Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

ARMIN FÜGENSCHUH, CHRISTINE HAYN, DENNIS MICHAELS

Mixed-Integer Linear Methods for Layout-Optimization of Screening Systems in Recovered Paper Production

ZIB-Report 12-44 (November 2012)

Herausgegeben vom
Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7
D-14195 Berlin-Dahlem

Telefon: 030-84185-0
Telefax: 030-84185-125

e-mail: bibliothek@zib.de
URL: http://www.zib.de

ZIB-Report (Print) ISSN 1438-0064
ZIB-Report (Internet) ISSN 2192-7782

Mixed-Integer Linear Methods for Layout-Optimization of Screening Systems in Recovered Paper Production

Armin Fügenschuh [*1], Christine Hayn [†2], and Dennis Michaels [‡3]

[1]Konrad-Zuse-Zentrum für Informationstechnik Berlin, Takustraße 7, 14195 Berlin, Germany
[2]Friedrich-Alexander Universität Erlangen-Nürnberg, Cauerstraße 11, 91058 Erlangen, Germany
[3]ETH Zürich, Institut für Operations Research, Raemistrasse 101, 8092 Zürich, Switzerland

August 17, 2012

Abstract

The industrial treatment of waste paper in order to regain valuable fibers from which recovered paper can be produced, involves several steps of preparation. One important step is the separation of stickies that are normally attached to the paper. If not properly separated, remaining stickies reduce the quality of the recovered paper or even disrupt the production process. For the mechanical separation process of fibers from stickies a separator screen is used. This machine has one input feed and two output streams, called the accept and the reject. In the accept the fibers are concentrated, whereas the reject has a higher concentration of stickies. The machine can be controlled by setting its reject rate. But even when the reject rate is set properly, after just a single screening step, the accept still has too many stickies, or the reject too many fibers. To get a proper separation, several separators have to be assembled into a network. From a mathematical point of view this problem can be seen as a multi-commodity network flow design problem with a nonlinear, controllable distribution function at each node. We present a nonlinear mixed-integer programming model for the simultaneous selection of a subset of separators, the network's topology, and the optimal setting of each separator. Numerical results are obtained via different types of linearization of the nonlinearities and the use of mixed-integer linear solvers, and compared with state-of-the-art global optimization software.

Keywords: Mixed-Integer Linear Programming, Nonlinear Programming, Piecewise Linear Approximation, Global Optimization, Linear Relaxation, Topology Optimization, Network Design.

1 Introduction

Paper and products made of paper we all use in our daily life, for example, in the form of books, newspapers, packages, or hygienic articles. In Germany, for example, about 21 million tons of paper, or 250 kg per head, are consumed each year; other industrialized countries have similar values. On the other hand, paper is one of the most collected and best recycled products in our industry. About 75% of the used paper is collected. Thus the most important resource in paper production are not trees, but recovered paper. About 67% of the raw material in paper production, that is, more than 15 million tons, consists of used paper [19]. Before waste paper can be used to produce new paper from it, it has to be prepared in several steps. In a first step, the paper is sorted manually and large containments are removed. Then the recovered paper is hackled into smaller pieces and resolved in water. This process step is called pulping. Afterwards, the suspension, also called pulp, is cleaned from paper clips, adhesives, and plastic material in several

[*]fuegenschuh@zib.de
[†]christine.hayn@math.uni-erlangen.de
[‡]dennis.michaels@ifor.math.ethz.ch

steps. If necessary, it is furthermore de-inked. Only after this preprocessing steps, the recovered paper suspension is ready to enter the actual paper machine, where the suspension is transferred over big wires, and dried. Finally, new paper is produced on rolls out of the recovered fibers.

This article deals with the optimization of one of these preprocessing steps, more precisely the fine screening of the pulp. This suspension consists of several components. Among the valuable components are fibers of different lengths. Furthermore, the pulp contains undesired components, most prominently stickies. These are small tacky particles arising, e.g., from book spines, labels, or tapes. If too many of them remain, they cause trouble in the later paper manufacturing process, as they, for example, stick on the cylinders, and thereby may cause breaks of the paper rolls, and thus lead to production losses. Valkama [80] estimates that the production losses due to stickies in the German paper industry were about 265 million Euro in 2004, when about 13.2 million tons of paper were recovered in Germany.

The main goal of the fine screening process is to clean the pulp from these stickies. In practice this process is realized with multi-stage screening systems, consisting of three up to six different pressure screens, or screens for short. For the set-up of a multi-stage screening system several configurations are possible. For systems of three screens a feed forward, a feed-back partial cascade, or a feed-back full cascade are typically used by engineers, see Figure 1, but one can think of several other ways to connect sorters to networks.

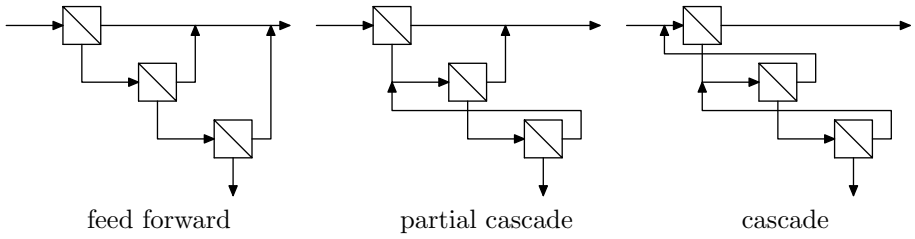

Figure 1: Examples of multi-stage-screening systems.

Essentially, a pressure screen consists of a cylindric screening basket and a rotor. The accepted pulp passes through the screening basket, whereas the inside remaining particles are rejected. Figure 2 shows a schematic illustration of a screen.

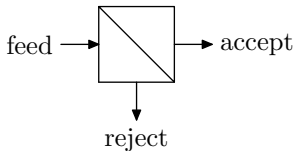

Figure 2: Schematic illustration of a screen.

The process within each screen can be described mathematically, for example by the so-called plug-flow model (see Almin and Steenberg [3], Kubat and Steenberg [42], Steenberg [69], and also Valkama [80]). The model assumes that the pulp is completely transferred over the screen, i.e., no material losses occur in screening:

$$m_{in} = m_{acc} + m_{rej}, \tag{1}$$

where m_{in} denotes the mass flow into the inlet of the pressure screen, and m_{acc}, m_{rej} the mass flows leaving accept and reject, respectively. The separation efficiency T_k of the separation of component k in the pulp, e.g., fibers and stickies, is defined by

$$T_k = \frac{m^k_{rej}}{m^k_{in}} = 1 - \frac{m^k_{acc}}{m^k_{in}}, \tag{2}$$

and describes the fraction of component k that is separated in the screen. On the other hand the mass reject rate defines the share of total separation:

$$R = \frac{m_{rej}}{m_{in}} = 1 - \frac{m_{acc}}{m_{in}}. \tag{3}$$

The reject rate may be adjusted at each screen individually, and serves as process variable in our later model. Physically it relates to the nominal pressure at which the screen operates.

According to the plug flow model, the separation efficiency and the reject rate are related in the following way:

$$T = R_k^{\beta_k}, \tag{4}$$

where $\beta_k \in (0, 1)$ is a device-specific factor that includes all the factors like basket geometry and rotor velocity. This parameter is specific for all components in the pulp and may be obtained by measurements, see Valkama [79, 80]. If β_k is close to zero for some component k then this component is separated efficiently from the pulp, whereas for values of β_k close to one no screening effect occurs.

Our aim is to optimize the screening result by simultaneously optimizing the reject rates as process variables and the installation type. Basically, the screening result is considered to be good, if as much stickies as possible are rejected, whereas as much as possible valuable particles like fibers are accepted. Hence, we are dealing with a multi-criteria optimization problem. The problem is mathematically challenging because of the combination of the nonlinearities arising from the screening process model, and its combinatorial nature originating from the choice of the layout. Structurally, it can be considered as a multi-commodity network flow design problem with nonlinear constraints.

Our optimization routine can be applied in many ways in practice. First of all, there is the task of building a new multi-stage screening system. At this point, one has as well the choice of the configuration of the system as the choice of different screens used in the system at ones disposal. During operation, it is possible to measure the total feed of the system online, and then determine the optimal adjustment of the reject rate. Here the network structure is considered as being fixed. However, it might be possible to change the connection of the screens online by switching certain valves in the system, depending on the actual feed. The optimal connections can also be computed by our methods.

Our further scientific contribution is a comparison of different techniques to handle nonlinear constraints within mixed-integer optimization problems. Although the reformulation of nonlinear constraints via a piecewise linear approximation is known for more than five decades [50, 16] they are still popular today [38, 59, 30]. One of their advantages is that their use results in a pure mixed-integer programming problem that can be handled by specialized solvers for this problem structure. Since those solvers have seen a tremendous performance boost over the last decades [11], one immediately can benefit from this for solving also piecewise linear approximations of nonlinear problems. On the other hand, also global optimization methods have been improved, e.g., see [74, 9]. In our present work we compare both approaches computationally on the basis of the sticky sorting and topology planning problem. We are interested to learn if some method clearly dominates some other method here.

The remainder of the paper is organized as follows. We start with a survey of the relevant literature in Section 2. Section 3 introduces nonlinear models for the optimization of the setting of a multi-stage screening system and the layout decision. Additionally, properties of the models are discussed. In the following Section 4, we explain the solution techniques used to solve the models. In Section 5, we display and discuss computational results for the case of a fixed cascade as well as for the layout decision. Finally, concluding remarks and ideas for future work are given in Section 6.

2 Survey of the Literature

The optimization of the fine screening process in recycled paper production has recently been treated by Valkama [79, 80]. In this work the screening process in a fixed installation is simulated using Matlab [76]. Valkama introduces an economical cost function in order to evaluate and compare the screening results. The reject rate of each screen is discretized, usually in 100 steps, and for each combination of reject rate values the screening process is simulated dynamically using the plug flow model. The simulation quickly reaches an equilibrium, typically after 10-15 iterations. Finally, all results at equilibrium are compared, and the best one regarding the cost function is said to be optimal. Valkama applies his algorithm to industrial paper machines. At every investigated paper mill an optimization potential was detected.

Since the simulation has to be carried out for each combination of values for the reject rates, this procedure is quite time-consuming, e.g., if the reject rates of three screens are discretized in

100 steps, one million (100^3) simulation runs have to be carried out. Thus, it is not practicable to use this method for additionally determining the optimal layout of the machine, because one would have to run the simulation for each possible layout and each combination of values for the reject rates.

More generally, our problem can be seen as the quest for an optimal arrangement of machines to fulfill separation tasks, which belongs to the field of process synthesis, see Nishida, Stephanopoulos, and Westerberg [61] for a survey. A special area of process synthesis, which was described by Nath and Motard [58], is the synthesis of separation sequences, where for a given feed stream of known conditions (i.e., composition, flow rate, temperature, pressure), the goal is to synthesize systematically a process that can isolate the desired (specified) products from the feed at minimum venture cost.

A lot of studies are dealing with the task of selecting such an optimal layout. Basically, the approaches can be divided into two classes. On the one hand *heuristics* like evolutionary methods were developed by Nath and Motard [58] or Muraki and Hayakawa [56]. On the other hand a *superstructure* for the system containing all reasonable links between machine elements was derived and the process was formulated as a nonlinear program by Floudas [20, 21], Friedler et al. [24], or Heckl et al. [34]. Methods for solving the nonlinear problems include heuristics, e.g., for reducing the superstructure, as well as standard nonlinear programming algorithms. Many of these publications, e.g. [58, 61, 56, 20], deal with simple sharp separators only. A simple separator separates a single feed into two products, and in sharp separators each component entering in the feed stream leaves in only one product. For example, some two product distillation columns are simple sharp separation units.

Contrarily, Muraki and Hayakawa [57] and Floudas [21] regard the separation sharpness as a variable. They deal with the process of distillation, allowing for separators and dividers in the system. The authors model the problem by linear and nonlinear equations. In [57] the nonlinearities are dealt with by randomly changing the variables associated with the recovery fraction. In [21] a modification of the *Generalized Benders Decomposition* for solving the resulting non-convex MINLP is suggested.

Baur and Stichlmair [8], Frey et al. [23], or Franke, Górak, and Strube [22] have addressed the topic of using MINLP optimization in process synthesis, mainly in the context of rectification. They consider the possibility of optimizing the operating variables and the process structure simultaneously as the main advantage of mixed integer nonlinear programming. Their aim is to gain products of minimum cleanness, while minimizing the energy demand, operating and total costs of the processes. To this end, they define superstructures and introduce MINLPs based on them. For solving the MINLPs they use standard algorithms like outer-approximation algorithms [8, 23], or Nelder-Mead algorithms and SQP-methods [22].

In general, standard algorithms as mentioned above are not able to guarantee terminating at a globally optimal solution. To overcome this drawback, a hierarchy of mixed-integer linear relaxations for the MINLPs modeling an underlying chemical engineering process have been constructed and solved in [28, 33, 5]. This way, the authors have obtained a sequence of globally valid bounds on the optimal objective function value of the original MINLP. Such bounds are used in [28] to determine an optimal process candidate for a concrete binary reactive distillation process, while infeasibility statements for some continuous counter-current chromatography processes have been derived in [33, 5].

Contrary to all reviewed publications, we are concerned with a separation, where the separation behavior itself is given by a nonlinear function. To the best of our knowledge, there is no mathematical literature dealing with this problem so far. Furthermore, our main objective is to receive the best possible separation result, and not as in the reviewed literature, to minimize the operating or installation costs of the system. To this end, we develop and utilize an optimization method, which can guarantee to find the globally optimal solution.

3 Nonlinear Models and their Properties

In this section we present a nonlinear programming (NLP) model for the case of a given fixed layout of a screening system and a mixed integer nonlinear model (MINLP) for the task of simultaneously finding an optimal layout and adjustment of the screening system. We analyze some properties of these problems, show that they are non-convex and that the model for a fixed layout agrees with the dynamical approach presented in Valkama [79, 80]. The examination also illustrates that the

system of equations resulting from fixing the layout and the reject rates has the property that it has exactly one solution.

As pointed out in the introduction we are concerned with a multi-criteria optimization problem. We want to minimize the amount of contaminations in the total accept, and at the same time minimize the loss of valuable fibers. There are at least two possibilities to handle this problem. On the one hand one might introduce weights for the different components and then a weighted objective function, and on the other hand one can introduce a constraint setting a bound on one (or several) objective(s) and minimize a single remaining one. For an introduction to multi-criteria optimization we refer to Ehrgott [18]. In practice it seems to be very difficult to obtain reasonable costs and gains for the different components, but it appears to be much easier to get meaningful bounds for the percentage of contaminations admissible in the accept of the machine. Thus we decided to use the latter approach.

3.1 NLP for a Fixed Layout

Let S denote the set of screens, $V = S \cup \{\mathbf{in}, \mathbf{acc}, \mathbf{rej}\}$ be the set of nodes in the network with source **in** denoting the total inlet into the machine, and sinks **acc** and **rej** denoting the total reject and accept of the machine, respectively. Let $K = K^+ \cup K^-$ name the set of components (valuable ones and contaminations) that shall be separated, e.g., 'fibers' $\in K^+$ and 'stickies' $\in K^-$. Let $P \subset V \times V$ be the set of arcs in the network, i.e., the possible pipes connecting the different screens among each other or with the total feed, accept and reject.

For each component $k \in K$ we assume a mass flow out of the source (total inlet) $m_{\mathbf{in},k} > 0$ to be given. Furthermore, parameters $\beta_{s,k}$, for $s \in S, k \in K$ associated with the plug flow model are given. Let $q_k \in [0,1]$ be the maximal fraction of contamination $k \in K^-$ allowed in the total accept, and let l_k, u_k be lower and upper bounds on the mass flow of component k in each pipe. Let parameters $p^{\text{acc}}_{v,w}$ and $p^{\text{rej}}_{v,w} \in \{0,1\}$ specify whether there is a pipe connecting the accept or reject of screen $v \in S$ and the inflow of $w \in V$, respectively. Similarly, parameters $p_{\mathbf{in},v} \in \{0,1\}$ indicate whether the mass flow out of the source is an inflow for screen $v \in S$.

We introduce variables $m^{\text{in}}_{s,k}, m^{\text{acc}}_{s,k}, m^{\text{rej}}_{s,k} \in [l_k, u_k]$ for the mass flow of component $k \in K$ into or from accept or reject of screen $s \in S$, respectively. Similarly, for the two sinks we denote the mass flow into the total accept and reject by $m^{\text{in}}_{\mathbf{acc},k}, m^{\text{in}}_{\mathbf{rej},k} \in [l_k, u_k]$. The lower and upper bounds are estimated from the total inlet of component k and the bounds on the reject rates. The reject rate that can be adjusted at each screen $s \in S$ is denoted by $r_s \in [l_s, u_s]$, where $0 < l_s < u_s < 1$ are machine-depending bounds on the reject rate. Typical values are $l_s := 0.1$ and $u_s := 0.9$; for smaller values of l_s or larger values of u_s the (empirical) validity of the plug flow model is not guaranteed.

Using the above definitions, an NLP formulation for the problem of optimal adjustment of the reject rates in the screening system can be stated as follows:

$$\min \sum_{k \in K^+} m^{\text{in}}_{k,\mathbf{rej}}, \tag{5a}$$

subject to

$$m^{\text{in}}_{\mathbf{acc},k} \le q_k \cdot m_{\mathbf{in},k}, \ \forall\, k \in K^-, \tag{5b}$$

$$m^{\text{in}}_{s,k} = m^{\text{acc}}_{s,k} + m^{\text{rej}}_{s,k}, \ \forall\, k \in K, s \in S, \tag{5c}$$

$$m^{\text{rej}}_{s,k} = m^{\text{in}}_{s,k} \cdot r_s^{\beta_{s,k}}, \ \forall\, k \in K, s \in S, \tag{5d}$$

$$m^{\text{in}}_{v,k} = p_{\mathbf{in},v} m_{\mathbf{in},k} + \sum_{\substack{s \in S:(s,v) \in P,\\ \sigma \in \{\text{acc,rej}\}}} p^{\sigma}_{s,v} m^{\sigma}_{s,k}, \ \forall\, k \in K, v \in V, \tag{5e}$$

$$l_s \le r_s \le u_s, \ \forall\, s \in S, \tag{5f}$$

$$l_k \le m^{\sigma}_{s,k} \le u_k, \ \forall\, k \in K, s \in S, \sigma \in \{\text{in,acc,rej}\}. \tag{5g}$$

The multi-criteria objective is modeled by equations (5a) and (5b). Equations (5c) guarantee the mass balance for every component in each screen, and equations (5d) are due to the separation process in each screen modeled via the plug-flow model. The given layout is integrated into the model by the equations (5e). Finally, trivial bound constraints are given by (5f) and (5g).

3.2 MINLP for Layout Optimization

There are a lot of possibilities to connect the screens. The number of "reasonable" layouts is increasing rapidly in the number of screens, see Table 1. These values were obtained by enumerating all combinations of screens satisfying the following requirements: Only accept (reject) streams were allowed to enter the total accept (reject), the accept and reject streams of one screen cannot flow together into one screen, direct back-flow into the same screen is forbidden if this is the only destination, and the whole screening system has to be connected.

number of screens	number of possible layouts
1	1
2	8
3	318
4	26,688
5	3,750,240

Table 1: Possible layouts

The layout decision can be taken into the model by replacing the parameters $p^{\sigma}_{v,w}$ by binary variables. The expressions $p^{\sigma}_{v,w} \cdot m^{\sigma}_{v,k}$ in equations (5e) then also become nonlinear, but they can be linearized, which we describe in the sequel.

We introduce new variables $m^{o}_{v,w,k}$ for $(v,w) \in P, k \in K, o \in \{\text{in,acc,rej}\}$ denoting the flow of component k from node v to node w on an o-pipe. That is, now m denotes mass streams on the arcs of the network, whereas the m-variables before were associated with the nodes.

Figure 3 shows an example for a superstructure for two screens. The gray arrows describe potential pipes.

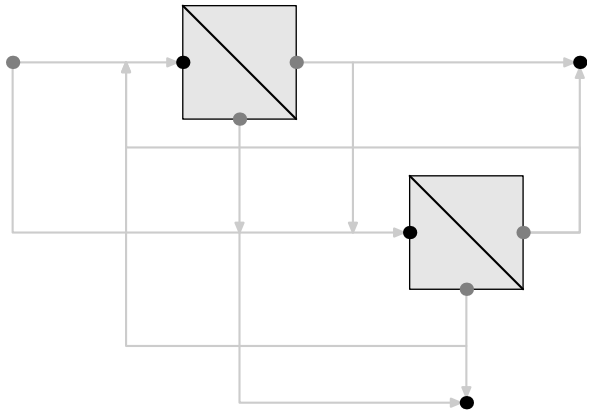

Figure 3: Example of a superstructure of two screens.

$$\min \sum_{\substack{k\in K^+,\\ v:(v,\mathbf{rej})\in P}} m^{\text{rej}}_{v,\mathbf{rej},k}, \tag{6a}$$

subject to

$$\sum_{s\in S:(s,\mathbf{acc})\in P} m^{\text{acc}}_{s,\mathbf{acc},k} \le q_k \cdot m_{\mathbf{in},k},\ \forall\, k \in K^-, \tag{6b}$$

$$\sum_{s\in S} m_{\mathbf{in},s,k} = m_{\mathbf{in},k},\ \forall\, k \in K, \tag{6c}$$

$$m_{\mathbf{in},s,k} + \sum_{\substack{v\in V:(v,s)\in P,\\ o\in\{\text{acc,rej}\}}} m^{o}_{v,s,k} = \sum_{\substack{w\in V:(s,w)\in P,\\ \sigma\in\{\text{acc,rej}\}}} m^{\sigma}_{s,w,k},\ \forall\, k \in K, s \in S, \tag{6d}$$

$$\sum_{w:(s,w)\in P} m^{\text{rej}}_{s,w,k} = (m_{\mathbf{in},s,k} + \sum_{\substack{v\in V:(v,s)\in P,\\ \sigma\in\{\text{acc,rej}\}}} m^{\sigma}_{v,s,k}) \cdot r_s^{\beta_{s,k}},\ \forall\, k \in K, s \in S, \tag{6e}$$

$$l_k \cdot p^{\sigma}_{v,w} \le m^{\sigma}_{v,w,k} \le u_k \cdot p^{\sigma}_{v,w},\ \forall\, k \in K, (v,w) \in P, \sigma \in \{\text{acc,rej}\}, \tag{6f}$$

$$u_k \cdot p_{\mathbf{in},s} \le m_{\mathbf{in},s,k} \le u_k \cdot p_{\mathbf{in},s},\ \forall\, k \in K, s \in S : (\mathbf{in}, s) \in P, \tag{6g}$$

$$\sum_{w\in V:(s,w)\in P} p^{\sigma}_{s,w} = 1,\ \forall\, s \in S, \sigma \in \{\text{acc,rej}\}, \tag{6h}$$

$$\sum_{s\in S:(\mathbf{in},s)\in P} p_{\mathbf{in},s} = 1, \tag{6i}$$

$$l_s \le r_s \le u_s,\ \forall\, s \in S, \tag{6j}$$

$$l_k \le m^{\sigma}_{v,w,k} \le u_k,\ \forall\, k \in K, (v,w) \in P, \sigma \in \{\text{in,acc,rej}\}, \tag{6k}$$

$$p^{\sigma}_{v,w} \in \{0,1\},\ \forall\, (v,w) \in P, \sigma \in \{\text{in,acc,rej}\}. \tag{6l}$$

Equations (6a) and (6b) model the multicriteria objective. Constraints (6c) ensure that the total inlet to the screening system is let into the screening unit connected to the feed. Equations (6d) and (6e) ensure the mass balance and the separation process via the plug flow model, respectively. Constraints (6f) to (6i) give the linearized layout description. Inequalities (6f) and (6g) ensure that there is only nonzero mass flow from node v to node w, if and only if there is a pipe connecting v and w. Equalities (6h) and (6i) guarantee that there is exactly one accept and one reject pipe leaving each screen, and that exactly one pipe is leaving the total feed. Note that the set P defines only meaningful possible pipes, for example, no pipes between the reject of some screen and the total accept do exist. If we fix the variables p concerning the layout, we obtain again a model for a given layout similar to the one given in Section 3.

3.3 Properties of the Models

In the following we discuss some properties of the introduced models.

3.3.1 Non-Convexity

The nonlinear functions $f(x,y) = xy^\beta$ occurring in the models as equations (5d) and (6e) are neither convex nor concave. The hessian of $f, -f$, respectively, is not positive definite, as

$$H_f = \begin{pmatrix} 0 & \pm\beta y^{\beta-1} \\ \pm\beta y^{\beta-1} & \pm\beta(\beta-1)x\, y^{\beta-2} \end{pmatrix} \tag{7}$$

has the determinant

$$\det(H_f) = -(\beta y^{\beta-1})^2 < 0, \tag{8}$$

for $y, \beta \neq 0$. Note that f is a positive monotone transformation of a concave function, i.e.,

$$f(x,y) = \exp(\ln(x) + \beta \ln(y)), \tag{9}$$

and therefore quasi-concave (cf. Arrow and Enthoven [4]), i.e., the level sets

$$\{(x,y) \in \mathbb{R}^2 : \{f(x,y) \ge z\}\} \tag{10}$$

are convex for every $z \in \mathbb{R}$.

3.3.2 Relation to Dynamical Models

In the introduction we already mentioned that the problem of optimizing the operation of a fixed screening system has already been studied by Valkama [79, 80] using a simulation-based approach. There, the reject rates are discretized, and a simulation for each combination of reject rate values is run. The simulation is done in discrete time steps, and the objective function is evaluated when the system is in balance. By comparison the most beneficial reject rate combinations are then determined. In the following, we show that our model only considering the system in its steady-state results in the same set of feasible solutions as the dynamical approach in [80]. For the reader's convenience we briefly outline this dynamical model in the sequel.

Let $x_i^t, i \in \{1, \dots, n+2\}, t \geq 0$ denote the mass stream entering node i at time t, where the first n indices correspond to the screens installed in the screening system and the second last index, $n+1$, and the last index, $n+2$, to the total accept and reject, respectively. For easier notation, we here omit the index k indicating the component. As before, let $p_{j,i}^{\text{acc/rej}}$ indicate whether there is a pipe connecting j and i via an accept or reject pipe, respectively. Let r_j and β_j be the reject rate and β-parameter of screen j, respectively. Let $b_i = p_{\mathbf{in},j} m_{\mathbf{in},k}$ for $i \in \{1, \dots, n\}$ be the mass flowing from outside into the system. Then the dynamical system introduced in [80] can be written as follows:

$$x_i^{t+1} = \sum_{j \in \{1,\dots,n\}} \left(p_{j,i}^{\text{acc}} (1 - r_j^{\beta_j}) + p_{j,i}^{\text{rej}} r_j^{\beta_j} \right) x_j^t, \qquad \text{for } i = 1, \dots, n, \tag{11}$$

$$x_{n+1}^{t+1} = \sum_{j \in \{1,\dots,n\}} p_{j,\mathbf{acc}}^{\text{acc}} (1 - r_j^{\beta_j}) x_j^t, \tag{12}$$

$$x_{n+2}^{t+1} = \sum_{j \in \{1,\dots,n\}} p_{j,\mathbf{rej}}^{\text{rej}} r_j^{\beta_j} x_j^t. \tag{13}$$

In matrix notation the system can be rewritten as

$$x^{t+1} = T^t x^t + b, \quad t \in \mathbb{N}, \quad x_0 = 0, \tag{14}$$

where

$$T = \begin{pmatrix} C & 0 \\ D & 0 \end{pmatrix} \in \mathbb{R}^{(n+2)\times(n+2)}, \tag{15}$$

and $C \in \mathbb{R}^{n \times n}$ with $c_{ij} = p_{j,i}^{\text{acc}}(1 - r_j^{\beta_j}) + p_{j,i}^{\text{rej}} r_j^{\beta_j}$ being the fraction of mass floating from screen j to screen i for $i \neq j$, and $c_{ii} = 0$. Moreover, $D \in \mathbb{R}^{2 \times n}$ with $d_{1j} = p_{j,\mathbf{acc}}^{\text{acc}}(1 - r_j^{\beta_j})$ and $d_{2j} = p_{j,\mathbf{rej}}^{\text{rej}} r_j^{\beta_j}$ being the fraction of mass floating from screen j to the total accept or reject, respectively. The vector $b \in \mathbb{R}^{n+2}$ with $b_i = p_{\mathbf{in},j} m_{\mathbf{in},k}$ for $i \in \{1 \dots n\}$ and $b_{n+1} = b_{n+2} = 0$ gives the mass flowing from outside into the system. Finally, the vector x^t gives the mass floating into the nodes at time t. Note that this simulation model itself does not include an objective function. To find the best reject rate assignment, [80] proposes an optimization-by-simulation approach.

A steady-state equilibrium of the system is given by $(I-T)^{-1}b$, if and only $(I-T)$ is nonsingular (cf. Galor [27]). Note that this dynamical system is an explicit Euler discretization (cf. Schwarz and Klöckler [68]) with step size equal one of the differential equation $x' = (T-1)x + b, x(0) = 0$, which is in balance if $x' = 0$, i.e., $(I-T)x = b$.

We rewrite the constraints of the steady-state NLP model introduced in Section 3.1 disregarding the ones from the objective. Using the notation introduced above we obtain

$$x = Tx + b, \tag{16}$$

or equivalently

$$(I - T)x = b. \tag{17}$$

Observe that equation (17) is equal to the steady-state equilibrium equation of the dynamical system (14).

In the following we show that the matrix $(I-T)$ is nonsingular for r, p, and β fixed. Therefore, a unique steady-state equilibrium of the dynamical system (14) exists which agrees with the solution of the model (5).

Theorem 3.1 *Let $A \in \mathbb{R}^{n \times n}$ be a matrix with the following properties:*

a) The diagonal entries of A are nonzero.

b) A^T is diagonally dominant, and for at least one row strictly diagonally dominant.

c) For any permutation P with

$$PAP^{-1} = \begin{pmatrix} B_{11} & 0 \\ B_{21} & B_{22} \end{pmatrix},$$

the square matrices B_{11}^T and B_{22}^T are diagonally dominant, and for at least one row strictly diagonally dominant.

Then A is nonsingular.

Proof. We will prove the theorem via induction over n. If $n = 1$, then $A = a_{11} \neq 0$, by assumption a). Assume that any matrix $A \in \mathbb{R}^{k\times k}, k \leq n$ fulfilling assumptions a), b), c), is nonsingular. Let $A \in \mathbb{R}^{(n+1)\times(n+1)}$. We distinguish the following two cases:

1. The matrix A^T is irreducible and hence irreducibly diagonal dominant, and therefore nonsingular by the Diagonal Dominance Theorem (cf. Ortega [62]).

2. There exists a permutation matrix P with

$$PAP^{-1} = \begin{pmatrix} B_{11} & 0 \\ B_{21} & B_{22} \end{pmatrix},$$

where B_{11} and B_{22} are square matrices. Then, assumption c) implies that B_{11}^T and B_{22}^T are also diagonally dominant, and for at least one row strictly diagonally dominant. Furthermore, the diagonal entries of B_{11} and B_{22} are nonzero, since P permutes the columns and rows simultaneously and hence the diagonal entries of PAP^{-1} are a permutation of the diagonal entries of A.

In addition, if for B_{11} (and similarly for B_{22}) there exists a permutation $\tilde{P}$ with

$$\tilde{P}B_{11}\tilde{P}^{-1} = \begin{pmatrix} \tilde{B}_{11} & 0 \\ \tilde{B}_{21} & \tilde{B}_{22} \end{pmatrix},$$

then

$$\operatorname{diag}(\tilde{P}, I)PAP^{-1}\operatorname{diag}(\tilde{P}^{-1}, I) = \begin{pmatrix} \tilde{B}_{11} & 0 & 0 \\ \tilde{B}_{21} & \tilde{B}_{22} & 0 \\ B_{21} & & B_{22} \end{pmatrix},$$

is also a permutation of A with $\tilde{B}_{11}$ being a square matrix. Hence $\tilde{B}_{11}^\mathsf{T}$, and similarly $\tilde{B}_{22}^\mathsf{T}$ are diagonally dominant, and for at least one row strictly diagonally dominant by c).

Therefore, a), b), c) hold for B_{11} and B_{22}. Hence B_{11} and B_{22} are nonsingular by the assumption since $\dim(B_{ii}) \leq n$ for $i = 1, 2$.

So $\det(A) = \det(B_{11}) \det(B_{22}) \neq 0$.

Therefore, Theorem 3.1 holds for all $n \in \mathbb{N}$. □

Corollary 3.2 *The matrix $I - T$ defined as in* (14) *is nonsingular.*

Proof. Remember the block structure of T and define $A := I - C$. Then

$$I - T = \begin{pmatrix} A & 0 \\ -D & I \end{pmatrix} \tag{18}$$

is nonsingular if and only if A is nonsingular. In the following we show that A satisfies the assumptions of Theorem 3.1. Without loss of generality, we may assume that the system is connected.

a) By definition of C the diagonal entries c_{ii} for all $i \in \{1, \ldots, n\}$ are zero, and hence the diagonal entries of A are one.

b) By definition, $c_{ij} \geq 0$ for $i \neq j$ is the fraction of the feed of screen j which is floating into screen i. These fractions together with the fractions of the feed of screen j floating from j to the sinks have to sum up to one. As both sinks have to be fed, the sum of at least one column has to be strictly smaller than one. Hence $\sum_{i \neq j} |a_{ij}| = \sum_i c_{ij} \leq 1$, and for at least one column the inequality strictly holds.

c) Let P be a permutation matrix with

$$PCP^{-1} = F = \begin{pmatrix} F_{11} & 0 \\ F_{21} & F_{22} \end{pmatrix}$$

and F_{11}, F_{22} square matrices. This permutation of the rows and columns corresponds to a permutation of the screens in the system, since the permutation permutes the rows and columns of C simultaneously. Hence as in b) we have that $\sum_i f_{ij} \leq 1$ for all columns j of F, and hence for each column of F_{11} and F_{22}.

Let $\{1, \ldots, n_1\}$ denote the indices of the rows and columns of F_{11}, and $\{n_1 + 1, \ldots, n_1 + n_2\}$ denote the indices of the rows and columns of F_{22}. Now, suppose $\sum_{i \neq j} f_{ij} = 1$ for all columns $j \in \{n_1 + 1, \ldots, n_1 + n_2\}$ of F_{22}. Since we assume that the system is connected, there will be mass entering the subsystem corresponding to the rows $i \in \{n_1 + 1, \ldots, n_1 + n_2\}$. But then, the supposition means that the mass entering the subsystem does not leave it anymore, as then for at least one column j we would have $\sum_{i \neq j} f_{ij} < 1$. But this contradicts the overall conservation of mass. With the same argument we have that $\sum_{i \in \{1,\ldots,n_1\} \setminus \{j\}} f_{ij} < 1$ for at least one column $j \in \{1, \ldots, n_1\}$.

Since the non-diagonal entries of A and C only differ in the sign, and P permutes the rows and columns simultaneously, the assumption also holds for A.

□

We have shown that for every choice of reject rates and layout, our steady state model agrees with the dynamical model. That means, it is sufficient to analyze the system in its steady-state, the consideration of time steps is not necessary.

4 Linear Approximations and Linear Relaxations

The techniques of mixed-integer linear programming (MILP), in particular, linear programming based branch-and-cut methods, have already shown their merits for many discrete-combinatorial decision problems in general, see Nemhauser and Wolsey [60], or Wolsey [83] for an introduction, and topology optimization problems in particular, for example, in telecommunication (cf. Bley [12]) or mechanical engineering (cf. Fügenschuh and Fügenschuh [25]). They are able to find global optima in finite time or estimate the optimality gap for suboptimal solutions. However, these methods are not able to deal with nonlinear constraints in the beginning. Therefore, we will discuss methods to transform nonlinearities to piecewise linear functions that can be incorporated in MILPs. Methods for including piecewise linear functions of one dimension in an MILP have already been introduced by Markowitz and Manne [50], Dantzig [16], or Nemhauser and Wolsey [60]. These ideas have been extended to higher dimensions by Lee [43], Wilson [82], Lee and Wilson [44], or Moritz [55]. Recent developments and an overview over so far known methods are presented by Nemhauser and Vielma [59], or Ahmed, Nemhauser, and Vielma [2]. We shortly give the different formulations in the Appendix and compare the approaches computationally in Section 5. In many previous applications, e.g., by Dantzig, Johnson, and White [15], Fügenschuh et al. [26], or Moritz [55], the given nonlinear functions have been interpolated, and the resulting interpolation function has been integrated in an MILP. We will generalize the ideas to arbitrary piecewise linear approximations. While an interpolation function coincides with the value of the given function in the vertices of the grid, a general approximation function does not necessarily have this property.

We will follow two approaches. On the one hand, the bivariate function resulting from the plug flow model is approximated directly over a triangulation. On the other hand, a logarithmic transformation is applied to end up with the approximation of several univariate functions. Details of the latter can be found in Section 4.1.

Current state-of-the-art global optimization software make also use of the algorithmic advances in solving (mixed-integer) linear programming problems (e.g., see [74, 75, 9, 1]). They typically

consist of methods for finding good feasible solutions and of procedures computing global bounds on the optimal objective function value of the given MINLP (see also [51]). The bounds can be obtained by constructing and solving (mixed-integer) linear relaxations for the MINLP. Note that, on the contrary to the linear approximation models we discuss in Section 4.1, relaxations are required to contain all feasible points of the original MINLP in their solution set. Moreover, the objective functions of the relaxations have to underestimate the original objective function over the relevant domain. A common approach to construct such relaxations is to replace each nonlinear term appearing in the MINLP by a finite set of linear under- and overestimating functions (e.g., see [74, 9]). In Section 4.2, we briefly discuss how to derive linear relaxations for our layout-optimization model (6). In Section 5 we apply global optimization software to our numerical examples and compare the results with the results obtained by our linear approximation models.

4.1 Transformation of Bivariate to Univariate Functions

Recall that the nonlinear equalities occurring in the models (5) and (6) are of the form

$$z = x\, y^{\beta}. \tag{19}$$

Let us first assume $x, y > 0$. Then applying the logarithm on both sides yields

$$\ln(z) = \ln(x) + \beta \ln(y). \tag{20}$$

Now we introduce new variables

$$Z = \ln(z), \quad X = \ln(x), \quad Y = \ln(y) \tag{21}$$

and the constraint

$$Z = X + \beta\, Y. \tag{22}$$

This way the problem is reduced to three nonlinear functions in one variable (21) and one linear constraint (22).

More generally, a transformation of inseparable functions to a separable one has already been proposed by Tomlin [78] or Wilson [82]. A separable function is of the form $f(x_1, x_2, \ldots, x_n) = \sum_{j=1}^{n} f_j(x_j)$, where each f_j is a function of one variable. The approximation of such a higher-dimensional function f can be reduced to the approximation of several one-dimensional functions f_j.

This transformation gives rise to the following question: How exact do we have to approximate the logarithm to guarantee that the two-dimensional function $f : \mathbb{R}_+^2 \mapsto \mathbb{R}_+$ with $f(x, y) := x\, y^{\beta}$ is approximated within a given tolerance when we use the above transformation? The answer allows us to compare the approaches: On the one hand the direct approximation of the bivariate function and on the other the transformation method. To this end we introduce the following error measure.

Definition 4.1 *Let $D \subset \mathbb{R}^n$ be a compact set. Let $f : D \to \mathbb{R}_+$ be a continuous function and $\tilde{f} : D \to \mathbb{R}$ an approximation to f. Then we define the* absolute error *on D as*

$$\mathrm{err}_{\mathrm{abs}}(f, \tilde{f}, D) := \max_{x \in D} |f(x) - \tilde{f}(x)|, \tag{23}$$

the relative error *on D as*

$$\mathrm{err}_{\mathrm{rel}}(f, \tilde{f}, D) := \max_{x \in D} \left|\frac{f(x) - \tilde{f}(x)}{f(x)}\right|, \tag{24}$$

and for $\delta > 0$ the mixed error *on D as*

$$\mathrm{err}_{\mathrm{mix}}(f, \tilde{f}, D, \delta) := \max_{x \in D} \frac{|f(x) - \tilde{f}(x)|}{|f(x)| + \delta}. \tag{25}$$

Since the relative error is huge for $f(x)$ close to zero, we introduce $\delta > 0$ as a small margin value in the above definition of the mixed error. If $f(x)$ is large, the mixed error is close to the relative error, whereas it is closer to the absolute error if $f(x)$ is small. As abbreviation we write 'err' or η instead of $\mathrm{err}_{\mathrm{mix}}(f, \tilde{f}, D, \delta)$ if the respective arguments are clear from the context.

Theorem 4.1 *Let $\delta > 0$. Let $h(x)$ be a function approximating the logarithm with error ε, i.e., $\max_x |h(x) - \ln(x)| = \varepsilon$. Assume h and h^{-1} are monotonically increasing. Then the overall error made by approximating the function $(x, y) \mapsto x\, y^\beta > 0$ for $x, y \in \mathbb{R}, x, y > 0$, by the logarithmic transformation is bounded by*

$$\text{err} := \max \frac{|h^{-1}(h(x) + \beta\, h(y)) - x\, y^\beta|}{x\, y^\beta + \delta} \leq \exp\left((2 + \beta)\varepsilon\right) - 1. \tag{26}$$

Proof. We distinguish two cases with respect to the sign of the numerator.
Case 1:

$$\begin{aligned}
\text{err} &= \frac{h^{-1}(h(x) + \beta\, h(y)) - x\, y^\beta}{x\, y^\beta + \delta} \\
&\leq \frac{h^{-1}(\ln(x) + \varepsilon + \beta(\ln(y) + \varepsilon)) - x\, y^\beta}{x\, y^\beta} \\
&\leq \frac{\exp(\ln(x) + \varepsilon + \beta(\ln(y) + \varepsilon) + \varepsilon) - x\, y^\beta}{x\, y^\beta} \\
&\leq \exp\left((2 + \beta)\varepsilon\right) - 1.
\end{aligned}$$

Case 2:

$$\begin{aligned}
\text{err} &= \frac{-h^{-1}(h(x) + \beta\, h(y)) + x\, y^\beta}{x\, y^\beta + \delta} \\
&\leq \frac{-h^{-1}(\ln(x) - \varepsilon + \beta(\ln(y) - \varepsilon)) + x\, y^\beta}{x\, y^\beta} \\
&\leq \frac{-\exp(\ln(x) - \varepsilon + \beta(\ln(y) - \varepsilon) - \varepsilon) + x\, y^\beta}{x\, y^\beta} \\
&\leq 1 - \exp\left((2 + \beta)(-\varepsilon)\right).
\end{aligned}$$

The first term dominates the second one:

$$\begin{aligned}
& \left[\exp\left((2 + \beta)\varepsilon\right) - 1\right] - \left[1 - \exp\left((2 + \beta)(-\varepsilon)\right)\right] \\
= \;& \exp\left((2 + \beta)\varepsilon\right) + \exp\left((2 + \beta)(-\varepsilon)\right)) - 2 \\
= \;& 2 \cdot \cosh((2 + \beta)\varepsilon) - 2 \\
\leq \;& 2 \cdot \cosh(0) - 2 = 2 - 2 = 0.
\end{aligned}$$

So the overall error is bounded by $\exp\left((2 + \beta)\varepsilon\right) - 1$. □

Note that we use the absolute (and not the relative) error in the univariate logarithm functions to determine the relative error of the bivariate plug flow function. This is due to the fact that in the proof we could make use of a relationship between the absolute error in the approximation of the logarithm and the relative error in the approximation of the exponential function by the inverse of the approximating function of the logarithm.

We further point out that the result is independent of the actual choice of $\delta > 0$.

As an immediate consequence of Theorem 4.1 the tolerance ε for the approximation of the logarithms has to be bounded by

$$\varepsilon \leq \frac{\ln(\eta + 1)}{2 + \beta}, \tag{27}$$

in order to ensure an overall tolerance of η for the plug flow equation.

4.1.1 Shifting the Argument

Recall that we assumed $x, y > 0$ in the previous section. In our model, we can be sure that the reject rate y is strictly positive. But assuming this for the mass stream x might be too restrictive. Nevertheless, we modify the derivation from above using a shifting of the argument, so that the case of $x \geq 0$ is also covered.

Let us assume that the variable y is bounded below by $l_y > 0$ and $x \geq l_x > -\alpha$, with $\alpha > 0$. Consider the expansion

$$(x + \alpha)y^\beta = x\, y^\beta + \alpha\, y^\beta. \tag{28}$$

By assumption both sides of the equation are positive, so we can take the logarithms on both sides:

$$\ln(x+\alpha) + \beta \cdot \ln(y) = \ln(x\, y^\beta + \alpha\, y^\beta). \tag{29}$$

Let $z = x\, y^\beta, v = z + \alpha y^\beta$, and $w = v - z = \alpha y^\beta > 0$. Taking the logarithms we obtain

$$\ln(w) = \ln(\alpha) + \beta \ln(y). \tag{30}$$

Let h be a piecewise linear approximation of the logarithm. Then the function $(x, y) \mapsto x\, y^\beta =: z$ can be approximated by the following set of equalities.

$$h(w) = \ln(\alpha) + \beta h(y), \quad w = v - z, \quad h(v) = h(x+\alpha) + \beta h(y). \tag{31}$$

Theorem 4.2 *Let $\alpha, \delta > 0$. Let $h(x)$ be a function approximating the logarithm with error ε, i.e., $\max_x |h(x) - \ln(x)| = \varepsilon$. Assume h and h^{-1} are monotonically increasing and $y^\beta \leq 1$. Then the overall error made by approximating the function $(x, y) \mapsto x\, y^\beta \geq 0$ for $x, y \in \mathbb{R}, x \geq 0, y > 0$, by the shifted logarithmic transformation is bounded by*

$$\begin{aligned} \text{err} &:= \max \frac{|h^{-1}(h(x+\alpha) + \beta h(y)) - h^{-1}(\ln(\alpha) + \beta h(y)) - x\, y^\beta|}{x\, y^\beta + \delta} \\ &\leq \frac{\alpha}{\delta} \left[\exp((2+\beta)\varepsilon) - \exp((-1-\beta)\varepsilon)\right] + \exp((2+\beta)\varepsilon) - 1. \end{aligned}$$

Proof. Similar to the proof of Theorem 4.1 one distinguishes two cases, and shows that the result of the first case dominates the second. Since the derivation of these cases is similar, we only present the first one here:

$$\begin{aligned} & \frac{h^{-1}(h(x+\alpha) + \beta\, h(y)) - h^{-1}(\ln(\alpha) + \beta\, h(y)) - x\, y^\beta}{x\, y^\beta + \delta} \\ \leq\ & \frac{\exp(\ln(x+\alpha) + \varepsilon + \beta(\ln(y) + \varepsilon) + \varepsilon) - \exp(\ln(\alpha) + \beta(h(y) - \varepsilon) - \varepsilon) - x\, y^\beta}{x\, y^\beta + \delta} \\ =\ & \frac{(x+\alpha) y^\beta \exp\left((2+\beta)\varepsilon\right) - \alpha y^\beta \exp\left((-1-\beta)\varepsilon\right) - x\, y^\beta}{x\, y^\delta + \delta} \\ =\ & \frac{x\, y^\beta \left[\exp\left((2+\beta)\varepsilon\right) - 1\right] + \alpha y^\beta \left[\exp\left((2+\beta)\varepsilon\right) - \exp\left((-1-\beta)\varepsilon\right)\right]}{x\, y^\beta + \delta} \\ =\ & \frac{x\, y^\beta \left[\exp\left((2+\beta)\varepsilon\right) - 1\right]}{x\, y^\beta + \delta} + \frac{\alpha y^\beta \left[\exp\left((2+\beta)\varepsilon\right) - \exp\left((-1-\beta)\varepsilon\right)\right]}{x\, y^\beta + \delta} \\ \leq\ & \exp\left((2+\beta)\varepsilon\right) - 1 + \frac{\alpha}{\delta} \exp\left((2+\beta)\varepsilon\right) - \exp\left((-1-\beta)\varepsilon\right) \\ =\ & \frac{\alpha}{\delta} \left[\exp((2+\beta)\varepsilon) - \exp((-1-\beta)\varepsilon)\right] + \exp((2+\beta)\varepsilon) - 1. \end{aligned}$$

□

4.2 Linear Relaxations

A widely used approach to construct (mixed-integer) linear relaxations for a given MINLP is, for each nonlinear function $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$, to introduce a new variable $z_f \in \mathbb{R}$ which is bounded from below by finitely many linear functions $\gamma_f^k : D \to \mathbb{R}$ underestimating f over D, $k = 1, \ldots, K_1$, and from above by finitely many linear functions $\gamma_f^k : D \to \mathbb{R}$ overestimating f over D, $k = 1, \ldots, K_2$ (e.g., see [51, 74, 9]).

To obtain tight relaxations, it is desirable to chose the linear estimators best possible. For this, the *convex* and *concave* envelopes of functions are investigated in the literature. They are defined to be the tightest convex under- and the tightest concave overestimating functions for the given function f over the relevant domain (e.g., see [51, 73]). Then, linear estimators can be, in principle, computed from supporting hyperplanes on the graphs of the envelopes (e.g., see [74, 75]).

In the following, we denote the convex and the concave envelope of f over D by $\text{vex}_D[f]$ and $\text{cave}_D[f]$, respectively. Furthermore, we assume that the function $f : D \subseteq \mathbb{R}^n \to \mathbb{R}$ under consideration is continuous over a convex compact domain $D \subseteq \mathbb{R}^n$. The envelopes are then given

by (e.g. see Rockafellar [64]).

$$\begin{array}{rcll} \mathrm{vex}_D[f](x) \ \big(\mathrm{cave}_D[f](x)\big) & = & \min\,(\max) & \sum_{k=1}^{n+1} \lambda_k \cdot f(x^k) \\ & & \text{s. t.} & \sum_{k=1}^{n+1} \lambda_k x^k = x, \\ & & & \sum_{k=1}^{n+1} \lambda_k \ \ = 1, \\ & & & \lambda_k \geq 0, \quad k = 1, \ldots, n+1, \\ & & & x^k \in D, \quad k = 1, \ldots, n+1. \end{array} \tag{32}$$

In order to determine linear estimators from the envelopes, we are interested in an algorithmically utilizable description of the envelopes. This can be achieved by solving the corresponding optimization problems given in (32) analytically, if possible, or by deriving structural results based on the specific properties of f over D that allow to transform the optimization problems into algorithmically tractable optimization problems and descriptions, respectively (e.g., see [72, 52, 37, 47, 46, 71, 39]).

Explicit formulas for the convex and/or the concave envelopes, mostly restricted to boxes in the non-negative orthant, are available for some important low-dimensional functions, including the product terms xy (cf. [51, 45]) and xyz ([53, 54]), and bivariate functions of the form $\frac{x}{y}$, xy^2 (cf. [72]), $\exp(xy)$ and $ax^2 + bxy + cy^2$ (cf. [37]), trivariate component-wise concave functions [52] and some well-structured convex (concave) extendable functions of arbitrary dimension (cf. [71]). Recently, convex envelopes for products of a convex function with a component-wise concave function have been investigated, and analytical formulas have been provided for many relevant subclasses (cf. [39, 40]). The subclasses, in particulary, involve functions of the form $x^a g(y)$, where $a \in \mathbb{R} \setminus [0, 1)]$, x^a is nonnegative and convex over its domain, and $g(y)$ is a univariate, positive and concave function (cf. [39, Cor. 2]).

Further structural results have been, for instance, developed for functions

- $f(x, y) : [l_x, u_x] \times [l_y, u_y] \subseteq \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$, where f is convex in $x \in [l_x, u_x]$, for every fixed $y \in [l_x, u_x]$, and f is concave in $y \in [l_y, u_y]$, for every fixed $x \in [l_y, u_y]$ (cf. [72]),
- $f(x) : [l, u] \subseteq \mathbb{R}^n \to \mathbb{R}$ being convex on $[l, u]$ whenever all but one variable are fixed to one of their values, and indefinite at any point in $[l, u]$ (cf. [37]),
- $f(x, y) : P \subseteq \mathbb{R}^2 \to \mathbb{R}$, where P is a triangle, and f is indefinite in the interior of P, each restriction of f to an edge of P is convex or concave, and, in case that f is strictly convex over each facets of P, then f is, in addition, strictly convex over a certain family of line segments contained in P (see [47], and see also [46] for extentions).

Recap that the only nonlinearities occurring in our model (6) are described by the function

$$f : [l_x, u_x] \times [l_y, u_y] \subseteq \mathbb{R}^2_{\geq 0} \to \mathbb{R}, \quad (x, y) \mapsto x \cdot y^\beta, \qquad \text{for fixed } \beta \in (0, 1).$$

In Section 3.3.1, it has been already pointed out that f is an indefinite function. Moreover, it is easy to check that f is linear in x (for every fixed $y \in [l_y, u_y] \subseteq \mathbb{R}_{\geq 0}$) and concave in y (for every fixed $x \in [l_x, u_x] \subseteq \mathbb{R}_{\geq 0}$).

Exploiting these properties the optimization problems corresponding to the convex and concave envelopes of our function f on $[l, u]$ can be simplified and solved analytically. This yields the following explicit formulas for the envelopes.

Lemma 4.3 *For an arbitrary number $\beta \in (0, 1)$, consider the bivariate function $f : \mathbb{R}^2 \to \mathbb{R}$, $(x, y) \mapsto xy^\beta$ restricted to a full-dimensional box $[l, u] := [l_x, u_x] \times [l_y, u_y] \subseteq \mathbb{R}^2_{\geq 0}$.*

(a) The convex envelope $\mathrm{vex}_{[l,u]}[f] : [l, u] \to \mathbb{R}$ of f on $[l, u]$ reads as

$$\begin{array}{rcl} \mathrm{vex}_{[l,u]}[f](x, y) & = & \max\big\{(l_y)^\beta x + l_x \frac{(u_y)^\beta - (l_y)^\beta}{u_y - l_y} y - l_x \frac{(u_y)^\beta - (l_y)^\beta}{u_y - l_y} l_y, \\ & & \qquad (u_y)^\beta x + u_x \frac{(u_y)^\beta - (l_y)^\beta}{u_y - l_y} y - u_x \frac{(u_y)^\beta - (l_y)^\beta}{u_y - l_y} u_y \big\}. \end{array} \tag{33}$$

(b) The concave envelope $\mathrm{cave}_{[l,u]}[f] : [l, u] \to \mathbb{R}$ of f on $[l, u]$ is given by

$$\mathrm{cave}_{[l,u]}[f](x, y) \quad = \quad \begin{cases} x \cdot y^\beta, & \text{if } x \in \{l_x, u_x\}, \\ \lambda l_x (r^\star)^\beta + \lambda u_x \left(\frac{y}{1-\lambda} - \frac{\lambda}{1-\lambda} r^\star\right)^\beta, & \text{if } l_x < x < u_x, \end{cases} \tag{34}$$

where $\lambda = \frac{u_x - x}{u_x - l_x}$ *and* $r^\star = \text{med}\{r_{\min}, \bar{r}, r_{\max}\}$ *with* $r_{\min} = \max\{l_y, \frac{y}{\lambda} - \frac{1-\lambda}{\lambda} u_y\}$,

$$\bar{r} = \frac{l_x^{1/(1-\beta)}}{(1-\lambda)\ u_x^{1/(1-\beta)} + \lambda\ l_x^{1/(1-\beta)}} \quad \text{and} \quad r_{\max} = \min\{l_y, \tfrac{y}{\lambda} - \tfrac{1-\lambda}{\lambda} l_y\},$$

and the operator $\text{med}\{\cdot,\cdot,\cdot\}$ *selects the middle value out of three numbers.*

Results and techniques to derive the explicit formulas for the envelopes of our function $f(x, y) = xy^\beta$ are well-established and have been demonstrated on very similar functions in the literature. Therefore, we will not present a proof here and refer to the corresponding literature, instead. For the convex envelope given in Lemma 4.3 (a), we refer to [72, 10] and references therein. For the concave envelope given in Lemma 4.3 (b), see [72, 73], [37, Obs. 2], [39, Thm. 2], and the examples discussed therein.

Note that the explicit formula for the convex envelope as given in Lemma 4.3(a) already provides two linear functions that underestimate $x \cdot y^\beta$ on $[l_x, u_x] \times [l_y, u_y]$. Linear functions overestimating $x \cdot y^\beta$ on $[l_x, u_x] \times [l_y, u_y]$ are given by supporting hyperplanes on the graph of the concave envelope. At a given point $(x, y) \in [l_x, u_x]$, such a supporting hyperplane can be easily computed with the help of Equation (34) using elementary linear algebra (e.g., see [6] and also [47]). For bivariate functions of the form $x^p y^q$, $(p, q \in \mathbb{R})$, on $\mathbb{R}^2_{\geq 0}$, and $ax^2 + bxy + cy^2$, $(a, b, c \in \mathbb{R})$, a constraint handler determining linear over- and under-estimators this way is available in the software package SCIP 2.1.0 [1]. We refer to [6], for more details. We will use this constraint handler to solve our numerical examples in the next section.

5 Computational Results

In the sequel we present and compare computational results for both, the NLP model with a fixed layout in Section 5.1 and the MINLP model for layout optimization in Section 5.2. These results are achieved by a direct approximation of the bivariate nonlinear function via triangulation, by two different transformations to univariate functions. We compare the CPU times of different linearization approaches and approximation accuracies. As linearization approaches to integrate piecewise linear function in the formulation (PWL-method, for short) we make use of the logarithmic (disaggregated) convex combination method (Log or DLog, for short), the convex combination method (CC), the special ordered set method (SOS2), and the incremental model (Inc). A detailed description of these techniques is given in the appendix. The approximation accuracy is set to 0.1, 0.01, and 0.001, respectively. Further we use two different MILP solvers, the commercial CPLEX 12.4.0.0 [35] and the academic SCIP 2.1.0 [1] with SoPlex 1.6.0 as underlying LP solver [84]. The modeling language Zimpl 3.2.0 [41] is used for translating the models into LP-input files for both solvers. The time limit was set to 12 hours per run, and the relative duality gap (i.e., the difference between upper and lower bound divided by the primal bound) was set to 0.0%. Other than that we use the default settings for both solvers. We also make use of two different MINLP solvers, the commercial Baron 10.2.0 [74, 66], and again the academic SCIP 2.1.0. The modeling language GAMS 23.8.1 [65] is used to set up the models for these nonlinear solvers. Our computational experiments were carried out on a Intel Core i7 CPU 870 running at 2.93 GHz on 4 cores, 8 MB cache, and 16 GB RAM. As operation system, openSUSE 12.1 Linux with kernel version 3.1.10-1.9 was used.

In all our computations we consider two components in the flow, namely R14, the most important fiber class, and macro-stickies. The system is made up of three screens. The corresponding parameters for the plug flow model are taken from [79, 80] and shown in Table 2. The mass inflow to the screening system is measured as 6.75t/h of R14 fibers. For our computations this value is scaled to 0.675 [10t/h]. Furthermore we have 100000 mm^2/kg of stickies, scaled to 1 [10^5mm^2/kg]. The fraction of incoming stickies that is maximally allowed in the total accept is set to 10%. The bounds for the reject rates are set to $[0.1, 0.9]$. The bounds of the mass flows are set to $[0, 10]$ for each component.

Table 3 shows the scale of the used approximations. For both univariate approaches, the shifted and the non-shifted, the total sum of points needed for the approximation of the logarithms is given. For the bivariate approach the average over all betas of the total number of triangles and points, respectively, needed for the approximation of one two-dimensional function is given. Figure 4 exemplary shows two bivariate piecewise linear approximations.

component	screen 1	screen 2	screen 3
macro-stickies area	0.29	0.13	0.06
R14	0.74	0.79	0.71

Table 2: β-parameters used in the plug flow equation.

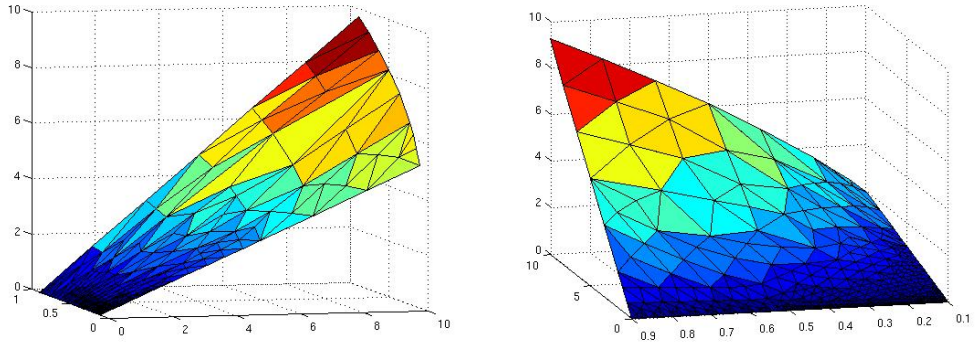

Figure 4: Bivariate piecewise linear approximations for error tolerance 0.01 for $\beta = 0.29$ and $\beta = 0.71$, respectively.

All tables below containing the computational results (i.e., Tables 4 – 6 and Tables 9 – 11) show the running time in seconds (in the respective upper row) and processed nodes in the branch and bound tree (in the lower row) for each PWL-method and solver used, if the solver was able to finish the computations within the given time limit of 12 hours. If not, the computations are terminated, and in the upper row we give the relative duality gap as a percentage value, whereas the lower row shows the number of processed nodes. If no feasible solution was found within the given time limit, the gap is infinity ($+\infty$). If not even the first (root) LP relaxation of the model could be solved, we mark it with a dash ($-$). Values in boldface letters show the best (that is, the one with the least CPU time or the least number of processed branch-and-bound nodes) results per column.

5.1 Partial Cascade

In this part the results obtained by optimizing a fixed layout are given. That is, we compute the best reject rates for a partial cascade of three screens, see Figure 1. Tables 4 – 6 show the running times and processed nodes for the different MILP approaches.

We start with the bivariate approach, see Table 4, and compare the convex combination (CC), the incremental (Inc), and the disaggregated convex combination method (DLog). For both solvers, CPLEX and SCIP, the CC method yields the shortest running times and the least number of branch-and-bound nodes if a coarse $\eta = 0.1$ approximation level was chosen. For medium (0.01) and fine (0.001) approximation levels there is no clear winner among these two methods. The relatively good performance of the convex combination method is somehow surprising since the incremental method is known to be theoretically superior, see Padberg [63], or Wilson [82]. The computationally worst method by far is the disaggregated convex combination method (DLog). For coarse approximations (0.1), this method needs the largest number of branch-and-bound nodes, and for fine approximations (0.001) it does not even finish the LP root relaxation within the time limit.

η	1d	1d shift	2d	
	#points	#points	#triangles	#points
0.1	28	35	41	29
0.01	88	105	535	303
0.001	280	336	6149	3324

Table 3: Number of points/triangles for linear approximations within given tolerances.

method		CPLEX			SCIP		
	$\eta =$	0.1	0.01	0.001	0.1	0.01	0.001
CC	time	**0.04**	**34.22**	288.59%	**0.82**	1208.32	2336.77%
	nodes	**47**	**1622**	586935	**12**	77744	153954
Inc	time	0.47	50.86	**7585.13**	9.6	**994.26**	$+\infty$
	nodes	313	1712	**7550**	389	**3173**	334
DLog	time	0.8	117.88%	$+\infty$	1.94	43051.96	$+\infty$
	nodes	1394	2329071	0	1144	2217241	0

Table 4: Running time (in sec) and processed branch&bound-nodes for bivariate approach and partial cascade layout.

For the univariate method without shifting we have four different linearization methods at hand, see Table 5. For CPLEX the fastest method is SOS2, although the difference to the second-fastest method, Inc, is quite small. The method having the least number of branch-and-bound nodes is the incremental method. The latter is also true for SCIP. However, among the three methods, CC, Log, and Inc, there is no clear winner in terms of CPU time. In contrast to the bivariate approach, all methods for all approximation degrees (from coarsest to finest) terminate with an optimal solution within the time limit on both solvers. As expected, the solution times and the number of branch-and-bound nodes grows when going from a coarse to a fine linear approximation.

method		CPLEX			SCIP		
	$\eta =$	0.1	0.01	0.001	0.1	0.01	0.001
CC	time	0.1	0.42	3.88	0.34	**1.67**	11.31
	nodes	187	480	1548	30	68	565
Inc	time	0.06	0.04	0.49	**0.25**	1.78	18.47
	nodes	**9**	**12**	**48**	**4**	**1**	**1**
Log	time	**0.05**	0.27	1.64	0.34	2.14	**5.94**
	nodes	106	413	1006	84	30	76
SOS2	time	**0.05**	**0.03**	**0.4**	0.72	1.78	180.56
	nodes	581	287	2013	4321	7073	284747

Table 5: Running time (in sec) and processed nodes for univariate approach without shifting and partial cascade layout.

The results for the univariate method with shifting, presented in Table 6, are similar to those without shifting. We use the same four methods. For CPLEX, the incremental method not only has the least number of branch-and-bound nodes but is now also the fastest. The difference to the Log method in terms of running times is small for coarse and medium approximations, but gets large for a fine approximation level. For SCIP, also the incremental method has the smallest number of nodes, but SOS2 and the logarithmic convex combination method are faster to solve.

method		CPLEX			SCIP		
	$\eta =$	0.1	0.01	0.001	0.1	0.01	0.001
CC	time	0.16	1.64	38.13	0.51	2.84	31.67
	nodes	361	2564	3330	10	73	8874
Inc	time	**0.04**	**0.06**	**0.41**	0.32	3.22	30.05
	nodes	**20**	**25**	**67**	**2**	**2**	**210**
Log	time	**0.04**	0.58	3.67	0.45	**2.09**	**8.58**
	nodes	61	869	1986	32	131	504
SOS2	time	0.08	0.22	0.84	**0.26**	16.67	541.49
	nodes	995	2119	4376	1151	43403	578410

Table 6: Running time (in sec) and processed nodes for univariate approach with shifting and partial cascade layout.

In a final run we solve the NLP with a nonlinear solver, which is either SCIP or Baron, see Table 7. These solvers use outer approximation and spatial branching techniques to compute global optimal solutions. No additional binary variables need to be introduced to approximate the nonlinear constraints. We run our numerical example twice with SCIP: In the first run, we simply use the standard version of SCIP as it is publicly available. For the second run, we additionally switch the specific constraint handler generating linear estimators from the envelopes of our bivariate nonlinear functions on. The solver Baron is faster than SCIP and also needs less branch-and-bound nodes. Both solvers are faster than most of the linearization techniques from Tables 4–6. It should be noted that the numerical accuracy for the NLP solvers is higher than those of the linear approximation methods.

	Baron	"SCIP"/"plain"	"SCIP"/"bivariate"
time	0.05	0.23	0.24
nodes	1	33	121

Table 7: Running time (in sec) and processed nodes for partial cascade layout and nonlinear solver.

So far we only compared the solutions in terms of CPU times and branch-and-bound tree sizes. It is of course also important to know what a numerical solution looks like, and whether all solvers yield more or less the same solution (up to the selected approximation degree). These values are shown in Table 8, for each method and for each approximation degree. We give the values of the objective function and the three reject rates r_1, r_2, r_3. Except for the bivariate approach with a coarse linearization of 0.1 the values for r_1 and r_3 are always more or less the same. For a suitable high level of approximation (0.001), there is no big difference between the two univariate methods, shifted or not shifted.

	η	Obj.	r_1	r_2	r_3
bivariate	0.1	0.11660	0.58	0.5277	0.1
	0.01	0.17669	0.9	0.5981	0.1
	0.001	n.a.	n.a.	n.a.	n.a.
univariate	0.1	0.18515	0.9	0.6490	0.1
	0.01	0.16753	0.9	0.5832	0.1
	0.001	0.17962	0.9	0.6073	0.1
univariate shifted	0.1	0.15830	0.9	0.5360	0.1
	0.01	0.16804	0.9	0.5840	0.1
	0.001	0.18004	0.9	0.6079	0.1
nonlinear		0.17124	0.9	0.6044	0.1

Table 8: Solution for the partial cascade layout for different solvers and accuracies.

5.2 Including the Layout Decision

In the sequel we report on our results for the simultaneous layout and process optimization. Again, this problem has been tackled by different linearization approaches and MILP solvers (CPLEX and SCIP) as well as by the MINLP solvers Baron and SCIP.

Table 9 shows the running times and finished branch-and-bound nodes for the bivariate approach. Due to the inclusion of the topology the running times are now much higher, compared to the running times of a sole reject rate computation and a given layout, cf. Table 4. It turns out that the incremental method 'Inc' is best for the solver CPLEX, whereas SCIP can better handle the convex combination method 'CC'. However both solvers were only able to find proven global optimal solutions for a coarse and medium discretization ($\eta = 0.1$ and 0.01).

method		CPLEX			SCIP		
	$\eta =$	0.1	0.01	0.001	0.1	0.01	0.001
CC	time	62.01	14704.35	11280.58%	**39.03**	**10529.44**	$+\infty$
	nodes	197610	3221304	164684	**42455**	1575875	168545
Inc	time	**32.88**	**6744.97**	$+\infty$	163.43	30822.65	$+\infty$
	nodes	**22342**	**138917**	7240	44838	**239125**	864
DLog	time	151.35	670.75%	$+\infty$	1838.57	$+\infty$	$+\infty$
	nodes	340989	3083955	0	1284994	3878744	0

Table 9: Running time (in sec) and processed nodes for bivariate approach and layout optimization.

Table 10 displays the running times and finished branch-and-bound nodes for the univariate approach without shifting. Compared to the bivariate approach, this univariate approach is faster. Both solvers, CPLEX and SCIP, favor the 'Inc' and the 'Log' method, where the incremental method produces mainly the smallest branch-and-bound trees, and the logarithmical method leads to significantly lower running times.

method		CPLEX			SCIP		
	$\eta =$	0.1	0.01	0.001	0.1	0.01	0.001
CC	time	26.28	2760.2	11083.16%	60.42	795.66	22293.09
	nodes	48157	1102466	9556223	51611	379923	4256432
Inc	time	**9.79**	77.1	1984.87	32.12	145.99	815.73
	nodes	**15465**	**49918**	389975	**15013**	**23106**	**25715**
Log	time	12.12	**70.93**	**168.26**	**23.9**	**134.06**	**804.41**
	nodes	27011	118302	**96997**	27421	91081	260560
SOS2	time	3824.04	2740.73%	11083.26%	29753.09	$+\infty$	$+\infty$
	nodes	25193528	84277601	30118379	88361448	113177681	73688781

Table 10: Running time (in sec) and processed nodes for the univariate approach without shifting and layout optimization.

Table 11 displays the running times and finished branch-and-bound nodes for the univariate approach with shifting. The running times are higher and the branch-and-bound trees are bigger compared to the univariate approach without shifting. Still, CPLEX and SCIP favor the incremental and the logarithmical method, where the incremental method produces slightly better results in terms of CPU times and size of branch-and-bound trees. Additionally, the SOS2 approach works quite poor. This may be due to the binary variables from modeling the layout decision. The solvers have especially problems to find a meaningful dual bound; it stays at zero.

method		CPLEX			SCIP		
	$\eta =$	0.1	0.01	0.001	0.1	0.01	0.001
CC	time	61.16	5339.96	$+\infty$	68.6	727.11	38207.66
	nodes	100298	2163000	6766392	38300	224313	4057337
Inc	time	**16.14**	**120.89**	1452.81	**29.17**	233.24	**894.29**
	nodes	**28696**	**57487**	181933	**13002**	**27919**	**21613**
Log	time	19.77	231.76	**529.45**	37.86	**152.29**	2164.55
	nodes	46911	155240	**230197**	34074	74706	472147
SOS2	time	5577.18	$+\infty$	$+\infty$	8511844981.78%	$+\infty$	$+\infty$
	nodes	30121991	225933704	36142743	123553202	0	0

Table 11: Running time (in sec) and processed nodes for univariate approach with shifting and layout optimization.

Now turning to the nonlinear solvers Baron and SCIP we found that these solvers are faster then most of the above linearization techniques, in particular when it come to finer approximations. The results are shown in Table 12. Interestingly SCIP is faster than Baron, whereas the latter produces a smaller branch-and-bound tree on our test instance. The detection and special treatment of bivariate constraints leads to a further reduction of the running time and size of the branch-and-bound tree by a factor of approximately two.

	Baron	"SCIP"/"plain"	"SCIP"/"bivariate"
time	35.53	9.05	5.6
nodes	3966	11086	5737

Table 12: Running time (in sec) and processed nodes for layout optimization and nonlinear solver.

The results for all approaches in terms of objective value and optimal values of the reject rate variables are shown in Table 13. Despite the bivariate approach all methods determine the (full) cascade with screens ordered 3-2-1 as the optimal one, c.f. Figure 1. Both univariate approaches (shifted/unshifted) produce this configuration as optimal layout and appropriate settings for the reject rate, already for low accuracies ($\eta = 0.1$). The bivariate approaches on the other hand resulted in the topologies shown in Figure 5. The nonlinear solvers compute the same layout, a full-cascade using the screens ordered 3, 2, 1.

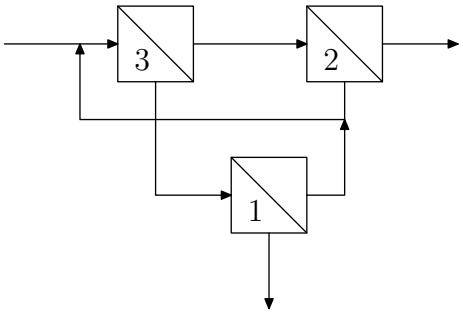

Figure 5: Topology determined by the bivariate approaches.

	η	Obj.	r_1	r_2	r_3
bivariate	0.1	0.0283	0.1	0.1	0.1
	0.01	0.0197	0.1	0.1	0.3464
	0.001	n.a.	n.a.	n.a	n.a.
univariate nonshift	0.1	0.0189	0.1	0.1	0.3396
	0.01	0.0211	0.1	0.1	0.3598
	0.001	0.0198	0.1	0.1	0.3461
univariate shift	0.1	0.0213	0.1	0.1	0.3558
	0.01	0.0204	0.1	0.1	0.3518
	0.001	0.0199	0.1	0.1	0.3467
nonlinear		0.0189	0.1	0.1	0.3440

Table 13: Results for the layout optimization for different solvers and accuracies.

6 Conclusion

We modeled the process and layout decision of a multi-stage screening system occurring in recovered paper production in a steady-state model. We showed that this model agrees with the former used dynamical model for each given layout. Using piecewise linear approximation techniques, the model can be solved numerically. We analyzed the impact of the approximation error for several univariate functions resulting from a transformation to the error of the non-transformed bivariate function. This allows us to compare the approach of directly approximating bivariate functions to the method of transforming the functions to several univariate functions. Beside an easier applicability of the piecewise linear methods, the advantage of the transformation approach

is supported by the computational results. Furthermore, the corresponding problems were also solved by recent nonlinear solvers such as Baron and SCIP. All approaches potentially lead to proven global optimal solutions (when finishing within the given time limit). It turns out that the nonlinear solvers are significantly faster than the piecewise linear approaches.

Acknowledgement. We thank Prof. Dr.-Ing. Samuel Schabel and Dipl.-Ing. Klaus Villforth of the chair of paper technology and mechanical process engineering at Technische Universität Darmstadt for posing the problem and fruitful discussions. We also thank Björn Geißler and Antonio Morsi for providing an implementation for ordering triangulations. The work of Christine Hayn was partly supported by the 'Excellence Initiative' of the German Federal and State Governments and the Graduate School of Computational Engineering at Technische Universität Darmstadt. The third author, Dennis Michaels, thanks the Deutsche Forschungsgemeinschaft (DFG) for their financial support through the Collaborative Research Centre "Integrated Chemical Processes in Liquid Multiphase Systems" (TRR 63).

References

[1] T. Achterberg. Scip: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41, July 2009. `http://mpc.zib.de/index.php/MPC/article/view/4`.

[2] S. Ahmed, J. P. Vielma, and G. Nemhauser. Mixed-integer models for nonseparable piecewise linear optimization: Unifying framework and extensions. *Operations Research*, 2009. to appear.

[3] K. E. Almin and B. Steenberg. The capacity problem in single series screen cascades – Studies in screening theory ii. *Svensk Papperstidning*, 57(2):37 – 40, 1954.

[4] K. J. Arrow and A. C. Enthoven. Quasi-concave programming. *Econometrica*, 29(4):779 – 800, 1961.

[5] M. Ballerstein, D. Michaels, A. Seidel-Morgenstern, and R. Weismantel. A theoretical study of continuous counter-current chromatography for adsorption isotherms with inflection points. *Computers & Chemical Engineering*, 34(4):447–459, 2010.

[6] M. Ballerstein, D. Michaels, and S. Vigerske. Global optimization of nonlinear optimization problems with bivariate functions with fixed convexity behaivor: a case study. Manuscript in preparation, 2012.

[7] J. J. Bartholdi and P. Goldsman. Continuous spatial indexing of surfaces. Part 1: Standard triangulations. Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia, 2001.

[8] M. H. Bauer and J. Stichlmair. Struktursynthese und Optimierung nicht-idealer Rektifizierprozesse. *Chemie Ingenieur Technik*, 68:911 – 916, 1996.

[9] P. Belotti, J. Lee, L. Liberti, F. Margot, and A. Wächter. Branching and bounds tightening techniques for non-convex MINLP. *Optimization Methods and Software (Special Issue: GLOBAL OPTIMIZATION)*, 24(4-5):597–634, 2009.

[10] H. P. Benson. On the Construction of Convex and Concave Envelope Formulas for Bilinear and Fractional Functions on Quadrilaterals. *Computational Optimization and Applications*, 27:5–22, 2004.

[11] R. E. Bixby. Solving Real-World Linear Programs: A Decade and More of Progress. *Operations Research*, 50(1):1–13, 2002.

[12] A. Bley. *Routing and capacitiy optimization for IP networks*. PhD thesis, Technische Universität Berlin, 2007.

[13] S. H. Cameron. Piece-wise linear approximations. Technical report, IIT Research Institute, 1966.

[14] A. Cantoni. Optimal curve fitting with piecewise-linear functions. *IEEE Transactions on Computers C*, 20:59 – 67, 1971.

[15] G. Dantzig, S. Johnson, and W. White. A linear programming approach to the chemical equilibrium problem. *Management Science*, 5:38 – 43, 1959.

[16] G. B. Dantzig. *Linear programming and extensions*. Princeton University Press, 1963.

[17] C. de Boor. *A practical guide to splines*. Springer, 2001.

[18] M. Ehrgott. *Multicriteria optimization*. Springer, 2000.

[19] Verband Deutscher Papierfabriken e.V. (VDP). Papier Kompass. http://www.vdp-online.de/pdf/Kompassdeutsch.pdf, 2007.

[20] C. A. Floudas. Separation synthesis of multicomponent feed streams into multicomponent product streams. *AIChE Journal*, 33:540 – 550, 1987.

[21] C. A. Floudas. *Nonlinear and mixed-integer optimization. Fundamentals and applications*. Oxford University Press, 1995.

[22] M. Franke, A. Górak, and J. Strube. Auslegung und Optimierung von hybriden Trennverfahren. *Chemie Ingenieur Technik*, 76:199 – 210, 2004.

[23] T. Frey, D. Brusis, J. Stichlmair, M. H. Bauer, and S. Glanz. Systematische Prozesssynthese mit Hilfe mathematischer Methoden. *Chemie Ingenieur Technik*, 72:812 – 821, 2000.

[24] F. Friedler, K. Tarjan, Y. W. Huang, and L. T. Fan. Graph-theoretic approach to process synthesis: Polynomial algorithm for maximal structure generation. *Computers chem. Engng.*, 17:929 – 942, 1993.

[25] A. Fügenschuh and M. Fügenschuh. Integer linear programming models for topology optimization in sheet metal design. *Mathematical Methods of Operations Research*, 68(2):313 – 331, 2008.

[26] A. Fügenschuh, M. Herty, A. Klar, and A. Martin. Combinatorial and continuous models for the optimization of traffic flows on networks. *SIAM J. OPTIM.*, 16:1155 – 1176, 2006.

[27] O. Galor. *Discrete dynamical systems*. Springer, New York, 2006.

[28] J. Gangadwala, A. Kienle, U.-U. Haus, D. Michaels, and R. Weismantel. Global Bounds on Optimal Solutions for the Production of 2,3-Dimethylbutene-1. *Industrial & Engineering Chemistry Research*, 45(7):2261–2271, February 2006.

[29] M. M. Gavrilovic. Optimal approximation of convex curves by functions which are piecewise linear. *Journal of Mathematical Analysis and Applications*, 52:260 – 282, 1975.

[30] B. Geißler, A. Martin, A. Morsi, and L. Schewe. *IMA Volume on MINLP*, chapter Using piecewise linear functions for solving MINLPs. Springer, 2010.

[31] D. A. Gürth. Approximation von bivariaten Funktionen mittels Orthogonalitätsrelationen für lineare Splines. Master's thesis, TU Darmstadt, Fachbereich Mathematik, 2007.

[32] B. Hamman and J. Chen. Data point selection for piecewise linear curve approximation. *Comput. Aided Geom. Des.*, 11(3):289 – 301, 1994.

[33] U.-U. Haus, D. Michaels, A. Seidel-Morgenstern, and R. Weismantel. A method to evaluate the feasibility of TMB chromatography for reduced efficiency and purity requirements based on discrete optimization. *Computers & Chemical Engineering*, 31(11):1525–1534, November 2007.

[34] I. Heckl, Z. Kovacs, F. Friedler, and L. T. Fan. Super-structure generation for separation network synthesis involving different separation methods. *Chemical Engineering Transactions*, 3:1209 – 1214, 2003.

[35] IBM ILOG CPLEX. Information available at `http://www.ibm.com/software/integration/optimization/cplex/`, 2010.

[36] H. Imai and M. Iri. An optimal algorithm for approximating a piecewise linear function. *Journal of Information Processing*, 9:159 – 162, 1986.

[37] M. Jach, D. Michaels, and R. Weismantel. The convex envelope of (n–1)-convex functions. *SIAM Journal on Optimization*, 19(3):1451–1466, 2008.

[38] A. B. Keha, I. R. de Farias, and G. L. Nemhauser. Models for representing piecewise linear cost functions. *Operations Research Letters*, 32(1):44 – 48, 2004.

[39] A. Khajavirad and N. V. Sahidinidis. Convex envelopes generated from finitely many compact convex sets. *Mathematical Programming A*, 2011. online available at `http://dx.doi.org/10.1007/s10107-011-0496-5`.

[40] A. Khajavirad and N. V. Sahidinidis. Convex envelopes of products of convex and component-wise concave functions. *Journal of Global Optimization*, 52:391–409, 2012.

[41] T. Koch. *Rapid mathematical programming.* PhD thesis, Technische Universität Berlin, 2004. ZIB-Report 04-58.

[42] J. Kubat and B. Steenberg. Screening at low particle concentrations – Studies in screening theory III. *Svensk Papperstidning*, 58(9):319 – 324, 1955.

[43] G. S. Lee. Piecewise linear approximation of multivariate functions. *Bell Syst. Tech. J.*, 61:1463 – 1486, 1982.

[44] J. Lee and D. Wilson. Polyhedral methods for piecewise-linear functions. I: The lambda method. *Discrete Applied Mathematics*, 108(3):269 – 285, 2001.

[45] J. Linderoth. A simplicial branch-and-bound algorithm for solving quadratically constrained quadratic programs. *Math. Program.*, 103(2, Ser. B):251–282, 2005.

[46] M. Locatelli. Convex envelopes for quadratic and polynomial functions over polytopes. Manuscript, 11/03/2010, available at `http://www.optimization-online.org/DB_FILE/2010/11/2788.pdf`, 2009.

[47] M. Locatelli and F. Schoen. On the convex envelopes and underestimators for bivariate functions. Manuscript, 11/17/2009, available at `http://www.optimization-online.org/DB_FILE/2009/11/2462.pdf`, 2009.

[48] J. K. Lowe. *Modelling with integer variables.* PhD thesis, Georgia Institute of Technology, 1984.

[49] G. Manis, G. Papakonstantinou, and P. Tsanakas. Optimal piecewise linear approximation of digitized curves. *IEEE*, 1977.

[50] H. Markowitz and A. Manne. On the solution of discrete programming-problems. *Ecometrica*, 25:84 – 110, 1957.

[51] G. P. McCormick. Computability of global solutions to factorable nonconvex programs. I: Convex underestimating problems. *Mathematical Programing*, 10:147–175, 1976.

[52] C. A. Meyer and C. A. Floudas. Convex envelopes for edge-concave functions. *Mathematical Programming*, Ser. B, 103:207–224, 2005.

[53] Clifford A. Meyer and Christodoulos A. Floudas. Trilinear Monomials with Positive or Negative Domains: Facets of the Convex and Concave Envelopes. In Christodoulos A. Floudas and P. M. Pardalos, editors, *Frontiers in Global Optimization*, pages 327–352. Kluwer Academic Publishers, Dordrecht, 2003.

[54] Clifford A. Meyer and Christodoulos A. Floudas. Trilinear Monomials with Mixed Sign Domains: Facets of the Convex and Concave Envelopes. *Journal of Global Optimization*, 29:125–155, 2004.

[55] S. Moritz. *A mixed integer approach for the transient case of gas network optimization.* PhD thesis, TU Darmstadt, Fachbereich Mathematik, 2006.

[56] M. Muraki and T. Hayakawa. Separation process synthesis for multicomponent products. *Journal of Chemical Engineering of Japan*, 17:533, 1984.

[57] M. Muraki and T. Hayakawa. Multicomponent product separation synthesis with separation sharpness. *Journal of Chemical Engineering of Japan*, 20:195 – 198, 1987.

[58] R. Nath and R. L. Motard. Evolutionary synthesis of separation processes. *AIChE Journal*, 27:578 – 587, 1981.

[59] G. L. Nemhauser and J. P. Vielma. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Lecture Notes in Computer Science*, 5035:199 – 213, 2008.

[60] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*. Wiley Interscience, 1988.

[61] N. Nishida, G. Stephanopoulos, and A. W. Westerberg. A review of process synthesis. *AIChE Journal*, 27:321 – 351, 1981.

[62] J. M. Ortega. *Iterative solution of nonlinear equations in several variables*. Academic Press, 1972.

[63] M. Padberg. Approximating separable nonlinear functions via mixed zero-one programs. *Operations Research Letters*, 27:1 – 5, 2000.

[64] R. T. Rockafellar. *Convex analysis*. Princeton Landmarks in Mathematics. Princeton, NJ: Princeton University Press, 1970.

[65] R. E. Rosenthal. *GAMS: A user's guide*. The Scientific Press, Redwood City, California, 1988.

[66] N. V. Sahinidis and M. Tawarmalani. *BARON 7.2.5: Global optimization of mixed-integer nonlinear programs*, 2005.

[67] C. Schönberger. Linearization methods for the optimization of screening processes in the recovered paper production. Master's thesis, TU Darmstadt, 2007.

[68] H. R. Schwarz and N. Köckler. *Numerische Mathematik*. Teubner Verlag, 2006.

[69] B. Steenberg. Principles of screening system design – Studies in screening theory I. *Svensk Papperstidning*, 56:771 – 778, 1953.

[70] H. Stone. Approximation of curves by line segments. *Mathematics of Computation*, 15:40 – 47, 1961.

[71] M. Tawarmalani, J.-P. Richard, and C. Xiong. Explicit convex and concave envelopes through polyhedral subdivisions. Manuscript June 1, 2010, available at `http://www.optimization-online.org/DB_HTML/2010/06/2640.html`, 2010.

[72] M. Tawarmalani and N. V. Sahinidis. Semidefinite Relaxations of Fractional Programs via Novel Convexification Techniques. *Journal of Global Optimization*, 20:137–158, 2001.

[73] M. Tawarmalani and N. V. Sahinidis. Convex extensions and envelopes of lower semi-continuous functions. *Mathematical Programming A*, 93:247–263, 2002.

[74] M. Tawarmalani and N. V. Sahinidis. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99:563 – 591, 2004.

[75] M. Tawarmalani and N. V. Sahinidis. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103(2, Ser. B):225–249, 2005.

[76] The MathWorks. Matlab version 7.9 (R2009b), 2009.

[77] I. Tomek. Two algorithms for piecewise-linear continuous approximation of functions of one variable. *IEEE Trans. Comput.*, 23:445 – 448, 1974.

[78] J. A. Tomlin. A suggested extension of special ordered sets to nonseparable nonconvex programming problems. In *Studies on graphs and discrete programming (Brussels, 1979)*, volume 11 of *Ann. Discrete Math.*, pages 359 – 370. North-Holland, 1981.

[79] J.-P. Valkama. Erabeitung eines Analysewerkzeugs für Altpapier verarbeitende Papierfabriken zur objektiven Bewertung der Grenzen der Stickyabtrennung durch Sortierprozesse. Abschlussbericht AiF-Projekt 18990 N. Technical report, Fachgebiet Papierfabrikation und Mechanische Verfahrenstechnik, TU Darmstadt, 2006.

[80] J. P. Valkama. *Optimisation of Low Consistency Fine Screening Processes in Recycled Paper Production*, volume 1 of *Fortschritt-Berichte Papiertechnik /Progress in Paper Technology 1*. Shaker, 2007.

[81] J. Vandewalle. On the calculation of the piecewise linear approximation to a discrete function. *IEEE Trans. Comput.*, 24:843 – 846, 1975.

[82] D. L. Wilson. *Polyhedral methods for piecewise-linear functions*. PhD thesis, University of Kentucky, Department of Mathematics, 1998.

[83] L. A. Wolsey. *Integer programming*. Wiley Interscience, 1998.

[84] R. Wunderling. *Paralleler und objektorientierter Simplex-Algorithmus*. PhD thesis, Technische Universität Berlin, 1996.

A Linear Approximation

Before any nonlinear function can be used in a MILP formulation, it has to be approximated piecewise linearly. The number of binary variables involved in the resulting MILP directly corresponds to the number of intervals, or triangles, respectively, used for the approximation. Therefore, we aim to find a sufficiently good approximation with a small number of intervals (or triangles).

A.1 Linearization of Univariate Functions

Several different approaches concerning approximation by piecewise linear functions in one dimension are discussed in the literature, e.g., finding the best approximation for a given number of fixed nodes [17, 70] or free nodes [17, 29, 81, 14, 70], or finding an approximation with the minimum number of line segments necessary, such that the error is smaller than a specified tolerance [13, 77, 49, 32, 36].

A piecewise linear function that coincides with a continuous function $g : [a, b] \to \mathbb{R}$ in its breakpoints is called its *interpolant* and denoted by I_g. Note that the absolute approximation error

$$\max_{x\in[a,b]} |h(x) - g(x)| \tag{35}$$

can at best be halved by going over from the interpolation to the best possible approximation by broken lines [17]. Halving the error is especially possible for convex and concave functions. In these cases the piecewise linear interpolant completely lies above (below) g, cf. Figure 6. Therefore, the error can be reduced by moving the interpolant downwards (upwards).

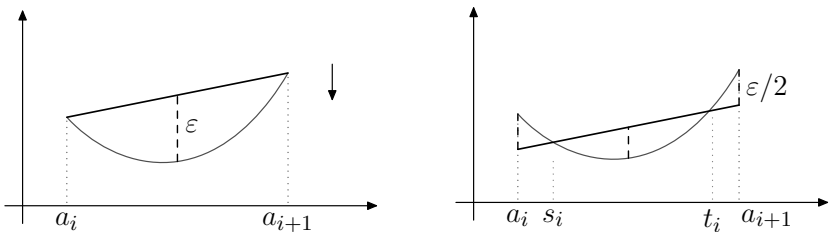

Figure 6: Shifting an interpolant to a convex function downwards by $\varepsilon/2$.

Let g be convex or concave and

$$\varepsilon := \max_{x\in[a,b]} |g - I_g|. \tag{36}$$

Then

$$\max_{x\in[a,b]} |g - (I_g \mp \frac{\varepsilon}{2})| \leq \frac{\varepsilon}{2}. \tag{37}$$

For finding a good interpolant [17] gives a useful formula for node placement in the following theorem.

Theorem A.1 *([17] p.36) Let $g \in C^2(a,b)$ and $|g''|$ be monotone near a and b, and $\int_a^b |g''(x)^{1/2}|dx < \infty$. Then , if $a_2, \ldots, a_{n-1}$ are chosen such that*

$$\int_a^{a_i} |g''(x)|^{1/2}dx = \frac{i-1}{n-1}\int_a^b |g''(x)|^{1/2}dx, \quad \text{for } i = 2, \ldots, n-1, \tag{38}$$

then

$$\|g - I_g\| = O(n^{-2}),$$

where $\|f\| := \max_{x\in[a,b]} |f|$ denotes the uniform norm of $f \in C[a,b]$.

Amongst others, [13, 49] treat the task of finding a broken line that approximates a given function within a given error tolerance with as few line segments as possible. Their algorithms define a tunnel of radius ε around the original curve (or around sample points of the curve) and find the farthest point visible through this tunnel. The solution received by the algorithms of [13, 49] is optimal with respect to the number of line segment. But there is no guarantee that the solution yielding the minimal error is chosen among those solutions with minimal number of line segments. In contrast, generally, the constructed solution is not optimal with respect to the error.

We implemented an algorithm based on the ideas of [13] in [67]. Examples in [67] show that for approximating the logarithm, de Boor's formula (38) combined with shifting the piecewise linear function upwards by half of the interpolation error results in the same number of intervals needed for a given error tolerance as by using the above techniques. Since the first approach is less time consuming, we used this method to obtain the piecewise linear functions for our further computational results.

A.2 Linearization of Bivariate Functions

For determining an approximation of the bivariate function given by the application, we used an algorithm developed by [31]. Given an error tolerance it determines an L_2-approximation, or an interpolation, respectively, of functions in two variables over triangulations using $\sqrt{3}$-subdivision [31] as refinement algorithm for triangulations. For calculating the L_2-approximation hat functions and a discretized scalar product induced by a weighted Sobolev norm are used. A basic implementation of the algorithm in Matlab was made available to us by [31]. Computational experiments in [67] suggest that this approach leads to less triangles than for example the use of Delaunay triangulations always refining at the point with the currently largest error, when taken into account the absolute error. The algorithm was adapted to fit to our error definition.

B Modeling Piecewise Linear Functions

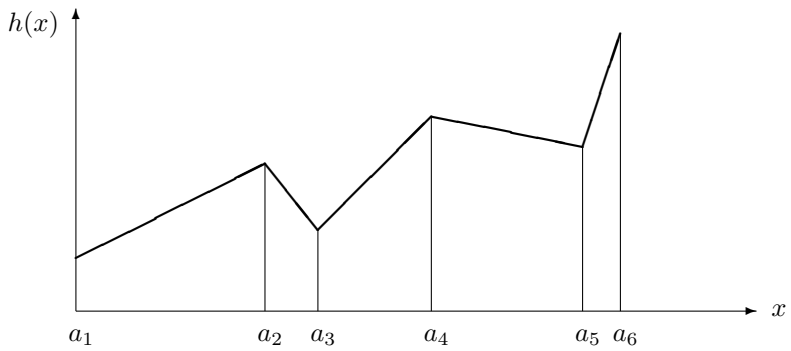

Figure 7: Piecewise linear function $h(x)$.

Consider a continuous piecewise linear function $h : D \subseteq \mathbb{R}^n \to \mathbb{R} : x \mapsto h(x)$, defined on a triangulation $\mathcal{T}$ of D, where the vertices of the triangulation correspond to the breakpoints of the

piecewise linear function. Let $V(\mathcal{T})$ denote the set of vertices of $\mathcal{T}$, and accordingly $V(T)$ the set of vertices of a simplex $T \in \mathcal{T}$.

B.1 Disaggregated convex combination models

B.1.1 Basic model

$$\sum_{T\in\mathcal{T}}\sum_{v\in V(T)} \lambda_{T,v} v = x, \qquad \sum_{T\in\mathcal{T}}\sum_{v\in V(T)} \lambda_{T,v} h(v) = h(x)$$
$$\lambda_{T,v} \geq 0, \quad \text{for } T \in \mathcal{T}, v \in V(T)$$
$$\sum_{v\in V(T)} \lambda_{T,v} = y_T, \qquad \sum_{T\in\mathcal{T}} y_T = 1, \qquad y_T \in \{0,1\}, \quad \text{for } T \in \mathcal{T}.$$

We refer to this model as *disaggregated convex combination model* and denote it by DCC. Inter alia, it has been studied by [48].

B.1.2 Logarithmic model

The idea is to reduce the number of binary variables and constraints by introducing a unique binary code for every simplex and just binary variables for every digit in this binary code.

$$\sum_{T\in\mathcal{T}}\sum_{v\in V(T)} \lambda_{T,v} v = x, \qquad \sum_{T\in\mathcal{T}}\sum_{v\in V(T)} \lambda_{T,v} h(v) = h(x),$$
$$\sum_{T\in\mathcal{T}}\sum_{v\in V(T)} \lambda_{T,v} = 1, \quad \lambda_{T,v} \geq 0, \quad \text{for } T \in \mathcal{T}, v \in V(T),$$
$$\sum_{T\in\mathcal{T}^0(B,l)}\sum_{v\in V(T)} \lambda_{T,v} \leq y_l, \qquad \sum_{T\in\mathcal{T}^+(B,l)}\sum_{v\in V(T)} \lambda_{T,v} \leq 1 - y_l,$$
$$y_l \in \{0,1\}, \quad \text{for } l \in L(\mathcal{T}),$$

where $B : \mathcal{T} \to \{0,1\}^{\lceil \log_2 |\mathcal{T}| \rceil}$ is any injective function, $\mathcal{T}^0(B,l) := \{T \in \mathcal{T} : B(T)_l = 0\}$, $\mathcal{T}^+(B,l) := \{T \in \mathcal{T} : B(T)_l = 1\}$ and $L(\mathcal{T}) = \{1, \ldots, \log_2(\lceil |\mathcal{T}| \rceil)\}$. We refer to this model as *logarithmic disaggregated convex combination model* and denote it by Dlog. This formulation has been introduced by [59, 2].

B.2 Convex combination models

Again a point $(x, h(x))$ is represented as convex combination of its neighbored grid-points. In contrast to above the number of continuous variables is reduced by aggregating the variables associated with the same vertex (belonging to more than one simplex) in the triangulation.

B.2.1 Basic model

$$\sum_{v\in V(\mathcal{T})} \lambda_v v = x, \qquad \sum_{v\in V(\mathcal{T})} \lambda_v h(v) = h(x),$$
$$\sum_{v\in V(\mathcal{T})} \lambda_v = 1, \quad \lambda_v \geq 0, \quad \lambda_v \leq \sum_{T:v\in V(T)} y_T, \quad \text{for } v \in V(\mathcal{T}),$$
$$\sum_{T\in\mathcal{T}} y_T = 1, \qquad y_T \in \{0,1\}, \quad \text{for } T \in \mathcal{T}.$$

This approach is known as the *convex combination* or *lambda method* [16, 60, 48, 82, 63, 44, 38, 55]. We refer to it as *convex combination model* and denote it by CC.

B.2.2 Logarithmic model

Similar to Dlog, the number of binary variables of CC can be reduced by identifying each simplex with a binary code through an injective function $B : \mathcal{T} \to \{0,1\}^{\lceil \log_2(|\mathcal{T}|) \rceil}$. But this time B has to own special properties in order to ensure the *adjacency condition*, i.e., the positive λ_v's have to

correspond to the vertices of the same simplex. Generally speaking, a binary branching scheme complying with the adjacency condition is a family of dichotomies $\{L_s, R_s\}_s \in S$ with S finite and $L_s, R_s \subset V(\mathcal{T})$ such that for every $T \in \mathcal{T}$ we have $V(T) = \bigcap_{s \in S}(V(\mathcal{T}) \setminus A_s)$, where $A_s = L_s$ or $A_s = R_s$ for each $s \in S$ [2]. Given such a branching scheme, the piecewise linear function may be modeled as follows.

$$\begin{aligned} \sum_{v \in V(\mathcal{T})} \lambda_v v = x, \qquad & \sum_{v \in V(\mathcal{T})} \lambda_v h(v) = h(x), \\ \sum_{v \in V(\mathcal{T})} \lambda_v = 1, \qquad & \lambda_v \geq 0, \quad \text{for } v \in V(\mathcal{T}), \\ \sum_{v \in L_s} \lambda_v \leq y_s, \quad \sum_{v \in R_s} \lambda_v \leq 1 - y_s, \quad & y_s \in \{0,1\}, \quad \text{for } s \in S. \end{aligned}$$

It is possible to construct such a branching scheme with a logarithmic number of dichotomies for every so-called Union-Jack-triangulation in $\mathbb{R}^2$ [2]. For a general triangulation this is not always possible.

For a triangulation in $\mathbb{R}$, however, such a branching scheme inducing a logarithmic number of variables can always be constructed using the so-called grey code, i.e., an injective function $B : \{1, \dots, r\} \to \{0,1\}^{\lceil \log_2(r) \rceil}$, where $r = |V(\mathcal{T})|$ is the number of vertices, such that for all $i \in \{2, \dots, r\}$ $B(i-1)$ and $B(i)$ differ in exactly one digit [59]. Then the dichotomies are given by

$$L_s = \{i \in \{2, \dots, r\} : B(i)_s = B(i-1)_s = 0\},$$

$$R_s = \{1\} \cup \{i \in \{2, \dots, r\} : B(i)_s = B(i-1)_s = 1\}.$$

We refer to this formulation as *logarithmic convex combination model* and denote it by Log.

B.2.3 SOS-approach

$$\begin{aligned} \sum_{v \in V(\mathcal{T})} \lambda_v v = x, \qquad & \sum_{v \in V(\mathcal{T})} \lambda_v h(v) = h(x), \\ \sum_{v \in V(\mathcal{T})} \lambda_v = 1, \qquad & \lambda_v \geq 0, \quad \text{for } v \in V(\mathcal{T}) \\ \lambda \text{ satisfies the adjacency condition.} & \end{aligned}$$

Instead of introducing binary variables to ensure the adjacency condition, the constraint may also be indirectly implemented by integrating it in a branch-and-bound framework [55]. For a triangulation in $\mathbb{R}$ the adjacency condition complies with the the so-called *Special Ordered Set of Type 2 Condition* (SOS 2 condition), i.e., at most two of the variables corresponding to the set of vertices are positive and if they are positive they are neighbored. The integration into a branch-and-bound framework may also be possible for higher dimensions, but one has to define its own branching scheme relying on the triangulation [55].

For the one-dimensional case, we refer to this approach as *SOS2-approach* and denote it by SOS2.

B.3 Incremental model

This formulation requires the vertices of the triangulation to be ordered in a special way [82].

- $T_i \cap T_{i-1} \neq \emptyset$, for $i = 2, \dots, |\mathcal{T}|$,
- for each simplex T_i, we can label its k_i vertices as $v_i^0, v_i^1, \dots, v_i^{k_i}$ so that $v_{i-1}^{k_{i-1}} = v_i^0$, for $i = 2, \dots, |\mathcal{T}|$.

Notice that for univariate functions this assumption is always fulfilled by the natural order in $\mathbb{R}$.

Wilson [82] shows, that the ordering assumption holds for any triangulation of a domain D that is a topological disk in $\mathbb{R}^2$. [7] give an algorithm to compute such an order fast. [30] pose an algorithm to order triangulations in higher dimensions.

Given a triangulation ordered in this way, we may describe a point x by filling-up all simplices prior to T_i, where $x \in T_i$, and then presenting x as v_i^0 plus a conical combination of the rays $v_i^j - v_i^0$, with $j = 1, \ldots, k_i$, compare Figure 8.

$$v_1^0 + \sum_{i=1}^{|\mathcal{T}|} \sum_{j=1}^{k_i} \delta_i^j (v_i^j - v_i^0) = x, \qquad h(v_1^0) + \sum_{i=1}^{|\mathcal{T}|} \sum_{j=1}^{k_i} \delta_i^j (h(v_i^j) - h(v_i^0)) = h(x),$$

$$\sum_{j=1}^{k_i} \delta_i^j \leq 1, \quad \delta_i^j \geq 0 \quad \text{for } i = 1, \ldots, |\mathcal{T}| \text{ and } j = 1, \ldots, k_i,$$

$$w_i \leq \delta_i^{k_i}, \quad \sum_{j=1}^{k_i} \delta_{i+1}^j \leq w_i, \qquad\qquad w_i \in \{0, 1\}, \quad \text{for } i = 1, \ldots, |\mathcal{T}| - 1.$$

We refer to this formulation as *incremental model* and denote it by Inc. In literature this model is sometimes also referred to as delta formulation. This formulation has been studied in [50, 16, 63, 82, 38].

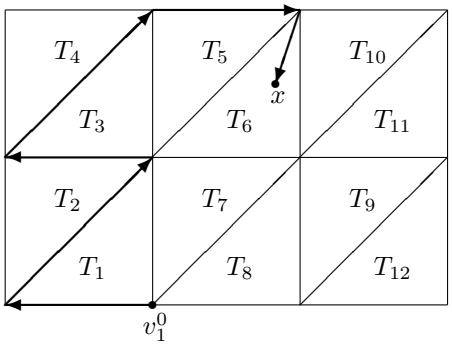

Figure 8: The point x written as v_1^0 plus a sum of vectors.

Note that for the (disaggregated) convex combination models a triangulation is not necessary. These approaches may directly be generalized to a finite family of general polytops [2], although the number of continuous variables may increase and ambiguities will occur.