $x^* \in X$ such that

$$-\nabla f(x^*) \in N_X(x^*), \tag{1}$$

where $N_X(x^*)$ denotes the normal cone to the set $X$ at the point $x^*$.

*Example 2* (variational inequality problem). The variational inequality problem (e.g., Harker and Pang 1990) is to find an $x^* \in X$ such that $-F(x^*) \in N_X(x^*)$, or, equivalently, such that

$$F(x^*)^{\mathrm{T}}(x - x^*) \geq 0, \qquad \forall x \in X, \quad [VIP]$$

where $F : X \to \mathbb{R}^n$ is continuous on the non-empty, compact and convex set $X \subset \mathbb{R}^n$.

*Example 3* (saddle-point problem). Given a non-empty, compact and convex Cartesian product set $X = X_1 \times X_2$, where $X_i \subset \mathbb{R}^{n_i}$, $i = 1, 2$, and a continuous saddle-function $L : X_1 \times X_2 \to \mathbb{R}$, a pair $(x_1^*, x_2^*) \in X_1 \times X_2$ is sought such that

$$L(x_1^*, x_2) \leq L(x_1^*, x_2^*) \leq L(x_1, x_2^*), \qquad \forall (x_1, x_2) \in X_1 \times X_2, \quad [SPP]$$

which describes the saddle-point problem (e.g., Danskin 1967; Rockafellar 1970; Dem'yanov and Malozemov 1974).

*Remark 1* (generality and flexibility of the archetype problem). (a) If the archetype problem $P(X)$ is an instance of a nonlinear program, then the Condition $\mathcal{C}(X)$ may describe *local* requirements on a point $x^* \in X$, such as a condition like (1), or it may describe *global* requirements such as

$$x^* \in \left\{ x \in X \,\middle|\, f(x) \leq \min_{z \in X} f(z) \right\}. \tag{2}$$

From these examples we may conclude that different formulations of the Condition $\mathcal{C}(X)$ (from the same original problem) will obviously then yield archetype problems that may differ substantially in their tractability. Further, the algorithm class to be devised and analyzed in this paper in order to find a point satisfying this condition includes iterative steps that will be more or less tractable and realizable.

(b) The versions of the archetype problem given above are of a *primal* nature, while they may also be chosen to be of a *dual* or *primal–dual* character, describing, for example, the KKT conditions of a nonlinear program or of a variational inequality. We may also envisage an archetype problem based on a reformulation of an objective function in terms of its epigraph, etc. The result of such reformulations is that the algorithm to be devised will yield different types of approximations of the original problem (such as outer approximations of constraints and/or the epigraph of the objective function), and also that the sequence

[1] The requirement that $f$ be convex is only made for reasons of simplicity of the presentation.

of iterates will be defined in different spaces of the problem formulation. We shall henceforth remark on opportunities along these lines for some instances of the archetype problem, but will concentrate on inner approximations of the admissible set in our basic convergence results.

(c) Further examples of problems which can be stated as instances of the archetype problem include Nash and Stackelberg equilibrium problems, and mathematical programs with equilibrium constraints (MPEC); see Luo et al. (1996). Further, the Condition $\mathcal{C}(X)$ also may describe requirements that are not of an optimality-describing character, but may describe requirements like complementarity, efficiency (in the context of multi-objective optimization), or feasibility with respect to side constraints, or a combination of different kinds of requirements.

We will also require the problem $P(X)$ to have a continuity property. Consider any sequence $\{X_k\}$ of non-empty, compact, convex and increasing subsets of $X$. The sequence then has a *set limit* (e.g., Mosco 1969; Salinetti and Wets 1979), say $\widetilde{X} \subseteq X$, which is also non-empty, compact and convex. Further, let, for all $k$, $x_k \in X_k^*$. Then, since, for all $k$, $x_k \in X_k$, it directly follows (e.g., Aubin and Frankowska 1990, Proposition 1.1.2) that any accumulation point $\widetilde{x}$ of the sequence $\{x_k\}$, belongs to $\widetilde{X}$. We impose the following *problem continuity* property.

**Assumption 2** (*problem continuity*). Let the sequence $\{X_k\}$ consist of non-empty, compact, convex and increasing subsets of $X$, and with the non-empty, compact and convex set limit $\widetilde{X} \subseteq X$. Let, for all $k$, $x_k \in X_k^*$, and suppose that $\widetilde{x}$ is an accumulation point of the sequence $\{x_k\}$. Then, $\widetilde{x} \in \widetilde{X}^*$.

The fulfillment of Assumption 2 for the Examples 1–3 discussed above will be verified in Sect. 4.

### 1.2 Background and motivation

The solution strategy known as the column generation principle—and its dual correspondence, constraint generation—is one of the standard tools of mathematical programming, and has since the pioneering work on the maximal multicommodity network flow problem by Ford and Fulkerson (1958) been developed and applied in a variety of contexts. Early, classical, applications of this strategy include economic lot sizing (Manne 1958), the cutting stock problem (Gilmore and Gomory 1961, 1963), and ship scheduling and routing (Appelgren 1969, 1971). The probably most widely known column generation method is the Dantzig–Wolfe decomposition method (Dantzig and Wolfe 1960, 1961) for block-angular linear programs. In all these applications of the general principle, the column generation is based on the pricing-out mechanism of the simplex method. Included in the class of column generation methods are also the inner linearization/restriction type algorithms defined by Geoffrion (1970); these include the simplicial decomposition method (Holloway 1974; Hohenbalken 1975, 1977) for nonlinear programming, in which the column generation is not based on any pricing mechanism, but on the solution of

a subproblem which is constructed through a linear approximation of the objective function of the nonlinear problem.

The origin of the work which led to the development of the *generic column generation principle*, to be presented in this paper, was the idea to generalize simplicial decomposition type methods to include nonlinear subproblems, the reason being that the use of subproblems arising from more accurate approximations of the original problem might enhance the overall performance of such methods. (Some numerical results for this generalization are presented in Larsson et al. (1996) as well as in García et al. (2011).) A further motivation for this work was also the interest to extend the previous analysis to cover also *non-differentiable* convex objective functions. The simplicial decomposition strategy is also a natural background to and setting for the generic column generation principle, and we are therefore inclined to proceed the introduction and the presentation from this angle of approach, even though the results obtained in this paper reach far beyond the original scope.

One origin of simplicial decomposition is the *Frank–Wolfe* algorithm (Frank and Wolfe 1956) for quadratic programs and its extension to general differentiable objective functions known as the *conditional gradient algorithm* (Pshenichny and Danilin 1978, Section III.3). Applied to the problem *CP* when $X$ is polyhedral, this method alternates between the solution of a feasible direction-finding subproblem, which is a linear program constructed through a first-order Taylor series expansion of $f$ at the current iterate, and a line search towards the solution to this linear program. The optima of the linear programs provide (convergent) lower bounds on the optimal value of the original problem, and are therefore useful to monitor the progress of the solution process. A further feature of the method is the ability to utilize problem structures, like feasible sets being Cartesian products or defining (generalized) network flows. Because of these nice features, the Frank–Wolfe method has reached some popularity, in particular within the field of traffic planning, although its convergence performance might be poor. (See, e.g., Canon and Cullum 1968; Wolfe 1970; Hearn et al. 1987; Patriksson 1994; Bar-Gera 2002; Boyce et al. 2004; Nie 2009; Bar-Gera 2010 on the latter issue.) There have been many suggestions for improvements, and among these we distinguish two important categories.

In a first category, one seeks to improve the quality of the search directions. An inherent weakness of the Frank–Wolfe method is that the search directions will eventually become arbitrarily poor, in the sense that they eventually become almost perpendicular to the direction of steepest descent (e.g., Wolfe 1970). This property is a direct consequence of the linearity of the direction-finding subproblem, and a natural strategy for improvements is therefore the introduction of a *nonlinearity* in this problem. Examples of this strategy are the constrained Newton method (e.g., Pshenichny and Danilin 1978) and the regularized Frank–Wolfe method of Migdalas (1994). (The latter method employs a direction-finding subproblem devised by augmenting the usual linear objective function with a term which has the effect of a trust region.) Similar strategies have, of course, been frequently used in other settings; an example of this is the auxiliary problem principle of Cohen (1980). The principle of using nonlinear direction-finding subproblems in descent

algorithms for nonlinear programs and variational inequality problems is in Patriksson (1998a, b, c, d) analyzed within the framework of the class of cost approximation algorithms, which includes all of the above as special cases.

In a second category of improvements of the Frank–Wolfe method, the line search is replaced by a multi-dimensional search. These *simplicial decomposition* algorithms are founded on *Carathéodory's Theorem* (e.g., Bazaraa et al. 2006, Theorem 2.1.6), which states that any point in the convex hull of an arbitrary subset, $S$, of $\mathbb{R}^n$ can be expressed as a convex combination of, at most, $1 + \dim S$ points in the set, where $\dim S$ refers to the dimension of the affine hull of $S$. (The convexity weights are sometimes referred to as *barycentric coordinates*.) A consequence of this result is that any feasible solution to an optimization problem with a bounded and polyhedral feasible set can be represented as a convex combination of the extreme points of the feasible polyhedron. This fact is exploited in the simplicial decomposition algorithms, which alternate between a *master problem*, which is the restriction of the original program to an inner approximation of the feasible set, defined by a restricted set of extreme points, and of the solution of the linear program of the Frank–Wolfe method.

Consider a convex program of the form *CP* and with a polyhedral feasible set. Given a feasible iterate, $x^{k-1}$ ($k \geq 1$), and $k$ stored extreme points, $y^i$, $i = 1, \ldots, k$, the master problem is given by

$$\begin{aligned} \min \quad & f\left(x^{k-1} + \sum_{i=0}^{k} \lambda_i \left(y^i - x^{k-1}\right)\right) \\ \text{s.t.} \quad & \sum_{i=0}^{k} \lambda_i \leq 1, \\ & \lambda_i \geq 0, \quad i = 0, \cdots, k, \end{aligned}$$

with $y^0 = x^0$. This problem provides the new iterate, $x^k$, and an upper bound on the optimal value of the nonlinear program. It obviously generalizes the line search of the Frank–Wolfe method, but its special constraint structure enables its solution by efficient specialized methods, as long as the number of columns retained is relatively low. The generation of a new column (i.e., an extreme point of the feasible polyhedron) to be included in the master problem is made through the solution of the linear programming subproblem of the Frank–Wolfe method, that is,

$$\min_{y \in X} \nabla f(x^k)^{\mathrm{T}} y.$$

The simplicial decomposition strategy has been applied mainly to certain classes of structured linearly constrained convex programs, and it has then shown to be successful. Especially, for nonlinear network flow problems, the simplicial decomposition methods have shown to be efficient computational tools (e.g., Hearn et al. 1987; Mulvey et al. 1990; Larsson and Patriksson 1992).

von Hohenbalken (1977), who gave the method its name and gave its first complete description, shows that the convergence of the simplicial decomposition algorithm is finite in the number of master problems even if extreme points with

zero weights are removed from one master problem to the next.[2] This convergence result allows for the use of column dropping (that is, the elimination of columns that have received a zero weight in the solution to a master problem, see Murphy 1973a, b), which is essential to gain computational efficiency in large-scale applications. In fact, by applying Carathéodory's theorem to the optimal face, $F^*$, of the feasible set, the number of extreme points needed to express any optimal solution is bounded from above by $1 + \dim F^*$. This fact is exploited in the restricted simplicial decomposition algorithm, devised by Hearn et al. (1985), in which the number of stored extreme points is bounded by a number, $r$; the convergence remains finite provided that $r \geq 1 + \dim F^*$.

The remaining discussion in this section on extensions of the simplicial decomposition algorithm is pertinent to those of the algorithm to be presented and analyzed in this paper.

The basic works described so far have, of course, been extended in various directions. Larsson and Patriksson (1992) extend the simplicial decomposition strategy to take full advantage of *Cartesian product* structures, resulting in the disaggregate simplicial decomposition (DSD) algorithm. Ventura and Hearn (1993) extend the restricted simplicial decomposition method to convexly constrained problems, and Feng and Li (2001) analyze the effect of approximating the restricted master problem by a quadratic one. In Lawphongpanich and Hearn (1984) the simplicial decomposition strategy is applied to a variational inequality formulation of the traffic equilibrium problem. The latter algorithm includes a column dropping scheme which is governed by the primal gap function (e.g., Hearn et al. 1984).

Simplicial decomposition may also be based on the pricing-out of a subset of the (linear) constraints. Identifying a subset of the constraints defining $X$ as complicating, these may be priced-out (that is, Lagrangian relaxed) in the column generation subproblem, and instead included in the master problem, just as in Dantzig–Wolfe decomposition for linear and non-linear programming problems; see Marín (1995), Stefek (1989). It should be noted, however, that just as in the original (primal) simplicial decomposition method, the column generation subproblems in these methods are based on the linearization of the original objective function, and are therefore linear programs, and their master problems are non-linear; this is precisely the opposite to the case of non-linear Dantzig–Wolfe decomposition (e.g., Lasdon 1970).

As noted earlier, in the Frank–Wolfe method, and therefore also in simplicial decomposition methods, the direction towards the latest generated extreme point might be arbitrarily close to being perpendicular to the direction of steepest descent, and there is therefore no guarantee that the inclusion of this extreme point in the master problem leads to any significant improvement in the objective value. These methods might therefore actually suffer from a weakness that is similar to that of the Frank–Wolfe method, and such a behaviour has indeed been observed in applications to some large-scale traffic equilibrium models (Hearn et al. 1987; Larsson and Patriksson 1992). As for the Frank–Wolfe method, there might also be

[2] As Higgins and Polak (1990) have pointed out, von Hohenbalken's version of the algorithm is guaranteed convergent for polyhedral feasible sets only.

a potential for enhancements of simplicial decomposition methods through the introduction of a nonlinear direction-finding subproblem. This was the original motivation for the work in Larsson et al. (1996) and the present one.

### 1.3 Preview

We present a conceptual column generation scheme for an archetype problem which encompasses a wide variety of problem classes from the field of mathematical programming. The admissible set of the archetype problem is required to be compact and convex, and its solution set is characterized by an a priori specified condition. The column generation problem of the generic scheme is not required to have any particular form; when applied to linearly constrained nonlinear programs, the generic column generation principle thus allows combinations of the two strategies for enhancements of the Frank–Wolfe method discussed above. The main contribution of our work is that the generic column generation principle provides a theoretical foundation for the introduction of multi-dimensional searches, instead of the traditional line searches, in a variety of existing solution methods for nonlinear programs, variational inequality problems, etc., while also suggesting new and interesting methodologies. We believe that this strategy will be computationally beneficial if the inner approximated problem is much more easily solved than is the original one.

The outline of the remainder of the paper is as follows. In the next section we introduce the general column generation problem of the conceptual scheme, and in the section that follows, we state the scheme formally and give the basic convergence theorem. In Sect. 4, it is shown that convergence is ensured when the algorithm is applied to nonlinear programs, variational inequalities and saddle-point problems. Next, we present a version of the algorithm in which both master and column generation problems are solved inexactly; we also extend the basic convergence result to this *truncated* version of the algorithm. In the same section, we also introduce a very general rule for updating the restriction from one iteration to the next, which will in particular allow for the construction of column dropping rules. Also this version is theoretically validated. Then in Sect. 6 we establish that the Dantzig–Wolfe decomposition method is a special case of the generic column generation principle, by applying the method to the primal–dual saddle-point problem arising from the original linear program. We also introduce a sample of new algorithms that may be derived from the use of the column generation principle: a sequential linear programming (SLP) method, a simplicial decomposition algorithm for constrained non-smooth optimization, and a Newton method for variational inequalities with multi-dimensional searches, and also briefly suggest some others. The paper is concluded with some suggestions for further work and applications.

## 2 Requirements on the column generation problem

We assume that we have at hand a principle for constructing a *column generation problem*, that is, an auxiliary problem which is used to guide the search for a

solution to the original problem, $P(X)$. The column generation problem constructed for some $x \in X$ and its solution set are denoted $CG(x)$ and $Y(x) \subseteq X$, respectively.

**Assumption 3** (*solution set of* $CG(x)$). For any $x \in X$, the solution set $Y(x) \subseteq X$ is non-empty.

We further suppose that the mapping associated with the column generation problem is *closed*, in the sense defined by Zangwill (1969, Chapter 4).

**Assumption 4** (*closedness of* $Y(x)$). The principle for constructing the column generation problem results in a point–to–set solution set mapping $Y : X \to 2^X$ which is *closed* on $X$ [i.e., if, for all $k$, $x^k \in X$, $\{x^k\} \to \widetilde{x}$, and if, for all $k$, $y^k \in Y(x^k)$, and $\{y^k\} \to \widetilde{y}$ holds, then $\widetilde{y} \in Y(\widetilde{x})$].

We further assume that the column generation problem is devised so that it can be used to establish if a solution to the original problem has been found. That is, we assume that the column generation problem has the following *fixed point property*.

**Assumption 5** (*fixed-point property of* $Y(x)$). Let $x \in X$. Then $Y(x) \ni x \Longleftrightarrow x \in X^*$.

We note in reference to the convergence results to follow, that the implication in the *right* direction is the only one actually needed, although the instances of column generation subproblems discussed in the paper satisfy Assumption 5.

Further, whenever a solution is not at hand, the column generation problems shall have the following *set augmentation property*.

**Assumption 6** (*set augmentation property*). Let the set $\widehat{X} \subset X$ be non-empty, compact and convex, and let $\widehat{x}^* \in \widehat{X}^*$. If $\widehat{x}^* \notin Y(\widehat{x}^*)$, then $Y(\widehat{x}^*) \subseteq X \setminus \widehat{X}$.

Hence, when the solution to the proper restriction $P(\widehat{X})$ does not solve its resulting column generation problem, then the column generation problem will supply candidate solutions (i.e., columns) to the original problem that are *strictly separated* from the restricted admissible set, that is, which have not already been considered; algorithmically, any column found by the column generation problem can thus be used to augment the set $\widehat{X}$, that is, improve the inner approximation of the original admissible set. (In this sense, the column generation problem may be viewed as a separation oracle.)

*Example 4* (algorithmic maps fulfilling Assumptions 3–6). An example of a column generation problem that fulfills the Assumptions 3–6 is the linear programming subproblem of the simplicial decomposition algorithm, as applied to linearly constrained convex programs; the column generation problem $CG(x)$ may thus be regarded as a generalization of that subproblem. The column generation problem arising in the classical column generation algorithm in linear programming (e.g., Lasdon 1970, Chapters 3–4), which in turn includes that of Dantzig and Wolfe (1960) and the cutting stock application in Gilmore and Gomory (1961, 1963), also satisfies Assumptions 3–6.

## 3 Conceptual algorithm

### 3.1 The algorithm and its convergence

The conceptual column generation algorithm for the solution of the archetype problem $P(X)$ is stated in Table 1.

Note that Steps 1, 2, and 4 of the algorithm are well-defined thanks to Assumption 5, Assumptions 3 and 6, and Assumption 1, respectively. (In realizations of the algorithm the Steps 1 and 2 are often naturally integrated.) Assumptions 2 and 4 are needed to invoke in order to make the algorithm convergent.

*Remark 2* (initiation). The initiation that is stated in the algorithm may, of course, be replaced by an advanced start, which amounts to choosing a non-trivial, closed and convex set $X^0 \subseteq X$ and finding $x^0$ as a solution to the restricted problem $P(X^0)$. Notice also that the restricted admissible sets $X^k$, $k = 0, 1, \ldots$, do not need to be polyhedral.

To state the main convergence result, let $d_{X^*}(x)$ denote the Euclidean distance between some $x \in X$ and the solution set $X^*$.

**Theorem 1** (convergence of the conceptual algorithm). *Suppose that Assumptions 1 through 6 hold. If the sequence $\{x^k\}$ of iterates generated by the algorithm is finite, then it has been established that the last iterate solves the problem $P(X)$. If the sequence of iterates is infinite, then*

$$\left\{d_{X^*}(x^k)\right\} \to 0.$$

*Proof* If the sequence of iterates is finite, then the algorithm has been interrupted in Step 1, and the conclusion follows from Assumption 5. In the remainder of the proof, we thus assume that the sequence of iterates is infinite.

The sequence $\{X^k\}$ (which exists thanks to Assumption 1) consists of non-empty, compact, convex and increasing sets. Thus, it has a set limit (e.g., Mosco 1969; Salinetti and Wets 1979), say $\widetilde{X} \subseteq X$, which is also non-empty, compact and convex.

Let $\varepsilon \geq 0$ be such that the sequence of iterates (which exists by Assumption 1) contains an (infinite) subsequence, say $\{x^k\}_{k \in \mathcal{K}}$, where $\mathcal{K} \subseteq \mathcal{N} := \{0, 1, \ldots\}$, with

**Table 1** Conceptual column generation algorithm

0 (*initiation*): Find an $x^0 \in X$, let $X^0 = \{x^0\}$ and $k = 0$.

1 (*termination check*): If the iterate $x^k$ solves the column generation problem $CG(x^k)$, then terminate with the same conclusion for the original problem $P(X)$.

2 (*column generation*): Find $y^{k+1}$ as a solution to the column generation problem $CG(x^k)$, and construct a closed and convex set $X^{k+1} \subseteq X$ such that $y^{k+1} \in X^{k+1} \supset X^k$.

3 (*iteration*): Let $k := k + 1$.

4 (*solution of restriction*): Find a new iterate $x^k$ as a solution to the restricted problem $P(X^k)$.

5 (*repetition*): Return to Step 1.

$d_{X^*}(x^k) \geq \varepsilon$ for all $k \in \mathcal{K}$. Since the sequence $\{x^k\}_{k\in\mathcal{K}}$ belongs to the compact set $X$, it has at least one accumulation point, say $\widetilde{x} \in X$, which is the limit point of a convergent subsequence, say $\{x^k\}_{k\in\widetilde{\mathcal{K}}}$, where $\widetilde{\mathcal{K}} \subseteq \mathcal{K}$. Then, from Assumption 2, $\widetilde{x} \in \widetilde{X}^*$.

Since the sequence $\{y^{k+1}\}_{k\in\widetilde{\mathcal{K}}}$ (which exists by Assumption 3) belongs to the compact set $X$, it has at least one accumulation point, say $\widetilde{y} \in X$. From the closedness of the column generation mapping (i.e., Assumption 4), it follows that $\widetilde{y} \in Y(\widetilde{x})$. Since $y^{k+1} \in X^{k+1}$ for all $k \in \widetilde{\mathcal{K}}$, it follows (e.g., Aubin and Frankowska 1990, Proposition 1.1.2) that $\widetilde{y} \in \widetilde{X}$. From Assumption 6 follows that $\widetilde{x} \in Y(\widetilde{x})$, and from Assumption 5 then follows that $\widetilde{x} \in X^*$.

Hence, $\varepsilon = 0$, and the result of the theorem follows. □

The traditional simplicial decomposition method is clearly a special case of the generic scheme when a linearly constrained nonlinear optimization problem (with a bounded feasible set) is to be solved, and set augmentation is made through the exact solution of a linearized version of the given problem.

### 3.2 Remarks

The restricted problem $P(X^k)$ is preferably dealt with in actual realizations as follows. Assume that the iterate $x^{k-1}$ is given and that the current inner approximation of the admissible set is given by $X^k = \text{conv}\,\{p^0, \ldots, p^k\}$, where $p^0 = x^0$ and the points $p^i \in X$, $i = 1, \ldots, k$, have been generated through some set augmentation principle. (One such principle is given below.) Introducing the $(k+1)$-dimensional unit simplex

$$S^{k+1} := \left\{ \lambda \in \mathbb{R}^{k+1} \,\middle|\, \sum_{i=0}^{k} \lambda_i \leq 1, \quad \lambda_i \geq 0, \quad i = 0, \ldots, k \right\}$$

and the admissible point

$$x(\lambda) := x^{k-1} + \sum_{i=0}^{k} \lambda_i \left(p^i - x^{k-1}\right), \qquad \lambda \in S^{k+1},$$

we obtain

$$X^k = x\left(S^{k+1}\right) := \left\{ x(\lambda) \,\middle|\, \lambda \in S^{k+1} \right\},$$

so that the restriction $P(X^k)$ can preferably be stated and solved as

$$\text{find a } \lambda \in S^{k+1} \text{ such that } x(\lambda) \text{ satisfies the Condition } \mathcal{C}\left(x\left(S^{k+1}\right)\right). \quad [P(S^{k+1})]$$

This equivalent problem, or (*restricted*) *master problem*, might in practice be significantly less expensive to solve because of the simplicity of its admissible set. (Note that the dependency between the convexity variables and the original

variables should be handled implicitly.) Furthermore, it is often natural and beneficial to re-optimize the master problem (from $0 \in S^{k+1}$).

We next discuss some possible relaxations of the requirements for convergence of the algorithm, and some generalizations of it.

### *3.2.1 On the boundedness assumption on X*

The boundedness requirement on the set $X$ can be replaced by any other assumption that implies that the sequences $\{x^k\}$ and $\{y^k\}$ both are bounded. If, for example, the optimal solution set is bounded and if the generic scheme is implemented so that the sequence $\{f(x^k)\}$ of objective values is descending, then since the (closed) lower level set of the objective function (restricted to $X$) corresponding to the initial iterate, $L_X^f(x^0)$, is bounded (e.g., Rockafellar 1970, Corollary 8.7.1), it follows that the sequence $\{x^k\}$ is bounded. If, further, the column generation problem is designed so that the solution set $Y(x)$ is non-empty and compact for any $x \in L_X^f(x^0)$ and the point–to–set solution set mapping $Y : X \to 2^X$ is upper semi-continuous on the compact set $L_X^f(x^0)$, then the set $Y(L_X^f(x^0)) := \cup_{x \in L_X^f(x^0)} Y(x)$ is also compact (e.g., Nikaido 1968, Lemma 4.5). Since, for all $k$, $y^k \in Y(x^k)$, and $\{x^k\} \subseteq L_X^f(x^0)$, it then follows that the sequence $\{y^k\} \subseteq Y(L_X^f(x^0))$ is bounded.

Other means to obtain a bounded (working) admissible set include the addition of redundant constraints. The algorithm may also be combined with trust region methods (e.g., Conn et al. 2000; Bazaraa et al. 2006, Section 8.7), although the convergence properties of such a method remains to be analyzed.

Another alternative is to extend the method described in Table 1 so that it deals explicitly with the possibility of the column generation problem having an unbounded solution. In such a case, Step 2 would generate a direction in the recession cone of $X$, and the inner approximation of $X$ utilized in Step 4 then describes the sum of the convex hull and cone of the columns and directions generated, respectively, so-far in the course of the algorithm.

### *3.2.2 On the closedness of the column generation step*

Consider the possibility of applying a column generation principle described by some mapping $x \mapsto Y(x)$ which we cannot establish to be closed (so, Assumption 4 is violated), but which satisfies Assumption 6. Provided that provably convergence-inducing mappings are applied an *infinite* number of times in any infinite sequence of iterations, the resulting sequence of iterates can be shown to still satisfy the conclusions of Theorem 1, since the property $X^{k+1} \supset X^k$ still holds for all $k \geq 1$. [See further the discussion in Remark 8 on spacer steps, for the case where a merit function exists for the problem $P(X)$.] Such a column generation mapping could, for example, be the result of the application of some heuristic procedure which is of interest to invoke occasionally.

### 3.2.3 *On the realization of the set augmentation step*

The augmentation of the inner approximation of the admissible set might be implemented in various ways; a proper generalization of the principle used in simplicial decomposition would be to choose a $t_k \geq 1$ such that $x^k + t_k(y^{k+1} - x^k) \in X$, and set $X^{k+1} = \text{conv}\,(X^k \cup \{x^k + t_k(y^{k+1} - x^k)\})$, where conv is the convexification operator. The choice $t_k = \max\{\, t \mid x^k + t(y^{k+1} - x^k) \in X \,\}$, which gives the largest augmentation, has in Larsson et al. (1996) been observed to yield a substantial improvement over the choice $t_k = 1$ for some cases of nonlinear column generation problems $CG(x)$.

The column generation step of the algorithm may also be replaced by a *closed set augmentation mapping*, that is, a closed mapping $M : 2^X \to 2^{2^X}$, with the property that for any non-empty, compact and convex set $V \subseteq X$ it holds that any set $W \in M(V)$ is also non-empty, compact and convex, and fulfill that $V \subseteq W \subseteq X$. Then the set augmentation step of the algorithm is to let $X^{k+1} \in M(X^k)$, of which the given, composite, set augmentation step is a special case. Convergence is guaranteed if for any non-empty, compact and convex set $V \subseteq X$ it holds that (cf. Assumptions 5 and 6)

$$M(V) \ni V \Longrightarrow V \cap X^* \neq \emptyset.$$

We have chosen to consider set augmentation through the *exact* solution of a column generation problem, since it is from a conceptual point of view a natural way to implement this step. Another natural way to obtain a closed set augmentation mapping is through the *approximate* solution of a column generation problem, with a closed solution set mapping, using a solution method with a closed algorithmic map.

We finally remark that to the algorithm described in Table 1 there is a corresponding *dual* methodology, where inner representation is replaced by outer representation, and column generation is replaced by constraint generation. A large class of such methods is in fact established automatically through the convergence analysis performed in this paper (as, for example, Benders decomposition of a linear program is equivalent to Dantzig–Wolfe decomposition of its dual). An example of the constraint generation methods that can be derived through our framework is given in Sect. 6.2.1. (See also the discussion in Remark 1.)

## 4 Realizations of sufficient conditions

We will in this section give a simple realization of Assumption 2 by means of a *merit function*, and show that this realization is readily applicable to three familiar problem classes. It is further shown that Assumptions 5 and 6 can be realized by the same means.

**Lemma 1** (a sufficient condition for problem continuity). *Suppose that there exists a function $\psi : 2^X \times X \to \mathbb{R}$ with the following properties.*

(a) (continuity) *Let the sequence* $\{X_k\}$ *consist of non-empty, compact, convex and increasing subsets of* $X$, *with set limit* $\widetilde{X} \subseteq X$. *Further, consider a convergent sequence* $\{x_k\}$, *where, for all* $k$, $x_k \in X_k$, *with limit* $\widetilde{x}$. *Then,*

$$\{\psi(X_k, x_k)\} \to \psi(\widetilde{X}, \widetilde{x}).$$

(b) (merit property) *Let* $\widehat{X} \subseteq X$ *be non-empty, compact and convex. Then,*

$$\widehat{X}^* = \arg\min_{y \in \widehat{X}} \psi(\widehat{X}, y).$$

*Then Assumption 2 is fulfilled.*

*Proof* Consider a sequence $\{x_k\}$ where, for all $k$, $x_k \in X_k^*$, and let $\widetilde{x}$ be one of its accumulation points, which is then the limit of a convergent subsequence, say $\{x_k\}_{k \in \mathcal{K}}$, where $\mathcal{K} \subseteq \mathcal{N}$. Consider some arbitrary $\widetilde{y} \in \widetilde{X}$. Since $\widetilde{X}$ is the set limit of the sequence $\{X_k\}$, $\widetilde{y}$ is then (e.g., Aubin and Frankowska 1990, Proposition 1.1.2) the limit of a convergent sequence $\{y_k\}$, where, for all $k$, $y_k \in X_k$. From the merit property, we have that, for all $k$, $\psi(X_k, y_k) \geq \psi(X_k, x_k)$. Taking the limit corresponding to the subsequence $\mathcal{K}$, using that $\{X_k\}_{k \in \mathcal{K}} \to \widetilde{X}$, $\{y_k\}_{k \in \mathcal{K}} \to \widetilde{y}$, and the continuity property, we obtain that $\psi(\widetilde{X}, \widetilde{y}) \geq \psi(\widetilde{X}, \widetilde{x})$. Finally, by recalling that $\widetilde{y} \in \widetilde{X}$ is arbitrary and again invoking the merit property, the result of the lemma follows. □

The continuity property stated in this lemma holds in particular for any function $\psi$ that is continuous on an open neighbourhood of $2^X \times X$.

*Example 5* (convex programming (continued)). To show that Assumption 2 is fulfilled for the nonlinear program *CP*, we invoke Lemma 1 with the choice

$$\psi(\widehat{X}, x) := f(x), \qquad x \in \widehat{X} \subseteq X,$$

whose fulfillment of the continuity and merit properties is obvious.

*Example 6* (variational inequality problem (continued)). For the variational inequality problem *VIP*, Lemma 1 can be invoked by choosing the function $\psi$ as, for example, the *primal gap function* (e.g., Auslender 1976; Hearn et al. 1984; Larsson and Patriksson 1994), that is,

$$\psi(\widehat{X}, x) := \max_{y \in \widehat{X}} F(x)^{\mathrm{T}}(x - y), \qquad x \in \widehat{X} \subseteq X. \tag{3}$$

The continuity property required follows from the compactness of $X$. Further, it has the merit property since it can be shown that, for all $\widehat{X} \subseteq X$, $\psi(\widehat{X}, x) \geq 0$ for all $x \in \widehat{X}$, and $\psi(\widehat{X}, x) = 0$ if and only if $x \in \widehat{X}^*$. Assumption 2 is thus fulfilled for the problem *VIP*.

*Example 7* (saddle-point problem (continued)). To show that Assumption 2 is fulfilled for the saddle-point problem *SPP*, we may choose

$$\psi(\widehat{X},x) := \max_{y_2\in\widehat{X}_2} L(x_1,y_2) - \min_{y_1\in\widehat{X}_1} L(y_1,x_2), \qquad x=(x_1,x_2)\in\widehat{X}_1\times\widehat{X}_2=\widehat{X}\subseteq X. \tag{4}$$

The function $\psi$ is continuous in the sense of Lemma 1, and the merit property follows from that, for all $\widehat{X}\subseteq X$, $\psi(\widehat{X},x)\geq 0$ for all $x\in\widehat{X}$, and $\psi(\widehat{X},x)=0$ if and only if $x\in\widehat{X}^*$.

When there exists a continuous merit function $\psi$, it may also be used to establish that a column generation problem has the fixed point and set augmentation properties, as defined in Assumptions 5 and 6.

**Lemma 2** (a sufficient condition for the fixed point property). *Suppose that there exists a function $\psi: 2^X\times X\to\mathbb{R}$ having the properties that are stated in Lemma 1, and the following additional ones.*

(a) (descent at non-solutions) *If $x\notin Y(x)$, then for all $y\in Y(x)$ and all sufficiently small $t\in(0,1]$,*

$$\psi(X,x+t(y-x))<\psi(X,x).$$

(b) (non-descent at solutions) *If $x\in Y(x)$, then*

$$\psi(X,y)\geq\psi(X,x), \qquad \forall y\in X.$$

*Then Assumption 5 is fulfilled.*

*Proof* Immediate from the merit property of the function $\psi$. □

*Remark 3* (fixed point property). From the descent property of the lemma it directly follows that if for some $y\in Y(x)$, $\psi(X,x+t(y-x))\geq\psi(X,x)$ holds for arbitrarily small $t>0$, then $Y(x)\ni x$. Hence, an admissible point then solves its corresponding column generation problem (and, consequently, the original one) if *any* of its solutions (e.g., the one produced by some algorithm) *does not* provide descent with respect to the merit function $\psi(X,\cdot)$.

**Lemma 3** (a sufficient condition for the set augmentation property). *Suppose that there exists a function $\psi: 2^X\times X\to\mathbb{R}$ having the properties that are stated in Lemma 1, and the additional property that for all non-empty, compact and convex sets $\widehat{X}\subset X$, any point $\widehat{x}^*\in\widehat{X}^*$ such that $\widehat{x}^*\notin Y(\widehat{x}^*)$, and any $y\in Y(\widehat{x}^*)$, it holds that the merit function $\psi(\widehat{X},\cdot)$ is descending in the direction of $y-\widehat{x}^*$ from the point $\widehat{x}^*$. Then Assumption 6 is fulfilled.*

*Proof* If Assumption 6 is not fulfilled, then there exists a non-empty, compact and convex set $\widehat{X}\subset X$, a point $\widehat{x}^*\in\widehat{X}^*$ such that $\widehat{x}^*\notin Y(\widehat{x}^*)$, and a $y\in Y(\widehat{x}^*)$ such that

$y \in \widehat{X}$. Then, for all $t \in (0, 1]$, $\widehat{x}^* + t(y - \widehat{x}^*) \in \widehat{X}$, and from the merit property of the function $\psi$ it follows that, for all $t \in (0, 1]$,

$$\psi\Big(\widehat{X}, \widehat{x}^* + t(y - \widehat{x}^*)\Big) \geq \psi\Big(\widehat{X}, \widehat{x}^*\Big),$$

which contradicts the descent assumption, and the result follows. □

*Example 8* (convex programming (continued)). With the choice $\psi := f$ for the nonlinear program *CP*, the descent property in Lemma 2 reduces to the well-known descent condition

$$\nabla f(x)^{\mathrm{T}}(y - x) < 0, \qquad y \in Y(x), \qquad x \in X.$$

This condition, as well as the non-descent property in Lemma 2, the set augmentation property in Lemma 3, and Assumptions 3 and 4, are typically fulfilled by iterative descent methods for the problem *CP*, such as the Frank–Wolfe, constrained Newton (e.g., Pshenichny and Danilin 1978), gradient projection (Goldstein 1964; Levitin and Polyak 1966), and proximal point (e.g., Rockafellar 1976) methods. [See Patriksson (1998d) for many further examples.] The direction-finding subproblem of any of these methods may thus be used as a column generation problem in the generic scheme, as applied to the problem *CP*. In particular, the traditional simplicial decomposition scheme (for the special case of *CP* where the feasible set, $X$, is polyhedral) might therefore be generalized through the use any of the above-mentioned direction-finding strategies, while still being convergent.

*Example 9* (variational inequality problem (continued)). As has been surveyed, for example, in Larsson and Patriksson (1994) and Patriksson (1998d), there exist several column generation problems and merit functions that, in combination, satisfy the conditions of Lemma 2. Consider, for example, the column generation problem which, given an $x \in X$ and under the assumption that $F$ is continuously differentiable on $X$, defines $Y(x)$ to be the set of vectors $y \in X$ satisfying the linearized problem

$$[F(x) + \nabla F(x)^{\mathrm{T}}(y - x)]^{\mathrm{T}}(y - z) \geq 0, \qquad \forall z \in X. \tag{5}$$

If $F$ is monotone on $X$, that is, if,

$$[F(x) - F(y)]^{\mathrm{T}}(x - y) \geq 0, \qquad \forall x, y \in X,$$

then a result of Marcotte and Dussault (1989) is that the direction of $y - x$ defines a direction of descent with respect to the primal gap function (3) if and only if $X$ is not a solution to *VIP*. Its application in the context of the present algorithmic framework yields a Newton method with multi-dimensional searches for the problem *VIP*, and it is easily verified from the above that the conditions of Lemma 2 are satisfied. (See Sect. 6.2.3 for further discussions on this class of algorithms.) The references Larsson and Patriksson (1994), Patriksson (1998d) contain many other examples of column generation problems which yield descent directions for differentiable merit functions for the problem *VIP*, under additional assumptions on $F$ (such as strict or strong monotonicity on $X$) and on the column generation problem itself.

*Example 10* (saddle-point problem (continued)). Assume that the saddle-function $L$ is *strictly* convex–concave on $X$. Then, it is easily established (e.g., Dem'yanov and Malozemov 1974) that the vector $(y_1, y_2)$ which defines the value of the merit function (4) previously defined for the problem *SPP*, defines a direction of descent, $y - x$, for the said function. In the absence of strict convex–concavity, in some instance of the saddle-point problem the merit function may be augmented with strictly convex–concave terms in order to retain the descent property, for example as follows:

$$\psi(\widehat{X}, x) := \max_{y_2 \in \widehat{X}_2} \left[ L(x_1, y_2) - \frac{1}{2}\|y_2 - x_2\|^2 \right] - \min_{y_1 \in \widehat{X}_1} \left[ L(y_1, x_2) + \frac{1}{2}\|y_1 - x_1\|^2 \right],$$
$$x \in \widehat{X} \subseteq X.$$

This merit function, which is differentiable on $X$, can be shown (cf. Patriksson 1998d, Section 8.8) to satisfy the conditions of Lemmas 2 and 3 whenever $L$ is extended linear–quadratic (i.e., of the form $L(x, y) := c^T x + \frac{1}{2} x^T C x - b^T y - \frac{1}{2} y^T B y - x^T Q y$ for vectors and matrices of the appropriate dimensions) and convex–concave on $X$.

The above examples suggest that it may be a natural strategy to first find a suitable merit function for the problem class under consideration and then devise a column generation problem which is related to the merit function in accordance with the above lemmas.

## 5 A truncated version of the algorithm

### 5.1 Additional requirements

In the truncated version of the generic column generation algorithm, the restricted and column generation problems are allowed to be solved only approximately, using iterative algorithms, denoted $\mathcal{A}_r$ and $\mathcal{A}_c$, respectively, with closed iteration maps (e.g., Zangwill 1969, Chapter 4). The convergence analysis relies on the existence of merit functions both for the restricted and column generation problems:

**Assumption 7** (*merit function for* $P(X)$). There exists a merit function, $\psi$, having the properties that are stated in Lemma 1.

**Assumption 8** (*merit function for* $CG(x)$). There exists a continuous merit function $\Pi : X \times X \to \mathbb{R}$ for the column generation problem $CG(\cdot)$, that is, for any $x \in X$,

$$Y(x) = \arg\min_{z \in X} \Pi(x, z),$$

with the additional property that, for any non-empty, compact and convex set $\widehat{X} \subseteq X$ and any $\widehat{x}^* \in \widehat{X}^*$,

$$\widehat{X}^* \subseteq \arg\min_{y \in \widehat{X}} \Pi(\widehat{x}^*, y).$$

Assumption 7 implies the fulfillment of Assumption 2 by Lemma 1, and Assumption 8 implies the fulfillment of Assumption 4.

*Remark 4* From Assumptions 5 and 8 follow that

$$\Pi(x, x) = \min_{y \in X} \Pi(x, y) \Longleftrightarrow x \in X^*.$$

This result is analogous to that discussed in Remark 3, and provides an efficient termination criterion for use in Step 1.

**Assumption 9** (*algorithm for* $P(\widehat{X})$). As applied to a restriction $P(\widehat{X})$, where $\widehat{X} \subseteq X$ is non-empty, compact and convex, the algorithm $\mathcal{A}_r$ has a closed iteration map and all its iterates belong to the set $\widehat{X}$. Further, for any such restriction and any point in the restricted admissible set, one iteration of the algorithm gives descent with respect to the merit function $\psi(\widehat{X}, \cdot)$, unless a solution to the restriction is at hand.

**Assumption 10** (*algorithm for* $CG(x)$). As applied to a column generation problem $CG(x)$, where $x \in X$, the algorithm $\mathcal{A}_c$ has a closed iteration map and all its iterates belong to the set $X$. Further, for any point in the admissible set, one iteration of the algorithm applied to the problem $CG(x)$ gives descent with respect to the merit function $\Pi(x, \cdot)$, unless a solution to $CG(x)$ is at hand.

Observe that the algorithms $\mathcal{A}_r$ and $\mathcal{A}_c$ both are equipped with sufficient properties for being (asymptotically) convergent for the problems $\min_{x \in \widehat{X}} \psi(\widehat{X}, x)$ and $\min_{y \in X} \Pi(x, y)$, respectively, by Zangwill's Theorem A (Zangwill 1969, p. 239).

Assumptions 6, 8, and 10 together imply that a set augmentation will take place whenever the restricted problem has been solved to a sufficient accuracy, unless the restricted admissible set contains a solution to the original problem. In an application to a nonlinear program of the form *CP*, the merit function, $\Pi$, for the column generation problem is often given directly through the choice of (typically linear or quadratic) approximation of $f$, as in the Frank–Wolfe, Newton and gradient projection methods.

### 5.2 The truncated algorithm and its convergence

The truncated version of the conceptual column generation algorithm for the solution of the archetype problem $P(X)$ is stated in Table 2.

**Theorem 2** (convergence of the truncated algorithm). *Suppose that Assumptions 1 through 10 hold. If the sequence* $\{x^k\}$ *of iterates generated by the truncated algorithm is finite, then it has been established that the last iterate solves the problem* $P(X)$. *If the sequence of iterates is infinite, then*

**Table 2** Truncated column generation algorithm

0 (*initiation*): Find an $x^0 \in X$, let $X^0 = \{x^0\}$ and $k = 0$.

1 (*termination check*): If the iterate $x^k$ solves the column generation problem $CG(x^k)$, then terminate with the same conclusion for the original problem $P(X)$.

2 (*column generation*): Find $y^{k+1}$ by performing one or more iterations with the algorithm $\mathcal{A}_c$ on the column generation problem $CG(x^k)$, starting from any point $v^k \in X$ such that $\Pi(x^k, v^k) \leq \Pi(x^k, x^k)$ holds, and construct a closed and convex set $X^{k+1} \subseteq X$ such that $y^{k+1} \in X^{k+1} \supset X^k$.

3 (*iteration*): Let $k := k + 1$.

4 (*solution of restriction*): Find a new iterate $x^k$ by performing one or more iterations with the algorithm $\mathcal{A}_r$ on the restricted problem $P(X^k)$, starting from $x^{k-1}$.

5 (*repetition*): Return to Step 1.

$$\left\{d_{X^*}(x^k)\right\} \to 0.$$

*Proof* If the sequence of iterates is finite, then the algorithm has been interrupted in Step 1, and the conclusion follows from Assumptions 5 and 8 (cf. Remark 4). In the remainder of the proof, we thus assume that the sequence of iterates is infinite.

The sequence $\{X^k\}$ consists of non-empty, compact, convex and increasing sets (Assumptions 3 and 6). Thus, it has a set limit (e.g., see Mosco 1969; Salinetti and Wets 1979), say $\widetilde{X} \subseteq X$, which is also non-empty, compact and convex.

Since the sequence $\{x^k\}$ is contained in the compact set $X$, it has a non-empty and compact set of accumulation points (e.g., Rudin 1976, Theorem 3.7), say $\overline{X} \subseteq X$. Further, $\overline{X} \subseteq \widetilde{X}$ (e.g., Aubin and Frankowska 1990, Proposition 1.1.2). From the continuity of the merit function $\psi$ (Assumption 7) and the compactness of the set $\overline{X}$ follow that there is a point in $\overline{X}$ where the function $\psi(\widetilde{X}, x)$ attains a maximal value on this set. Consider a subsequence $\{x^k\}_{k\in\mathcal{K}}$, where $\mathcal{K} \subseteq \mathcal{N}$, which converges to such a point; denoting the limit point by $x^{\mathcal{K}}$, we thus have that $\psi(\widetilde{X}, x^{\mathcal{K}}) \geq \psi(\widetilde{X}, x)$ holds for all $x \in \overline{X}$.

Denote by $z^k$, $k \in \mathcal{K}$, the first iterate produced by the algorithm $\mathcal{A}_r$ applied to the restricted problem $P(X^k)$, starting from the point $x^{k-1} \in X^k$. Since each iteration of the algorithm gives descent with respect to the merit function $\psi(X^k, \cdot)$, unless a solution to the restriction is already at hand (Assumption 9), it follows that $\psi(X^k, x^{k-1}) > \psi(X^k, z^k) \geq \psi(X^k, x^k)$ holds for all $k \in \mathcal{K}$. Let $x^{\mathcal{K}-1} \in \widetilde{X}$ be the limit point of a convergent subsequence of $\{x^{k-1}\}_{k\in\mathcal{K}}$, and let $z^{\mathcal{K}} \in \widetilde{X}$ be an accumulation point of the corresponding subsequence of $\{z^k\}_{k\in\mathcal{K}}$. Taking the limit, corresponding to this accumulation point, of the above inequality, the continuity of the merit function yields that $\psi(\widetilde{X}, x^{\mathcal{K}-1}) \geq \psi(\widetilde{X}, z^{\mathcal{K}}) \geq \psi(\widetilde{X}, x^{\mathcal{K}})$.

The fact that $x^{\mathcal{K}-1} \in \overline{X}$ gives that $\psi(\widetilde{X}, x^{\mathcal{K}}) \geq \psi(\widetilde{X}, x^{\mathcal{K}-1})$ holds, and we conclude that $\psi(\widetilde{X}, x^{\mathcal{K}-1}) = \psi(\widetilde{X}, z^{\mathcal{K}}) = \psi(\widetilde{X}, x^{\mathcal{K}})$. The former equality together with the closedness and descent properties of the iteration mapping of the algorithm $\mathcal{A}_r$, and Assumption 1, then gives that $x^{\mathcal{K}-1} \in \widetilde{X}^*$. Further, the relation $\psi(\widetilde{X}, x^{\mathcal{K}-1}) = \psi(\widetilde{X}, x^{\mathcal{K}})$ and the merit property of the function $\psi(\widetilde{X}, \cdot)$ imply that $x^{\mathcal{K}} \in \widetilde{X}^*$

(Assumption 7). Using the merit property again, and the construction of the point $x^{\mathcal{K}} \in \widetilde{X}^*$, we obtain that $\psi(\widetilde{X}, y) \geq \psi(\widetilde{X}, x^{\mathcal{K}}) \geq \psi(\widetilde{X}, x)$ holds for all $y \in \widetilde{X}$ and all $x \in \overline{X}$. Hence, $\overline{X} \subseteq \widetilde{X}^*$.

Now, let $\varepsilon \geq 0$ be such that there is an infinite number of iterates $x^k$ with $d_{X^*}(x^k) \geq \varepsilon$. This subsequence of iterates has some accumulation point, say $\widetilde{x}$, which is then the limit point of some convergent sequence $\{x^k\}_{k \in \widetilde{\mathcal{K}}}$, where $\widetilde{\mathcal{K}} \subseteq \mathcal{N}$. From the above we then know that $\widetilde{x} \in \widetilde{X}^*$.

Since each iteration of the algorithm $\mathcal{A}_c$ gives descent with respect to the merit function $\Pi(x^k, \cdot)$, unless the current iterate is a solution to the column generation problem $CG(x^k)$ (Assumption 10), it follows that, for all $k \in \widetilde{\mathcal{K}}$, $\Pi(x^k, x^k) \geq \Pi(x^k, v^k) > \Pi(x^k, y^{k+1})$. Taking the limit corresponding to a suitable subsequence, the continuity of the merit function (Assumption 8) yields that $\Pi(\widetilde{x}, \widetilde{x}) \geq \Pi(\widetilde{x}, \widetilde{y})$, where $\widetilde{y} \in X$ denotes an accumulation point of the sequence $\{y^{k+1}\}_{k \in \widetilde{\mathcal{K}}}$.

Since $y^{k+1} \in X^{k+1}$ for all $k \in \widetilde{\mathcal{K}}$, $\widetilde{y} \in \widetilde{X}$ holds (e.g., Aubin and Frankowska 1990, Proposition 1.1.2). From the fact that $\widetilde{x} \in \widetilde{X}^*$ and Assumption 8 follow that $\Pi(\widetilde{x}, \widetilde{y}) \geq \Pi(\widetilde{x}, \widetilde{x})$. Therefore, $\Pi(\widetilde{x}, \widetilde{x}) = \Pi(\widetilde{x}, \widetilde{y})$ holds, which together with the closedness and descent properties of the iteration mapping of the algorithm $\mathcal{A}_c$ imply that $\widetilde{x} \in X^*$ (cf. Remark 4).

Hence, $\varepsilon = 0$, and the result of the theorem follows. □

*Remark 5* Note that the choice $v^k = x^k$ satisfies the merit value condition in Step 2. This choice might, however, be practically infeasible in certain realizations of the generic scheme, due to the nature of the column generation problem and the algorithm used for its solution. (This is, for example, the case if the point $x^k$ is not an extreme point of the set $X$, and the column generation problem is a linear program which is solved by the simplex method.) Note also that the re-optimization strategy that is suggested in Step 4 is natural, and computationally advantageous, in most applications. A feasible alternative is, however, to initiate the solution of the restricted problem in a way that is analogous to that used for the column generation problem.

### 5.3 Column dropping

Computational efficiency in large-scale applications may require a column dropping facility. Such a facility might be especially advantageous when the column generation problem constitutes a high-quality approximation of the original problem (e.g., Newton-type approximations in nonlinear programming applications), since the restricted problem will then (at least in the late iterations) have a solution which is close to the solution to the latest column generation problem. (See also the discussion in Section 9.2 in Patriksson 1998d.)

We next establish the convergence of a version of the generic column generation algorithm which includes column dropping. We specifically consider the truncated version of the algorithm, and replace the set augmentation in Step 2 with:

construct a closed and convex set $X^{k+1} \subseteq X$ such that $X^{k+1} \supseteq \text{conv}\{x^k, y^{k+1}\}$.

This construction (found, e.g., in Bertsekas 1999, p. 221; Patriksson 1998d) of the new restricted set of course permits a very extensive column dropping, and also allows for a large degree of freedom when devising realizations of the scheme. (It indeed includes traditional line search methods.) The resulting algorithm can quite easily be shown to be convergent, provided that we add the following condition on the merit function $\psi$.

**Assumption 11** (*merit function*). The merit function $\psi$ is independent of its first argument, that is, $\psi(\widetilde{X}, x) \equiv \psi(X, x)$ holds for all $\widetilde{X} \subseteq X$ and all $x \in X$. Further, if for some $x, y \in X$ it holds that $\Pi(x, y) < \Pi(x, x)$, then the merit function $\psi(X, \cdot)$ is descending in the direction of $y - x$ from $X$.

Note that, under this assumption, it is equivalent to solve the problem $P(X^k)$ and to minimize $\psi$ over $X^k$.

An example of a merit function $\psi$ which satisfies Assumption 11 is $\psi := f$ in the case of the problem *CP*. Examples of merit functions $\Pi$ which satisfy Assumption 11 together with this merit function $\psi$ are those which describe the direction-finding problem in several descent (line search) methods in the solution of the problem *CP*, such as the Frank–Wolfe, Newton, and gradient projection methods, among many others. (See, for example, Proposition 2.14.b in Patriksson 1998d.)

**Theorem 3** (convergence of the truncated algorithm under column dropping). *Suppose that Assumptions 1 through 11 hold. If the sequence* $\{x^k\}$ *of iterates generated by the truncated algorithm is finite, then it has been established that the last iterate solves the problem* $P(X)$. *If the sequence of iterates is infinite, then*

$$\left\{d_{X^*}(x^k)\right\} \to 0.$$

*Proof* The only difference in the proof of this result and that of Theorem 2 is the following. First, we remark that the sequence $\{X^k\}$ need not converge. This is immaterial in this context, where the function $\psi(\widetilde{X}, \cdot)$ is to be replaced throughout with $\psi(X, \cdot)$. [We note however that it does have accumulation sets that are non-empty, compact and convex, by the results of Rockafellar and Wets (1997), Theorem 4.18, and Propositions 4.4 and 4.15, and the proof may proceed in a subsequence corresponding to any one of those limit sets.]

Since $X^k \supseteq \text{conv}\{x^{k-1}, y^k\}$ and since the merit function is descending in the direction of $y^k - x^{k-1}$ from $x^{k-1}$ by Assumption 11, this point is clearly not a minimizer of the merit function $\psi(X, \cdot)$ on the set $X^k$. Hence, $\psi(X, x^{k-1}) > \psi(X, z^k) \geq \psi(X, x^k)$ holds. The rest of the proof follows the identical pattern to that of Theorem 2. □

*Remark 6* (column dropping and merit functions). The merit functions $\psi$ utilized in the Examples 2 and 3 on variational inequality and saddle-point problems do not satisfy the requirements of Assumption 11, since they are set-dependent. The

simplicial decomposition algorithm of Lawphongpanich and Hearn (1984) utilizes column dropping through a rule governed by the decrease in the primal gap function (3), but in order to establish convergence, column dropping is only allowed to be performed a finite number of times. In order to extend column dropping rules to set-dependent merit functions such as the primal gap function, additional requirements on the column generation problem may have to be introduced. (An example of a column generation problem for variational inequalities where very flexible column dropping rules are allowed is however given in Sect. 6.2.3.)

*Remark 7* (instances of the algorithmic framework). A special case of the exact algorithm validated in Theorem 1 is the rudimentary simplicial decomposition algorithm of von Hohenbalken; among the truncated methods validated in Theorem 2, we find approximate simplicial decomposition methods, such as those defined by Hearn et al. (1987); and among the truncated methods which allow for column dropping, validated in Theorem 3, we find both the Frank–Wolfe algorithm and truncated versions of it both in the linear subproblem and in the line search step.

*Remark 8* (spacer steps). In the case where a merit function $\psi$ exists for the problem $P(X)$, and convergence relies on the descent of this merit function in each step (cf. Sect. 5.1), we may take a different approach to establish convergence than was sketched in Sect. 3.2, by applying the Spacer Step Theorem (Luenberger 1984, p. 231). Assume that a column generation principle exists, for which closedness can not be established, but where the resulting column can be shown to yield a direction of descent with respect to the merit function unless a solution to the problem $P(X)$ is at hand (cf. Lemma 2). In short, the strategy for establishing convergence is as follows. Under the requirement that an infinite number of the problems $CG(x^k)$ are constructed from a closed algorithmic map of the form described hitherto in this paper, and that the overall algorithm is such that the entire sequence of iterates is descending with respect to the merit function for the original problem, the overall algorithm may be described by a closed algorithmic map, by describing the iterations corresponding to the use of the non-closed column generation problems as elements of the closed algorithmic map $x \mapsto L_X^{\psi}(x)$. Since $X^{k+1} \supseteq \operatorname{conv}\{x^k, y^{k+1}\}$ is still satisfied, convergence holds.

From the convergence analysis, we finally note the interesting fact that as the algorithm framework allows for a greater and greater flexibility, the requirements on the merit functions involved in monitoring and guiding the convergence of it also increases. In the basic convergence result (Theorem 1), no merit function is needed. In order to establish the convergence of the truncated algorithm (Theorem 2) we however rely on the existence of a merit function for the column generation and master problems. Finally, when considering the possibility to drop columns (Theorem 3), the merit functions must obey an even stronger requirement.

## 6 Instances of the generic principle

In this section, we provide a sample of instances of both existing methods and of potentially interesting, new, methods, within the algorithmic framework of the paper. Mostly, we shall for simplicity deal with the conceptual framework, but we will occasionally refer to the possibility of using truncated steps and/or column dropping.

### 6.1 Known methods

We here give examples of how some known solution methods can be seen as special cases of the generic column generation principle.

#### *6.1.1 Dantzig–Wolfe decomposition*

As is very well-known, both the simplicial decomposition algorithms, for nonlinear programs and variational inequality problems, and the classical Dantzig–Wolfe decomposition method, for linear programs (e.g., Lasdon 1970), are founded on Carathéodory's theorem (e.g., Bazaraa et al. 2006, Theorem 2.1.6), but no further relation has, to the best of our knowledge, been shown. We will here establish a precise relationship between these two solution procedures by showing that the Dantzig–Wolfe decomposition method can be derived as a special case of the generic column generation principle. (Recall that the simplicial decomposition algorithms are also included as special cases.) This derivation also illustrates how special problem structures might be taken into account and exploited within the frame of the generic scheme.

Consider the linear program

$$\max_{x \in X} \{\, c^{\mathrm{T}} x \mid Ax \leq b \,\}, \quad [LP]$$

where $X \subset \mathbb{R}^n$ is a bounded polyhedron, $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. We assume that the problem is feasible, so that its set of optimal solutions is non-empty and compact.

Introducing a vector $u \in \mathbb{R}^m_+$ of multipliers (or linear programming dual variables) and the Lagrangian function $L : X \times \mathbb{R}^m_+ \to \mathbb{R}$, with $L(x, u) = c^{\mathrm{T}} x + b^{\mathrm{T}} u - u^{\mathrm{T}} A x$, the linear program is equivalent to the saddle-point problem

$$\min_{u \in \mathbb{R}^m_+} \max_{x \in X} L(x, u),$$

which is a special case of the archetype problem.

Now, suppose that we attack this saddle-point problem with the generic column generation principle and let the column generation problem be defined through a linearization of the saddle-function. Because of the Cartesian product structure of the feasible set $X \times \mathbb{R}^m_+$, each of the two sets can be treated separately. Furthermore, because of the simplicity of the dual feasible set $\mathbb{R}^m_+$, it is not approximated but treated explicitly.

Assume, without any loss of generality, that we have at hand a feasible solution to the linear program *LP*, that is, a point $x^0 \in X$ such that $Ax^0 \leq b$. Assume further that $k$ (distinct) extreme points of the set $X$, denoted $y^i, i = 1, \ldots, k$, are known explicitly. Letting

$$\Lambda^{k+1} = \left\{ \lambda \in \mathbb{R}^{k+1} \middle| \sum_{i=0}^{k} \lambda_i = 1, \quad \lambda_i \geq 0, \quad i = 0, \ldots, k \right\}$$

and defining

$$x(\lambda) = \sum_{i=0}^{k} \lambda_i y^i, \qquad \lambda \in \Lambda^{k+1},$$

with $y^0 = x^0$, the restricted saddle-point problem is to

$$\min_{u \geq 0} \max_{\lambda \in \Lambda^{k+1}} L(x(\lambda), u).$$

This problem is equivalent to the linear program

$$\max_{\lambda \in \Lambda^{k+1}} \left\{ \sum_{i=0}^{k} \left(c^{\mathrm{T}} y^i\right) \lambda_i \middle| \sum_{i=0}^{k} \left(A y^i\right) \lambda_i \leq b \right\},$$

which is recognized as the *restricted master problem* of the Dantzig–Wolfe decomposition method, as applied to the problem *LP*.

Let $(\lambda^k, u^k)$ be a solution to the restricted saddle-point problem, and let $x^k = x(\lambda^k)$. The linearization based column generation problem defined at $(x^k, u^k)$ is to find

$$\min_{u \geq 0} \max_{x \in X} L\left(x^k, u^k\right) + \nabla_x L\left(x^k, u^k\right)^{\mathrm{T}} \left(x - x^k\right) + \nabla_u L\left(x^k, u^k\right)^{\mathrm{T}} \left(u - u^k\right),$$

which reduces and separates into

$$\max_{x \in X} \left(c - A^{\mathrm{T}} u^k\right)^{\mathrm{T}} x \qquad \text{and} \qquad \min_{u \geq 0} \left(b - A x^k\right)^{\mathrm{T}} u.$$

The former problem is recognized as the *column generation problem* (or, *subproblem*) of the Dantzig–Wolfe decomposition method, as applied to the problem *LP*, and it provides a new extreme point of the set $X$, $y^{k+1}$. (The latter problem, which is of no interest since the dual feasible set is treated explicitly in the restricted saddle-point problem, is trivial since $Ax^k \leq b$ holds.)

The Dantzig–Wolfe decomposition method is thus a special case of the generic column generation principle; applied to the primal–dual saddle-point formulation of the original linear program, in fact it is a simplicial decomposition method.

To establish that Assumptions 1–10 are satisfied is straightforward, and hence the generic Dantzig–Wolfe algorithm as well as truncated versions of it are validated. The latter enables the use of well-known and practical truncation possibilities, as follows: in the column generation step, we do not search for the best new column,

but notice that it suffices to generate *any* column (that is, extreme point of $X$) with a negative reduced cost; in the master problem phase, we notice that it is sufficient, for example, to perform one pivoting step of the simplex method, starting from the previous solution. (The latter, however, reduces the algorithm to a version of the revised simplex method.) Several possibilities to devise "inexact" versions of the Dantzig–Wolfe algorithm are discussed in Lübbecke and Desrosiers (2005), Fuller and Chung (2008), Çelebi and Fuller (2013).

How to realistically fulfill Assumption 11 to establish the convergence of versions of the Dantzig–Wolfe algorithm which include column dropping rules that are as general as that utilized in the scheme of Theorem 3 is beyond the scope of this paper.

#### *6.1.2 Nonlinear simplicial decomposition*

As has been remarked upon already, the motivation for the work in Larsson et al. (1996) and the present one was the potential for improving the convergence of simplicial decomposition schemes through the use of nonlinear column generation subproblems. In Larsson et al. (1996) convergence is established for an *exact* version of the *nonlinear simplicial decomposition* method for the solution of *CP*, in which the column generation problem is assumed to be constructed through the approximation of $f$ with a function $\Pi$ of the form $\Pi(x,y) = \nabla f(x)^T(y-x) + \varphi(x,y)$, with the properties that $\varphi$ is strictly convex and continuously differentiable in $y$ for each fixed $x \in X$ and further with $\varphi(x,x) = 0$ and $\nabla_y \varphi(x,x) = 0$ being true for every $x \in X$. [This subproblem is equivalent to that of the regularized Frank–Wolfe method (Migdalas 1994; Karakitsiou et al. 2004), and a special case of that of the cost approximation method (Patriksson 1998a, b, c, d).] With the appropriate choices of the function $\varphi$, multi-dimensional search versions of gradient projection and Newton-type methods, for example, are possible to construct. Applications to two types of nonlinear network flow problems established the numerical efficiency of the approach; in one of the cases, the function $\varphi$ was chosen to adapt to a Cartesian product structure of the feasible set.

Convergence of this scheme, also in its truncated version and with column dropping, follows from the above results by making the immediate choices of $\psi$ and $\Pi$; the fulfillment of Assumptions 1–11 follows immediately. We note finally that in Patriksson (1998d), the convergence of this type of methods has been taken some steps further, and has for example removed the requirement of closedness of the mapping $Y$. Further extensions of the scheme and results on its properties can also be found in García et al. (2003, 2011), the first mainly on its finite convergence, the second on numerical investigations.

### 6.2 New methods

#### *6.2.1 A sequential linear programming method*

We will here use the generic column generation scheme to develop a novel *sequential linear programming* (SLP) type method for constrained nonlinear

optimization. It is based on inner approximations of both the primal and dual spaces, which yields a method which in the primal space combines column and constraint generation.

Consider a convex and differentiable optimization problem of the form

$$\min_{x\in X}\{f(x) \mid g(x)\le 0\}, \quad [CDP]$$

where $X \subset \mathbb{R}^n$ is a bounded polyhedron and $g : X \to \mathbb{R}^m$. We assume that the problem is feasible, so that its set of optimal solutions is non-empty and compact, and that an appropriate constraint qualification holds. (We also presuppose that the numbers of variables and explicit constraints are large, since the method to be presented is otherwise not advantageous.)

Letting $u \in \mathbb{R}^m_+$ be a vector of multipliers associated with the explicit constraints and introducing the Lagrangian function $L : X \times \mathbb{R}^m_+ \to \mathbb{R}$, with $L(x,u) = f(x) + u^{\mathrm{T}}g(x)$, the above problem can (under a suitable constraint qualification, cf., for example, Bazaraa et al. 2006, Chapter 5) be equivalently restated as the saddle-point problem

$$\max_{u\in B^R_+}\min_{x\in X} L(x,u),$$

where $B^R_+ = \{u \in \mathbb{R}^m_+ \mid \|u\|_2 \le R\}$, with $R$ being a very large positive constant.

Assume, for simplicity, that a feasible solution, $x^0$, to *CDP* is available. Let $x^{k-1} \in X$ be the current (primal) iterate, and suppose that $k$ extreme points of the set $X$, denoted $y^i$, $i = 1,\ldots,k$, and $k$ points in the dual feasible set, $\mathbb{R}^m_+$, say $d^i$, $i = 1,\ldots,k$, such that $\|d^i\|_2 = 1$, are known explicitly. With $S^{k+1}$ denoting the $(k+1)$-dimensional unit simplex, define

$$x(\lambda) = x^{k-1} + \sum_{i=0}^{k}\lambda_i\left(y^i - x^{k-1}\right), \qquad \lambda \in S^{k+1},$$

where $y^0 = x^0$, and

$$u(\mu) = R\sum_{i=1}^{k}\mu_i d^i, \qquad \mu \in S^{k+1}.$$

The restricted saddle-point problem then is to find

$$\max_{\mu\in S^{k+1}}\min_{\lambda\in S^{k+1}} L(x(\lambda),u(\mu)).$$

Let $(\lambda^k,\mu^k)$ be a solution and define $x^k = x(\lambda^k)$ and $u^k = u(\mu^k)$.

Note that the restricted saddle-point problem is equivalent to

$$\max_{\mu\in\mathbb{R}^k_+}\min_{\lambda\in S^{k+1}} L(x(\lambda),u(\mu)),$$

which can be stated and solved as the *primal problem*

$$\min_{\lambda \in S^{k+1}} \{ f(x(\lambda)) \mid D^{\mathrm{T}} g(x(\lambda)) \leq 0 \}, \tag{6}$$

where $D = (d^1, \ldots, d^k)$. This problem is, clearly, of the same type as the original one, but it typically has fewer (actual) variables and fewer constraints. Further, it is (typically) a restriction of the original problem in the respect that only a subset of the points in the set $X$ are spanned by the points $y^i$, $i = 0, \ldots, k$, while it is (typically) a relaxation in the respect that the original explicit constraints are replaced by the *surrogate constraints* $(d^i)^{\mathrm{T}} g(x(\lambda)) \leq 0$, $i = 1, \ldots, k$. Because of the fewer variables and constraints, it should be computationally less demanding than the original problem. It is easily verified that the problem (6) is feasible, under the assumptions made above.

The column generation problem is constructed through a linear approximation of the saddle-function at the point $(x^k, u^k)$, and amounts to finding

$$\max_{u \in B_+^R} \min_{x \in X} L(x^k, u^k) + \nabla_x L(x^k, u^k)^{\mathrm{T}} (x - x^k) + \nabla_u L(x^k, u^k)^{\mathrm{T}} (u - u^k),$$

which reduces and separates into

$$\min_{x \in X} (\nabla f(x^k) + \nabla g(x^k) u^k)^{\mathrm{T}} (x - x^k) \qquad \text{and} \qquad \max_{u \in B_+^R} g(x^k)^{\mathrm{T}} (u - u^k). \tag{7}$$

The former problem is a linear program and it provides a new extreme point, $y^{k+1}$, of the set $X$. Assuming that $x^k$ is not feasible in the original problem, that is, that $g(x^k) \not\leq 0$ holds, the latter problem is solved by $u = Rd^{k+1}$, with $d^{k+1} = g_+(x^k)/\|g_+(x^k)\|_2$ (where $g_+$ denotes component-wise maximum of $g$ at comparison with zero). Note that the new surrogate constraint, that is, $(d^{k+1})^{\mathrm{T}} g(x) \leq 0$, is the *most violated* at the point $x^k$.

Regarding the convergence of this algorithm, the critical stepping-stone is the fulfillment of the set augmentation property (Assumption 6), which we establish next; the argument is similar to that for simplicial decomposition methods.

Suppose that the pair $(x^k, u^k)$ does not solve the saddle-point problem *SPP*. By the separability of the saddle-function it must then hold that either $x^k$ does not fulfill the saddle-point inequality over $X$ for the given value of $u^k$, or $u^k$ does not fulfill the saddle-point inequality over $u$ for the given value of $x^k$, or both. We consider each case separately below.

The vector $x^k$ is feasible in the min-problem over $X$ in (7), and its objective value in this problem is zero. Hence, the optimal value in this problem must then be negative. Consider next the optimality conditions of the master problem in terms of the conditions on $X$: every point in the convex hull of the vectors $y^i$ generated and stored so-far must have a non-negative objective value in the subproblem over $X$ in (7). Therefore, it must be that the negative optimal value in this problem corresponds to an optimal extreme point solution (we consider only extreme points of the set $X$) that was not in their convex hull, and it is therefore new.

The vector $u^k$ is feasible in the max-problem over $u$ in (7), and its objective value in this problem is zero. By the above, the optimal value in this problem must then be

positive. Consider next the optimality conditions of the master problem in terms of the conditions on $u$: the complementarity condition $g(x^k)^{\mathrm{T}}u^k = 0$ must hold, thanks to complementarity being fulfilled for each constraint in the master problem. Further, every vector $u$ which is feasible in the master problem, or is a non-negative linear combination thereof, satisfies that $g(x^k)^{\mathrm{T}}u \leq 0$, thanks to the fact that the vector $x^k$ is primal feasible. Therefore, it must be that the positive optimal value in the max-problem in (7) corresponds to a vector which is new.

This completes the proof that Assumption 6 holds.

*Remark 9* The LP subproblem (7) can of course be supplied with trust-region type constraints. Such constraints can also be incorporated implicitly as proximity penalty terms in the objective function. These approaches have the effect that non-extremal subproblem solutions are found, which may enhance convergence in practice.

*Remark 10* The linear programming subproblem of the SLP method presented differs from the one constructed in traditional methods of this type, in the respect that the explicit constraints are taken into account implicitly in a Lagrangian-dual (priced-out) fashion, instead of being imposed explicitly. An SLP method of the traditional type, but with the line-search step replaced by a multi-dimensional search, is reached if the generic column generation scheme is applied to an exact penalty reformulation (that is, a purely primal formulation) of the nonlinear program. A method for linear programming with complicating constraints and variables can also be constructed in this manner.

### 6.2.2 *Simplicial decomposition for constrained non-smooth optimization*

Consider a convex non-smooth problem of the form

$$\min_{x \in X} f(x), \quad [NSP]$$

where $X \subset \mathbb{R}^n$ is a non-empty, compact and convex set, and $f : S \to \mathbb{R}$ is convex but not everywhere differentiable; the set $S \subseteq \mathbb{R}^n$ represents the effective domain of $f$ and is for simplicity of the presentation assumed to include the feasible set $X$ in its relative interior. This problem can be stated as an instance of the archetype problem $P(X)$ as to find an $x^* \in X$ and a $\gamma^* \in \mathbb{R}^n$ such that

$$\gamma^* \in \partial f(x^*) \cap -N_X(x^*),$$

where $\partial f$ denotes the subdifferential of $f$.

Suppose that the algorithm was initiated at $x^0 \in X$ and that $k$ points in the set $X$, say $y^i$, $i = 1, \ldots, k$, have been generated by its column generation problem (to be defined below). Defining $X^k = \mathrm{conv}\,\{x^0, y^1, \cdots, y^k\}$, the restricted non-smooth problem then is to find

$$\min_{x \in X^k} f(x).$$

Let $x^k$ be a solution and let $\gamma^k \in \partial f(x^k) \cap -N_{X^k}(x^k)$. (As usual, the restricted problem might of course preferably be stated and solved in terms of the convexity variables for the points which span the set $X^k$.)

The column generation might be accomplished by constructing a continuously differentiable and convex function $\varphi : X \to \mathbb{R}$ which has the property that $\nabla\varphi(x^k) = \gamma^k$ (one may, for example, choose $\varphi$ as a linear or convex quadratic approximation of the objective function $f$ at the point $x^k$), and the solution of the problem

$$\min_{x \in X} \varphi(x).$$

Let a solution to this problem define $y^{k+1}$. To verify the set augmentation property of this column generation problem, we observe that $x^k$ is a minimizer of $\varphi$ over the restricted feasible set $X^k$, since $\nabla\varphi(x^k) = \gamma^k \in -N_{X^k}(x^k)$. Therefore $y^{k+1} \in X^k$ holds if and only if $x^k$ is also a minimizer of $\varphi$ over the feasible set $X$, which, in turn, holds exactly when $\nabla\varphi(x^k) = \gamma^k \in -N_X(x^k)$. Using that $\gamma^k \in \partial f(x^k)$, we thus conclude that $y^{k+1} \in X^k$ if and only if $x^k$ solves *NSP*.

*Remark 11* A vector $\gamma^k \in \partial f(x^k) \cap -N_{X^k}(x^k)$ is found automatically whenever the restricted problem is solved by a bundle method, but this is not the case if it is solved by a subgradient optimization scheme. Such a method might, however, be augmented with the computation of a weighted average of all subgradient directions used (an *ergodic* subgradient), which asymptotically has the requested property (see Strömberg 1997, Part II.6; Larsson et al. 1998, Theorem 3.9; Larsson et al. 2012, Section 5). (If $\gamma^k \in \partial f(x^k)$ does not also satisfy $\gamma^k \in -N_X(x^k)$ then the algorithm may halt at a non-optimal solution, as shown by Strömberg 1997, Example 6.3, p. 83.) The paper by Bertsekas and Yu (2011, Section 6.2) contains several additional variations of simplicial decomposition methods for non-differentiable convex programs, and illustrates how the above normal cone inclusion can be verified in special cases, such as when the objective is a max-function over convex differentiable functions, or if the constraints are described through such constructions.

Assumptions 1–6 follow immediately, which means that convergence is established of a generic simplicial decomposition scheme for convex, non-differentiable optimization. The fulfillment of further assumptions for the use of inexact computations and column dropping require further assumptions on the function $\varphi$ and on the algorithms utilized for solving the sub- and master problems.

### 6.2.3 Newton methods with multi-dimensional searches

Consider the subproblem (5) corresponding to the affine approximation of the mapping $F$ of the variational inequality problem *VIP* at the point $X$, and consider its use as a column generation problem in the generic scheme. By the assumptions stated in Example 2, Assumptions 1–6 follow immediately, and so it is straightforward to extend the convergence of the Newton method of Marcotte and Dussault

(1989) to the case of multi-dimensional searches (see also Patriksson 1998d, Chapter 9).

The restriction of the primal gap function (3) to the current subset of columns, $\psi(\hat{X}, \cdot)$, can be used as a merit function for the restriction, and the algorithm for $P(\hat{X})$ stated in Assumption 9 could, for example, be the original Newton method in Marcotte and Dussault (1989). The choice of a merit function $\Pi$ for the affine subproblem (5) is less straightforward to make, but could, for example, be the natural result of the choice of a particular splitting method for its solution. With such a choice, also the conditions of Theorem 2 are fulfilled.

When considering the use of very general column dropping rules, we note that the primal gap function is not set-independent, and so the convergence of gap-minimizing algorithms with multi-dimensional searches is difficult to establish. (We note, however, that if the flexibility of the set augmentation is reduced to require that $X^{k+1} = \text{conv}\,\{x^k, y^{k+1}\}$ then convergence according to Theorem 3 may be established, with the use of the global merit function also in the restricted (line search) problem; this however in effect reduces the algorithm to the original one in Marcotte and Dussault (1989).)

## 7 Concluding remarks

A goal when deriving the generic column generation principle has been to make it widely applicable and easy to modify and extend. We have therefore throughout the derivation strived at choosing the assumptions so that they are reasonable, from the point of view of our experience and intuition, and realistic, from the point of view of possible realizations, and also as weak as possible while still being able to guarantee convergence.

Clearly, the algorithm framework of the generic column generation principle contains a large variety of possible realizations since its steps can be implemented in many ways; in an optimization context, one might for instance construct column generation problems with or without a pricing mechanism, which, in turn, might be linear or nonlinear (e.g., an augmented Lagrangian dualization as in Ruszczyński 1989), and one might choose between keeping some of the constraints of the original problem in the restricted problem, or not. However, it is important to observe that an extra degree of freedom in the construction of an algorithm for a specific class of problems is achieved by considering different, but equivalent, formulations of the original problem. In particular, one can in an optimization context consider primal, dual, and primal–dual formulations. Moreover, one may also obtain different algorithms for a given problem by considering alternative equivalent formulations through transformations of the original one. An example of such a transformation is the (well-known) replacement of a nonlinear objective function with an auxiliary variable and an additional constraint (made in order to create an equivalent problem with a linear objective function), which can be shown to be a means for introducing an approximation of the objective function in the restricted problem of an instance of the generic column generation principle.

(Compare with the different treatment of the objective function in a simplicial decomposition method and in nonlinear Dantzig–Wolfe decomposition, respectively.)

Whenever the admissible set $X$ is a Cartesian product of sets, it is likely to be advantageous, from the point of view of computational efficiency, to employ a *disaggregated* representation, that is, to approximate each of the sets in the product separately. [In Larsson and Patriksson (1992), the advantage of using this strategy was established for an application of a simplicial decomposition method to a specially structured nonlinear multicommodity network flow problem. See also Jones et al. (1993), where it is similarly observed that a disaggregated representation is advantageous in the application of Dantzig–Wolfe decomposition to linear multicommodity network flow problems.] It can in the Cartesian product case also be natural to use a column generation problem that separates over the individual sets; the resulting overall scheme can then be interpreted as a Jacobi-type method with multi-dimensional searches.

In case $X = G \cap R$, where the set $G$ has a favourable structure (e.g., a Cartesian product structure) and the set $R$ is associated with some additional requirements for overall feasibility (e.g., described by a set of *side constraints*), the application of the generic scheme may be based on an inner approximation of the set $G$ only, while the additional requirements are treated explicitly in the master problem. Such an extended scheme makes use of a column generation problem with a solution set mapping $Y : G \cap R \to 2^G$, and a (side constrained) restricted problem $P(G^k \cap R)$, where, for example, $G^k = \operatorname{conv}\{p^0, \ldots, p^k\}$ with $p^0 = x^0 \in G \cap R$ and $p^i \in G$, $i = 1, \ldots, k$, and which is solved as

$$\begin{aligned} &\text{find a } \lambda \in S^{k+1} \text{ such that } x(\lambda) \in R \text{ and satisfies the Condition} \\ &\quad \mathcal{C}\big(x\big(S^{k+1}\big) \cap R\big), \quad [P(S^{k+1} \cap \lambda(R))] \end{aligned}$$

where $\lambda(R) = \{\, \lambda \mid x(\lambda) \in R \,\}$. (Note that the mapping $Y$ takes on as values subsets of $G$, and is in this sense relaxed compared to the column generation mapping of the conceptual scheme, which would take on as values subsets of $G \cap R$.) In the context of simplicial decomposition applied to nonlinear network flows, Stefek (1989) and Marín (1995) have developed methods of that type; see also Larsson and Patriksson (1995), Zenios et al. (1995), Larsson and Patriksson (1999), Lawphongpanich (2000), Larsson et al. (2004), Nie et al. (2004), Mijangos (2005), Shahpar et al. (2008), and Çelebi and Fuller (2013). The convergence of this methodology requires a column generation principle which includes a *pricing mechanism* for the side constraints (cf. the strategy employed in the Dantzig–Wolfe decomposition method, see Sect. 6.1.1).

We finally note that convergence of this scheme in its full generality does not immediately follow from Theorem 1, although it does for some special cases, such as Dantzig–Wolfe decomposition.

The exploitation of the generic column generation principle might lead to new, and hopefully also efficient, solution methods in many of the fields of mathematical programming. As concluded in Sect. 4, the generic scheme is directly applicable to large classes of nonlinear programming, variational inequality, and saddle-point

problems, but it should also be applicable to related problem classes, such as for example complementarity problems and non-smooth optimization problems. Because of the degree of freedom of our algorithm framework, there is with no doubt a large number of instances that are worth studying.

For example, within the field of nonlinear programming, a frequently employed solution principle is to alternate, in an iterative manner, between the solution of an approximate problem and a line search with respect to a merit function, which measures the degree of non-optimality of any tentative solution. Methods included in this class are the reduced gradient method (e.g., Bazaraa et al. 2006, Section 10.6), the Frank–Wolfe algorithm, the method of Zoutendijk (e.g., Bazaraa et al. 2006, Section 10.1–2), the linearization method (e.g., Pshenichnyi and Sosnovsky 1993), the constrained Newton method (e.g., Pshenichny and Danilin 1978), and the successive quadratic programming approach (e.g., Bazaraa et al. 2006, Section 10.4), among others. Further, this solution principle is also employed for the numerical treatment of variational inequality and saddle-point problems. An important area for continued research is, in our opinion, to use the framework of the generic column generation principle to enhance methods within this class through replacing their line searches with multi-dimensional searches. An example method from this class of methods was proposed in Feng and Cui (2003), combining a trust-region method with simplicial decomposition for an optimization problem with explicit nonlinear constraints, and thus extending the method in Ventura and Hearn (1993). It is also our opinion that the introduction of a master problem can relax the need for penalty functions in nonlinear programming algorithms; this idea is complementary to that of introducing "filters," which has become very popular in combination with SLP and sequential quadratic programming (SQP) methods (see, e.g., Fletcher et al. 2002; Fletcher and Leyffer 2002). An example SQP algorithm without a penalty function is obtained from our SLP algorithm of Sect. 6.2.1 if we construct the column generation problem based upon a second-order approximation. We are currently investigating the extension of the SLP type method validated in Sect. 6.2.1 to general, non-convex, problems, in which case the saddle-point based analysis is replaced by a KKT based one.

The use of nonlinear pricing mechanisms (e.g., an augmented Lagrangian dualization) in the Dantzig–Wolfe decomposition method would permit the subproblem to have non-extreme point solutions. [Cf., e.g., Kim and Nazareth (1991), in which non-extreme subproblem solutions are generated by truncating an interior-point method, Feinberg (1989), in which a special class of nonlinear price functions are used, and O'Neill and Widhelm (1976), in which an augmented Lagrangian pricing is used to accelerate convergence in the extension of the Dantzig-Wolfe method to nonlinear problems.] Convergence of such a method is immediate from that of the generic scheme. Nonlinear pricing in the Dantzig–Wolfe method is of interest since it can probably enhance the method's practical performance (which is otherwise often rather poor), but also because it can be interpreted as a mathematical means to obtain an optimal utilization of scarce resources in an economic system with decentralized planning (e.g., Dirickx and Jennergren 1979; Feinberg 1989).

Further questions for continued research is the establishment of (non-trivial) sufficient conditions (e.g., on the properties of the original problem or the column generation problem, on the implementation of the steps of the algorithm, or combinations thereof) for finite convergence. In case the admissible set is polyhedral and represented by a set of linear constraints, it is also of interest to establish conditions under which the constraints that are active at a solution are identified finitely. (Cf. the results given by Burke and Moré (1988, 1994) for nonlinear programming algorithms. See also Patriksson (1998d), García et al. (2003) on this issue.)

## References


Appelgren LH (1969) A column generation algorithm for a ship scheduling problem. Transp Sci 3:53–68

Appelgren LH (1971) Integer programming methods for a vessel scheduling problem. Transp Sci 5:64–78

Aubin J-P, Frankowska H (1990) Set-valued analysis. Birkhäuser, Boston

Auslender A (1976) Optimisation: Méthodes Numériques. Masson, Paris

Bar-Gera H (2002) Origin-based algorithm for the traffic assignment problem. Transp Sci 36:398–417

Bar-Gera H (2010) Traffic assignment by paired alternative segments. Transp Sci 44:1022–1046

Bazaraa MS, Sherali HD, Shetty CM (2006) Nonlinear programming: theory and algorithms, 3rd edn. Wiley, New York

Bertsekas DP (1999) Nonlinear programming. Athena Scientific, Belmont

Bertsekas DP, Yu H (2011) A unifying polyhedral approximation framework for convex optimization. SIAM J Optim 21:333–360

Boyce D, Ralevic-Dekic B, Bar-Gera H (2004) Convergence of traffic assignments: How much is enough? J Transp Eng 130:49–55

Burke JV, Moré JJ (1988) On the identification of active constraints. SIAM J Numer Anal 25:1197–1211

Burke JV, Moré JJ (1994) Exposing constraints. SIAM J Optim 3:573–595

Canon MD, Cullum CD (1968) A tight upper bound on the rate of convergence of the Frank–Wolfe algorithm. SIAM J Control 6:509–516

Çelebi E, Fuller JD (2013) Master problem approximations in Dantzig–Wolfe decomposition of variational inequality problems with applications to two energy market models. Comput Oper Res 40:2724–2739

Cohen G (1980) Auxiliary problem principle and decomposition of optimization problems. J Optim Theory Appl 32:277–305

Conn AR, Gould NIM, Toint PhL (2000) Trust-region methods., Vol. 1 of MPS-SIAM series on optimization, SIAM, Philadelphia

Danskin JM (1967) The theory of max–min. Springer, Berlin

Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. Oper Res 8:101–111

Dantzig GB, Wolfe P (1961) The decomposition algorithm for linear programs. Econometrica 29:767–778

Dem'yanov VF, Malozemov VN (1974) Introduction to minimax. Wiley, New York

Dirickx YMI, Jennergren LP (1979) Systems analysis by multilevel methods. Wiley, Chichester

Feinberg B (1989) Coercion functions and decentralized linear programming. Math Oper Res 177–187

Feng G, Cui Z (2003) Combination of trust region method and simplicial decomposition for convex constrained nonlinear optimization. Optimization 52:459–466

Feng G, Li M (2001) Quadratic approximation in the restricted simplicial decomposition. OR Trans 6:43–49

Fletcher R, Gould NIM, Leyffer S, Toint PL, Wächter A (2002) Global convergence of a trust-region SQP-filter algorithm for general nonlinear programming. SIAM J Optim 13:635–659

Fletcher R, Leyffer S (2002) Nonlinear programming without a penalty function. Math Program 91:239–269

Ford LR Jr, Fulkerson DR (1958) A suggested computation for maximal multi-commodity network flows. Manag Sci 5:97–101
Frank M, Wolfe P (1956) An algorithm for quadratic programming. Nav Res Logist Q 3:95–110
Fuller JD, Chung W (2008) Benders decomposition for a class of variational inequalities. Eur J Oper Res 185:76–91
García R, Marín A, Patriksson M (2003) Column generation algorithms for nonlinear optimization, I: convergence analysis. Optimization 52:171–200
García R, Marín A, Patriksson M (2011) Column generation algorithms for nonlinear optimization, II: numerical investigations. Comput Oper Res 38:591–604
Geoffrion AM (1970) Elements of large scale mathematical programming. Manag Sci 16:652–691
Gilmore P, Gomory R (1961) A linear programming approach to the cutting stock problem. Oper Res 9:849–859
Gilmore P, Gomory R (1963) A linear programming approach to the cutting stock problem—part ii. Oper Res 11:863–888
Goldstein AA (1964) Convex programming in Hilbert space. Bull Am Math Soc 70:709–710
Harker PT, Pang J-S (1990) Finite-dimensional variational inequality and nonlinear complementarity problems: a survey of theory, algorithms and applications. Math Program 48:161–220
Hearn DW, Lawphongpanich S, Nguyen S (1984) Convex programming formulations of the asymmetric traffic assignment problem. Transp Res B Methodol 18:357–365
Hearn DW, Lawphongpanich S, Ventura JA (1985) Finiteness in restricted simplicial decomposition. Oper Res Lett 4:125–130
Hearn DW, Lawphongpanich S, Ventura JA (1987) Restricted simplicial decomposition: computation and extensions. Math Program Study 31:99–118
Higgins JE, Polak E (1990) Minimizing pseudoconvex functions on convex compact sets. J Optim Theory Appl 65:1–27
Holloway CA (1974) An extension of the Frank and Wolfe method of feasible directions. Math Program 6:14–27
Jones KL, Lustig IJ, Farvolden JM, Powell WB (1993) Multicommodity network flows: the impact of formulation on decomposition. Math Program 62:95–117
Karakitsiou A, Mavrommati A, Migdalas A (2004) Efficient minimization over products of simplices and its application to nonlinear multicommodity network flows. Oper Res 4:99–118
Kim K, Nazareth JL (1991) The decomposition principle and algorithms for linear programming. Linear Algebra Appl 152:119–133
Larsson T, Patriksson M (1992) Simplicial decomposition with disaggregated representation for the traffic assignment problem. Transp Sci 26:4–17
Larsson T, Patriksson M (1994) A class of gap functions for variational inequalities. Math Program 64:53–79
Larsson T, Patriksson M (1995) An augmented Lagrangian dual algorithm for link capacity side constrained traffic assignment problems. Transp Res B Methodol 29:433–455
Larsson T, Patriksson M (1999) Side constrained traffic equilibrium models—analysis, computation and applications. Transp Res B Methodol 33:233–264
Larsson T, Patriksson M, Rydergren C (1996) Applications of simplicial decomposition with nonlinear column generation to nonlinear network flows. In: Pardalos PM, Hearn DW, Hager WW (eds) Network optimization. Proceedings of the network optimization conference. Lecture notes in economics and mathematical systems, University of Florida, Gainesville, vol 450. Springer, Berlin, pp 346–373
Larsson T, Patriksson M, Rydergren C (2004) A column generation procedure for the side constrained traffic equilibrium problem. Transp Res B Methodol 38:17–38
Larsson T, Patriksson M, Strömberg A-B (1998) Ergodic convergence in subgradient optimization. Optim Methods Softw 9:93–120
Larsson T, Patriksson M, Strömberg A-B (2012) Ergodic convergence in subgradient optimization—with application to simplicial decomposition of convex programs. Contemp Math 568:159–189
Lasdon LS (1970) Optimization theory for large systems. Macmillan, New York
Lawphongpanich S (2000) Simplicial with truncated Dantzig–Wolfe decomposition for nonlinear network flow problems with side constraints. Oper Res Lett 26:33–41
Lawphongpanich S, Hearn D (1984) Simplicial decomposition of the asymmetric traffic assignment problem. Transp Res B Methodol 18:123–133

Levitin ES, Polyak BT (1966) Constrained minimization methods. USSR Comput Math Math Phys 6:1–50
Lübbecke ME, Desrosiers J (2005) Selected topics in column generation. Oper Res 53:1007–1023
Luenberger DG (1984) Linear and nonlinear programming, 2nd edn. Addison-Wesley, Reading
Luo Z-Q, Pang J-S, Ralph D (1996) Mathematical programs with equilibrium constraints. Cambridge University Press, Cambridge
Manne AS (1958) Programming of economic lot sizes. Manag Sci 4:115–135
Marcotte P, Dussault J-P (1989) A sequential linear programming algorithm for solving monotone variational inequalities. SIAM J Control Optim 27:1260–1278
Marín A (1995) Restricted simplicial decomposition with side constraints. Networks 26:199–215
Migdalas A (1994) A regularization of the Frank–Wolfe method and unification of certain nonlinear programming methods. Math Program 65:331–345
Mijangos E (2005) An efficient method for nonlinearly constrained networks. Eur J Oper Res 161:618–635
Mosco U (1969) Convergence of convex sets and of solutions of variational inequalities. Adv Math 3:510–585
Mulvey JM, Zenios SA, Ahlfeld DP (1990) Simplicial decomposition for convex generalized networks. J Inf Optim Sci 11:359–387
Murphy FH (1973a) Column dropping procedures for the generalized programming algorithm. Manag Sci 19:1310–1321
Murphy FH (1973b) A column generation algorithm for nonlinear programming. Math Program 5:286–298
Nie Y (2009) A class of bush-based algorithms for the traffic assignment problem. Transp Res B Methodol 44:73–89
Nie Y, Zhang HM, Lee D-H (2004) Models and algorithms for the traffic assignment problem with link capacity constraints. Transp Res B Methodol 38:285–312
Nikaido H (1968) Convex structures and economic theory. Academic Press, New York
O'Neill RP, Widhelm WB (1976) Acceleration of Lagrangian column-generation algorithms by penalty function methods. Manag Sci 23:50–58
Patriksson M (1994) The traffic assignment problem—Models and methods, VSP BV, Utrecht, The Netherlands. Facsimile republication 2015 by Dover Publications, Mineola, NY
Patriksson M (1998a) Cost approximation: a unified framework of descent algorithms for nonlinear programs. SIAM J Optim 8:561–582
Patriksson M (1998b) Cost approximation algorithms with nonmonotone line searches for nonlinear programs. Optimization 44:199–217
Patriksson M (1998c) Decomposition methods for differentiable optimization problems on Cartesian product sets. Comput Optim Appl 9:5–42
Patriksson M (1998d) Nonlinear programming and variational inequality problems—a unified approach., Vol. 23 of applied optimization. Kluwer Academic Publishers, Dordrecht
Pshenichny BN, Danilin YM (1978) Numerical methods in extremal problems. MIR Publishers, Moscow
Pshenichnyi BN, Sosnovsky AA (1993) The linearization method: principal concepts and perspective directions. J Glob Optim 3:483–500
Rockafellar RT (1970) Convex analysis. Princeton University Press, Princeton
Rockafellar RT (1976) Monotone operators and the proximal point algorithm. SIAM J Control Optim 14:877–898
Rockafellar RT, Wets RJ-B (1997) Variational analysis. Springer, Berlin
Rudin W (1976) Principles of mathematical analysis, 3rd edn. McGraw-Hill, Auckland
Ruszczyński A (1989) An augmented Lagrangian decomposition method for block diagonal linear programming problems. Oper Res Lett 8:287–294
Salinetti G, Wets RJ-B (1979) On the convergence of sequences of convex sets in finite dimensions. SIAM Rev 21:18–33
Shahpar AH, Aashtiani HZ, Babazadeh A (2008) Dynamic penalty function method for the side constrained traffic assignment problem. Appl Math Comput 206:332–345
Stefek D (1989) Extensions of simplicial decomposition for solving the multicommodity flow problem with bounded arc flows and convex costs. PhD thesis, University of Pennsylvania, Philadelphia, PN, USA

Strömberg A-B (1997) Conditional subgradient methods and ergodic convergence in nonsmooth optimization. PhD thesis, Department of Mathematics, Linköping Institute of Technology, Linköping, Sweden

Ventura JA, Hearn DW (1993) Restricted simplicial decomposition for convex constrained problems. Math Program 59:71–85

von Hohenbalken B (1975) A finite algorithm to maximize certain pseudo-concave functions on polytopes. Math Program 9:189–206

von Hohenbalken B (1977) Simplicial decomposition in nonlinear programming algorithms. Math Program 13:49–68

Wolfe P (1970) Convergence theory in nonlinear programming. In: Abadie J (ed) Integer and nonlinear programming. Prentice-Hall, Englewood Cliffs, pp 1–36

Zangwill WI (1969) Nonlinear programming: a unified approach. Prentice-Hall, Englewood Cliffs

Zenios SA, Pınar MÇ, Dembo RS (1995) A smooth penalty function algorithm for network-structured problems. Eur J Oper Res 83:220–236