Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Bi-objective optimization of dynamic systems by continuation methods

Tobias Keßler[a,*], Filip Logist[b], Michael Mangold[c]

[a] *Max Planck Institute for Dynamics of Complex Technical Systems, Process Synthesis and Process Dynamics, Sandtorstraße 1, 39106 Magdeburg, Germany*
[b] *KU Leuven, Department of Chemical Engineering, BioTeC+ - Chemical and Biochemical Process Technology and Control, Odisee Campus KU Leuven, University of Leuven, Gebroeders De Smetstraat 1 L020, B-9000 Gent, Belgium*
[c] *Technische Hochschule Bingen, Berlinstraße 109, 55411 Bingen*

Abstract

As a proof of concept the properties of path-following methods are studied for multi-objective optimization problems involving dynamic systems (also called multi-objective dynamic optimization or multi-objective optimal control problems), which have never been presented before. Two case studies with two objectives are considered to cover convex, as well as non-convex trade-off curves or Pareto sets. In order for the method to be applicable, the infinite dimensional dynamic problems have to be discretized and scalarization parameters have to be introduced, which leads to large-scale parametric nonlinear optimization problems. For both the chemical tubular reactor and the fed-batch bioreactor case study it is found that a path-following continuation approach is able to compute the Pareto fronts accurately and efficiently. A branch switching technique is required whenever a constraint switches from active to inactive or vice versa. When dealing with non-convex problems, a technique for detecting inflection points is required. Simple switching techniques are suggested and have been tested successfully.

Keywords:
multi-objective optimization, continuation techniques, Karush Kuhn Tucker conditions, Fritz-John conditions, scalarization, Pareto front

*Corresponding author
Email address: kesslert@mpi-magdeburg.mpg.de (Tobias Keßler)

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Nomenclature

Theoretical sections

β_0	Fritz-John parameter
ε	tolerance
ε_1, ε_2	disturbance
Γ	curve length parameter
$\mathcal{L}$	Lagrangian
λ, φ	lagrange multipliers
ν	Fritz-John state
ω	continuation parameter
Φ	adaption parameter
Ψ	line through solution space
τ	tangent vector
$\tilde{x}$	augmented state vector
Υ	step size
$\varkappa$	arc-length
ξ	number of iteration steps
ζ	slope of line through solution space
e	unit vector
F	vector of system equations
g	inequality constraint
h	equality constraint
J	cost functional
p	auxiliary scalar equation
s	slack variable
w	weighting factor
w^*	branch switching point
x	state vector
y	solution of algebraic system

Case study 1

α, β	reaction kinetic constant
γ, δ	reaction kinetic constant
c	reactant concentration, mol/l
c_f	feed concentration, mol/l

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

K	scaling factor
L	length of reactor, m
N	number of grid points
T_f	feed temperature, K
T_w	jacket temperature, K
T_{max}	maximum reactor temperature, K
T_{min}	minimum reactor temperature, K
$T_{w,max}$	maximum jacket temperature, K
$T_{w,min}$	minimum jacket temperature, K
v	flow velocity, m/s
x_1	dimensionless reactant concentration
x_2	dimensionless reactor temperature
z	spatial coordinate, m
Case study 2	
μ	growth rate, 1/h
π	production rate, g/gh
σ	substrate consumption rate, g/gh
c_s	substrate concentration, g/l
$c_{s,F}$	feed substrate concentration, g/l
N	number of grid points
t	time coordinate, h
t_e	terminal time, h
u	volumetric rate of the feed stream, l/h
x_1	biomass, g
x_2	substrate, g
x_3	product (lysine), g
x_4	fermenter volume, l

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

1. Introduction

Multi-objective or multi-criterion optimization deals with problems of the form:

$$\begin{aligned} \min_{x} \quad & (J_1(x), \ldots, J_q(x)), \\ \text{s.t. } & h_i(x) = 0, \quad i = 1, \ldots, n, \\ & g_j(x) \leq 0, \quad j = 1, \ldots, m, \end{aligned} \tag{1}$$

in which x are the optimization variables, J_k the objective functions, h_i the equality constraints and g_j the inequality constraints. Most decisions of everyday life can be regarded as multi-objective optimization problems, because in most cases our decisions are trade-offs between two or more possibilities. These possibilities, or objectives, can very well be contradictory (Collette and Siarry, 2003). Solutions of these problems are always trade-offs between the objectives. When comparing the solutions, improvement of one objective is only possible at the cost of deterioration of another objective. Such solutions are called *Pareto optimal*. The main difference between single-objective and multi-objective optimization is, that there is not only one optimal solution, but a set of optimal solutions (Deb, 2014). This set is called *Pareto front*. In practice, however, the user or decision maker can only use one of these solutions. The user's choice depends on other, higher level information (Deb, 2014). Therefore it is the main goal of multi-objective optimization to generate many solutions, in order to give the user a good overview about what can be chosen from (Deb, 2014). To achieve this goal, Pareto fronts are usually calculated by (i) turning the multi-objective optimization problem into a sequence of single-objective optimization problems or (ii) exploiting evolutionary methods in which a set of candidate solutions gradually evolves to the Pareto set (Miettinen, 1999; Deb, 2002). For methods from the former class, various scalarization techniques, for example weighted sum method, hyperboxing scheme and normalized normal constraint, are known in literature (Marler and Arora, 2004; Logist et al., 2009; Bortz et al., 2014). Typical challenges these methods face are ensuring a homogeneous distribution of the computed points on the Pareto front, as well as capturing non-convex Pareto fronts. An alternative approach are path-following methods resulting from numerical continuation theory, which have been suggested in literature (Rakowska et al., 1991; Lundberg and Poore, 1993; Seferlis and Hrymak, 1996; Hillermeier, 2001; Gudat et al., 2007; Harada et al., 2007;

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Potschka et al., 2011; Ringkamp et al., 2012) but so far hardly have been applied to large-scale optimization problems resulting from the optimization of dynamic systems. Path-following methods are able to easily calculate non-convex Pareto fronts. Further, they can be combined with established predictor corrector continuation methods from bifurcation analysis to solve bi-criterial optimization problems with a large number of optimization variables (Thompson Hale, 2005; Pérez, 2014). One of the major challenges of this approach is the occurrence of bifurcations due to constraints (Rao and Papalambros, 1989b,a; Guddat et al., 1990), which has been tackled recently (Martin et al., 2016), but not yet solved for large-scale problems.
The idea discussed in the following is an extension of the predictor corrector continuation algorithm reported in the conference paper (Keßler et al., 2016). It is used as a path-following method for large-scale bi-criterial optimization problems. The application to dynamic models illustrates the feasibility of the method for optimization problems with differential equations as constraints, i.e. dynamic optimization or optimal control problems. Normally such problems are solved using (i) direct optimal control approaches using gradient based methods such as the sequential approach/single shooting and simultaneous approach/multiple shooting (Abo-Ghander et al., 2010; Logist et al., 2012) and (ii) stochastic approaches (Bhaskar et al., 2000; Mitra et al., 2004; Patel and Padhiyar, 2016).

2. Theoretical background

This section introduces the theoretical background of the methods used to produce the results of this work. We will explain the weighted sum scalarization method for solving multi-objective optimization problems and outline its drawbacks and we will show how predictor corrector continuation algorithms work and how they can be used to overcome these drawbacks.

2.1. Weighted sum method

A traditional approach in multi-objective optimization is the weighted sum method (Marler and Arora, 2004). In this approach, the multi-objective optimization problem is reformulated, such that the objectives are combined in a weighted sum, which then gets minimized, to find a Pareto optimal

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

solution (Marler and Arora, 2004)

$$\min_x J(x) = \sum_{i=1}^{q} w_k \cdot J_k(x), \tag{2a}$$

$$\text{s.t. } h_i(x) = 0, \quad i = 1, \ldots, n \tag{2b}$$

$$g_j(x) \leq 0, \quad j = 1, \ldots, m, \tag{2c}$$

with w_k being the weight or scalarization parameter of the k-th objective.

2.2. Optimality conditions

The numerical continuation algorithm outlined in section 2.3 is able to solve algebraic systems. In order for this method to be applicable to optimization problems, we need to transform the optimization problem into an algebraic problem. To do this, we make use of optimality conditions.
The most commonly used necessary conditions for an optimum of a constrained optimization problem are called Karush-Kuhn-Tucker (KKT) conditions. Inequality constraints can be transformed into equality constraints, by introducing so called *slack variables* s_j (Boyd and Vandenberghe, 2004). That is possible, because $g_j(x) \leq 0$ only holds, iff there is a $s_j \in \Re$, such that $g_j(x)+s_j^2 = 0$. If the inequality constraints are transformed, the optimization problem has m additional unknowns.
It is worth noting that this traditional approach may result in numerical disadvantages for conventional optimization algorithms (Gill et al., 1981; Jongen and Stein, 2003). Especially for poor initial guesses, the slack variable approach may have convergence problems (Armand and Orban, 2012). In our case, this is not critical, as we assume that one point on the Pareto front or one optimal solution is known. Hence, local convergence of the algorithm is sufficient for our purposes. Furthermore, the use of slack variables can introduce instabilities and singularities, if it is not done correctly (Robinson, 1976). Therefore squared slack variables were avoided by experts in optimization theory very early (Tapia, 1980). The singularities occur on bifurcation points, where at least two solution branches intersect. In some cases common optimization algorithms will not be able to switch to the correct branch and thus produce incorrect or suboptimal solutions (Armand and Orban, 2012). One way to overcome this problem is to introduce some conditions which guarantee the non-negativity of the slack variables (Robinson, 1976; Ohtsuka, 2004).

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

On the other hand, the slack variable approach has advantages in our case, because we can directly exploit the strict complementarity to check whether or not we reached a point where one of the constraints switches from active to inactive or vice versa by simply checking the sign of the slack variables and their corresponding Lagrange multipliers. Thus we can also guarantee the non-negativity of the slack variables and their corresponding Lagrange multipliers, hence we do not have the convergence problems due to singularities. A thorough discussion concerning the bifurcations and our branch switching strategies is presented in Section 3. Furthermore the approach is computationally less expensive than other approaches, for example when using a log-barrier approach the weighting factor for the logarithmic part has to be decreased iteratively until it is near zero. Therefore we would have to implement an underlying Newton-iteration for that. Whereas we don't need to do that when using the squared slack variable approach.
For the calculation of a starting point, i.e. one point on the Pareto front, it may, however, be more advantageous to use other reformulations such as log-barriers or interior point methods.
The basic idea of the KKT conditions is to augment the objective function with a weighted sum of the constraints (Boyd and Vandenberghe, 2004), yielding the Lagrangian

$$\mathcal{L}(x, \lambda, \varphi) = J(x) + \sum_{i=1}^{m} \lambda_i \cdot g_i(x) + \sum_{i=1}^{n} \varphi_i \cdot h_i(x), \tag{3}$$

where λ_i and φ_i are called Lagrange multipliers.
The KKT conditions are defined as

$$g_i(x^*) \leq 0, \quad \text{for all } i = 1, \ldots, m \tag{4a}$$
$$h_i(x^*) = 0, \quad \text{for all } i = 1, \ldots, n \tag{4b}$$
$$\lambda_i^* \geq 0, \quad \text{for all } i = 1, \ldots, m \tag{4c}$$
$$\lambda_i^* \cdot g_i(x^*) = 0, \quad \text{for all } i = 1, \ldots, m \tag{4d}$$
$$0 = \nabla J(x^*) + \sum_{i=1}^{m} \lambda_i^* \cdot \nabla g_i(x^*) + \sum_{i=1}^{n} \varphi_i^* \cdot \nabla h_i(x^*). \tag{4e}$$

An alternative set of optimality conditions, which proves to be useful for our study, are the so called Fritz-John conditions (Poore and Tiahrt, 1987). Their difference to the KKT conditions is that the cost functional gets multiplied by

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

an additional unknown parameter ν. For $\nu = 1$, the Lagrangian is identical to the Lagrangian obtained for the KKT conditions. For $\nu = 0$ the cost functionals are not taken into account. To achieve $\nu > 0$, an additional equation

$$\nu^2 + \lambda^T \cdot \lambda - \beta_0^2 = 0, \tag{5}$$

is defined, where β_0 is a real parameter (Lundberg and Poore, 1993). For a properly chosen β_0, the Lagrange multipliers will take larger numerical values, resulting in a better scaled numerical problem.
The biggest difference between the KKT and FJ conditions is, that the KKT conditions need a set of constraint qualifications, whereas the FJ conditions do not need it (Poore and Tiahrt, 1987). They can, however, also be used when the constraint qualification holds. Thus every KKT point is also a FJ point, but not every FJ point is a KKT point.
The computational effort of the FJ conditions is only slightly higher than that of the KKT conditions (one additional equation), but it offers an additional tuning parameter β_0. Therefore, it seems worthwhile testing both sets of conditions. The case studies will show, that the FJ conditions indeed have the mentioned scaling effect on the Lagrange multipliers and improve numerical robustness in some cases.

2.3. Numerical continuation

Numerical continuation algorithms are used to approximate a solution of an underdetermined system of the form

$$F(\omega, x) = 0, \tag{6}$$

where $x \in \Re^a$ is the state vector, $\omega \in \Re$ is a free parameter and $F : \Re \times \Re^a \mapsto \Re^a$ (Keller, 1977). To solve the whole curve numerically, it is needed to generate a set of solutions $y^{(i)}$, with $i = 0, 1, 2, \ldots$, which satisfy a given tolerance $\left|F(y^{(i)})\right| \leq \varepsilon$, with $\varepsilon > 0$ (Allgower and Georg, 1990). To do this, a predictor-corrector approach is used.

Using the *tangent predictor*, a tangent to the current solution $y^{(k)}$ is calculated by solving the linear algebraic system

$$F_y\left(y^{(k)}\right) \cdot \tau^{(k)} = 0, \tag{7}$$

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

with $\tau^{(k)}$ being the tangent and $F_y\left(y^{(k)}\right)$ being the Jacobian of the solution $y^{(k)}$.
The Jacobian $F_y\left(y^{(k)}\right)$ needs to be extended by a scalar function to make it a $(a+1)\times(a+1)$ matrix, in order to make equation (7) solvable. Equation (7) can be rewritten and solved as (Seydel, 2009)

$$\begin{pmatrix} F_y\left(y^{(k)}\right) \\ e_j^{\mathsf{T}} \end{pmatrix} \cdot \tau^{(k)} = e_{a+1}, \tag{8}$$

where each element of $e_j \in \Re^{a+1}$ equals zero, except for the j-th one, which equals unity. With the tangent $\tau^{(k)}$ it is then possible to predict a new solution (Seydel, 2009)

$$\hat{y}^{(k+1)} = y^{(k)} + \Upsilon^{(k)} \cdot \tau^{(k)}. \tag{9}$$

In this notation $\Upsilon^{(k)}$ denotes the step size at point k.

One simple approach for a *step size adaption* is to rely the adaption Φ on the number of Newton iterations ξ needed in the previously calculated step (Krasnyk, 2008).
The new step size is defined as

$$\Upsilon^{(k+1)} = \Upsilon^{(k)} \cdot \Phi\left(\xi^{(k)}\right). \tag{10}$$

The *corrector step* is needed because the predicted solution $\hat{y}^{(k+1)}$ usually does not satisfy the desired tolerance $\left|F(y^{(k+i)})\right| \leq \varepsilon$.
The considered system is underdetermined and an additional auxiliary scalar equation,

$$p(\omega, x, \Gamma) = 0, \tag{11}$$

is needed, in order to make the extended system

$$\begin{pmatrix} F(\omega, x, \Gamma) \\ p(\omega, x, \Gamma) \end{pmatrix} = 0 \tag{12}$$

regular. This process is called parametrization. In the algorithm we use in this work the *local parametrization* is implemented, which parametrizes the curve along the locally most rapidly changing parameter $y_i^{(k)}$ (Krasnyk, 2008). The algorithm is illustrated in the pseudo-code Algorithm 1 (Krasnyk, 2008)

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Algorithm 1 Continuation algorithm

Input: x_0, ω_0, $\Upsilon^{(0)}$
Output: $y^{(1)}, \ldots, y^{(k_{max})}$
$y^{(0)} \leftarrow$ `initial_correction`(x_0, ω_0)
$\vec{y}^{(0)} \leftarrow$ `predictor`$(y^{(0)})$
$k \leftarrow 0$
repeat
    $\tilde{y}^{(k+1)} = y^{(k)} + \Upsilon^{(k)} \cdot \vec{y}^{(k)}$
    $y^{(k+1)} \leftarrow$ `corrector`$(\tilde{y}^{(k+1)})$
    $\vec{y}^{(k+1)} \leftarrow$ `predictor`$(y^{(k+1)})$
    $\Upsilon^{(k+1)} \leftarrow$ `stepsize`$(y^{(k+1)}, \Upsilon^{(k)})$
    `compute_test_functions`$(y^{(k+1)})$
    if *test condition occurs* **then**
        return ContiOkTestFunction
    $k \leftarrow k + 1$
    if $k \geq k_{max}$ **then**
        return ContiOkMaxStepsMade
until *boundary achieved*
return ContiOkAchievedBoundary

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

3. Scalarization of bi-criterial optimization problems

The optimality conditions from section 2.2 lead to sets of algebraic equations, which can be solved using numerical continuation methods introduced in section 2.3.

Our approach to bi-criterial optimization problems is a direct application of the weighted sum method. We scalarize the bi-objective optimization problem and use the weighting factor w as continuation parameter. Doing this has some advantages over the normal weighted sum method, but there also arise some difficulties which have to be addressed.
Subsection 3.1 will explain the general problems occurring when applying continuation algorithms to constrained optimization problems due to bifurcations. The following subsection will give insight into the problems arising from non-convex Pareto fronts. Finally, subsection 3.3 will suggest how to solve these problems with branch switching algorithms.
Note that there may also occur disconnected Pareto sets. Our approach would be able to compute disconnected Pareto sets part-wise, if it is possible to generate a starting point on each subset. It is, however, not meant to do that automatically and is currently not able to detect them. There are, however, other algorithms which are able to cope with these problems, i.e. methods based on set-inversion (Kubica and Wozniak, 2007, 2012) and evolutionary methods (Miettinen, 1999; Deb, 2002).

3.1. Scalarization of constrained problems

The approach presented in this work is applicable to optimization problems with an arbitrary number of equality and inequality constraints. However, for better readability, we restrict the discussion in this section to a simplified optimization problem with a single inequality constraint

$$\begin{aligned} &\min_x J(x,w) = (1-w)\cdot J_1(x) + w \cdot J_2(x), \\ &\text{s.t. } g(x) \leq 0. \end{aligned} \tag{13}$$

We do not include equality constraints, because they are always active and therefore do not pose any problems for our method. After applying the KKT conditions and replacing the inequality constraint with an equality constraint by introducing a slack variable s we get the following set of equations for the

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

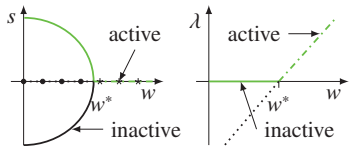

Figure 1: Solution branches for s and λ. The solid line shows the valid solution branch and the dotted line, marked with circles in the first graph, shows the invalid solution branch for $w < w^*$, the dashed line, marked with stars in the first graph, shows the valid solution branch for $w > w^*$. The valid solution path is shown in green.

Lagrangian and it's partial derivatives with respect to all states

$$\mathcal{L} = J(x, w) + \lambda \cdot (g(x) + s^2), \tag{14a}$$

$$\frac{\partial \mathcal{L}}{\partial x} = \frac{\partial J}{\partial x} + \lambda \cdot \frac{\partial g}{\partial x} = 0, \tag{14b}$$

$$\frac{\partial \mathcal{L}}{\partial s} = 2 \cdot \lambda \cdot s = 0, \tag{14c}$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = g(x) + s^2 = 0. \tag{14d}$$

There are two cases we have to take into account. Under the assumption of strict complementary slackness from equation (14c) follows that $s = 0$, $\lambda > 0$ for active inequality constraints and $\lambda = 0$, $s \neq 0$ for inactive inequality constraints.
Now, assume without loss of generality that

$$\begin{aligned} &\lambda = 0, \text{ for } w = w^*, \\ &\lambda \leq 0, \text{ for } w < w^*, \\ &\lambda > 0, \text{ for } w > w^*. \end{aligned} \tag{15}$$

The examination of the sign of λ yields, that, though there exist solutions for all w, the only relevant solutions for $s = 0$ are the solutions where $\lambda > 0$ and, hence, $w > w^*$ (active inequality constraints).
For inactive inequality constraints ($w \leq w^*$) we get

$$s^2 = -g(x). \tag{16}$$

Because the constraint is inactive and $g(x) < 0$ holds, there exist two real solutions, namely

$$s = \pm\sqrt{-g(x)}. \tag{17}$$

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Figure 1 illustrates the feasible solutions of the optimization problem.
In summary the inequality constraints require a switching strategy between solution branches at points where an inequality constraint becomes active or inactive.
A more thorough discussion on which bifurcations may occur can be found in (Guddat et al., 1990).

3.2. Treatment of non-convex Pareto fronts

For an optimization problem with a convex Pareto front there exists only one feasible solution for a given value of w, but if we deal with optimization problems with non-convex Pareto fronts there is the possibility that more than one solution point on the Pareto front belong to a given value of w.
Figure 2 depicts an example of a non-convex Pareto front. The weighted sum can be interpreted as a line Ψ through the solution space, with w defining the slope ζ of this line. If the Pareto front is non-convex, it has inflection points and thus more than one point with a given slope. These inflection points on the Pareto front lead to turning points in the curves of the objective functions when plotted against w. Predictor corrector continuation algorithms, which have their origin in numerical bifurcation analysis, were designed to continue solution branches around turning points and, hence, can be tailored to the computation of non-convex Pareto fronts. If equipped with a proper step size control, they avoid the problem of clustering. Further, many implementations of continuation algorithms are available that are applicable to large-scale algebraic equation systems.
It is possible, that a turning point and a switching point of an inequality constraint coincide. The bifurcations occurring in such cases are shown in Figure 3. All the feasible solution branches in such a case lie in the region $w < w^*$.

3.3. Branch switching strategy

Like shown in section 3.1 we have to take two scenarios into account, for which a branch has to be switched. These scenarios are switching the branch $s = 0$, $\lambda \neq 0$ to the branch $s \neq 0$, $\lambda = 0$ and vice versa, which correspond to an inequality constraint switching from active to inactive and from inactive to active, respectively. For each of the two cases we implemented a different branch switching approach.
Firstly, we will describe the approach for inequality constraints switching from active to inactive. We detect the critical point by monitoring the sign

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

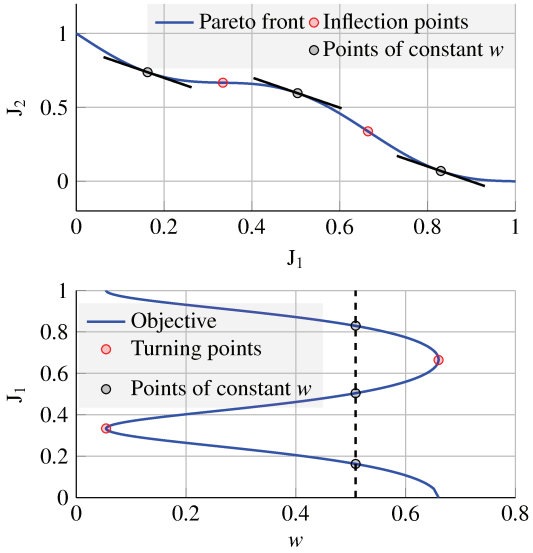

Figure 2: Example of a non-convex Pareto front with one corresponding objective profile and illustrated turning and inflection points.

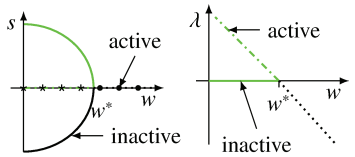

Figure 3: Solution branches for s and λ for a non-convex Pareto front at turning points. The solid line and the dashed line, marked with stars in the first graph, show the valid solution branches for $w < w^*$. The dotted line, marked with circles in the first graph, shows the invalid solution branch for $w > w^*$. The valid solution path is shown in green.

of λ. If a change in the sign is detected, we set $w = w^* + \varepsilon_1$, $x = x^*$, $\lambda = 0$ and $s = \varepsilon_2$ as a guess for the starting point on the new branch and continue with a corrector step. The parameters $\varepsilon_{1/2}$ are used to push the new starting point far enough into the new branch, so that the numerical solver does not return to the old branch.
When an inequality constraint switches from inactive to active we make use of an approach which was inspired by the arc-length continuation introduced by Chan and Keller (1982). We have to calculate the tangent to the new solution branch from

$$F_{\tilde{x}} \cdot \frac{\partial \tilde{x}}{\partial \varkappa} + F_w \cdot \frac{\partial w}{\partial \varkappa} = 0, \tag{18}$$

where $F = \left(\frac{\partial \mathcal{L}}{\partial x}, \frac{\partial \mathcal{L}}{\partial s}, \frac{\partial \mathcal{L}}{\partial \lambda} \right)$, $\tilde{x} = (x, s, \lambda)$ and $\varkappa$ is the arc-length which specifies how far we move on the curve. To generate a new starting point, we use the correlation

$$\Delta \tilde{x} = -F_{\tilde{x}}^{-1} \cdot F_w \cdot \Delta w, \tag{19}$$

which is obtained from equation (18). This correlation is used to calculate a predicted solution on the new solution branch [$w = w^* + \Delta w$, $x = x^* + \Delta x$, $s = 0$, $\lambda = \Delta\lambda$]. The precise new solution on the new branch is calculated by Newton's method, using the predicted solution as starting point.
These switching techniques are augmented by an algorithm which checks the solutions for possible turning points due to non-convexity of the Pareto front. To check whether or not we

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Table 1: Parameter values of constrained PFR-equations.

Parameter	Meaning	Value
c_f	Feed concentration	$0.02\,\frac{\text{mol}}{\text{l}}$
K	Scaling factor	250 000
L	Length of reactor	1 m
N	Number of grid points	50
T_f	Feed temperature	340 K
T_{max}	Maximum reactor temperature	400 K
T_{min}	Minimum reactor temperature	280 K
$T_{w,max}$	Maximum jacket temperature	400 K
$T_{w,min}$	Minimum jacket temperature	280 K
v	Flow velocity	$0.1\,\frac{\text{m}}{\text{s}}$
α	Reaction kinetic constant	$0.0582\,\frac{\text{l}}{\text{s}}$
β	Reaction kinetic constant	$0.2\,\frac{\text{l}}{\text{s}}$
γ	Reaction kinetic constant	16.659
δ	Reaction kinetic constant	0.25

accidentally passed a turning point we make use of the duality of the Lagrange multipliers and slack variables. A flowsheet of the algorithm is depicted in Figure 4. If we detect a change in the sign of λ or s we save the current states before switching the branch. Then we check if the corresponding other, dual, costate (in the case of a zero passing λ we check s and in the case of a zero passing s we check λ) holds a negative value after switching the branch. If so, we load the states and do a branch switching in the other direction. If not, we proceed with the continuation. That way we are able to overcome the problem of inflection points on non-convex Pareto fronts and are therefore not only able to compute a convex hull, but the whole Pareto front. The full algorithm is shown in the pseudo-code Algorithm 2. Whether ε_1 is added or subtracted from w depends on the continuation direction, therefore it is denoted as $\pm\varepsilon_1$.

4. Case studies

Two case studies illustrate the feasibility of our approach. The first case study considers a Plug-Flow-Reactor with a convex Pareto front, the second case study considers a Fed-Batch bioreactor with a non-convex Pareto front. Both problems are taken from Logist et al. (2009). They involve a dynamic

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

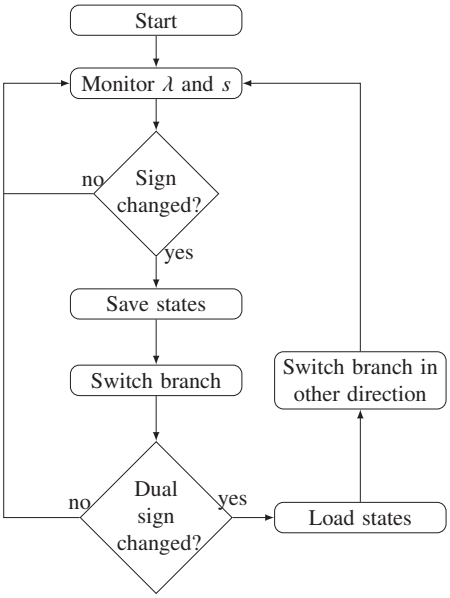

Figure 4: Branch switching algorithm, including detection of inflection points

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

Algorithm 2 Branch switching

Input: $w^{(0)}$, $x^{(0)}$, $\lambda_j^{(0)}$, $s_j^{(0)}$, `initial_direction`
repeat
  call CONTINUATION_ALGORITHM
  for all j **do**
    if $\lambda_j^{(k)} < 0$ **then**
      save $w^{(k)}, x^{(k)}, \lambda^{(k)}_{(1,\dots,n)}, s^{(k)}_{(1,\dots,n)}$
      set $\lambda_j^{(k)} = 0$, $w^{(k)} = w^{(k)} \pm \varepsilon_1$, $s_j^{(k)} = \varepsilon_2$
      call NEWTON_ITERATION
      if $s_j^{(k)} < 0$ **then**
        `inflection_detected`
        load $w^{(k)}, x^{(k)}, \lambda^{(k)}_{(1,\dots,n)}, s^{(k)}_{(1,\dots,n)}$
        set `direction` $=$ `direction` $\cdot\,(-1)$
        set $\lambda_j^{(k)} = 0$, $w^{(k)} = w^{(k)} \pm \varepsilon_1$, $s_j^{(k)} = \varepsilon_2$
      break
    if $s_j^{(k)} < 0$ **then**
      save $w^{(k)}, x^{(k)}, \lambda^{(k)}_{(1,\dots,n)}, s^{(k)}_{(1,\dots,n)}$
      predict `solution_on_new_branch`
      call NEWTON_ITERATION
      if $\lambda_j^{(k)} < 0$ **then**
        `inflection_detected`
        load $w^{(k)}, x^{(k)}, \lambda^{(k)}_{(1,\dots,n)}, s^{(k)}_{(1,\dots,n)}$
        set `direction` $=$ `direction` $\cdot\,(-1)$
        predict `solution_on_new_branch`
        call NEWTON_ITERATION
      break
until *boundary achieved*

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

optimization problem with differential equations as constraints and optimal control profiles to be determined. In our work, we use the ProMoT and Diana simulation environment (Krasnyk et al., 2007) to solve the problems numerically.

4.1. Tubular reactor

The model equations of the Plug-Flow-Reactor (PFR) are defined as

$$\frac{\partial x_1}{\partial t} = -\frac{\partial x_1}{\partial z} + \frac{\alpha}{v} \cdot (1 - x_1) \cdot e^{\gamma \cdot \frac{x_2}{1 + x_2}}, \tag{20a}$$

$$\frac{\partial x_2}{\partial t} = -\frac{\partial x_2}{\partial z} + \frac{\alpha \cdot \delta}{v} \cdot (1 - x_1) \cdot e^{\gamma \cdot \frac{x_2}{1 + x_2}} + \frac{\beta}{v} \cdot \left(\frac{T_w - T_f}{T_f} - x_2 \right), \tag{20b}$$

with the boundary conditions

$$x_1(0) = 0, \tag{21a}$$

$$x_2(0) = 0. \tag{21b}$$

The states of the system are the dimensionless reactant concentration, x_1, and the dimensionless reactor temperature, x_2, which are defined as

$$x_1 = \frac{c_f - c}{c_f}, \tag{22a}$$

$$x_2 = \frac{T - T_f}{T_f}. \tag{22b}$$

There is only one input to control the system, which is the spatially distributed jacket temperature T_w, with the spatial coordinate z defined in $[0, L]$. The parameters used in equations (20) are given in Table 1.

4.1.1. Scalarization

In order to make the problem solvable by a numerical continuation method, it gets discretized using the Finite Volume Method with $N = 50$ grid points. We only consider the steady-state of the system, therefore the right hand

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

side of the resulting 100 equations equals zero

$$0 = \frac{x_{1,i-1} - x_{1,i}}{\Delta z} + \frac{\alpha}{v} \cdot (1 - x_{1,i}) \cdot e^{\gamma \cdot \frac{x_{2,i}}{1 + x_{2,i}}}, \text{ for } i = 1, \ldots, N, \tag{23a}$$

$$0 = \frac{x_{2,i-1} - x_{2,i}}{\Delta z} + \frac{\alpha \cdot \delta}{v} \cdot (1 - x_{1,i}) \cdot e^{\gamma \cdot \frac{x_{2,i}}{1 + x_{2,i}}} + \frac{\beta}{v} \cdot \left(\frac{T_{w,i} - T_f}{T_f} - x_{2,i} \right), \text{ for } i = 1, \ldots, N, \tag{23b}$$

where the lower index i stands for the i-th grid point z_i. These grid points are equally distant spaced by Δz in the definition interval of z. The first grid point, $i = 0$, is the inlet, where the boundary conditions have to be fulfilled, resulting in $x_{1,0} = 0$ and $x_{2,0} = 0$.
The corresponding optimization problem is defined as

$$\begin{aligned} \min_{T_{w,i}} J &= (1 - w) \cdot J_1 + w \cdot J_2, \\ \text{s.t. } & \text{discretized system equations (23)}, \\ & x_{2,min} \leq x_{2,i} \leq x_{2,max}, \\ & T_{w,min} \leq T_{w,i} \leq T_{w,max}, \end{aligned} \tag{24}$$

with the objective functions J_1 and J_2 defined as

$$J_1 = c_f \cdot (1 - x_1(L)) = c_f \cdot (1 - x_{1,N}), \tag{25a}$$

$$J_2 = \frac{T_f^2 \cdot x_2^2(L)}{K} = \frac{T_f^2 \cdot x_{2,N}^2}{K}. \tag{25b}$$

The first objective is to minimize the reactant concentration at the outlet of the reactor, i.e. maximize the conversion, the second objective is to hold the reactor temperature at the outlet as close to the inlet temperature as possible, i.e. minimize the heat loss. To transform the constrained optimization problem into an algebraic problem solvable by the continuation algorithm, we apply the Karush-Kuhn-Tucker conditions. In the equation for the gradient of the Lagrangian

$$\begin{aligned} 0 = & -(1 - w) \cdot c_f \cdot \frac{\partial x_{1,N}}{\partial T_{w,j}} + 2 \cdot w \cdot \frac{T_f^2}{K} \cdot x_{2,N} \cdot \frac{\partial x_{2,N}}{\partial T_{w,j}} \\ & - \sum_{i=1}^{N} \mu_{1,i} \cdot \frac{\partial x_{1,i}}{\partial T_{w,j}} + \sum_{i=1}^{N} \mu_{2,i} \cdot \frac{\partial x_{2,i}}{\partial T_{w,j}} - \mu_{3,j} + \mu_{4,j}, \end{aligned} \tag{26}$$

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

there occur the unknown sensitivities $\partial x_{1,i}/\partial T_{w,j}$ and $\partial x_{2,i}/\partial T_{w,j}$, for which further equations have to be added to the equation set. These additional equations are calculated by introducing the derivative

$$\frac{\partial F}{\partial T_{w,j}} = \frac{\partial F}{\partial x_i} \cdot \frac{\partial x_i}{\partial T_{w,j}}, \tag{27a}$$

$$= \begin{bmatrix} \frac{\partial f_1}{\partial x_{1,i}} \cdot \frac{\partial x_{1,i}}{\partial T_{w,j}} + \frac{\partial f_1}{\partial x_{2,i}} \cdot \frac{\partial x_{2,i}}{\partial T_{w,j}} \\ \frac{\partial f_2}{\partial x_{1,i}} \cdot \frac{\partial x_{1,i}}{\partial T_{w,j}} + \frac{\partial f_2}{\partial x_{2,i}} \cdot \frac{\partial x_{2,i}}{\partial T_{w,j}} \end{bmatrix}, \text{ for } i,j = 1,\ldots,N. \tag{27b}$$

To generate a starting point for the continuation we solve equations (23) semi analytically for $w = 0$. For $w = 0$ the only objective is to minimize the reactant concentration at the outlet. To convert the most of the reactant, the reactor has to be held at the highest temperature possible. This is achieved by setting T_w to $T_{w,max}$ in the inlet region of the reactor, i.e. in a region $z < z^*$, where z^* is the point, where the reactor temperature reaches T_{max}. Hence, for $z < z^*$, equation (23) has to be solved numerically for $x_{1,i}$ and $x_{2,i}$ setting $T_{w,i} = T_{w,max}$. for $z > z^*$, equation (23) is solved analytically for $x_{1,i}$ and $T_{w,i}$, setting $x_{2,i} = x_{2,max}$.

4.1.2. Results

The results generated by using the continuation approach are shown in Figure 5. The branch switching algorithm had to trigger 36 times. The two upper graphs show the states with respect to z for different values of w. The third figure shows the jacket temperature profiles and the last figure shows the resulting Pareto front. For each value of w, the reactor gets heated up as fast as possible until it reaches T_{max}, to obtain a high conversion. From that point on the reactor has to be cooled, to compensate the heat of reaction, preventing the temperature to exceed T_{max}. Towards the right end of the reactor the cooling temperature has to be increased stepwise, because the more reactant is converted, the less energy is set free by the reaction. For $w = 0$ only J_1 is active, therefore the reactor temperature is held constant at T_{max} until the end of the reactor, to maximize the conversion. For $w = 1$ only J_2 is active, therefore the reactor has to be cooled down from a specific point on, to reach the inlet temperature at the output and to minimize the heat loss. For $w \in (0, 1)$ there are combinations of the behaviors of the extremes, two of them are depicted.

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

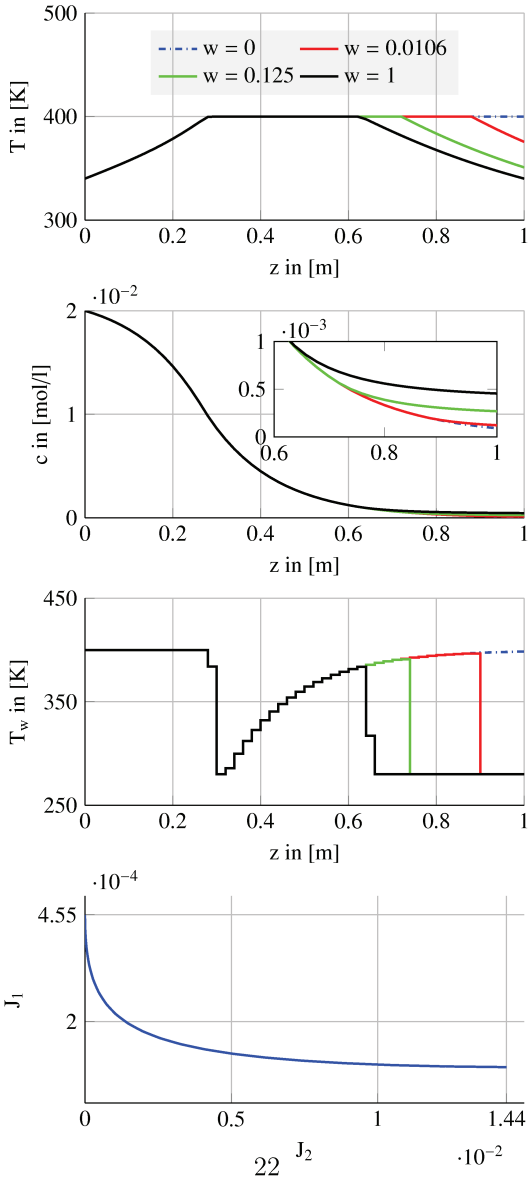

Figure 5: Results of the continuation: state profiles of the reactor temperature T, concentration c, together with a closeup of the end part of the reactor, and jacket temperature T_w. Each for various values of w. The Pareto front is shown at the bottom, the corresponding values for the weighting factor are on the right $w = 0$ and on the left $w = 1$.

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

The problem is rather complex, with 200 inequality constraints, 100 equality constraints and 50 optimization variables, and close to a real application of the scalarization. The drawback of the weighted sum method, that it may not yield uniformly distributed results on the Pareto front, does not affect the accuracy of the Pareto front. This can be seen when comparing our results with the results obtained by Logist et al. (2009). The numerical results we obtained agree well with those from Logist et al. (2009), using the more complex normalized normal constraint and normal boundary intersection methods. The algorithm calculated a total of 232 points on the Pareto front.

4.2. Biochemical fed-batch reactor

The second example illustrates the application of the approach to a problem with a non-convex Pareto front.
The model of a bioreactor is given by the following equations

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = \mu \cdot x_1, \tag{28a}$$
$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = -\sigma \cdot x_1 + u \cdot c_{s,F}, \tag{28b}$$
$$\frac{\mathrm{d}x_3}{\mathrm{d}t} = \pi \cdot x_1, \tag{28c}$$
$$\frac{\mathrm{d}x_4}{\mathrm{d}t} = u, \tag{28d}$$

with the initial conditions

$$x_1(0) = 0.1\,\mathrm{g}, \tag{29a}$$
$$x_2(0) = 14\,\mathrm{g}, \tag{29b}$$
$$x_3(0) = 0\,\mathrm{g}, \tag{29c}$$
$$x_4(0) = 5\,\mathrm{l}. \tag{29d}$$

In this model x_1 is the biomass in g, x_2 is the substrate in g, x_3 is the product (lysine) in g and x_4 is the fermenter volume in l. The system input u is the volumetric rate of the feed stream, which contains a feed substrate concentration $c_{s,F}$ of 2.8 g/l. The other parameters are the rates for growth μ in 1/h, the rate for substrate consumption σ in g/gh and the rate for

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

production π in g/gh. They are defined as

$$\mu = 0.125 \cdot c_s, \tag{30a}$$
$$\sigma = \mu / 0.135, \tag{30b}$$
$$\pi = -384 \cdot \mu^2 + 134 \cdot \mu, \tag{30c}$$

where c_s is the substrate concentration x_2/x_4 in g/l.

4.2.1. Scalarization

Similar to the previous case study, we have to discretize the problem. We use $N = 50$ grid points in time and obtain a system consisting of 200 equations. Taking the different rates into account, we derive the following equation system

$$\frac{x_{1,i} - x_{1,i-1}}{\Delta t} = 0.125 \cdot \frac{x_{2,i}}{x_{4,i}} \cdot x_{1,i}, \tag{31a}$$
$$\frac{x_{2,i} - x_{2,i-1}}{\Delta t} = -\frac{25}{27} \cdot \frac{x_{2,i}}{x_{4,i}} \cdot x_{1,i} + u_i \cdot c_{s,F}, \tag{31b}$$
$$\frac{x_{3,i} - x_{3,i-1}}{\Delta t} = \left(-6 \cdot \left(\frac{x_{2,i}}{x_{4,i}} \right)^2 + 16.75 \cdot \left(\frac{x_{2,i}}{x_{4,i}} \right) \right) \cdot x_{1,i}, \tag{31c}$$
$$\frac{x_{4,i} - x_{4,i-1}}{\Delta t} = u_i, \text{ for } i = 1, \ldots, N. \tag{31d}$$

The first grid point $i = 0$ is the starting point, at which the initial conditions have to be fulfilled. The grid points are equally spaced along the time domain, but the interval between grid points is changed during the optimization process, because the terminal time, t_e, is one of the optimization variables. The optimization problem is defined as

$$\begin{aligned} \min_{u_i, t_e} \; & J = (1 - w) \cdot J_1 + w \cdot J_2, \\ \text{s.t.} \; & \text{discretized system equations (31)}, \\ & 5\,\text{l} \leq x_{4,i} \leq 20\,\text{l}, \\ & 0\,\text{l/h} \leq u_i \leq 2\,\text{l/h}, \\ & 20\,\text{h} \leq t_e \leq 40\,\text{h}, \\ & 20\,\text{g} \leq (x_{4,N} - x_{4,0}) \cdot c_{s,F}, \end{aligned} \tag{32}$$

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

with the objective functions J_1 and J_2 given by

$$J_1 = -\frac{x_{3,N}}{t_e}, \tag{33a}$$

$$J_2 = -\frac{x_{3,N}}{(x_{4,N} - x_{4,0}) \cdot c_{s,F}}. \tag{33b}$$

The first objective is to minimize the negative ratio between the product at the terminal time and the process duration, e.g. maximize the productivity, and the second objective is to maximize the yield.
Like in the previous case study we apply optimality conditions to transform the optimization problem into an algebraic problem. The sensitivities for the control, $\mathrm{d}x_i/\mathrm{d}u_j$, can be calculated like the sensitivities for the jacket temperature in the previous example, found in equation (27). The sensitivity for the terminal time can be calculated with

$$\frac{\mathrm{d}x_{i,j}}{\mathrm{d}t_e} = \frac{\mathrm{d}x_i}{\mathrm{d}t_j} \cdot \frac{\mathrm{d}t_j}{\mathrm{d}t_e}, \text{ with:} \tag{34a}$$

$$\frac{\mathrm{d}x_i}{\mathrm{d}t_j} = \frac{x_{i,j} - x_{i,j-1}}{\Delta t}, \tag{34b}$$

$$t_j = \frac{t_e}{N} \cdot j, \tag{34c}$$

$$\frac{\mathrm{d}t_j}{\mathrm{d}t_e} = \frac{j}{N}, \text{ for } i = 1, \ldots, 4 \text{ and } j = 1, \ldots, N. \tag{34d}$$

In this case study we implemented the Fritz-John conditions, as well as the KKT conditions, to compare the computation times.

4.2.2. Results

The results are shown in Figures 6 and 7. The branch switching algorithm had to trigger 42 and detected two inflection points. The first graph in Figure 6 shows the optimal control sequences, the four graphs below show the according behaviors of the states. The state profiles are numbered in the order of their appearance during the continuation process. Due to the non-convex nature of this optimization problem, there occur ambiguities in the weighting factor w, which can be seen clearly in Figure 7, in which the Pareto front and the objective profiles are depicted and where the marked red points belong to the correspondingly numbered state profiles from Figure 6. Our starting point for the continuation is $w = 1$, where the optimal input profile consists of three parts. At the beginning the input equals zero, then

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

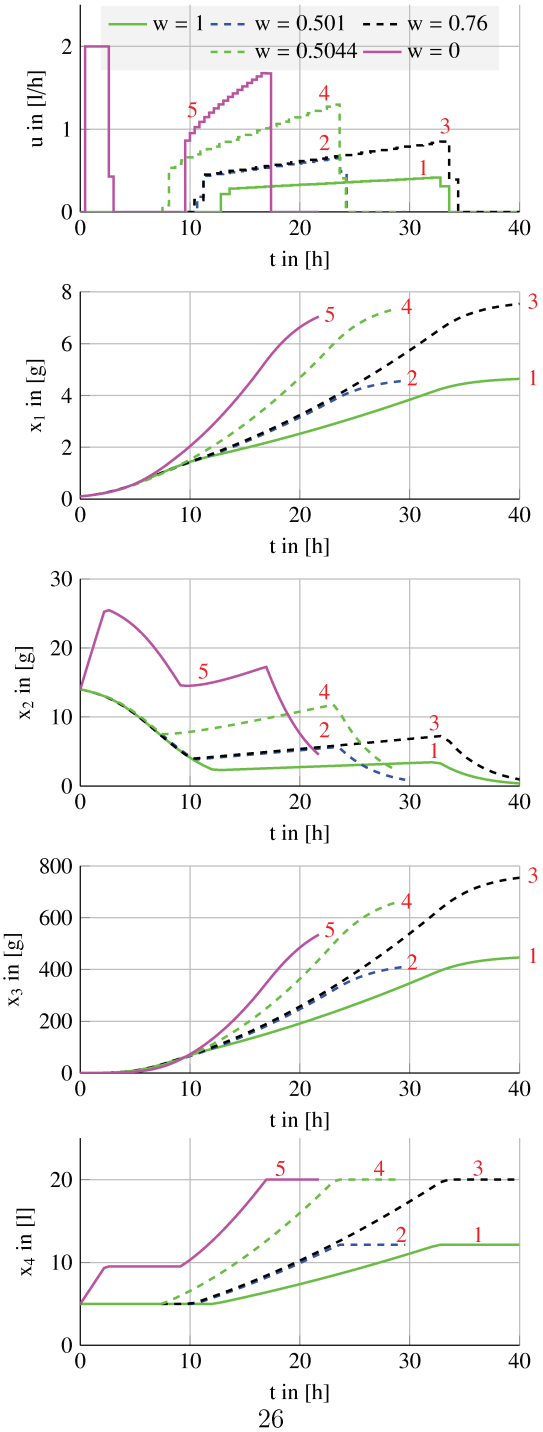

Figure 6: Results of the continuation: state profiles of the biomass x_1, substrate x_2, product (lysine) x_3 and fermenter volume x_4. Each for various values of w, with the corresponding input, u, profiles. The state profiles are numbered accordingly to their appearance.

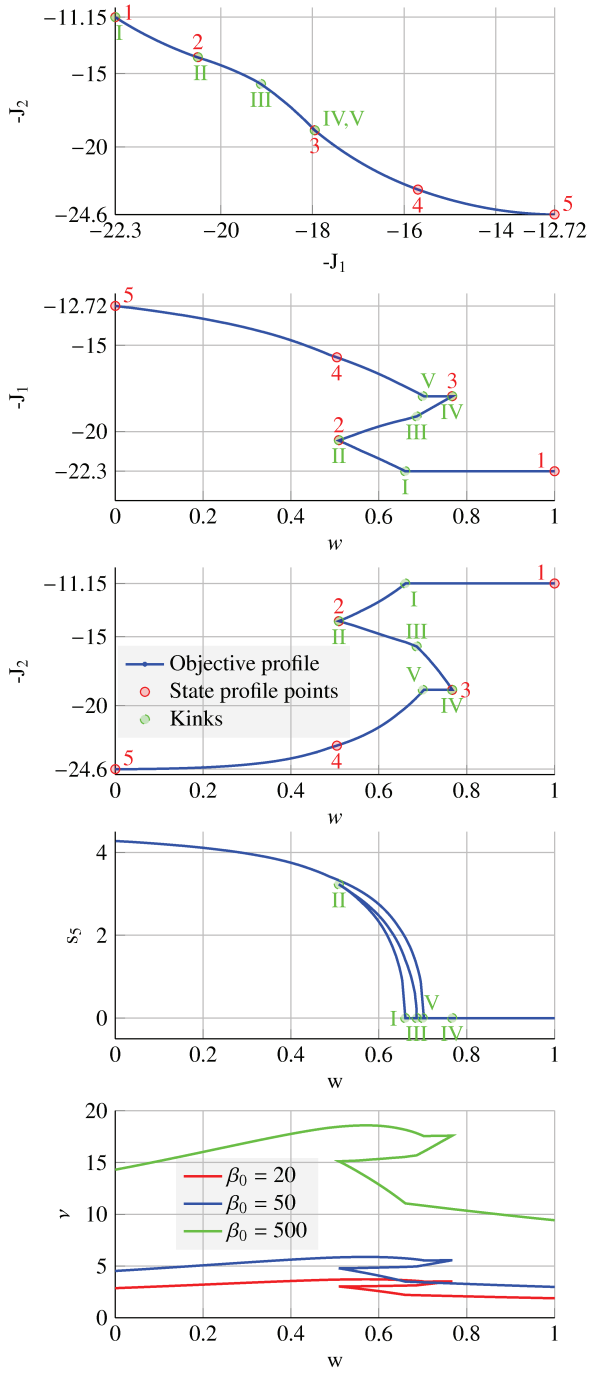

Figure 7: Non-convex Pareto front of the optimization problem in the upper graph. The corresponding values for the weighting factor are on the right $w = 0$ and on the left $w = 1$. The two graphs below show the objective profiles. The red latin numbers belong to the state profiles from Figure 6 and the roman numbers belong to the kinks. The fourth graph shows the slack variable belonging to the terminal time constraint and the last graph shows the profiles of the Fritz-John state.

Postprint version of paper published in Computers and Chemical Engineering 2017, vol. 98, p. 89–99.
The content is identical to the published paper, but without the final typesetting by the publisher.

it has a trapezoidal shape and then it equals zero again until the upper time limit of 40 h is reached. We start the continuation in a decreasing direction. On this solution branch the terminal time decreases and the trapezoid gets steeper. The first inflection point is reached at $w = 0.501$. From this point on the terminal time increases again and the steepness of the input trapezoid stays constant. The next inflection point is reached at $w = 0.76$, where the terminal time has again reached its upper limit of 40 h. From here on the terminal time decreases again and the input trapezoid gets steeper. At around $w = 0.48$ the behavior changes and a small input occurs around $t = 0$ h. This small input evolves gradually towards the end of the continuation and the input takes the maximum value for a short amount of time, until it returns to its minimum-trapezoid-minimum shape. This maximum stimulates the biomass growth and thus increases the productivity, which is advantageous because at $w = 0$ the productivity is the only objective. Note that this additional max arc has been automatically added to the sequence by the algorithm.

The objective profiles in Figure 7 have several kinks. These kinks are a result of the constraints switching from active to inactive. The slack variable belonging to the constraint of the terminal time t_e is shown in Figure 7 as an illustration. As a short reminder, the constraint is defined as: $t_e - t_{e,max} + s_5^2 = 0$. From the red 1 to the green I the terminal time t_e takes its maximum value $t_{e,max} = 40$ h and both objectives keep their values until the green I is reached. This behavior can also be seen for the classical weighted sum method, resulting in a cluster in this region. After the green I is passed, the terminal time begins decreasing. The first turning point and the second kink coincide. Until that point is crossed the substrate added to the process lies at its minimum value. The third kink is a result of the terminal time reaching its maximum again. At the second turning point and fourth kink the reactor volume reaches its maximum and from the fifth kink on the terminal time starts decreasing again.

The non-convex character of the optimization problem can be seen very well. Despite the fact that the weighted sum approach is not able to compute non-convex Pareto fronts, our approach is able to compute the Pareto front accurately with a total of 485 points on the Pareto front. Moreover, using our approach it is possible to get the exact position of the non-convex part of the Pareto front.

As stated above, we used this case study to compare the results and computational expenses of the KKT and Fritz-John conditions. The results for both

Journal homepage: https://www.journals.elsevier.com/computers-and-chemical-engineering
Original file available at: http://www.sciencedirect.com/science/article/pii/S0098135416303738?np=y

optimality conditions are the same. We evaluated the Fritz-John conditions for various values of β_0, the profiles of the Fritz-John state ν can be found in Figure 7. It can be seen, that the size of ν depends on β_0. Because the cost function values get magnified by ν, the costates also have to take larger values. Therefore it becomes easier for the continuation algorithm to find the branch switching points, requiring less computation steps. By using the Fritz-John conditions we could decrease the computation time for this case study by around 11%, from 7 minutes 46 seconds to 6 minutes 54 seconds.

5. Conclusions

In our work we investigated as a proof of concept the feasibility of using predictor corrector continuation methods from bifurcation analysis for solving multi-objective optimal control problems. These methods can be used, if the optimization problem gets transformed into an algebraic problem by applying optimality conditions. Our approach is a direct application of the weighted sum method, which has some major drawbacks. By using the numerical continuation, however, we are able to overcome the heavy clustering of solutions and it is possible to calculate non-convex Pareto fronts.
We demonstrate with two case studies, that the approach is able to calculate Pareto fronts accurately and efficiently, if the algorithm takes into account that there occur ambiguities, and thus inflection points, due to non-convexities and bifurcations due to constraints. The results agree well with those obtained by the normalized normal constraint and normal boundary intersection methods presented by Logist et al. (2009), which are much more complex. Further, it adaptively changes the step size and thus generates a detailed Pareto front with many more points than the more commonly used approaches would generate.
Even for complex and large-scale applications we are able to calculate accurate solutions within a reasonable amount of time. We found, that this amount of time can be lowered even more by using the Fritz-John optimality conditions.
The presented algorithm is applicable to bi-criterial optimization problems. An extension to multicriterial problems may be possible by using multi-parameter continuation algorithms like Multifario (Henderson, 2002).

Abo-Ghander, N. S., Logist, F., Grace, J. R., Van Impe, J., Elnashaie, S., Lim, C. J., 2010. Optimal design of an autothermal membrane reactor cou-

pling the dehydrogenation of ethylbenzene to styrene with the hydrogenation of nitrobenzene to aniline. Chemical Engineering Science 65, 3113–3127.

Allgower, E. L., Georg, K., 1990. Numerical continuation methods : an introduction. Springer.

Armand, P., Orban, D., 2012. The squared slack transformation in nonlinear programming. SQU Journal for Science 17 (1), 22–29.

Bhaskar, V., Gupta, S., Ray, A., 2000. Applications of multi-objective optimization in chemical engineering. Reviews in Chemical Engineering 16, 1–54.

Bortz, M., Burger, J., Asprion, N., Blagov, S., Böttcher, R., Nowak, U., Scheithauer, A., Welke, R., Küfer, K.-H., Hasse, H., 2014. Multi-criteria optimization in chemical process design and decision support by navigation of pareto sets. Computers & Chemical Engineering 60, 354–363.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.

Chan, T. F. C., Keller, H. B., 1982. Arc-length continuation and multi-grid techniques for nonlinear elliptic eigenvalue problems. SIAM Journal on Scientific and Statistical Computing 3, 173–194.

Collette, Y., Siarry, P., 2003. Multiobjective Optimization: Principles and Case Studies. Springer.

Deb, K., 2002. Multi-objective optimization using evolutionary algorithms. Chichester, Wiley.

Deb, K., 2014. Search Methodologies. Springer.

Gill, P. E., Murray, W., Wright, M. H., 1981. Practical Optimization. Academic Press Limited.

Gudat, J., Vazquez, F., Nowak, D., Rückmann, J., 2007. Pathfollowing methods for nonlinear multiobjective optimization problems. International Journal of Management Science and Engineering Management 2, 163–177.

Guddat, J., Guerra Vazquez, F., Jongen, H. T., 1990. Parametric Optimization: Singularities, Pathfollowing and Jumps. John Wiley & Sons, Chichester.

Harada, K., Sakuma, J., Kobayashi, S., Ono, I., 2007. Uniform sampling of local pareto-optimal solution curves by pareto path following and its applications in multi-objective ga. In: Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation. GECCO '07. ACM, New York, NY, USA, pp. 813–820.

Henderson, M., 2002. Multiple parameter continuation: Computing imlicitly defined k-manifolds. International Journal of Bifurcation and Chaos 12, 451–476.

Hillermeier, C., 2001. Generalized homotopy approach to multiobjective optimization. Journal of Optimization Theory and Applications 110, 557 – 583.

Jongen, H. T., Stein, O., 2003. On the complexity of equalizing inequalities. Journal of Global Optimization 27, 367–374.

Keller, H. B., 1977. Numerical solution of bifurcation and nonlinear eigenvalue problems. Applications of Bifurcation Theory, 359–384.

Keßler, T., Logist, F., Mangold, M., 2016. Use of predictor corrector methods for multi-objective optimization of dynamic systems. In: Proceedings of the 26th European Symposium on Computer Aided Process Engineering.

Krasnyk, M., 2008. DIANA - An object-oriented tool for nonlinear analysis of chemical processes. Shaker Verlag.

Krasnyk, M., Ginkel, M., Mangold, M., Kienle, A., 2007. Numerical analysis of higher order singularities in chemical process models. Computers & Chemical Engineering 31, 1100–1110.

Kubica, B. J., Wozniak, A., 2007. Interval methods for computing the pareto-front of a mmulticriteria problem. Parallel Processing and Applied Mathematics, 1382–1391.

Kubica, B. J., Wozniak, A., 2012. Tuning the interval algorithm for seeking pareto sets of multi-criteria problems. Parallel Processing and Applied Mathematics, 504–517.

Logist, F., Vallerio, M., Houska, B., Diehl, M., Van Impe, J., 2012. Multi-objective optimal control of chemical processes using acado toolkit. Computers & Chemical Engineering 37, 191–199.

Logist, F., Van Erdeghem, P., Van Impe, J., 2009. Efficient deterministic multiple objective optimal control of (bio)chemical processes. Chemical Engineering Science 64, 2527–2538.

Lundberg, B., Poore, A., 1993. Numerical continuation and singularity detection methods for parametric nonlinear programming. SIAM Journal on Optimization 3, 134–154.

Marler, R. T., Arora, J. S., 2004. Survey of multi-objective optimization methods for engineering. Structural and Multidisciplinary Optimization 26, 369–395.

Martin, B., Goldsztejn, A., Granvilliers, L., Jermann, C., 2016. On continuation methods for non-linear bi-objective optimization: towards a certified interval-based approach. Journal of Global Optimization 64, 3–16.

Miettinen, K., 1999. Nonlinear Multiobjective Optimization. Kluwer Academic Publishers.

Mitra, K., Majumdar, S., Raha, S., 2004. Multiobjective dynamic optimization of a semi-batch epoxy polymerization process. Computers & Chemical Engineering 28, 2583–2594.

Ohtsuka, T., 2004. A continuation/gmres method for fast computation of nonlinear receding horizon control. Automatica 40, 563–574.

Patel, N., Padhiyar, N., 2016. Multi-objective optimal control study of fed-batch bio-reactor,. In: Proceedings of the 11th IFAC Symposium on Dynamics and Control of Process Systems, including Biosystems. pp. 91–96.

Poore, A., Tiahrt, C., 1987. Bifurcation problems in nonlinear parametric programming. Mathematical Programming 39, 189–205.

Potschka, A., Logist, F., Impe, J. V., Bock, H., 2011. Tracing the pareto frontier in bi-objective optimization problems by ode techniques. Numerical Algorithms 57, 217–233.

Pérez, A. M., 2014. Pareto tracer: A predictor corrector method for multi-objective optimization problems. Master's thesis, Instituto Politecnico Nacional Mexico.

Rakowska, J., Haftka, R. T., Watson, L. T., 1991. Tracing the efficient curve for multi-objective control-structure optimization. Computational Structures Technology 2, 461–471.

Rao, J. R., Papalambros, P. Y., 1989a. Extremal behavior of one parameter families of optimal design models. Advances in Design Automation, 91–100.

Rao, J. R., Papalambros, P. Y., 1989b. A non-linear programming continuation strategy for one parameter design optimization problems. In: Proceedings of ASME Design Automation Conference.

Ringkamp, M., Ober-Blöbaum, S., Dellnitz, M., Schütze, O., 2012. Handling high-dimensional problems with multi-objective continuation methods via successive approximation of tangent space. Engineering Optimization, 1–30.

Robinson, S. M., 1976. Stability theory for systems of inequalities, part ii: Differentiable nonlinear systems. SIAM Journal on Numerical Analysis 13, 497–513.

Seferlis, P., Hrymak, A. N., 1996. Sensitivity analysis for chemical process optimization. Computers & Chemical Engineering 20, 1177–1200.

Seydel, R., 2009. Practical Bifurcation and Stability Analysis. Springer.

Tapia, R. A., 1980. On the role of slack variables in quasi-newton methods for constrained optimization. Numerical Optimization of Dynamic Systems, 235–246.

Thompson Hale, E., 2005. Numerical methods for d-parametric nonlinear programming with chemical process control and optimization applications. Ph.D. thesis, University of Texas at Austin.