Global and Local Optimal Control of a Resource Utilization Problem

I. Vierhaus[1], A. Fügenschuh[2]

[1] Zuse Institute Berlin, Germany

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

Abstract: System Dynamic models describe physical, technical, economical, or social systems using differential and algebraic equations. In their purest form, these models are intended to describe the evolution of a system from a given initial state. In many applications, it is possible to intervene with the system in order to obtain a desired dynamic or a certain outcome in the end. On the mathematical side, this leads to control problems, where aside from the simulation one has to find optimal intervention functions over time that maximize a specific objective function. Using a dynamical model for the utilization of a natural nonrenewable resource of Behrens as a demonstrator example, we present two main mathematical solution strategies. They are distinguished by the quality certificate on their respective solution: one leads to proven local optimal solution, and the other technique yields proven global optimal solutions. We present implementational and numerical issues, and a comparison of both methods.

Keywords:
System Dynamics; Optimal Control; Nonlinear Optimization; Bounds Strengthening.

1 Introduction

The French writer Gustave Flaubert wrote already in 1841 [11]: "Si la Société continue à aller de ce train il n'y aura plus dans deux mille ans ni un brin d'herbe ni un arbre; ils auront mangé la nature.[1]" Being then very much ahead of his time, his foreboding has become certainty in our days, as the World Model of Forrester [13] and Limits to Growth studies of Meadows et al. [21] showed, and which corrected Flaubert only in the sense that society will probably have much less than the proclaimed "deux mille ans".

The goal of our work is to take System Dynamics models as a basis, and try to move their dynamic behavior into a desired direction by introducing one or several control functions that interact with the model's structure. Furthermore, we introduce an objective function that measure success by assigning a single score value to the outcome of the controlled System Dynamics model. It is natural to ask for the best possible or optimal control that maximizes (or minimizes, depending on the purpose of the model) this score value. 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, see [6,9,10,14,16–20]. We intend to use methods that give a certain quality certificate for the computed solutions, which is either a proven local optimal solution or a proven global optimal solution.

We formulate an SDO problem as nonlinear program (NLP). 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 (local or global) optimality. For local optimal solutions, there are numerical nonlinear solvers, such as IPOPT [27] or CONOPT [8], which we briefly present below. For global optimal solutions, 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 first reformulate the nonlinear problem with non-smooth functions as a mixed-integer nonlinear program (MINLP). We then 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 [7]. The so obtained solution value defines a (lower) bound on the optimal value of the original NLP problem. In case this solution is NLP feasible, it would be a proven global optimal NLP solution. However, this rarely happens in practice. Hence we either add cutting planes to strengthen

[1] If society continues to proceed in its current pace, then there will be nothing left in two thousand years, not a blade of grass, not a single tree; they will have eaten the nature.

the relaxation, or we decide to branch on a variable. For more details on cutting planes and branch-and-bound for MILP we refer to Nemhauser and Wolsey [22], and for an application of this framework to global mixed-integer nonlinear programming to Smith and Pantelides [23], and Tawarmalani and Sahinidis [24,25]. Information on the MINLP framework SCIP which we apply is given in Achterberg [2], and in particular on nonlinear aspects of SCIP in Berthold, Heinz, and Vigerske [5].

2 Demonstrator: Natural Resource Utilization

In 1972, Behrens [4] formulated a System Dynamics model to describe *the dynamics of natural resource utilization*. It is based on the observation that the earth's minerals are a finite source, hence their future availability needs to be carefully planned. Three main interacting feedback loops were identified to describe the long-term behavior over time. All three are negative or goal-seeking loops.

1. If the *Actual Cost* rise, then the *Demand* decreases. The *Demand* lowers the amount of *Natural Resources*, and the *Actual Grade* of the resource will decrease, hence the *Actual Cost* of excavating the resource will rise more.

2. If the *Actual Cost* increases, then *Sales Revenue* will go up, hence the *Investment in R&D* rises, which leads to a *Technology Change*, so that more new *Technology* reduces the *Actual Cost*.

3. If the *Actual Cost* go up, then the *Potential Substitution Fraction* also increases, and with a delay the actual *Substitution Fraction* rises. Hence the *Demand* lowers, and so does the *Actual Cost*.

Behrens analyses this system's behavior. In a standard run, it is assumed that the annual growth rate of the resource consumption is a fixed value of 3%, which is said to be a conservative assumption for most resources. Hence the consumption grows exponentially over time, which leads to a fast decline in the still available amount of the nonrenewable resource. From its initial value at the beginning of the simulated time horizon in 1970, it only takes 150 years until the natural resource is fully consumed. The actual cost stays almost constant for the first 50 years (until 2020) (attributed to a change in technology), but afterwards the technology change cannot compensate the ever growing usage rate. Thus, the actual cost increases between 2020 and 2070 to a maximum, and, after reaching a peak in 2040, the usage rate steeply declines, until it reaches zero around 2120. At that time, all further demand must be satisfied from recycled products.

Behrens evaluates several strategies or scenarios in order to find out, whether this collapse of the natural resource is avoidable. For example, if one doubles the amount of resources (better excavation technology might be able to do so), then the collapse is shifted into the future by some 50 years only. Also a subsidy of research and development investment does not lead to a significantly different outcome, compared to the standard run. According to Behrens, the only chance is a significant reduction of the yearly demand growth rate from 3% to just 1%, so that the resource can be used for about 75 years longer (in comparison with the standard run).

We use this model as a demonstrator for our System Dynamics Optimization techniques. To this end, we introduce control functions (or controls, for short) that represent some interventions to the system, in order to move its dynamical behavior into a desired direction. Here, our goal is to move the decline of the usage rate as far as possible into the future. Equivalently, we aim to minimize the actual cost for the resource. As controls, we introduce a tax that artificially increases the actual cost. This revenue is used as R&D investment. The question is, what amount of tax should be charged in which year, in order to achieve this goal? This problem is solved by our optimization techniques.

3 Formulating System Dynamics Optimization Problems

3.1 System Dynamics Models as DAEs

From a mathematical point of view, any Sytem Dynamics model is a set of differential and algebraic equations. The differential equations correspond to the stocks integration statements, while the algebraic equations correspond to the models auxiliary equations. We denote the states (or stocks) by

$x_1(t), x_2(t), \ldots, x_p(t)$ and summarize them using the state vector $x(t)$. Analogously, we define the vector of all auxiliary variables (including rates) as $y(t) = (y_1(t), y_2(t), \ldots, y_q(t))$ and the set of (non time dependent) simulation parameters by $p = (p_1, p_2, \ldots, p_r)$. We consider simulations with a fixed time horizon $t \in [0, T]$.

Simulating one run of a system dynamics model is then equivalent to finding $x(t)$, $y(t)$ that satisfy the following system of differential algebraic equations (DAE) for given parameters p:

$$\dot{x}(t) = f(x(t), y(t), p) \tag{1a}$$
$$y(t) = g(x(t), y(t), p) \tag{1b}$$

3.2 Discretization

In system dynamics, this problem is usually solved numerically with an explicit discretization scheme like the Euler Method. The variables x, y are no longer defined continuously on t, but only on certain equidistiant time intervals. We denote discrete times as bracketed superscript and rewrite the system (1) as

$$\dot{x}^{(i)} = f\left(x^{(i)}, y^{(i)}, p\right), \tag{2a}$$
$$y^{(i)} = g\left(x^{(i)}, y^{(i)}, p\right), \tag{2b}$$
$$n = \frac{T}{\Delta t} \tag{2c}$$
$$i \in \{0, 1, 2, \ldots, n\}, \tag{2d}$$
$$t^{(i)} = i\Delta t. \tag{2e}$$

Using the euler step $x^{(i+1)} = x^{(i)} + \Delta t \dot{x}^{(i)}$, the model is then solved by succesively evaluating the explicit equations $y^{(0)}, x^{(0)}, y^{(1)}, x^{(1)}, \ldots, y^{(n)}, x^{(n)}$.

3.3 System Dynamics Optimization Problem

In this paper, we will consider a system dynamics optimization problem. This means, that we will equip the original model with an objective function, and a set of control parameters, which can be constant or change over time. In particular, we choose one or more parameters from the vector p, and consider them time dependent ($p \to p(t)$). The time dependence is either on the same scale as t, or on a coarser scale with a switching step $s\Delta t$, $s \in \mathbb{Z}^+$. With the above definitions, an arbitrary, discretized SDO using an explicit one-step integration scheme can be written as follows:

$$\max\ c(x, y, p), \tag{3a}$$
$$\text{s.t. } x^{(i+i)} = f\left(x^{(i)}, y^{(i)}, p^{(i)}\right), \quad i \in \{0, 1, \ldots, n-1\}, \tag{3b}$$
$$y^{(i)} = g\left(x^{(i)}, y^{(i)}, z^{(i)}\right), \quad \in \{0, 1, \ldots, n\}, \tag{3c}$$
$$x^{(i)} \in \mathbb{R}^p, \quad i \in \{0, 1, \ldots, n\}, \tag{3d}$$
$$y^{(i)} \in \mathbb{R}^q, \quad i \in \{0, 1, \ldots, n\}, \tag{3e}$$
$$p_j^{(i)} \in \mathbb{R}^r, \quad i \in \left\{0, 1, \ldots, \frac{n}{s}_j\right\}. \tag{3f}$$

The system of equations (3) has the form of a nonlinear program (NLP) and is theoretically accessible with standard optimization software. The instance we consider in this paper however, has a number of features that are typical elements of system dynamics simulations, but are not understood by standard optimization solvers. In the following sections, we describe how we modified the problem to make it accessible with standard solvers.

3.4 Handling Vensim Functions

The originam model considered in this paper contains two Vensim smoothing functions, `SMOOTHI` and `DELAY3`. As described in the Vansim manual, both of these functions can be rewritten by replacing the axuiliary variable with one (in the case of `SMOOTHI`) or three (in the case of `DELAY3`) stock variables.

The remaining non-standard functions that appear in the model are eight table functions. We will describe two approaches to the reformulation of table functions, one aimed at local and one at global optimization. As described in more detail in Section 4.2, the local solution approach we propose for this model relies heavily on derivative information. Since the derivative of a piecewise linear function is not defined everywhere, we interpolate the tabled data with smooth functions. The global approach on the other hand, allows the use of integer variables and special ordered sets, making it possible to reproduce the model exactly.

Smooth Interpolation of Tabled Data

For the local approach, we used an interpolation of the tabled data using cubic splines. The splines are then supplied to the modeling language GAMS [15] as external functions. The process of reading an `.mdl` file as input and creating a `.gms` file containing the lookup data was automated with the tool `mdlconv`. The `mdlconv` as well as the tool `lookuplib`, which is used to interpolate the data and evaluate the resulting splines for GAMS during the solution process are scheduled for release in source in July 2015. The interpolation is calculated, by fitting a cubic spline to the original piecewise linear functions. The number of sampling points is increased until the interpolation does not exceed a maximum deviation between the original function and the interpolation. More information on the interpolation can be found in [26]. As an example, the interpolation of the `demandtable` is shown in Figure 1b. Deviations between the original and the smooth interpolation are not visible by eye, even at the data points. The goal in calculating these interpolations, was to find a smooth function that reproduces the original model behavior as accurately as possible. A second approach, would be to allow for more deviations in order to find a more natural function. However, since we consider a literature model in this paper, the former approach was selected.

Piecewise Linear Functions using Special Ordered Sets

To demonstarte how to formulate table functions in terms of a MINLP, we consider an arbitrary table function $u = f(v)$ with n_p data points $(u_0, v_0), (u_1, v_1), \ldots, (u_{n_p}, v_{n_p-1})$.

We now use a set of constraints and introduce at each time new sets of positive variables $\lambda_{n,k}$ where $k \in \{0, 1, \ldots, n_p\}$ that are each part of a special ordered set of type two (SOS2), introduced by Beale and Forrest [3]. Out of a set of SOS2-Variables, at least two can be non-zero, and the two need to be adjacent.

We define two vectors l_u and l_v containing the ordered list of u and v values respectively:

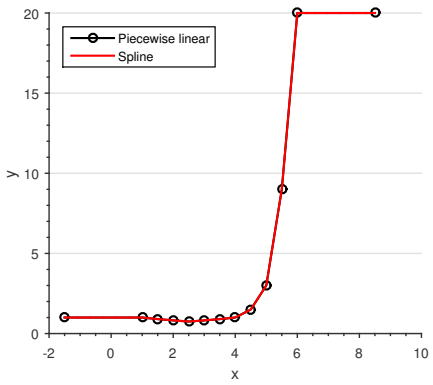
$$l_u = (u_0, u_1, \ldots, u_{n_p}) \tag{4a}$$
$$l_v = (v_0, v_1, \ldots, v_{n_p}) \tag{4b}$$

The set of constraints that we need to implement at each point in time then reads for $n(a)$:

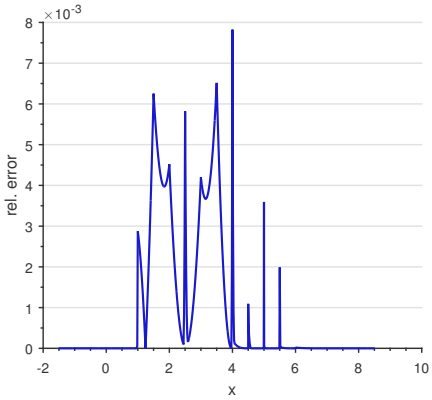
$$u = \sum_k l_{u,k} \lambda_{n,k} \tag{5a}$$
$$v = \sum_k l_{v,k} \lambda_{n,k} \tag{5b}$$
$$1 = \sum_k \lambda_{n,k} \tag{5c}$$

In system dynamics, it is not uncommon to query a table function with an argument that is outside of the defined argument interval $[u_0, u_{n_p}]$. In this case, the first or last function value is supplied. To reproduce this behavior, we added additional points at the beginning and end of the table, to create an interval with slope 0. The width of this interval was in this instance chosen manually, based on the expected maximal range of the parameter. Figure 1b shows a comparison of an original piecewise linear table function with

a smooth interpolation, as described in the next Section. The interval of the original table function was $[0, 10]$. The intervals $[-5, 0]$ as well as $[10, 15]$ were added to the model to allow for requests outside of the defined range.



(a) Interpolations of `Cost of Technology Advance Table` data



(b) Relative difference between linear and smooth interpolation of `Cost of Technology Advance Table` data

4 Local and Global Optimal Control

Optimization approaches can be roughly divided into local and global approaches. While the former are in many cases much faster, only the second yield proven quality indicators for the found solution. In this paper, we will attempt to solve the model with one global and one local method.

4.1 Global Optimal Control

After performing the reformulations described above, we can now attempt a solution of our problem using a standard branch and bound solver. However to our knowledge, there is currently no MINLP solver available, that exploits the special problem structure of a discretized control problem. However, taking this structure into account appears to be key in order to find feasible solutions as well as dual bounds to our test instance. We have implemented two methods tailored methods for MINLP formulations of SDOs:

Primal Heuristic

Finding feasible solutions is a requirement for an efficient branch-and-cut approach. To quickly produce feasible solutions, we implemented a simple heuristic, that reduces the control problem to a simulation problem, by fixing the control variables to their lower (or in a second run upper) bound. For our system, this will always yield a feasible solution, since there are not state bounds given.

Bound propagation

In a branch-and-cut algorithm, bound propagation describes the derivation of bounds from one variable to another. This is done usually along a single constraint.

Our tailored bound propagation is executed only once as part of presolving. Its goal is to determine, which values of the state and algebraic variables at each time are reachable with the given initial conditions and allowed control. In this context, the bound propagation can be considered a reachability analysis of the dynamic system.

In order to find the reachable values, we formulate subproblems $s_{i,h}$ that contain all constraints and variables of the times $t \in i - h, i - h + 1, \ldots, i$. Within this subproblem we consider finding the maximal and minimal values for all states and algebraic variables again as optimization problems.

In more detail, our 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 preset node limit. If the time limit is reached before the problem is solved to optimality, the considered bound is set to the best dual bound.

Linear Programming based Spatial-Branch-and-Cut

We relax the nonlinear constraints, and embed them in linear convex hull. The resulting linear program (LP), is solved efficiently using Dantzig's simplex algorithm [7]. The so obtained solution value defines a (lower) bound on the optimal value of the original NLP problem. In case this solution is NLP feasible, it would be a proven global optimal NLP 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 (spatial branching). For more details on cutting planes and branch-and-bound for MILP we refer to Nemhauser and Wolsey [22], and for an application of this framework to global mixed-integer nonlinear programming to Smith and Pantelides [23], and Tawarmalani and Sahinidis [24,25]. Information on the MINLP framework SCIP which we apply is given in Achterberg [2], and in particular on nonlinear aspects of SCIP in Berthold, Heinz, and Vigerske [5].

4.2 Local Methods

In the most abstract setting, we aim to solve a nonlinear optimization problem (NLP) of the general form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c(z), && (6)\\ \text{subject to} \quad & a(z) = 0, && (7)\\ & z \geq 0. && (8) \end{aligned}$$

Here the constraint function $a(z) = 0$ subsumes all constraints from (2a) and (2b), and z is the vector of all variables from (2c)–(2e).

IPOPT

As one of two options for the solution of the NLP (6), we use the solver IPOPT of Wächter and Biegler [27]. It implements a barrier method, that solves a sequence of barrier problems

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \varphi_\mu(z) := c(z) - \mu \sum_{i=1}^{N} \ln(z_i), && (9)\\ \text{subject to} \quad & a(z) = 0, && (10) \end{aligned}$$

where μ is converging to zero. For μ close to zero one obtains a Karush-Kuhn-Tucker (KKT) point as solution, which comes with a certificate of local optimality, if certain constraint qualification conditions are fulfilled. The solution of each barrier problem (9) is carried out by a variant of Newton's method, applied to the primal-dual equations

$$\begin{aligned} \nabla f(z) + \nabla a(z)\lambda - w &= 0, & (11)\\ a(z) &= 0, & (12)\\ ZWe - \mu e &= 0, & (13) \end{aligned}$$

where $Z := \mathrm{diag}(z), W := \mathrm{diag}(w), e := (1, 1, \ldots, 1)$. In order to ensure global convergence of Newton's method, a line-search variant of Fletcher and Leyffer's filter method [12] is applied. For further details, we refer to [27].

CONOPT

Our second option for the solution of the NLP (6) is the solver CONOPT of Drud [8]. This solver is particularly suitable for NLP having a periodic structure due to the dynamic constraints.

CONOPT is based on a clever implementation (i.e., choice of data structures and a careful implementation of the algorithm's components) of the generalized reduced gradient (GRG) algorithm of Abadie and Carpentier [1]. A generic GRG algorithm starts from a feasible solution of (6). Then the Jacobian matrix $J = \left(\frac{\partial a_i}{\partial z_j}\right)_{i,j}$ is computed. A basis is selected, which is a square non-singular submatrix of J, such that the basic variables are away from the bounds and the submatrix is well-conditioned. Multipliers and reduced gradients are computed. If the current solution is already a Karush-Kuhn-Tucker (KKT) point, the algorithm stops. Otherwise, a search direction is computed, and from the current solution one moves along this search direction for a certain step length. Hereto, a nonlinear subproblem is solved by Newton's method. This procedure is repeated, until it converges to a KKT point.

5 Computational Results

5.1 Definition of Control Parameters

Behrens [4] considered the possibility of introducing a subsidy into the model, i.e., doubling the investment in research and development. However, he concluded, that this measure does not succeed at keeping the variable `actualcost` lower then in the standard run. We examine, if there are other possibilities to invest money into research and development in a way that achieves a lower total actual cost then the standard run. We therefore choose the variable `percent invested in RD` as our single time dependent control variable:

$$p_0(t) = \texttt{percent invested in RD}(t) \in [0, 5] \quad (14)$$

For all other parameters we retain the values as set in the original model. Finding the optimal investment strategy to keep the `actualcost` down, is a nontrivial problem. A consistently high amount of investment will keep costs low for a longer period than in the base run, but the increase towards the end of the model will be stronger, leading to a higher accumulated `actualcost`. A consistently low investment will make the `actualcost` increase sooner, again leading to a higher accumulated value than in the base run.

5.2 Presolving and Analysis of the Global Optimization Problem

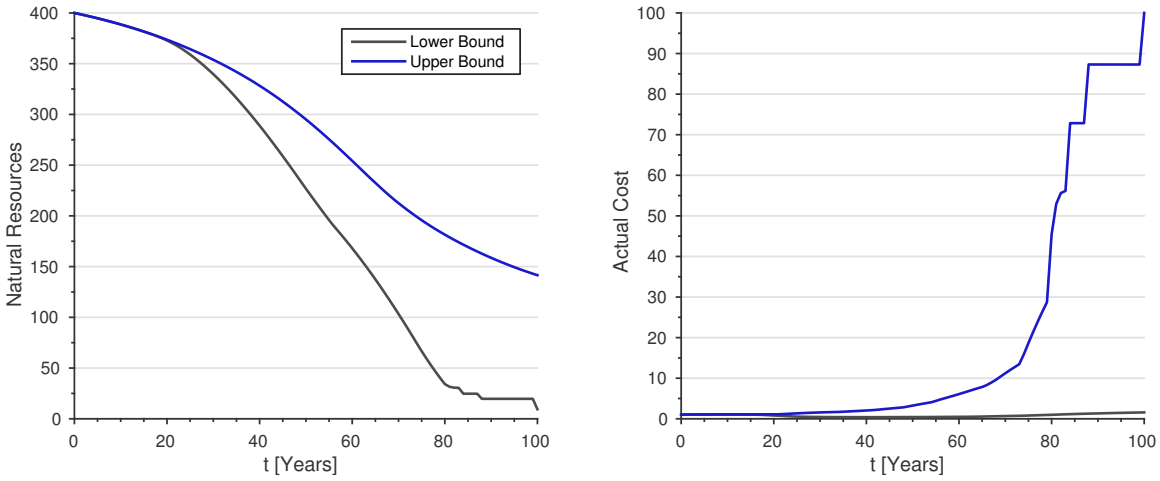
For the global approach, we implemented our tailored methods for global optimization as plug-ins to the branch-and-cut framework and solver SCIP [2,5]. For the solution of the subproblems in bound propagation and for the branch-and-cut process that follows presolving, we rely entirely on SCIP using the nonlinear solver IPOPT [27] and the linear solver CPLEX.

We run each of our calculations on one core of an HP machine, equipped with Intel Xeon E5-2690 2.90GHz processors and a total of 384 GB of memory.

Time [s]	Primal Bound	Dual Bound	Gap [%]
0	300.9	97.9	207.35%
500	300.9	101.9	195.33%
1000	300.9	102.2	194.35%

Table 1: Computational results for the global solution approach.

As a first step towards the solution of the problem, we applied our presolving bound propagation method to the exact NLP reformulation of the model. The bound propagation run took 3099 seconds. Note that the bounds computed in this presolving step do not depend on the definition of the objective function, but only on the control definition. As result of the presolving, we receive a set of bounds to the reachable states of the system. In Figure 2 we show the best computed bounds for `actualcost` and `naturalresources`. These bounds are outer approximations, i.e., not every point within the shown bounds is reachable. However, no point outside of the bounds is reachable with control values within the defined region. From the calculated bounds, we see that the behavior of the system during the first 20 years is



a) b)

Figure 2: Bounds for the state variables Natural Resources and Actual Cost computed as part of presolving in the global solution approach.

independent of our control decision. This is consistent with the delay contained in the model. Starting between year 20 and 30, it is clear that investment in R&D only has limited capabilities to change the system's behavior. The natural resource remaining will in any scenario have declined to less then 37 % after 100 years.

5.3 Global Solution Approach

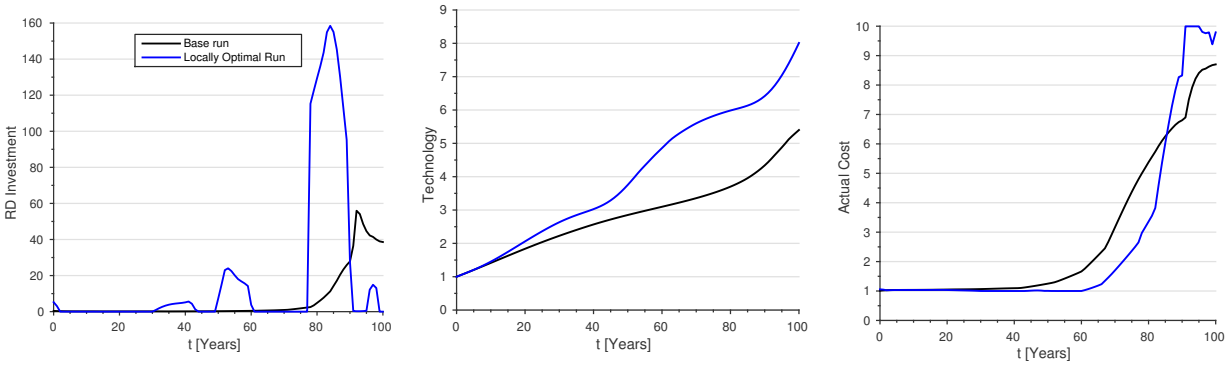
We now attempted to solve the preprocessed problem with SCIP using the tailored primal heuristic. The results of the solution process are summarized in Table 1.

With the precomputed bounds, we immediately find a lower bound to accumulated `actualcost` of 97.9. This bound can be improved slightly to 102.2 within 1000 seconds of branch and bound. However, this still leaves a gap of roughly 200%.

5.4 Local Solution Approach

For comparison, we now attempted to solve the optimization problem with interpolated table functions with the local solver CONOPT [8]. CONOPT converges towards a locally optimal solution with an objective

value of 260.2 within four seconds. To compare the local solution, we conducted a base run. For this base run, we chose as our control the values that the variable `percent invested in RD` takes in the base run of the original model. We then introduced the tax formulation and conducted one simulation run. In Figure 3, we show the locally optimal control over the selected time frame of the model. Through the first 30 years, there is no investment in research at all. Following this period, there are two periods of roughly 10 years duration, during which an increasing absolute part of the sales revenues is invested into research and development. Note, that the investment into RD is significantly higher in the optimal solution, than it is in the base run. As can be seen in panel b), this leads to a significantly better development of the subsitution technology, compared to the base run. When comparing the values of the variable `actualcost`, we see that the cost remain lower for a few years longer, but then increase more abruptly, reaching their maximum slightly before the base run. The objective value in the base run has a value of 282.0. The locally optimal solution has improved this value to 260.2.



a) b) c)

Figure 3: Comparison between the base run after introducing a tax with the optimal solution: a) absolute investment in RD, b) value of stock `technology`, c) comparison of `actualcost`.

6 Summary and Conclusions

In this paper, we outlined a global and a local solution approach to a system dynamics optimization problem. The global approach has the advantages of allowing an exact representation of the model, and of providing a proven performance indicator. However, for this very complex nonlinear model with a high number of non-smooth table functions, the gap that remains between primal and dual bound is still very large. A more extensive presolving of the problem, as well as special branching rules are currently being investigated, in order to close the gap further. However, the global approach yields state bounds as a side product, that allow for a first analysis of the effectiveness of chosen controls. The local approach, required the elimination of all non-smooth functions from the model. The resulting qualitatively equivalent model, can be solved to local optimality within a few seconds. The solution shows an interesting investment strategy with the overall goal of keeping the accumulated actual cost of the resource as low as possible. As is inherent to all local approaches, we can however not be sure how much better a globally optimal solution might be. This is currently our ongoing research.

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] J. Abadie and J. Carpentier. Generalization of the Wolfe reduced gradient method to the case of nonlinear constraints. In R. Fletcher, editor, *Optimization*, pages 37 – 47. Academic Press, New York, 1969.

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

[3] E.M.L. Beale and J.J.H. Forrest. Global Optimization Using Special Ordered Sets. *Mathematical Programming*, 10:52 – 69, 1976.

[4] William W. Behrens. The Dynamics of Natural Resource Utilization. *Simulation*, 19(2):91 – 99, 1972.

[5] Timo Berthold, Stefan Heinz, and Stefan Vigerske. Extending a CIP framework to solve MIQCPs. In Jon Lee and Sven Leyffer, editors, *Mixed Integer Nonlinear Programming*, volume 154, part 6 of *The IMA Volumes in Mathematics and its Applications*, pages 427–444. Springer Verlag, Berlin, 2012.

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

[7] George Dantzig. *Linear programming and extensions*. Princeton University Press and RAND Corporation, 1963.

[8] A. S. Drud. CONOPT: A GRG Code for large sparse dynamic nonlinear optimization problems. *Mathematical Programming*, 31(2):153 – 191, 1985.

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

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

[11] Gustave Flaubert. *Souvenirs, Notes et Pensées intimes*. 1841. Online available at URL http://gallica.bnf.fr/ark:/12148/btv1b60006475/f181.image.

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

[13] J. W. Forrester. *World Dynamics*. Wright-Allen Press, Boston, MA, 1971.

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

[15] GAMS Development Corporation. General Algebraic Modeling System (GAMS) Release 23.8.1. Washington, DC, USA, 2013.

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

[17] Caner Hamarat, Jan H Kwakkel, Erik Pruyt, and Erwin T Loonen. An exploratory approach for adaptive policymaking by using multi-objective robust optimization. *Simulation Modelling Practice and Theory*, 46:25–39, 2014.

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

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

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

[21] D.H. Meadows, D.L. Meadows, J. Randers, and W.W. Behrens III. *The Limits to Growth*. Universe Books, New York, NY, 1972.

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

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

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

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

[26] Ingmar Vierhaus, Armin Fuegenschuh, Robert Lion Gottwald, and Stefan Groesser. Modern Nonlinear Optimization Techniques for an Optimal Control of System Dynamics Models, 2015.

[27] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106:25 – 57, 2006.