

---

Konrad-Zuse-Zentrum  
für Informationstechnik Berlin

Takustraße 7  
D-14195 Berlin-Dahlem  
Germany

ARMIN FÜGENSCHUH, INGMAR VIERHAUS

**System Dynamic Optimization in the  
Sustainability Assessment of a  
World-Model**

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](mailto:bibliothek@zib.de)  
URL: <http://www.zib.de>

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

# System Dynamic Optimization in the Sustainability Assessment of a World-Model

A. Fügenschuh<sup>1</sup>, I. Vierhaus<sup>2</sup>

<sup>1</sup> Helmut Schmidt University / University of the Federal Armed Forces Hamburg, Germany

<sup>2</sup> Department Optimization, 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 nonlinear. Therefore seemingly simple systems can show a nonintuitive, nonpredictable behavior over time. Controlling a dynamical system means to define a desired final state in which the system should be, and to specify potential interventions from outside that should keep the system on the right track. The central question is how to compute such globally optimal control for a given SD model. We propose a branch-and-bound approach that is based on a bound propagation method, primal heuristics, and spatial branching. We apply our new SD-control method to a small System Dynamics model, that describes the evolution of a social-economic system over time. We examine the problem of steering this system on a sustainable consumption path.

## Keywords:

System Dynamics; Mixed-Integer Nonlinear Optimization

## 1 INTRODUCTION

In sustainability science, the object of examination and the manipulation are often complex, large and dynamical systems [1, 2]. These systems can be found on the macroscopic socioeconomic scale [3, 4] as well as on the microscopic level in individual manufacturing processes (see [5] and the references therein). The reaction of these systems to external interventions can be counter intuitive, so that even when a quantitative goal has been identified and resources have been dedicated to achieving this goal, its implementation is non trivial [6].

For this reason, the study of complex dynamical systems is an essential part of sustainability research. A traditional and successful approach to modelling dynamical systems is the System Dynamics (SD) methodology. SD was the basis for the report "Limits to Growth" [3], which introduced one of the first concepts of sustainability. For an introduction to SD and its applications we refer to Sterman [7].

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 models can typically be simulated with very little computational effort. The simulation with different parameter sets can be used to investigate different scenarios, as was done, for example, in the "Limits to Growth". When the question shifts from understanding the future behavior to understanding how to influence the system in the future, a systematic optimization approach is necessary. This entails the definition of an objective or goal function which maps 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 [8–15]. All these approaches have in common that they at best only provide feasible or local opti-

mal solutions. Moreover, as pointed out by Burns and Janamanchi [16], 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 by developing a computational method that is able to handle all kinds of SDO models and yields feasible solutions as well as 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. Therefore 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 [17]. 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 [18, 19]. Information on the MINLP framework SCIP which we apply is given in Achterberg [20], and in particular on nonlinear aspects of SCIP in Berthold, Heinz, and Vigerske [21].

## 2 Our Approach

In the following we describe our approach to solve mixed-integer dynamic optimization problems that originate from SDO models. As a foundation we use the linear and nonlinear programming based mixed-integer nonlinear solver SCIP [20]. 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, it became clear that the special constraint structure of the SDO is for the most part intractable for the solver. However, if this structure is known and exploited to a high degree, the solution process can be fundamentally improved. In this paper, we will address the two aspects of the branch-and-bound solution approach, with the highest potential of improvement.

### 2.1 Presolving Bound Propagation

The task of bound propagation on the states of the MINLP formulation of an SDO is identical to a reachability analysis [22, 23] in discrete time. States that cannot be reached using the given parameter and control intervals shall be excluded from the solution space. When applying this analysis, it is essential to take the discretization and the type of parameter limits in the problem into account. We will describe our method in detail in Section 4.2.

### 2.2 Specialized Primal Heuristics

Integer variables enter into the MINLP formulation of an SDO when a non-smooth function like  $\min, \max, |\cdot|$ , or a logical function needs to be implemented. For example, we formulate a minimum function  $x = \min(y, 0), y \in [\underline{y}, \bar{y}]$  with the following constraints and variables in the MINLP formulation:

$$\begin{aligned} y^+, y^- &\in \mathbb{R}^+, \\ y^b &\in \{0, 1\}, \\ y &= y^+ - y^-, \\ y^+ &\leq \bar{y}y^b, \\ y^- &\leq \underline{y}(y^b - 1), \\ x &= -y^-. \end{aligned} \tag{1}$$

The formulation of the minimum function requires the introduction of three additional variables and three constraints in addition to the constraint assigning the value to  $x$ . If  $x, y$  are time dependent variables, these numbers need to be multiplied by the number of discretized time steps. However, there is no degree of freedom in the values of the additional variables, once  $y$  is known (except the degenerate case  $y = 0$ ). Using the special structure of this formulation, we can easily construct a feasible solution for the additional variables depending on  $y$ . However, if the structure is unknown, a MINLP consisting of a number of system analogous to (1) is already a difficult feasibility problem. A primal heuristic that employs knowledge of the special structures of the problem is described in Section 4.1

Both procedures, the bound propagation and the primal heuristics, are used as plug-in modules in SCIP. Together, they make up the new solver SCIPSD. In Section 5 we demonstrate how SCIPSD improves the solution process by example.

## 3 Model

In this paper we will concentrate on one particular SDO which is based on a classical System Dynamic simulation model, taken from the literature (c.f. Bossel [24–26]). We will discuss the properties of the proposed algorithms using that model as an example. For the

general formulation of our bound propagation and primal heuristics, we refer the reader to [27].

### 3.1 Model description and goal

The selected base model is the *Miniworld* introduced in [26]. It is a simplification of the much more sophisticated World2 and World3 model of Forrester [28] and Meadows et al. [3]. 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.

As discussed in Section 1, we will extend this model to an SDO, by introducing an objective function and a control mechanism.

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  $\xi$  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  $\xi$ .

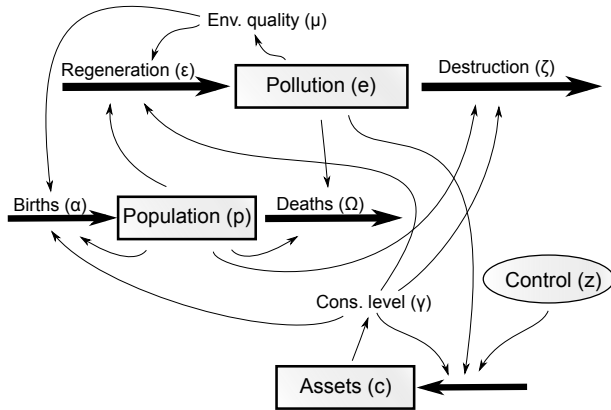


Figure 1: System Dynamics diagram for the mini-world control problem.

The SDO diagram is shown in Figure 1. 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  $(\Delta t)$  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  $i = 0, 1, \dots, T$  approximate the functions of the respective same name. Furthermore, we introduce the following real-valued continuous variables, each for  $i = 0, 1, \dots, T$ :  $\alpha_i$  are the number of births and  $\Omega_i$  are the number of deaths at time step  $i$ .  $\gamma_i$  is the consumption level.  $\mu_i$  describes the environmental quality. The environmental conditions change over time, and for this we introduce  $\varepsilon_i$  for the environmental recovery and  $\zeta_i$  for the environmental destruction.

### 3.2 MINLP Formulation

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

$$\max \sum_{i=0}^T z_i \quad (2a)$$

$$s.t. \frac{p_{i+1} - p_i}{\Delta t} = \alpha_i - \Omega_i, \quad i = 0, 1, \dots, T, \quad (2b)$$

$$\alpha_i = 0.03 \cdot p_i \cdot \mu_i \cdot \gamma_i, \quad i = 0, 1, \dots, T, \quad (2c)$$

$$\Omega_i = 0.01 \cdot p_i \cdot e_i, \quad i = 0, 1, \dots, T, \quad (2d)$$

$$\frac{e_{i+1} - e_i}{\Delta t} = \varepsilon_i - \zeta_i, \quad i = 0, 1, \dots, T, \quad (2e)$$

$$\zeta_i = 0.02 \cdot p_i \cdot \gamma_i, \quad i = 0, 1, \dots, T, \quad (2f)$$

$$e_i = e_i^+ - e_i^- + 1.0, \quad i = 0, 1, \dots, T, \quad (2g)$$

$$e_i^+ \leq 20.0 \cdot x_i, \quad i = 0, 1, \dots, T, \quad (2h)$$

$$e_i^- \leq 1.0 - x_i, \quad i = 0, 1, \dots, T, \quad (2i)$$

$$\varepsilon_i = 0.1 \cdot (1.0 - e_i^-), \quad i = 0, 1, \dots, T, \quad (2j)$$

$$1.0 = e_i \cdot \mu_i, \quad i = 0, 1, \dots, T, \quad (2k)$$

$$\frac{c_{i+1} - c_i}{\Delta t} = 0.05 \cdot \gamma_i \cdot e_i \cdot \left(1 - \left(\frac{\gamma_i \cdot e_i}{z_i}\right)\right), \quad (2l)$$

$$i = 0, 1, \dots, T,$$

$$c_i = \gamma_i, \quad i = 0, 1, \dots, T, \quad (2m)$$

$$p_i \geq \xi, \quad i = t, t+1, \dots, T, \quad (2n)$$

$$p_0 = \bar{p}, \quad (2o)$$

$$e_0 = \bar{e}, \quad (2p)$$

$$c_0 = \bar{c}, \quad (2q)$$

$$p_i, e_i, e_i^+, e_i^-, c_i, z_i \in \mathbb{R}_+, \quad i = 0, 1, \dots, T, \quad (2r)$$

$$\alpha_i, \Omega_i, \gamma_i, \mu_i, \theta_i, \varepsilon_i, \zeta_i \in \mathbb{R}_+, \quad i = 0, 1, \dots, T, \quad (2s)$$

$$x_i \in \{0, 1\}, \quad i = 0, 1, \dots, T. \quad (2t)$$

The change in the population  $p$  from time step  $i$  to  $i+1$  is a result of the births  $\alpha_i$  minus the deaths  $\Omega_i$  in time step  $i$  (2b). The number of births  $\alpha_i$  in time step  $i$  is proportional to size of the population  $p_i$ , the environmental quality  $\mu_i$ , and the consumption level  $\gamma_i$ , where 0.03 is the proportionality factor (birth rate) (2c). The number of deaths  $\Omega_i$  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 (2d).

The change in the environmental pollution  $e$  from time step  $i$  to  $i+1$  is a result of the negative environmental destruction  $\zeta_i$  and the positive environmental recovery

$\varepsilon_i$  in time step  $i$  (2e). The environmental destruction  $\zeta_i$  in time step  $i$  is proportional to the population size  $p_i$  and the consumption level  $\gamma_i$  (2f). Here 0.02 is the proportionality factor (environmental destruction rate). The environment is able to recover over time. If the environmental quality  $\mu_i$  is above a certain threshold value (here 1.0), then the recovery  $\varepsilon_i$  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  $\mu_i$  is below the threshold value, then the recovery rate is no larger than 0.1 (2j). This is equivalent to a minimum function  $0.1 \min(1, \mu_i)$ , which is implemented with the auxiliary variables  $e_i^+$ ,  $e_i^-$  and  $x_i$  and the additional constraints (2g), (2h) and (2i) as described in Section 2.2.

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 (2l), which depends on both the consumption level  $\gamma_i$  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  $\gamma_i$  equals the production  $c_i$  (2m).

The population  $p_i$  must not fall short of the given level  $\xi_i$  in each time step  $i$  (2n). Initially, in time step  $i = 0$ , the size of the population (2o), the environmental pollution (2p), and the consumption resp. production (2q) are given.

The system of equations and inequalities (2) completely defines the model. Setting  $z_i = 0$  for all times  $i$  and choosing initial values  $p_0, e_0$  and  $c_0$  reproduces the Miniworld simulation or initial value problem (IVP).

### 3.3 Constraint Classification

We distinguish five types of constraints, that are typical for any SDO:

The discretized differential equations (2l), (2b), (2e) define the state variables  $p_i, e_i, c_i$  at a given time, in terms of the previous time.

The *explicit algebraic equations* (2c), (2d), (2f), (2j), (2k), (2k), (2m), define the algebraic variables on the left hand side. These equations depend on coinciding states and other coinciding algebraic variables.

The equations (2g), (2h) and (2i) define a *system of algebraic equations and inequalities* which implicitly but uniquely defines the algebraic variables  $e_i^+$  and  $e_i^-$ , as described in Section 2.2.

Finally, equation (2n) defines a *linear state constraint*, and equations (2p), (2p) and (2q) defines *initial values* as mentioned above.

## 4 Improving the Solution of SDOs

In this section we will describe the proposed presolving bound propagation method and the proposed primal heuristic.

### 4.1 Primal Heuristics

Linear programming based branch-and-bound methods for solving mixed-integer linear and nonlinear problems are on the one hand today's method of choice for verifying the global optimality of a solution to a given instance of some model. On the other hand, they are also known for their actual weakness in creating such solutions. A pure branch-and-bound method can only come up with a feasible solution, if the linear programming relaxation of the current node is feasible for the non-relaxed model. This, however, is a rare exception. Thus it is necessary to provide feasible solutions from other sources. Having a good feasible solution early in the solution process allows to prune the search tree. Ideally, large portions of the otherwise exponential sized search tree do not have to be created and examined.

In modern MINLP solvers a large variety of different search strategies is used to construct feasible solutions, see [29–31] and the survey [32], for instance. These methods are designed and applied to general purpose MINLP problems. As reported by the respective authors, they are good in identifying solutions on a wide range of different problems. Some of them are readily available within SCIP. However, these have problems in finding solutions for dynamic MINLPs. This motivated our approach, which is presented in the following.

As described in Section 3, setting all controls in an SDO to zero, results in an IVP. If no path constraints like (2n) exist, then every solution of the IVP is a feasible solution of the SDO. From this, a fast primal heuristic for the model can be constructed. We set the control to a constant value for all times  $z_i = z_c$  and then iterate the following steps for each discretized time  $i$  except  $i = T$ . We start at  $i = 0$ , where the states are given by the initial values:

1. Calculate the values of  $\gamma_i, \mu_i$ , from the values of the state variable. Since every variable on the right hand side is fixed, we can treat the corresponding constraints as an equation with a unique solution.
2. Calculate the values of  $\alpha_i, \Omega_i, \epsilon_i, \xi_i$ , from the values of the states and the already calculated algebraic variables, by treating the corresponding constraints as equations as before.
3. Set  $e_i^- = -\min(1, \mu_i)$ ,  $e_i^+ = \mu_i - e_i^-$  and  $x_i$  to 1 if  $\mu_i > 1$ , otherwise to 0.
4. Calculate the values of the state variables in  $i+1$ , using the values computed in steps 1, 2, 3.

This procedure can be easily generalized to general SDOs.

In one run of our current implementation, we attempt to construct two solutions for the problem by setting  $z_i = \bar{z}$  in the first, and  $z_i = \underline{z}$  in the second run.

As mentioned above, this method may not yield a feasible solution if there are state constraints. To react to these cases, we add a check at the end of the propagation at each time if any state constraints are violated by

the states calculated in the current step. If a violation is detected, we flip the control from  $\bar{z}$  to  $\underline{z}$  in the first run, and from  $\underline{z}$  to  $\bar{z}$  in the second run. It is also possible to backtrack a given number of timesteps, and apply the flip at an earlier point.

## 4.2 Presolving Bound Propagation

From the modelers perspective, it appears unnatural to give bounds to the states of the system. The definition of initial states or bounds on the initial states and bounds on the control already defines a finite state space, that should not be reduced by giving additional bounds. However, finding the exact bounds defining that state space is usually not possible in finite time. On the other hand, tight bounds are essential in speeding up the branch-and-bound process 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.

A crucial step in solving mixed-integer linear or non-linear 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 [32–36].

In this section, we consider bound propagation in presolving. The usual approach for MINLPs is to consider each constraint individually, and to attempt the deduction of tighter bounds on each variable of the constraint from the bounds of the other variables [21]. For our model, this approach yields bounds that quickly diverge after the first few timesteps. This phenomenon of "exploding bounds" is characteristic for SDO problems, and could be avoided by considering the full problem and solving the maximum and minimum problems for each state variables. However, the computational effort would be comparable to the original control problem, rendering this approach useless.

We propose a compromise between these two ap-

proaches. Instead of considering a single constraint, we formulate subproblems, by selecting constraints at a given time. We introduce the *lookback parameter*  $h \geq 0$ , which defines how far the problem extends in time. We define the subproblem at time  $i$  with lookback parameter  $h$ , as the subproblem  $s_{t,h}$  consisting of the constraints (2b) through (2n), where  $i \in \{t-h, t-h+1, \dots, t\}$ .

Our proposed bound propagation method iterates the following steps for each discretized time  $i$  except  $i = T$ , starting at  $i = 0$ :

1. Formulate the subproblem  $s_{i,h}$ .
2. For each algebraic variable  $v_i$  at time  $i$ :
  - a) Solve the maximization problem  $\max v_i$  subject to the constraints of  $s_{i,h}$  to optimality and set the upper bound of the variable  $\bar{v}_i$  to the solution value.
  - b) Solve the minimization problem  $\min v_i$  subject to the constraints of  $s_{i,h}$  to optimality and set the lower bound of the variable  $\bar{v}_i$  to the solution value.
3. For each differential variable  $w_{i+1}$  at time  $i+1$ :
  - a) Solve the maximization problem  $\max w_{i+1}$  subject to the constraints of  $s_{i,h}$  to optimality and set the upper bound of the variable  $\bar{w}_{i+1}$  to the solution value.
  - b) Solve the minimization problem  $\min w_{i+1}$  subject to the constraints of  $s_{i,h}$  to optimality and set the lower bound of the variable  $\bar{w}_{i+1}$  to the solution value.

The subproblems are solved with a time limit. If the time limit is reached before the problem is solved to optimality, the considered bound is set to the best dual bound.

## 5 Computational Results

For our computational experiments, 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 4 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 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 (2n). 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.

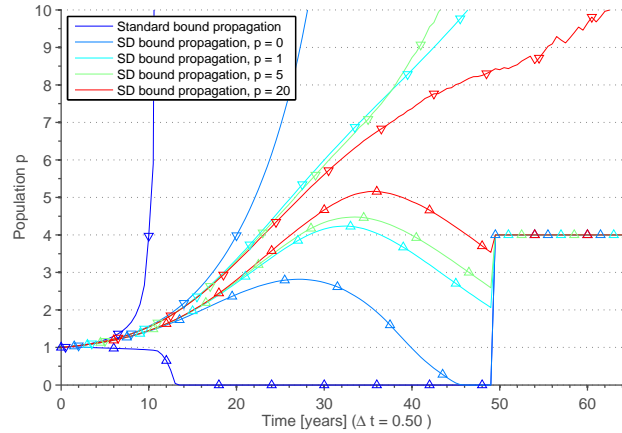


Figure 2: Presolved bounds for different look-back levels.

In Figure 3 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 4 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 cut off part.

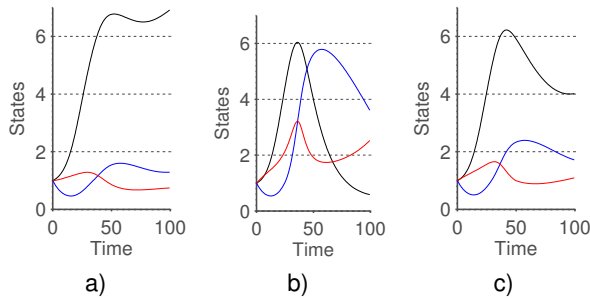


Figure 3: Black lines:  $p$ , blue:  $e$ , red:  $c$ . a) Feasible but suboptimal solution with control  $z_0 = z_1 = 1$ . b) Infeasible solution with control  $z_0 = z_1 = 10$ . c) Optimal solution with  $z_0 = 1.79$  and  $z_1 = 2.16$ .

We apply our bounds strengthening presolve routine. The results for different look-back levels are shown in Figure 2 and Table 1. The SCIP-level (first line in Table 1) 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.

level	presolve		branch-and-bound	
	total time [h]	dual	dual	gap
SCIP	0.002	–	–	–
0	1.33	$6 \cdot 10^9$	625	$\gg 100\%$
1	4.42	640	638	71.1%
2	2.702	581	579	68.1%
5	4.53	508	506	63.5%
10	9.01	479	476	61.3%

Table 1: Computational results for different presolve look-back levels. Primal bound during branch-and-bound was 184.655 for all instances. Running time of the branch-and-bound runs was one hour.

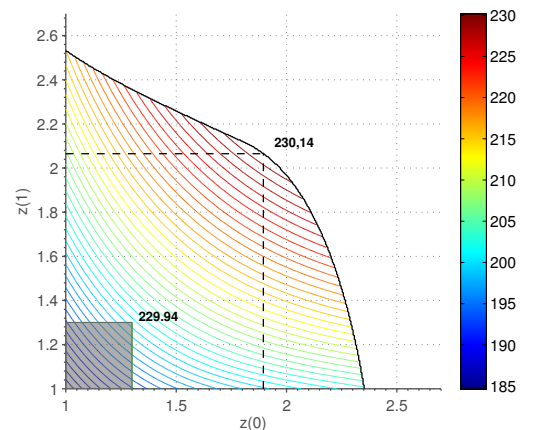


Figure 4: Objective function values for piecewise constant controls varying in  $[1, 10] \times [1, 10]$ . The (unique) optimal solution is in  $(1.79, 2.16)$ . The part of the control space with  $z_1 \leq 1.3$  and  $z_2 \leq 1.3$  marked by the gray box can be cut off. No better feasible solutions can be found in here.



A part of the search space that does not require further attention due to the bounding argument is shown in Figure 4. 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.

## 6 Summary and Conclusions

We presented a System Dynamics Optimization (SDO) as an extension of classical System Dynamics Simulation (SD) model, by introducing a control mechanism and an objective function into the system. To solve such an optimization problem, 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 an 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 have introduced a specially tailored bounds strengthening as preprocessing. We have also presented a primal heuristic to identify feasible solutions earlier.

We tested our methods on the mini-world SDO problem, and presented the results for several parameters. The presolving bound propagation yields better bounds for longer look-back-levels, which however comes at growing computational costs. With the bounds from presolving and our primal heuristic, we were able to

compute a primal solution and show that its gap is at most 61%, using our branch-and-bound approach. We also presented an example of a reduction of the search space by the bounding argument, that is the basis for our branch-and-bound approach.

We believe that our methods are generally applicable to all different kinds 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).

## References

- [1] Folke, C., Carpenter, S., Elmqvist, T., Gunderson, L., Holling, C., Walker, B., 2002. Resilience and Sustainable Development: Building Adaptive Capacity in a World of Transformations. *AMBIO: A Journal of the Human Environment*, 31:437–440.
- [2] Todorov, V., Marinova, D., 2011. Modelling sustainability. *Math. Comput. Simul.*, 81(7):1397–1408.
- [3] Meadows, D., Meadows, D., Randers, J., Behrens III, W., 1972. *The Limits to Growth*. Universe Books, New York, NY.
- [4] Scoones, I., 2007. Dynamic systems and the challenge of sustainability. STEPS working paper. STEPS Centre.
- [5] Kibira, D., Jain, S., McLean, C. R., 2009. A System Dynamics Modeling Framework for Sustainable Manufacturing. In *Proceedings of the 27th Annual System Dynamics Society Conference*.
- [6] Ghaffarzadegan, N., Lyneis, J., Richardson, G. P., 2011. How small system dynamics models can help the public policy process. *Syst. Dyn. Rev.*, 27(1):22–44.
- [7] Sterman, J. D., 2000. *Business Dynamics – Systems Thinking and Modeling for a Complex World*. Irwing McGraw-Hill, Boston.
- [8] 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.

- [9] 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.
- [10] Dangerfield, B., Roberts, C., 1996. An Overview of Strategy and Tactics in System Dynamics Optimization. *Journal of the Operational Research Society*, 47:405–423.
- [11] 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.
- [12] Fu, M., 1994. Optimization via simulation: A review. *Annals of Operations Research*, 53(1):199–247.
- [13] Keloharju, R., Wolstenholme, E., 1989. A Case Study in System Dynamics Optimization. *Journal of the Operational Research Society*, 40(3):221–230.
- [14] Keloharju, R., Wolstenholme, E., 1988. The basic concepts of system dynamics optimization. *Systemic Practice and Action Research*, 1(1):65–86.
- [15] Gustaffson, L., Wiechowski, M., 1986. Coupling DYNAMO and optimization software. *System Dynamics Review*, 2(1):62–66.
- [16] 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.
- [17] Dantzig, G., 1963. *Linear programming and extensions*. Princeton University Press and RAND Corporation.
- [18] Tawarmalani, M., Sahinidis, N., 2004. Global optimization of mixed-integer nonlinear programs: A theoretical and computational study. *Mathematical Programming*, 99:563–591.
- [19] Tawarmalani, M., Sahinidis, N., 2005. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, 103:225–249.
- [20] Achterberg, T., 2009. SCIP: Solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41.
- [21] 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.
- [22] Asarin, E., Dang, T., Girard, A., 2003. Reachability analysis of nonlinear systems using conservative approximation. In In Oded Maler and Amir Pnueli, editors, *Hybrid Systems: Computation and Control*, LNCS 2623. Springer-Verlag, 20–35.
- [23] Scott, J., of Technology. Dept. of Chemical Engineering, M. I., 2012. *Reachability Analysis and Deterministic Global Optimization of Differential-algebraic Systems*. Massachusetts Institute of Technology, Department of Chemical Engineering.
- [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] Fügenschuh, A., Vierhaus, I., 2013. *A Global Approach to the Optimal Control of System Dynamics Models*.
- [28] Forrester, J., 1973. *World Dynamics*. Waltham, MA, Pegasus Communications.
- [29] Nannicini, G., Belotti, P., 2012. Rounding-based heuristics for nonconvex MINLPs. *Mathematical Programming Computation*, 4:1–31.
- [30] Bonami, P., Cornuéjols, G., Lodi, A., Margot, F., 2009. A feasibility pump for mixed integer nonlinear programs. *Mathematical Programming*, 119:331–352.
- [31] Berthold, T., Gleixner, A., 2013. Undercover: a primal MINLP heuristic exploring a largest sub-MIP. *Mathematical Programming*.
- [32] 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.
- [33] 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.
- [34] 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.
- [35] 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.
- [36] Savelsbergh, M., 1994. Preprocessing and Probing Techniques for Mixed Integer Programming Problems. *INFORMS Journal on Computing*, 6:445–454.