A Global Approach to the Optimal Control of System Dynamics Models

A. Fügenschuh[1], I. Vierhaus[2]

[1] Helmut Schmidt University / University of the Federal Armed Forces Hamburg, Germany

[2] Zuse Institute Berlin, Germany

Abstract:
The System Dynamics (SD) methodology is a framework for modeling and simulating the dynamic behavior of socioeconomic systems. Characteristic for the description of such systems is the occurrence of feedback loops together with stocks and flows. The mathematical equations that describe the system are usually ordinary differential equations and nonlinear algebraic constraints. Therefore seemingly simple systems can show a nonintuitive, unpredictable behavior over time. Controlling a dynamical system means to specify potential interventions from outside that should keep the system on the desired track, and to define an evaluation schema to compare different controls among each other, so that a "best" control can be defined in a meaningful way. The central question is how to compute such globally optimal control for a given SD model, that allows the transition of the system into a desired state with minimum effort. We propose a mixed-integer nonlinear programming (MINLP) reformulation of the System Dynamics Optimization (SDO) problems. MINLP problems can be solved by linear programming based branch-and-bound approach. We demonstrate that standard MINLP solvers are not able to solve SDO problem. To overcome this obstacle, we introduce a special-tailored bound propagation method. We apply our new method to a predator-prey model with additional hunting activity as control, and to a mini-world model with the consumption level as control. Numerical results for these test cases are presented.

Keywords:
System Dynamics; Global Optimal Control; Mixed-Integer Nonlinear Optimization; Bounds Strengthening.

1 Introduction: From SD to MINLP

SD models describe the behavior of a system that consists of several interrelated stocks, flows and feedback loops. The relation between those is usually determined by ordinary differential equations, nonlinear functional relations, logical relations (such as if-then-else), or tabular data. Even if each of these relations is individually well understood, the interplay of several of these relations can show a surprising, unexpected behavior in a simulation over time. SD is nowadays used as a tool mainly in socio-economic and engineering applications. For an introduction to SD and its applications we refer to Sterman [1].

Simulation runs with varying input parameters can be used to test and evaluate different policies. Here a subset of meaningful input parameters from the model is selected. They can either be constant parameters which do not change over time, or they can also be time-dependent control functions. When two different runs lead to a different dynamical behavior of the system, it is natural to ask, which of the two solutions is better? To answer this question properly, one needs to introduce an objective or goal function. This function is a mapping of the simulated trajectories to a single real number. In this way, two different simulation runs become comparable: The higher the objective function value, the better. It is natural to ask, what are the parameters that lead to the best-possible objective function value? This question cannot be answered from just testing and simulating any finite number of different parameters, no matter how large this number would be.

A System Dynamics model together with control functions and a real-valued objective function is called a System Dynamics Optimization (SDO) problem in the sequel. The need for an integration of optimization methods into the SD framework has been recognized already in the past. In previous approaches, different methods are used, such as nonlinear local optimization (for example, gradient search methods) or heuristics (such as genetic algorithms), which are essentially based on an "optimization through repeated simulations" approach, see [2–9]. All these approaches have in common that they at best only provide feasible or local optimal solutions. Moreover, as pointed out by Burns and Janamanchi [10], nonlinear optimization methods rely on the availability of derivative information from sufficiently smooth functions. This restriction is in conflict with certain SD modeling elements, such as if-then-else clauses. In principle, all these methods

cannot give a proof of global optimality or any other quality certificate of the computed solutions. The ultimate goal of our research is to fill this gap and develop a computational method that is able to handle all kinds of SD models and does not only yield feasible solutions, but also a certificate of optimality.

We formulate the SDO problem as mixed-integer nonlinear program (MINLP). Solving optimization problems from this class is theoretically intractable and also known to be computationally difficult in general. By "solving" we mean to compute a feasible solution for a given instance of the problem together with a computational proof of its optimality. Therefor we apply the general framework of a branch-and-bound approach, where the bounds are obtained from relaxations of the original model. To this end, we relax the MINLP first to a mixed-integer linear program (MILP) and then further to a linear program (LP), which is solved efficiently using Dantzig's simplex algorithm [11]. The so obtained solution value defines a (lower) bound on the optimal value of the original MINLP problem. In case this solution is MINLP feasible, it would be a proven global optimal MINLP solution. However, this rarely happens in practice. Hence we either add cutting planes to strengthen the relaxation, or we decide to branch on a variable. As an example, consider the nonlinear constraint $f : x \mapsto x|x|$.In the LP relaxation this function is replaced by a polyhedral (linear) outer approximation, which is iteratively refined during the branch-and-bound process by branching on variables (spatial branching), see Figure 1. For more details on cutting planes and branch-and-bound for MILP we refer to Nemhauser and Wolsey [12], and for an application of this framework to global mixed-integer nonlinear programming to Smith and Pantelides [13], and Tawarmalani and Sahinidis [14, 15]. Information on the MINLP framework SCIP which we apply is given in Achterberg [16], and in particular on nonlinear aspects of SCIP in Berthold, Heinz, and Vigerske [17].

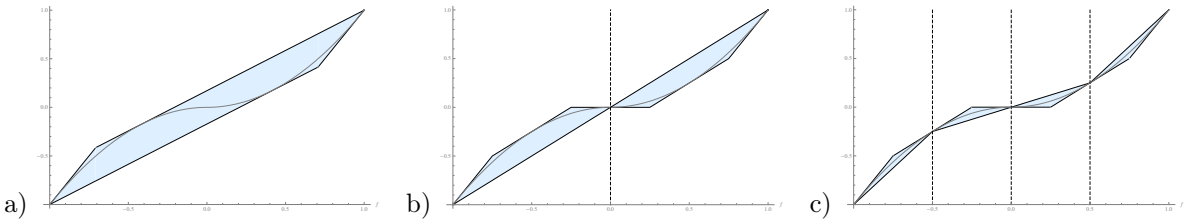

Figure 1: a) Polyhedral outer approximation of $f : x \mapsto x|x|$, b) initial spatial branching on zero, c) further spatial branching.

As a modeling language, mixed-integer programming offers a high flexibility in formulating various types of SD modeling elements. As an example, we demonstrate this flexibility by reformulating an if-then-else clause. Say, the SD models contains the following constraint: *if* $x \geq a$ *then* $y = 1$ *else* $y = 0$ *end*. Here x, y are variables to be determined in the simulation. We assume that there are lower and upper bounds known for x, that is, $x \in [\underline{x}, \overline{x}]$. The (binary) variable y is activated ($y = 1$), if the stock x becomes larger than a certain given quantity a. Otherwise, y is inactive ($y = 0$). Note that this condition constitutes a discontinuous relation between x and y. Due to this discontinuity, an optimization method based on derivatives would not be applicable (or, to be more precise, would only be applicable after some smoothing tricks, that come along with further numerical issues). In a mixed-integer formulation of this condition we introduce two auxiliary variables $\xi^+, \xi^- \geq 0$, and then state the following constraints:

$$
\begin{aligned}
x - a &= \xi^+ - \xi^-, && \text{(1a)}\\
\xi^+ &\leq (\overline{x} - a)y, && \text{(1b)}\\
\xi^- &\leq (a - \underline{x})(1 - y), && \text{(1c)}\\
y &\in \{0, 1\}. && \text{(1d)}
\end{aligned}
$$

Note that constraints (1a), (1b), (1c) are linear (in-) equalities. Hence Dantzig's simplex algorithm [11] is able to deal with them. The integrality constraint (1d) is then relaxed to its continuous counterpart $0 \leq y \leq 1$. If in an optimal solution to this LP relaxation the variable y has a fractional value, one can re-introduce the integrality condition either by a cutting plane approach or by branching on y. For details on these two approaches, we refer to the literature [12, 18].

2 Our Approach

In the following we describe our approach to solve mixed-integer dynamic optimization problems that originate from System Dynamics Optimization (SDO) models. As a foundation we use the linear and nonlinear programming based mixed-integer nonlinear solver SCIP [16]. This framework already handles the input of an instance, the set up of the model's variables and constraints, and the overall branch-and-cut solution process. Initially, we tried to solve SDO model instances using SCIP as a black-box solver off the shelf. Although SCIP is a highly capable general-purpose solver, the results in our case were devastating. It became clear that we had to enhance SCIP to detect and exploit the special structure that is inherent to SDO problems. That means, the time discretization leaves its traces in the variables and the constraints, and it pays off to use the natural (time) order of the variables to speed up the computation process. Certain general-purpose preprocessing techniques also needed to be specially tailored towards SDO. These issues are described in detail in Section 2.2. With their help it is possible to substantially reduce the upper bound of the linear relaxation (in case of a maximization problem which we consider here without loss of generality).

2.1 Problem Formulation

We formulate an SDO problem as a dynamic MINLP problem of the following form:

$$\max\ c(x,y,z), \tag{2a}$$
$$\text{s.t. } 0 = f(x_{j+1}, x_j, y_j, z_j), \quad j \in \{0,1,\dots,T-1\}, \tag{2b}$$
$$0 \le g(x_j, y_j, z_j), \quad j \in \{0,1,\dots,T\}, \tag{2c}$$
$$\underline{x}_j \le x_j \le \overline{x}_j, \quad j \in \{0,1,\dots,T\}, \tag{2d}$$
$$\underline{y}_j \le y_j \le \overline{y}_j, \quad j \in \{0,1,\dots,T\}, \tag{2e}$$
$$\underline{z}_j \le z_j \le \overline{z}_j, \quad j \in \{0,1,\dots,T\}, \tag{2f}$$
$$x_j \in \mathbb{R}^n, \quad j \in \{0,1,\dots,T\}, \tag{2g}$$
$$y_j \in \mathbb{Z}^q \times \mathbb{R}^{p-q}, \quad j \in \{0,1,\dots,T\}, \tag{2h}$$
$$z_j \in \mathbb{Z}^s \times \mathbb{R}^{r-s}, \quad j \in \{0,1,\dots,T\}. \tag{2i}$$

Here $x = (x_j)_{j=0,1,\dots,T}$ are the system's dynamic variables for all time steps j. Further, $y = (y_j)_{j=0,1,\dots,T}$ denotes the static variables at each time step j, and $z = (z_j)_{j=0,1,\dots,T}$ are the control variables at each time step j. A variable is said to be *dynamic*, if its time derivative occurs in the model. A *static* variable is an auxiliary variable that aggregates informations from other variables subject to a functional relationship. A *control* variable models the external influence to the system. The objective function c assigns a real value to each feasible solution (x, y, z), and thus allows to compare two solutions with each other. The goal is to identify a global optimal solution with respect to the measure of this function c. Without loss of generality we focus on maximization problems in equation (2a). The system's dynamical structure is described by u continuous nonlinear functions $f : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{Z}^q \times \mathbb{R}^{p-q} \times \mathbb{Z}^s \times \mathbb{R}^{r-s} \to \mathbb{R}^u$. Equations (2b) couple the dynamic variables of time step $j+1$ with all (static, dynamic, and control) variables of time step j. These equality constraints are used to describe discretized differential equations. The v continuous nonlinear functions $g : \mathbb{R}^n \times \mathbb{Z}^q \times \mathbb{R}^{p-q} \times \mathbb{Z}^s \times \mathbb{R}^{r-s} \to \mathbb{R}^v$ in equation (2c) model static relations for a single time step j. Similar to the distinction of static and dynamic variables, we say that a constraint is *dynamic*, if it contains (discretized) derivates, and *static* otherwise. Static constraints can be inequality constraints in general. (The special case of an equality constraint is covered by using two inequalities of opposite direction). Bounds on the range of the state variables are given for all time steps by constraint (2d) for the dynamic variables and by constraints (2e) for the static variables. The dynamic variables are all continuous variables as specified in equation (2g). Some of the static and control variables may carry additional integrality constraints, which is specified in equation (2h) and equation (2i), respectively.

2.2 Bound Propagation

Clearly, the bound constraints (2d), (2e), and (2f) are special cases of the static constraints (2c). Nevertheless, we stated them explicitly in our problem formulation (2), because they are crucial in the initial phase of our solution approach. The bound values $\underline{x}, \overline{x}, \underline{y}, \overline{y}, \underline{z}, \overline{z}$ that are given in (2) are usually weak in the following sense: They reflect the modeler's idea on the domain that the optimized system should never leave. It is known that already "normal" SD models (i.e., without the aspects of control and optimization) can have a surprising, unexpected behavior due to the nonlinear structure and feedback loops. This is even more true for SDO problems, where the unintuitive dynamic interacts with an external control. Giving tight bounds on the variables is practically impossible under these circumstances. So it is more likely to expect that initially some rough estimation will be given.

A crucial step in solving mixed-integer linear or nonlinear problems is the preprocessing phase. In this initial step of the solution process, one tries to extract as much information as possible from the problem at hand. These information are implicitly contained in the model formulation, but hidden for the solver. The goal is to make them visible, so that the solver can use them to shorten the solution process. Preprocessing or presolving refers not just to a single method, but a whole bunch of various techniques, where some are more general, and others are more specific for certain problem structures. For more details and general surveys to MILP and MINLP solution techniques including preprocessing, we refer to [18–22].

Bound propagation belongs to the class of general techniques. Assume that the problem contains a linear constraint of the form $\sum_{i=1}^{n} a_i w_i \leq b$, where a_i, b are constants and w_i are variables, and the bound constraints $\underline{w_i} \leq w_i \leq \overline{w_i}$ for $i = 1, 2, \ldots, n$. Then for any j with $a_j > 0$ we can try to update the upper bound via

$$\overline{w}_j := \min \left\{ \overline{w}_j, \frac{b}{a_j} - \sum_{i \neq j, a_i > 0} \frac{a_i}{a_j} \underline{w}_i - \sum_{i \neq j, a_i < 0} \frac{a_i}{a_j} \overline{w}_i \right\}, \tag{3}$$

and for any j with $a_j < 0$ we can try to update the lower bound via

$$\underline{w}_j := \max \left\{ \underline{w}_j, \frac{b}{a_j} - \sum_{i \neq j, a_i > 0} \frac{a_i}{a_j} \underline{w}_i - \sum_{i \neq j, a_i < 0} \frac{a_i}{a_j} \overline{w}_i \right\}. \tag{4}$$

In case of a continuous nonlinear constraint $f(w_1, \ldots, w_n) \leq b$ with $f : \mathbb{R}^n \to \mathbb{R}$, and bounds $\underline{w_i} \leq w_i \leq \overline{w_i}$ for $i = 1, 2, \ldots, n$ one needs to solve an unconstrained optimization problem to obtain improved bounds

$$\overline{w}_j := \min \left\{ \overline{w}_j, \max\{ f(w_1, \ldots, w_n) \mid \underline{w}_i \leq w_i \leq \overline{w}_i, i = 1, 2, \ldots, n \} \right\}, \tag{5a}$$

$$\underline{w}_j := \max \left\{ \underline{w}_j, \min\{ f(w_1, \ldots, w_n) \mid \underline{w}_i \leq w_i \leq \overline{w}_i, i = 1, 2, \ldots, n \} \right\}, \tag{5b}$$

The computation steps in equations (3), (4), and (5) are repeated for all variables j and all constraints of the model, until no further bound improvement is achieved.

Bound strengthening is an important step for two reasons. First, we apply linear programming to solve the subproblems in the branch-and-bound tree. If the variable bounds are weak, then the solution of the linear programs will also yield weak (upper) bounds for the solution of the MINLP. Second, when branching on a variable, it needs more branching decisions to fix a variable when the bounds are weak. Both issues are related to the solution speed and the memory requirement during the solution process. Hence in general one is interested in obtaining good (narrow) bounds on all variables, assuming that the time spend for computing them is marginal compared to the gain in solving the overall problem.

In Fügenschuh et al. [23] it was noted that this general bound strengthening procedure gives weak results on MILPs with a dynamic structure. In this work, a finite difference approach for a partial differential (transport) equation on a network was examined. It was concluded that the dynamic structure must be exploited in order to speed up the presolution process by some orders of magnitudes. Furthermore, dealing with a single constraint at a time yields weaker results compared to the case of multiple constraint preprocessing. The situation for dynamic MINLP problems approximating ordinary differential equations is somehow similar. Again, we use the induced structure of the time axis for an ordering of the preprocessing steps. And then, we do not use a single constraint in each iteration, but certain larger portions of the

model in order to arrive at better (more narrow) bounds. The details of this procedure are described below.

We define the problem $\overline{P}(w;h,k)$ for a variable $w \in \{x_1,\ldots,x_n,y_1,\ldots,y_p,z_1,\ldots,z_r\}$, and $h,k = 0,1,\ldots,T$ with $h-k \geq 0$ as

$$\begin{aligned}
\overline{P}(z;h,k) := \max\ & w, && \text{(6a)}\\
\text{s.t. } & 0 = f(x_{j+1},x_j,y_j,z_j), \quad j \in \{h-k,\ldots,h\}, && \text{(6b)}\\
& 0 \leq g(x_j,y_j,z_j), \quad j \in \{h-k,\ldots,h\}, && \text{(6c)}\\
& \underline{x}_j \leq x_j \leq \overline{x}_j, \quad j \in \{h-k,\ldots,h\}, && \text{(6d)}\\
& \underline{y}_j \leq y_j \leq \overline{y}_j, \quad j \in \{h-k,\ldots,h\}, && \text{(6e)}\\
& \underline{z}_j \leq z_j \leq \overline{z}_j, \quad j \in \{h-k,\ldots,h\}, && \text{(6f)}\\
& x_j \in \mathbb{R}^n, \quad j \in \{h-k,\ldots,h\}, && \text{(6g)}\\
& y_j \in \mathbb{Z}^q \times \mathbb{R}^{p-q}, \quad j \in \{h-k,\ldots,h\}, && \text{(6h)}\\
& z_j \in \mathbb{Z}^s \times \mathbb{R}^{r-s}, \quad j \in \{h-k,\ldots,h\}. && \text{(6i)}
\end{aligned}$$

Similar, we define the problem $\underline{P}(w;h,k) := \min\ w$, with the same constraints as before in (6). These values are used to update the upper and lower bounds on variable w, respectively. These two auxiliary problems are very similar to the original problem (2), where the main differences are the exchange of the objective function, and the smaller time horizon, from $h-k$ to h instead of 0 to T.

We use the solution of (6) in the following way. We select a value for the *look-back presolve level* k. For smaller values of k the solution of (6) is faster, but also the obtained bounds are weaker, compared to large values of k. Here a trade-off between solution speed and a quality of the results needs to be established. For each $h = k,k+1,\ldots,T$ and each variable $z \in \{x_1,\ldots,x_n,y_1,\ldots,y_p,z_1,\ldots,z_r\}$ we solve for $\overline{P}(w;h,k)$ and $\underline{P}(w;h,k)$, where these values are compared with the previously existing best bounds for w, and updated if necessary. These runs are in principle independent of each other. However, it pays off to solve in upwind direction of the time, that is, first for $h = k$, then $h = k+1$, and so on. The bounds obtained in earlier runs can directly be re-used in subsequent iterations, and speed up the solution process. Moreover, after one single run all bounds already obtained their respective final values, and it is not necessary to repeat this run a second time (this would not lead to better bounds).

3 Test Instances

As test instances we use two system dynamic models that we describe in the following. The models are taken from the literature (c.f. Bossel [24–26]). We add a control function to each model that allows for an external interception. The question is how to chose this control function with respect to some optimality criterion. A typical goal is to interfere with the models' inherent dynamics in the least possible way. We then transform each model into a nonlinear mixed-integer problem. In this form it can be used as input data for our solution algorithm SCIP and our presolve methods.

3.1 Predator-Prey

As a first test instance we use a predator-prey system with unlimited capacities. The mathematical formulation of predator-prey models can be dated back to the work of Lotka [27] and Volterra [28]. The basis of our model's description below are taken from Bossel [25]. Since Bossel describes a simulation problem only, we add a control function to the model: We aim to maximize the amount of prey that can be removed from the system without inducing its collapse. This can be thought of as a model for sustainable human hunting activity. Since we have to deal with a finite time horizon, it needs to be assured that not all prey is killed at the (somehow artificial) end-of-time, in order to simply maximize the objective function.

The SDO diagram is shown in Figure 2. In the mathematical model corresponding to this figure, we introduce three time-dependent functions: the prey population $x(t)$, the predator population $y(t)$, and the

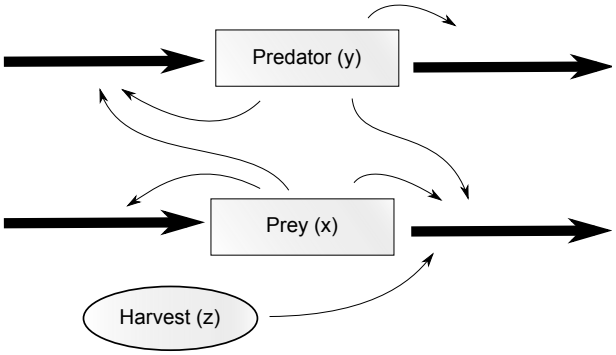

Figure 2: System Dynamics diagram for the predator-prey control problem.

control $z(t)$. If both populations, predator x and prey y, would be independent of each other, the prey population would grow proportional to its size and net growth rate (because of the assumed unlimited capacity). This would lead to an exponential growth with rate a, i.e., $\frac{dx}{dt} = ax$. Without prey, the predator population would decrease with an exponential rate $-d$, i.e., $\frac{dy}{dt} = -dy$. When both populations share the same habitat, they meet each other, and as a result, parts of the prey are consumed by the predators. The chance of a meeting is proportional to both population sizes xy. In case of a meeting the prey population is reduced by a factor b and the predator population increases by a factor c. The prey is additionally hunted by humans. We assume that these hunters can take out any desired part of the population, independent from its current size. Thus it is reduced by z in each time step. Hence we have the differential equation system $\frac{dx}{dt} = ax - bxy - z$ and $\frac{dy}{dt} = cxy - dy$. As boundary values, the initial population for predator and prey is given: $x(0) = \bar{x}, y(0) = \bar{y}$. Furthermore, the prey population is not allowed to fall below a certain limit over the whole time horizon: $x \geq \xi$. This in particular prevents to take out all remaining prey in the very last time step of the model.

We apply a forward discretization to the differentials. Let (Δt) denote a time discretization. We introduce continuous variable $x_i, y_i, z_i \in \mathbb{R}$ for $i = 0, 1, \ldots, T$. Then the discretized SDO problem is the following:

$$\max \sum_{i=0}^{T} z_i \tag{7a}$$

$$s.t. \ \frac{x_{i+1} - x_i}{\Delta t} = ax_i - bx_i y_i - z_i, \quad i = 0, 1, \ldots, T, \tag{7b}$$

$$\frac{y_{i+1} - y_i}{\Delta t} = -dy_i + cx_i y_i, \quad i = 0, 1, \ldots, T, \tag{7c}$$

$$x_i \geq \xi, \quad i = 0, 1, \ldots, T, \tag{7d}$$

$$x_0 = x_T, y_0 = y_T, z_0 = z_T, \tag{7e}$$

$$x_0 = \bar{x}, y_0 = \bar{y}, \tag{7f}$$

$$x_i, y_i, z_i \in \mathbb{R}, \quad i = 0, 1, \ldots, T. \tag{7g}$$

We emphasize equation (7e). These constraints enforce any feasible solution to the problem being periodic, that is, the final state must equal the initial state of the system. Then it is guaranteed that a sustainable (optimal) solution is reached, which can be continued forever.

3.2 Mini-World

The second test instance is a mini-world model, introduced by Bossel [26]. It is a simplification of the much more sophisticated World2 and World3 model of Forrester [29] and Meadows et al. [30]. World3 uses 18 stocks, 60 parameters, 52 tabular functions, and around 200 equations to model the system's relations. Bossel's Miniworld is an aggregated version, that comes with just 3 stocks: the world population, the production (industrial, commercial, agricultural), which equals the consumption of goods, and the

environmental pollution. Consequently, the number of parameters, tables, and equation relations are much lower. Interestingly, the model shows a qualitatively similar behavior compared to the much more evolved World3 model. Again, Miniworld is designed as a pure SD model for simulation. We add a control function to obtain an SDO problem. Our goal is to maximize the economic growth level, in order to provide a high standard of living to the world's population. However, if this standard would be too high, then a fast growing population would quickly consume its natural resources, and thus the population would collapse soon after. To prevent such behavior we introduce a population level ξ as a lower limit. The question we want to address is, how much sustainable growth can bear this mini-world at most for a population being at least ξ.

The SDO diagram is shown in Figure 3. The corresponding mathematical model has four time-dependent functions: $p(t)$ for the population, $e(t)$ for the environmental pollution, $c(t)$ for the consumption, and $z(t)$ for the control function (the growth level).

We apply a forward discretization to the differential terms, with (Δt) being the size of a step in the time discretization. The continuous variables $p_i, e_i, c_i, z_i \in \mathbb{R}$ for $i = 0, 1, \ldots, T$ approximate the functions of the respective same name. Furthermore, we introduce the following real-valued continuous variables, each for $i = 0, 1, \ldots, T$: α_i are the number of births and Ω_i are the number of deaths at time step i. γ_i is the consumption level. μ_i describes the environmental quality. The environmental conditions change over time, and for this we introduce ε_i for the environmental recovery and ζ_i for the environmental destruction.

Using a forward Euler schema, the discretized SDO problem reads as follows:

$$\max \sum_{i=0}^{T} z_i \tag{8a}$$

$$s.t.\ \frac{p_{i+1} - p_i}{\Delta t} = \alpha_i - \Omega_i, \quad i = 0, 1, \ldots, T, \tag{8b}$$

$$\alpha_i = 0.03 \cdot p_i \cdot \mu_i \cdot \gamma_i, \quad i = 0, 1, \ldots, T, \tag{8c}$$

$$\Omega_i = 0.01 \cdot p_i \cdot e_i, \quad i = 0, 1, \ldots, T, \tag{8d}$$

$$\frac{e_{i+1} - e_i}{\Delta t} = \varepsilon_i - \zeta_i, \quad i = 0, 1, \ldots, T, \tag{8e}$$

$$\zeta_i = 0.02 \cdot p_i \cdot \gamma_i, \quad i = 0, 1, \ldots, T, \tag{8f}$$

$$e_i = e_i^+ - e_i^- + 1.0, \quad i = 0, 1, \ldots, T, \tag{8g}$$

$$e_i^+ \leq 20.0 \cdot x_i, \quad i = 0, 1, \ldots, T, \tag{8h}$$

$$e_i^- \leq 1.0 - x_i, \quad i = 0, 1, \ldots, T, \tag{8i}$$

$$\varepsilon_i = 0.1 \cdot (1.0 - e_i^-), \quad i = 0, 1, \ldots, T, \tag{8j}$$

$$1.0 = e_i \cdot \mu_i, \quad i = 0, 1, \ldots, T, \tag{8k}$$

$$\frac{c_{i+1} - c_i}{\Delta t} = 0.05 \cdot \gamma_i \cdot e_i \cdot \left(1 - \left(\frac{\gamma_i \cdot e_i}{z_i}\right)\right), \quad i = 0, 1, \ldots, T, \tag{8l}$$

$$c_i = \gamma_i, \quad i = 0, 1, \ldots, T, \tag{8m}$$

$$p_i \geq \xi, \quad i = t, t+1, \ldots, T, \tag{8n}$$

$$p_0 = \bar{p}, \tag{8o}$$

$$e_0 = \bar{e}, \tag{8p}$$

$$c_0 = \bar{c}, \tag{8q}$$

$$p_i, e_i, e_i^+, e_i^-, c_i, z_i, \alpha_i, \Omega_i, \gamma_i, \mu_i, \theta_i, \varepsilon_i, \zeta_i \in \mathbb{R}_+, \quad i = 0, 1, \ldots, T, \tag{8r}$$

$$x_i \in \{0, 1\}, \quad i = 0, 1, \ldots, T. \tag{8s}$$

The change in the population p from time step i to $i + 1$ is a result of the births α_i minus the deaths Ω_i in time step i (8b). The number of births α_i in time step i is proportional to size of the population p_i, the environmental quality μ_i, and the consumption level γ_i, where 0.03 is the proportionality factor (birth rate) (8c). The number of deaths Ω_i in time step i is proportional to the population p_i and the environmental pollution e_i, with a proportionality factor (death rate) of 0.01 (8d).

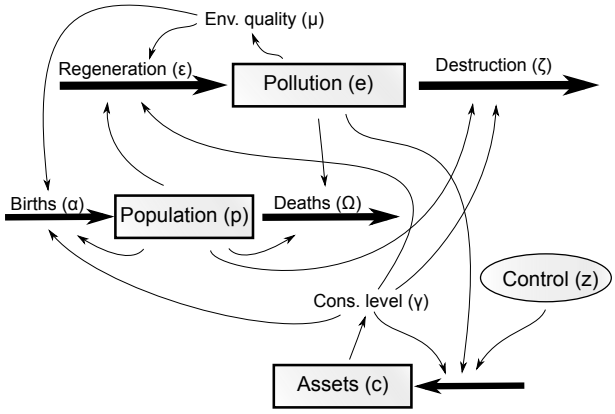

Figure 3: System Dynamics diagram for the mini-world control problem.

The change in the environmental pollution e from time step i to $i+1$ is a result of the negative environmental destruction ζ_i and the positive environmental recovery ε_i in time step i (8e). The environmental destruction ζ_i in time step i is proportional to the population size p_i and the consumption level γ_i (8f). Here 0.02 is the proportionality factor (environmental destruction rate). The environment is able to recover over time. If the environmental quality μ_i is above a certain threshold value (here 1.0), then the recovery ε_i in time step i is proportional to the environmental pollution e_i, with 0.1 being the proportionality factor (recovery rate). However, if the environmental quality μ_i is below the threshold value, then the recovery rate is no larger than 0.1 (8j). For a proper representation of this constraint in our model, we introduce a binary variable x_i for the decision if e_i is above or below the threshold value of 1.0 (8g). The part above and below is separately stored in two auxiliary variables e_i^+ (8h) and e_i^- (8i), respectively. Note that the variable μ_i does not appear in the equations (8j) directly. Equation (8k) reflects the dependency of the environmental pollution and its inverse, the environmental quality. This equation is responsible that μ_i is indirectly part of equations (8j).

The change in the production and consumption c from time step i to $i+1$ is the result of a logistic growth function of Verhulst type (8l), which depends on both the consumption level γ_i and the environmental pollution e_i. The control variable z_i plays the role of the system's capacity (this is a constant in the original Verhulst equation). The constant value of 0.05 is a growth rate for the consumption. The consumption level γ_i equals the production c_i (8m).

The population p_i must not fall short of the given level ξ_i in each time step i (8n). Initially, in time step $i = 0$, the size of the population (8o), the environmental pollution (8p), and the consumption resp. production (8q) are given.

4 Computational Results

We discuss the application of our methods to the two test problems. We start with the smaller and thus simpler predator-prey problem, and present the mini-world results afterwards.

4.1 Predator-Prey

Figure 4 shows the population sizes for different levels of constant hunting activity. That is, $z_i := \bar{z}$ for all $i = 0, 1, \ldots, T$, and $\bar{z} \in [0, 0.7]$. For $\bar{z} = 0$, i.e., no hunting activity, this reduces to the classical Lotka-Volterra model. The predator and prey populations (dark blue lines) follow a cyclic behavior with a period length of 7 time units (say, years). If positive hunting activity comes into play, i.e., $\bar{z} > 0$, then the prey population will sooner or later become extinct. This can be seen in detail in Figure 5, where even a small positive but constant hunting activity leads to an extinction after a few cycles. We conclude that a constant hunting level is not sustainable for this dynamical system.

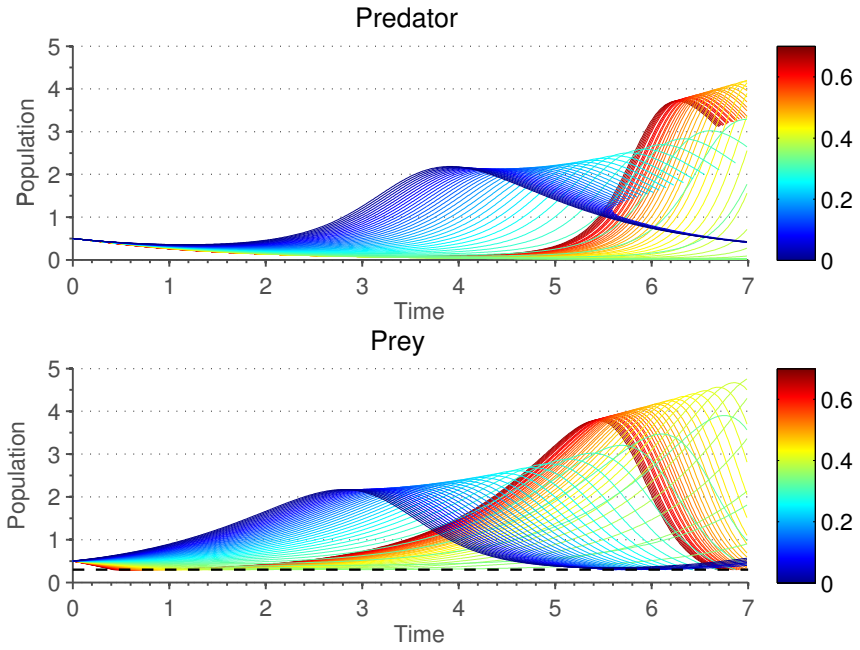

Figure 4: Trajectories for the predator and the prey for different levels of a constant control, varying in $[0, 0.7]$.

level	SCIP	0	10	20	40	80	120
time [min]	0.005	6	9	13	52	144	200

Table 1: Running times for different presolve look-back levels for the predator-prey SDO.

In order to apply mixed-integer nonlinear programming, the bounds on the variables need to be stengthened in a preprocessing step. The look-back level of this subroutine determines the time spent here, as well as the quality of the result. In general, a higher look-back level leads to higher running times, but also better results. In Figure 6 we compare the remaining bounds of the original presolve routines of the MINLP solver SCIP with our results for various look-back levels. In Table 1 we show the corresponding running times.

Figure 7 shows a solution to the predator-prey problem with human hunting activity. Using our presolve methods we could prove that this solution is at most 0.77% away from the lower bound. This means, this solution might be optimal, but in case it is not optimal, an optimal solution can only be 0.77% better. We have a high level of hunting for about 4 years, and a close season for about 3 years allowing the prey population to regenerate. At the end of the 7 years cycle both populations are of the same size as in the beginning. This means that the same trajectories can be repeated to infinity, hence a sustainable control schema of the system is achieved.

In Figure 8 we compare three phase diagrams in the predator-prey space. In black we show the population dynamic without hunting activity. The blue curve is the population dynamic for constant hunting activity. The red curve is the population dynamic for the sustainable hunting schema.

4.2 Mini-World

We further simplify the control z. To this end, we introduce a piecewise constant control with two states z_0 and z_1, where z_0 is valid for the first $T/2$ many time steps, an z_1 is valid for the second half of $T/2$ time steps. Then the objective function is a two-dimensional function depending on the values z_0, z_1 for the two constant controls. Figure 9 shows a contour plot of the objective function values for the two controls varying independently in the interval $[1, 10]$. Feasible solutions are in the lower-left corner. The

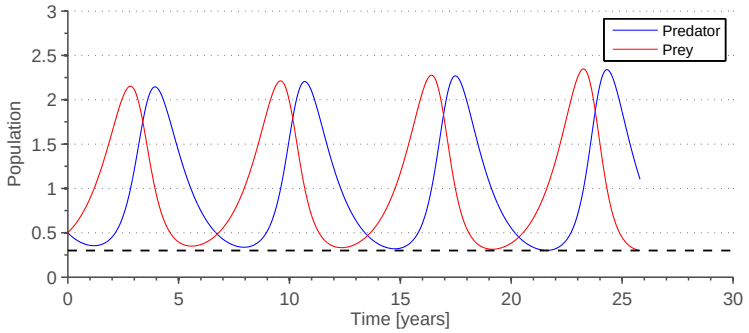

Figure 5: Trajectories for the predator and the prey for constant hunting level of 0.01. After a few cycles, the prey population becomes extinct.

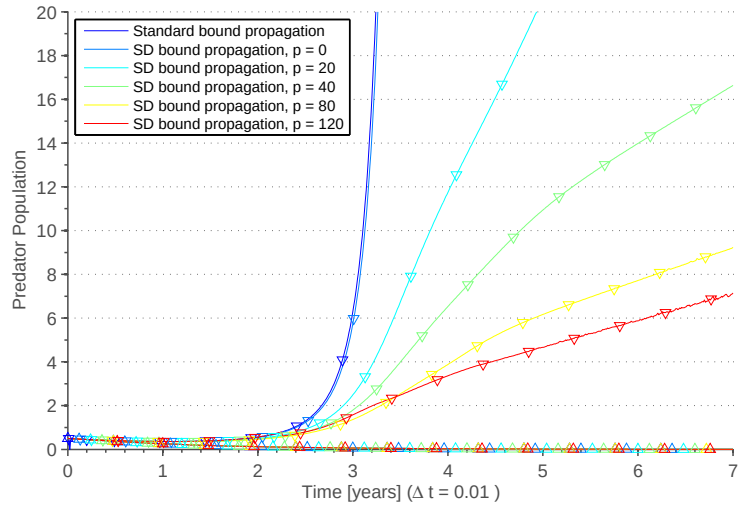

Figure 6: Presolved bounds for different look-back levels.

black curve marks the borderline between the feasible and the infeasible region. Our goal is to keep the mini-world population above $\xi \geq 4$ (billion inhabitants). Any solution having less inhabitants is infeasible. Solutions in the infeasible region all fall short of the population lower limit of constraint (8n). A possibly optimal solution occurs for $z_0 = 1.79$ and $z_1 = 2.16$. This solution yields the highest total consumption level, while keeping the population size above the specified limit.

In Figure 10 we show three different solutions. The first for $z_0 = z_1 = 1$ is feasible but not optimal. The second for $z_0 = z_1 = 10$ is infeasible. (These two reference plots can also be found in Bossel [26].) Our potential optimal solution for $z_0 = 1.79$ and $z_1 = 2.16$ is shown as third plot.

Note that the plot in Figure 9 and the identification of an optimal solution is based on a number of simulation runs. Even if we use a fine grid, we cannot be absolutely sure that there is not a better solution that is hidden between two neighboring grid points. Our goal is to demonstrate that by using our bounding techniques we can cut off large portions of the search space. It is thus guaranteed that no better solution can be found in such a cut off part. We apply our bounds strengthening presolve routine. The results for different look-back levels are shown in Figure 11 and Table 2. We vary the look-back level between 0 and 10. The row "SCIP" shows as reference values the results of the solver SCIP without using our bounds strengthening routines. For each run, we give the CPU running time and the dual bound for the presolve phase. After the presolve phase is finished, SCIP starts the branch-and-bound phase with a time limit of one hour. We give the best dual bound and the best feasible solution found in the end, and

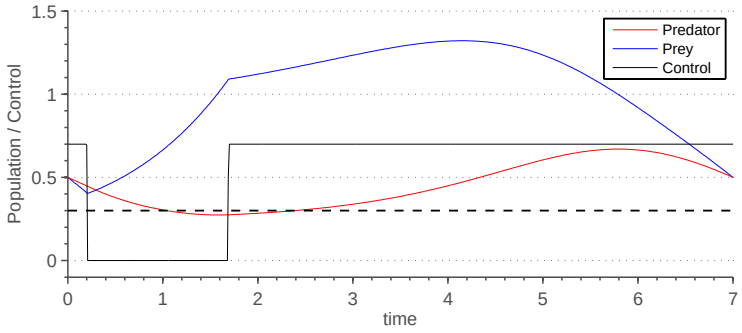

Figure 7: Optimal control for a sustainable hunting schema.

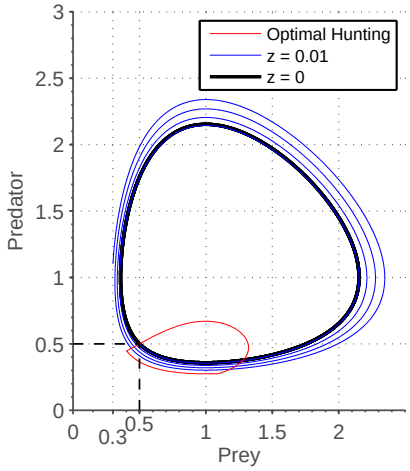

Figure 8: Phase diagrams for predator-prey population without hunting (black), constant hunting (blue), and sustainable hunting (red).

the gap, $g = \frac{d-p}{d} \cdot 100\%$, where p is the primal and d is the dual bound. The SCIP-level (first line in Table 2) is special: The solver was very fast, but due to its numerical instability "proved" that the problem is infeasible (i.e., does not have a feasible solution), which is of course not true.

A part of the search space that does not require further attention due to the bounding argument is shown in Figure 12. Here a look-back presolve level of 5 time steps was applied. Our optimal solution for $z_0 = 1.79$ and $z_1 = 2.16$ has an objective function value of 230.14. Any solution in the box defined by the green lines has an upper bound on the objective function value of less-or-equal 229.94, hence no better solution can be found in this part of the control space.

At the present stage of our implementation, we are, however, not able to prove that our potential optimal solution is indeed a global optimal one. The solver SCIP terminates, even in the highest presolve level, with a huge gap of 61.26%. Our ongoing research is devoted to close this gap to 0% within the given time limit, and to solve the problem for a control function with more than two constant segments.

5 Summary and Conclusions

We presented System Dynamics Optimization (SDO) models as an extension of classical System Dynamics Simulation (SD) models, where one (or several) control mechanism are introduced into the system. Moreover, a desired state must be specified that the modeler wants to achieve. This raises a theoretical as well as practical challenging question how to actually compute such control functions, and in case there

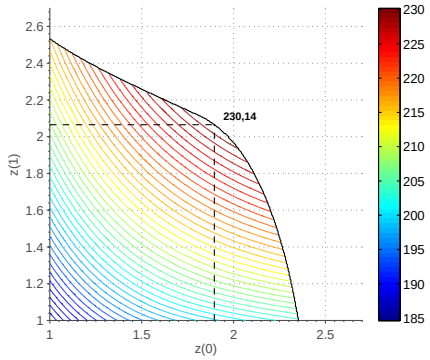

Figure 9: Objective function values for piecewise constant controls varying in $[1, 10] \times [1, 10]$. The (unique) optimal solution is in $(1.79, 2.16)$.

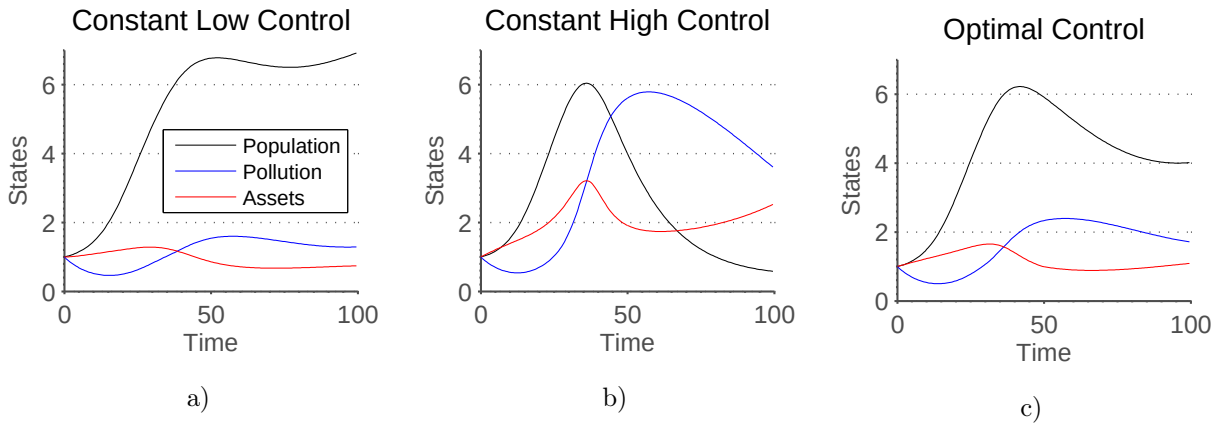

Figure 10: a) Feasible but suboptimal solution with control $z_0 = z_1 = 1$. b) Infeasible solution with control $z_0 = z_1 = 10$. c) Optimal solution with $z_0 = 1.79$ and $z_1 = 2.16$.

is more than one possible control, how to identify the globally best among all possibles, with respect to some optimality criterion. To answer this question one can simply resort to an optimization-by-simulation approach, where a certain number of simulation runs is carried out, and the control functions vary between each run. The mechanism to alter the control functions can be found in the literature, for example, Genetic Algorithms, Tabu Search, or Nelder-Mead Simplex. We demonstrated that these methods all have the principle disadvantage that they do not give any kind of performance guarantee on the computed result.

To overcome this weakness, we suggest mixed-integer nonlinear programming (MINLP) methods. To apply this method, the ordinary differential equations in the SDO problem are approximated by some discretization schema. Certain features that are common in the SD modeling language (such as tabular functions, or if-then-else constructs) also need to be transcribed and reformulated by introducing piecewise linear inequalities and binary or integer variables. Then this MINLP model can be relaxed to a linear programming problem, and solved by Dantzig's simplex method. This results in a bound on the objective function value to the SDO problem, which is then improved by adding suitable cutting planes and a branch-and-bound procedure. In order to achieve good bounds quickly, we further enhance this process by using a specially tailored bounds strengthening as preprocessing, and a primal heuristic to identify feasible solutions earlier.

We applied our method to two test problems from the literature. We showed that our bounds strengthening procedure significantly reduces that search space, which directly relates to the time spent to compute an optimal solution. In the predator-prey model with additional control by human hunting, we were able

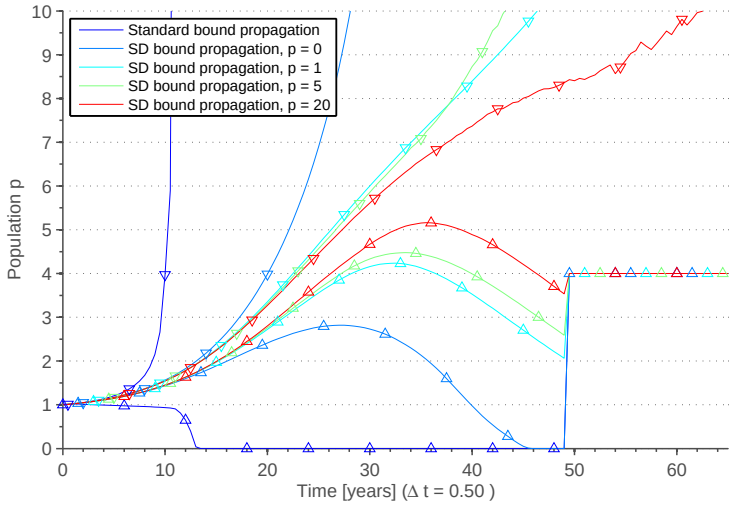

Figure 11: Presolved bounds for different look-back levels.

	presolve			branch-and-bound		
level	total time [h]	dual	time [h]	dual	primal	gap
SCIP	0.002	–	–	–	–	–
0	1.33	$6.253 \cdot 10^9$	1	$6.253 \cdot 10^9$	184.655	$\gg 100\%$
1	4.42	$6.395 \cdot 10^2$	1	$6.379 \cdot 10^2$	184.655	99.4
2	2.702	$5.806 \cdot 10^2$	1	$5.787 \cdot 10^2$	184.655	68.1
5	4.53	$5.082 \cdot 10^2$	1	$5.062 \cdot 10^2$	184.655	63.5
10	9.01	$4.786 \cdot 10^2$	1	$4.766 \cdot 10^2$	184.655	61.3

Table 2: Computational results for different presolve look-back levels.

to compute a control schema that guarantees a maximum yield of prey under the condition that we seek a sustainable solution, so that neither predator nor prey are dying out at any time in the future. The optimal control has both hunting season and a close season, whereas any constant schema that comes without such close season, no matter how little prey is hunted, will lead to an extinction of both species. For the mini-world control model we demonstrated that larger areas of the solution space do not need to be examined by a bounding argument, once a suitable good (perhaps optimal) solution is found. We computed a piecewise constant globally optimal control that achieves a maximum consumption level under the additional constraint that humankind shall not be perished from this earth.

We believe that our methods are generally applicable to all different kind of SDO models. At the present stage of our work, each model needs to be individually analyzed and treated. We are working towards a black-box solver that a model will be able to use for his or her models in the same way SD simulation tools are used today. We think that having not only good-looking solutions, but additionally a certificate of optimality, is a further asset in practical applications.

Acknowledgement

We gratefully acknowledge the funding of this work by the German Research Association (Deutsche Forschungsgemeinschaft, DFG), Collaborative Research Center CRC1026 (Sonderforschungsbereich SFB1026).

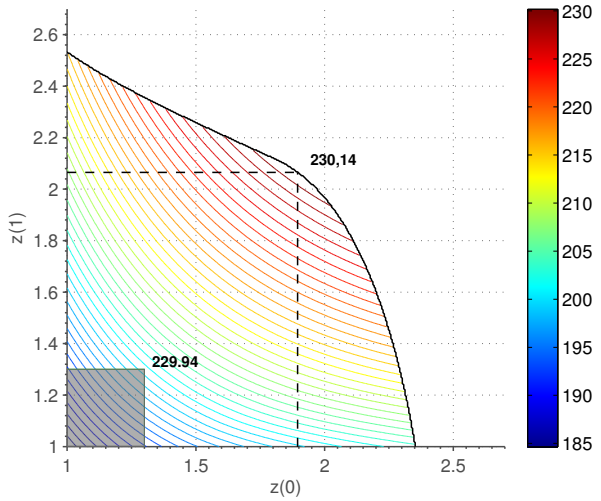

Figure 12: The part of the control space with $z_1 \leq 1.3$ and $z_2 \leq 1.3$ marked by the gray box can be cut off. No better feasible solutions can be found in here.

References

[1] Sterman, J. D., 2000. Business Dynamics – Systems Thinking and Modeling for a Complex World. Irwing McGraw-Hill, Boston.

[2] Duggan, J., 2008. Using System Dynamics and Multiple Objective Optimization to Support Policy Analysis for Complex Systems. In H. Qudrat-Ullah, J. Spector, P. Davidsen, editors, Complex Decision Making, Understanding Complex Systems. Springer Verlag, Berlin, 59–82.

[3] Elmahdi, A., Malano, H., Etchells, T., Khan, S., 2005. System Dynamics Optimisation Approach to Irrigation Demand Management. In Proceedings of the MODSIM 2005 International Congress on Modelling and Simulation. 196–202.

[4] Dangerfield, B., Roberts, C., 1996. An Overview of Strategy and Tactics in System Dynamics Optimization. Journal of the Operational Research Society, 47:405–423.

[5] Kleijnen, J., 1995. Sensitivity analysis and optimization of system dynamics models: Regression analysis and statistical design of experiments. System Dynamics Review, 11(4):275–288.

[6] Fu, M., 1994. Optimization via simulation: A review. Annals of Operations Research, 53(1):199–247.

[7] Keloharju, R., Wolstenholme, E., 1989. A Case Study in System Dynamics Optimization. Journal of the Operational Research Society, 40(3):221–230.

[8] Keloharju, R., Wolstenholme, E., 1988. The basic concepts of system dynamics optimization. Systemic Practice and Action Research, 1(1):65–86.

[9] Gustaffson, L., Wiechowski, M., 1986. Coupling DYNAMO and optimization software. System Dynamics Review, 2(1):62–66.

[10] Burns, J. R., Janamanchi, B., 2007. Optimal Control and Optimization of System Dynamics Models: Some Experiences and Recommendations. In Proceedings of the 2007 Meeting of the Southwest Region Decision Sciences Institute.

[11] Dantzig, G., 1963. Linear programming and extensions. Princeton University Press and RAND Corporation.

[12] Nemhauser, G. L., Wolsey, L. A., 1988. Integer and Combinatorial Optimization. John Wiley & Sons, New York.

[13] Smith, E., Pantelides, C., 1999. A Symbolic Reformulation/Spatial Branch-and-Bound Algorithm for the Global Optimization of nonconvex MINLPs. Computers & Chemical Engineering, 23:457–478.

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

[15] Tawarmalani, M., Sahinidis, N., 2005. A polyhedral branch-and-cut approach to global optimization. Mathematical Programming, 103:225–249.

[16] Achterberg, T., 2009. SCIP: Solving constraint integer programs. Mathematical Programming Computation, 1(1):1–41.

[17] Berthold, T., Heinz, S., Vigerske, S., 2012. Extending a CIP Framework to Solve MIQCPs. In J. Lee, S. Leyffer, editors, Mixed Integer Nonlinear Programming, volume 154, part 6 of *The IMA Volumes in Mathematics and its Applications*. Springer Verlag, Berlin, 427–444.

[18] Fügenschuh, A., Martin, A., 2005. Computational Integer Programming and Cutting Planes. In R. W. K. Aardal, G. Nemhauser, editor, Handbook on Discrete Optimization, Series Handbooks in Operations Research and Management Science, volume 12. Elsevier, 69 – 122.

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

[20] Belotti, P., Kirches, C., Leyffer, S., Linderoth, J., Luedtke, J., Mahajan, A., 2012. Mixed-Integer Nonlinear Optimization. Technical Report ANL/MCS-P3060-1112, Argonne National Laboratory, Argonne, Illinois.

[21] Burer, S., Letchford, A. N., 2012. Non-convex mixed-integer nonlinear programming: A survey. Surveys in Operations Research and Management Science, 17(2):97–106.

[22] Savelsbergh, M., 1994. Preprocessing and Probing Techniques for Mixed Integer Programming Problems. INFORMS Journal on Computing, 6:445–454.

[23] Dittel, A., Fügenschuh, A., Göttlich, S., Herty, M., 2009. MIP Presolve Techniques for a PDE-based Supply Chain Model. Optimization Methods and Software, 24(3):427 – 445.

[24] Bossel, H., 2007. Systems and Models: Complexity, Dynamics, Evolution, Sustainability. Books on Demand GmbH, Norderstedt.

[25] Bossel, H., 2007. System Zoo 2 Simulation Models: Climate, Ecosystems, Resources. Books on Demand GmbH, Norderstedt.

[26] Bossel, H., 2007. System Zoo 3 Simulation Models: Economy, Society, Development. Books on Demand GmbH, Norderstedt.

[27] Lotka, A. J., 1925. Elements of Physical Biology. Williams and Wilkins Company.

[28] Volterra, V., 1926. Variazioni e fluttuazioni del numero d'individui in specie animali conviventi. Memorie della Regia Accademia Nazionale dei Lincei. Ser. VI, 2:31 – 113.

[29] Forrester, J., 1973. World Dynamics. Waltham, MA, Pegasus Communications.

[30] Meadows, D., Meadows, D., Randers, J., Behrens III, W., 1972. The Limits to Growth. Universe Books, New York, NY.