Uses of Methods with Result Verification for Simplified Control-Oriented Solid Oxide Fuel Cell Models

Ekaterina Auer and Stefan Kiel
University of Technology, Business and Design, Wismar, Germany, ekaterina.auer@hs-wismar.de

Abstract: Solid oxide fuel cells (SOFCs) represent a promising technology in the energy sector. One of their advantages is high efficiency. Improvements are necessary, for example, to be able to reduce their operating temperatures or to speed up starting times. Developing efficient control-oriented models for the SOFC temperature has also been in the focus of ongoing research. The goal here is to devise (global) dynamic models valid for a wide range of operating conditions instead of obtaining (linear) approximations of the behavior only for a certain mode of operation. The use of methods with result verification helps to address reliability of SOFC models and to take into account bounded uncertainty.

Dynamic SOFC models are systems of differential equations, the parameters of which have to be fitted to the available sensor data (usually, thousands of measurements). Normally, such differential equations do not have analytical solutions if we keep in mind the goal of devising models suitable for a wide range of operating conditions. In this paper, we point out two simplification possibilities possessing closed-form solutions along with their areas of validity. Moreover, we study their potential with respect to uncertainty handling with the focus on simulation. We perform a sensitivity analysis for different variants of the suggested simplifications. Finally, we consider uncertainty of different magnitudes in heat capacities of gases, which we propagate through the system using interval, affine and Taylor model based methods.

Keywords: methods with result verification, optimization, simulation, UniVerMeC, SOFC

1. Introduction

Modern engineering tasks pose high requirements on safety and reliability of a manufactured system. To check such requirements, developers often have to work with computer simulations of the system instead of the system itself (or even its prototype) in order to keep the costs low. For that purpose, a mathematical model for the system in question is developed and implemented on a computer. However, both the mathematical model of a complex system and its implementation are in general inexact and could introduce a considerable inaccuracy into the simulated outcome. Ideally, this inaccuracy has to be taken into account while checking the above mentioned requirements so that system verification, validation and uncertainty quantification gain more and more importance. In the last decades, the techniques and methodologies for that purpose initially suggested, for example, in (AIAA, 1998), have been studied, improved and extended continuously (e.g., (Oberkampf at al., 2003; Auer and Luther, 2009; Henninger et al., 2010)).

Assuming that the mathematical model is accurate enough, verification answers the question of whether the computer-obtained (numerical) outcome is the correct result of the model. That is, the question of whether the mathematical model and its associated implementation actually produce realistic results remains unanswered during verification, to be handled separately during system validation. Uncertainty handling helps to account for inherent stochasticity in or lack of knowledge about the system. For example, it is well-known that physical quantities such as lengths or masses can only be measured with a certain (small) inaccuracy so that the result is actually not a single real number but a (continuous) set of numbers. Methods for forward uncertainty quantification are able to propagate such input through the system, providing (an approximation of) the resulting output set.

A less widely known possibility to deal with inaccuracies in the model is to use methods with result verification, for example, interval analysis (Moore et al., 2009). Such methods prove mathematically that the outcome of a computer simulation is correct with respect to the underlying mathematical model. As a result, they usually supply a set of machine numbers which with absolute certainty contains the exact model outcome. Almost as a by-product, methods with result verification allow us to propagate bounded uncertainty in parameters directly through the implementation of the mathematical model (provided that the appropriate implementation exists). A common drawback is the possibility of too conservative bounds for the solution sets (e.g., between $-\infty$ and $+\infty$) resulting from the dependency problem or the wrapping effect (Lohner, 2001).

Current research (Rauh et al., 2015; Auer et al., 2012) shows that interval analysis helps to improve and develop control-oriented models for solid oxide fuel cells (SOFC) with the focus on reliability and uncertainty quantification. SOFCs are devices converting chemical energy into electricity using ceramics as the electrolyte. Working at high temperatures, they are sensitive to overheating, but can theoretically achieve the overall efficiency of up to 85%. General research goals in this area are, for example, the reduction of operating temperatures or a speed-up in the starting times. In recent years, developing efficient control-oriented models for the temperature of SOFCs has also been in the focus of ongoing research, see (Huang et al., 2012; Rauh et al., 2014). The goal here is to devise (global) dynamic models valid for a wide range of operating conditions instead of obtaining (linear) approximations of the behavior only for a certain mode of operation.

Dynamic SOFC models are systems of differential equations with many unknown parameters, for example, heat capacities of the involved gases. These quantities have to be chosen in such a way as for the model to reflect the experimental data (usually, thousands of measurements). The traditional approach in this case is to use the least squares optimization for parameters. One difficulty is that the differential equations representing the mathematical SOFC model do not have analytical solutions. That is, to measure the difference between the modeled and experimental result, which is necessary for the objective function of the least squares method, the modeled solution is either approximated by an analytical expression or obtained numerically, increasing the overall imprecision (cf. Section 2). Another difficulty is the necessity to compute the sum of squares of such differences for a large number of measurements, leading to various numerical problems.

An obvious approach to solve the first problem is to simplify the available models. In principle, it could be contradictory to the goal of obtaining models valid under various operating conditions. In (Auer at al., 2015), we pointed out two simplification possibilities for general SOFC models along with their corresponding validity areas. For these simplifications, we derived a closed form solution

and assessed the performance. In this paper, we carry out a sensitivity analysis for both variants and study their potential with respect to uncertainty handling (corresponding to the second difficulty mentioned above) using methods with result verification, in particular, Taylor models (Berz, 1995) and affine forms (de Figueiredo and Stolfi, 2004). Throughout the paper, we rely on the framework UniVerMeC (Kiel, 2014) which allows us to use the same models with different kinds of verified or floating point methods in an efficient way.

The structure of the paper is as follows. In the next section, we give a very brief overview of the theory on parameter identification for SOFC models. This does not include the details on how SOFC models are actually obtained, which can be found elsewhere (Huang et al., 2012). Although methods with result verification are less established in the engineering community, they are well-known in general, so that we omit their description and refer interested readers to (Moore et al., 2009). We specify the kinds of possible SOFC models from the point of view of verification and validation analysis as well as our understanding of the sensitivity analysis in this case in the remainder of Section 2. Section 3 provides the details on the above-mentioned simplification possibilities. In Section 4, we describe the most important features of UniVerMeC and use this tool to actually carry out the sensitivity analysis along with uncertainty quantification. In particular, we explore the potential for overestimation reduction provided by result verification with dependency tracking. A summary of the achieved results and an outlook on our future work conclude this paper.

2. Parameter Identification for SOFC Models

A model of an actual SOFC test rig consists of several subsystems each of which is modeled separately. They describe the behavior of the SOFC stack with respect to its thermodynamics, fluid mechanics, or electrochemistry. In this paper, we consider only the temperature model (Rauh et al., 2014; Rauh et al., 2015) based on general techniques from (Huang et al., 2012; Bove and Ubertini, 2008).

2.1. Modeling the SOFC Temperature

The overall procedure consists of three phases. First, a system of equations is developed to describe the temperature based on heat flow/energy balances over finite domains ("early lumping"). This system is used in combination with the finite volume method to obtain a system of ordinary differential equations (ODEs) for the temperature of the stack (including the behaviour of the preheaters) in the second step. The dimension of the resulting initial value problem (IVP) for the ODEs depends on the number of preheaters and on the coarseness of the discretization we consider. The IVP is used as the basis for the subsequent simulation and control of the SOFC stack in order to obtain global behaviour, in contrast to the traditional methodology (Bove and Ubertini, 2008) locally simplifying the system even further.

In this section, we focus on the third step, the parameter identification for the ODE-based thermal model using the least squares minimization. In general, there are many parameters needing identification, for example, specific heat capacities of the involved gases, reaction enthalpies or coefficients of heat convection. The first parameter group is usually quite large, since it is assumed that

a certain heat capacity c_{gas} is not a constant, but itself depends on the temperature. Since the actual dependency is unknown, it is approximated by polynomials of the second order in temperature θ:

$$c_{gas}(\theta) = c_{gas,0} + c_{gas,1} \cdot \theta + c_{gas,2} \cdot \theta^2 \quad , \tag{1}$$

which leads to three constant parameters per specific heat capacity. If such approximations are used, the right hand side of the resulting ODE system is a third-order polynomial in θ because it contains products of the type $c_{gas} \cdot \theta$. The actual equation for the temperature of the stack $\theta(t)$ has the general form

$$\dot{\theta}(t) = c_m^{inv} \left(c_{gases,AG}(\theta) \cdot \left(\theta_{AG}^{in}(t) - \theta(t) \right) + c_{gases,CG}(\theta) \cdot \left(\theta_{CG}^{in}(t) - \theta(t) \right) + f(\theta(t)) \right) \quad , \tag{2}$$

explained in more detail, for example, in (Rauh et al., 2013). Here, $f(\theta)$ contains components related to Ohmic losses and the heat flows between the stack and environment and of the overall chemical reaction, which are at most quadratic in temperature in case the reaction enthalpies are modeled similarly to (1). The symbols θ_{AG}^{in} and θ_{CG}^{in} denote the temperatures of anode and cathode gases, respectively (parameters which can be measured and recorded by SOFC test rig sensors). The constant c_m^{inv} is explained in Table I, and $c_{gases,AG}(\theta)$, $c_{gases,CG}(\theta)$ are the heat capacities of gases at the anode and cathode, respectively, modeled as in Eq. (1) for each of the involved gases. We studied this kind of model in (Auer et al., 2012; Kiel at al., 2013; Auer et al., 2014). In this paper, we simplify (2) as explained in Section 3 and study the resulting models. The rest of this section contains theory which can be applied generally to all temperature models of the type (2), simplified or not.

2.2. Parameter Identification for the SOFC Temperature

We obtain values for the parameters of the SOFC temperature model using the least squares minimization. If the goal is to verify the obtained optimum, the interval technique of global optimization (Hansen and Walster, 2004) can be employed. A common principle is to minimize the objective function J (or its worst-case value under uncertainty – its upper bound $\overline{J}$ in the interval case) with respect to parameters p:

$$J = \sum_{k=T_b}^{T_e} \sum_{i=1}^{N} \left(y_i(t_k, p) - y_{m,i}(t_k) \right)^2 \quad . \tag{3}$$

Here, $y(t_k, p)$ is the solution to the model equations (e.g., as given in Eqs. (8)–(12)) at the time t_k, $k = T_b, \ldots, T_e$. The notation $y_{m,i}(t_k)$, $i = 1 \ldots N$, signifies the measured values for the temperature. Without loss of generality, we assume that the N states that can be measured using the sensors of the SOFC test rig are the first N ones in the vector $y(t_k, p)$. In our context, $t_k = T_b, T_b + 1, \ldots, T_e$. That is, J quantifies deviations between the simulated results and the measured output vector acquired with $T = T_e - T_b$ samples and a constant sampling time $h = 1$s.

The solution $y(t, p)$ is obtained from the corresponding IVP and optimised with respect to the measured values. Depending on the number of finite volume elements that are used to discretize

the differential equations for the stack temperature, the resulting ODE system for this quantity has different number of equations. Until now, we considered one, three, and nine volume elements. If the preheater models consisting of M equations are used additionally (cf. e.g. Eqs. (8)–(11)), this results in systems of $1+M$, $3+M$, and $9+M$ equations, cf. (Rauh et al., 2013). Finer discretisations are possible in principle but are expensive computationally and rather unstable numerically. An empirically motivated additional condition is induced by the accuracy of the measurements:

$$y_i(t_k, p) \subseteq [y_{m,i}(t_k) - K_i, y_{m,i}(t_k) + K_i] =: [\Delta y_m(t_k)] \quad \text{for} \quad t_k = T_b, \ldots, T_e,\ i = 1 \ldots N \ , \tag{4}$$

with constants K_i chosen in accordance with the actual SOFC test rig. A traditional measure for comparing the quality of the identified parameter sets (because there could be many possibilities) is the root mean square error e given by

$$e = \sqrt{\frac{\sum_{k=T_b}^{T_e} (y_i(t_k) - y_{m,i}(t_k))^2}{T_e - T_b}} \tag{5}$$

for each solution component with the index $i \in \{1 \ldots N\}$ of $y(t)$ we are interested in.

In the most of our recent publications (e.g., (Auer et al., 2012; Kiel at al., 2013)), we solved the global optimisation problem (3) by verified approximation: The true solution $y(t, p)$ of the IVP was approximated by the explicit Euler method as

$$[y^{(k)}] := [y^{(k-1)}] + h \cdot f([y^{(k-1)}], [p]) \ , \tag{6}$$

where f denotes the right side of the IVP (componentwise). The approximation $[y^{(k)}]$ at t_k was substituted for the exact solution $y(t_k)$ in the objective function (Eq. (3)) with the sampling time of $h = 1s$ and the discretisation error ignored. Although we could not verify the whole process by applying interval optimisation procedures in this case, the approximated cost function

$$J_{app} = \sum_{k=T_b+1}^{T_e} \sum_{i=1}^{N} \left(y_i^{(k-1)} - y_{m,i}(t_k) + h \cdot f(y_i^{(k-1)}, p) \right)^2 \ , \tag{7}$$

where $y^{(T_b)}$ is the initial condition, was evaluated in a verified way and optimised using an interval algorithm available in UniVerMeC. Note that the first (and second) derivatives of J_{app} were computed exactly with the help of algorithmic differentiation. Although the control-oriented SOFC models are less complex than those based on, for example, partial differential equations, they are still complex enough so that their treatment with general-purpose verified global optimisation software is next to impossible. In particular, the summation in Eq. (3) is carried out for $T = 16628$ measurements which causes numerical problems even in the usual floating point based case. High times T are explained by the already mentioned general goal to produce models valid for a wide range of operating conditions.

2.3. Anaylsis of the Model Types; Sensitivity

From the point of view of verification and validation analysis, the optimization problem described above might take different forms which are verified to different extents. Generally, we can differentiate between four verification categories (Auer and Luther, 2009) summarized from the lowest to the highest verification extent as follows.

(C4) The implementation of the mathematical model is based on fixed point arithmetic or non-standard floating point arithmetic.

(C3) The implementation of the mathematical model uses standardized IEEE floating point arithmetic.

(C2) The implementation of the mathematical model uses verified techniques for several subtasks.

(C1) The implementation of the mathematical model uses verified techniques everywhere.

In the case of SOFC parameter identification, there are two general classification characteristics. The first one (F1) is the way the simulated solution $y(t_k, p)$ of the IVP is obtained in (3). The second characteristic (F2) is the kind of techniques the implementation relies on (essentially, verified or usual floating-point). Overall, we can discern the following model types:

(F1) How is $y(t_k, p)$ obtained in (3)?

- (F1.a) $y(t_k, p)$ is computed analytically (a closed-form solution)
- (F1.b) $y(t_k, p)$ is approximated by an analytic expression (e.g., using the Euler or Heun method) and the approximation error is neglected, cf. Eq. (7)
- (F1.c) $y(t_k, p)$ is computed using a "black box" numerical solver (no explicit expression for the solution)

(F2) What is the underlying technique for the implementation?

- (F2.a) Traditional floating point methods
- (F2.b) Interval analysis (with result verification; possibility to propagate bounded uncertainty through the system)
- (F2.c) Other techniques with result verification (e.g., affine arithmetic, Taylor models, etc.)

Note that forms F1.a and F1.c combined with F2.b or F2.c correspond to the complete verification of the model (C1), if optimisation is carried out in a completely verified way. If it is not possible to verify the optimum and a certain best suitable interval vector is chosen from the list of candidates produced by global optimisation according to a heuristic technique, then the corresponding degree is C2. In (Auer et al., 2012; Kiel at al., 2013), we were able to achieve the verification degree of C2 for SOFC temperature models of dimensions one and three (without considering preheaters) by using the variant F1.b&F2.b.

There are two more important notions in the context of verification and validation analysis: uncertainty quantification and sensitivity analysis. The task of the former is to quantify the uncertainty in the the model output from the uncertainty in the input (forward propagation) or vice versa (inverse). Generally, it is easier to solve the first problem than the second, so that there are a variety of probabilistic (e.g. Monte-Carlo) or non-probabilistic methods (interval or fuzzy analysis) developed for this purpose. The task of sensitivity analysis is to apportion the uncertainty in the model output to different sources of uncertainty in the model input (Saltelli et al., 2008). Here, a good indicator is considered to be the first derivative of the output with respect to the input we are interested in. The already mentioned framework UniVerMeC allows us to solve both tasks for different kinds of thermal models for SOFCs.

3. Possibile Simplifications

In a realistic situation, engineers often need a trade-off between a high degree of system reliability (or verification) and an acceptable computing time (or overall costs). That is, the models might need simplification if high verification degrees are requested in a real-time simulation or control. In this section, two obvious possibilities to simplify SOFC temperature models are described. Additionally, we provide closed form solutions to the suggested models. Note that using the computer algebra system MATLAB did not work very well for obtaining closed-form expressions (which had to be derived "by hand"). Although it was possible to compute them in MATLAB in some cases, the expressions were numerically unstable, leading to overflows with increasing times t.

3.1. Common Settings

We make several simplifying assumptions, some of which are used only for the purpose of clearer presentation and are not mandatory:

(S1) We consider the nitrogen as the only anode gas for the heating phase without any chemical reactions of gases (not mandatory)

(S2) We work with a single volume element to describe the temperature of the stack as a whole (mandatory for obtaining a closed form solution, at least under assumption S3.b)

(S3) We consider different approximations to the heat capacities $c(\theta)$ of the nitrogen (the anode gas) and the cathode gas (mandatory)

(S3.a) $c_{gas}(\theta) := c_{gas,0}$ is constant (corresponds to the model we denote by MPC1 in this paper)

(S3.b) $c_{gas}(\theta) := c_{gas,0} + c_{gas,1} \cdot \theta$ is linear (corresponds to the model MPL1)

In several cases, we take two preheaters into consideration since it improves the overall quality of SOFC models from the point of view of their control. It is possible include more preheaters into the overall model or do not consider them al all.

The parameters, variables, and control states still present in MPC1 and MPL1 are given by Table I. Those denoted by $\dot{m}$ describe mass flows of, for example, nitrogen. All *control variables* from

Table I are assumed to be piecewise constant. That is, we use the information from measurements to describe these values instead of modeling them by appropriate initial value problems (IVPs). These measurements are recorded each second (i.e., with the stepsize $h = 1$s) between $T_b = 362$ and $T_e = 17000$ (the heating stage of the experiment). In the same way, it is possible to use sensor data for the preheater output instead of the IVPs (8)–(11). All *parameters* are considered to be constant; their actual values are to be identified by an appropriate optimization procedure (cf. Section 2). For all *variables*, the corresponding initial values at the beginning of integration T_b are supposed to be known.

Because the control variables are piecewise constant, the model IVPs are solved separately in each time interval $[t_{n-1}, t_n]$, $t_n - t_{n-1} = 1$, $n = T_b + 1, \ldots, T_e$, since the stepsize for taking measurements is $h = 1$s. The initial values in the next step are the results from the previous step so that $t_0 = t_{n-1}$ for the time step t_n. If the possibility F2.b is used, we have to work with the interval hull of the values of the control variables, because we do not know the exact value inside (t_{n-1}, t_n), only in t_{n-1} and t_n.

3.2. Simplified Model MPC1: Constant Heat Capacities

The system of ODEs resulting from the assumptions S1, S2, S3.a is as follows (as a comparison, the general form under S2 is given in (Auer et al., 2012)):

$$\dot{y}_0 = T_{AG}^{inv} \cdot (u_1 - y_0) \tag{8}$$
$$\dot{y}_1 = T_{SL,AG}^{inv} \cdot (y_0 - y_1) \tag{9}$$
$$\dot{y}_2 = T_{CG}^{inv} \cdot (u_2 - y_2) \tag{10}$$
$$\dot{y}_3 = T_{SL,CG}^{inv} \cdot (y_2 - y_3) \tag{11}$$
$$\dot{\theta} = -c_m^{inv} \cdot (k_{const} - (c_{CG,0} \cdot y_3 + c_{N_2,0} \cdot y_1) + k_{lin}\theta) \ , \tag{12}$$

where k_a, k_{const} and k_{lin} are defined according to

$$\begin{aligned} k_a &= 234000\alpha_i + 448500\alpha_j + 345000\alpha_k \ , \\ k_{const} &= -\theta_A k_a \ , \\ k_{lin} &= k_a + \dot{m}_{CG}^{in} \cdot c_{CG,0} + \dot{m}_{N_2}^{in} \cdot c_{N_2,0} \ , \end{aligned}$$

and initial conditions $y_i(T_b) = y_i^{ic}$, $i = 0, 1, 2, 3$, $\theta(T_b) = \theta^{ic}$. The obtained ODE system is linear in temperature (including the preheater states). The equations (8)–(9) describe the first preheater (for the nitrogen), Eqs. (10)–(11) the second (for the cathode gas), and Eq. (12) the stack temperature. We can derive the simple closed form solution to the model:

$$y_0(t) = u_1 - (u_1 - y_0^{ic})e^{-T_{AG}^{inv}(t-t_0)} \tag{13}$$
$$y_1(t) = u_1 - \frac{T_{SL,AG}^{inv}}{T_{SL,AG}^{inv} - T_{AG}^{inv}}(u_1 - y_0^{ic})e^{-T_{AG}^{inv}(t-t_0)} + k_{N_2}e^{-T_{SL,AG}^{inv}(t-t_0)} \tag{14}$$

for the anode gas preheater, where $k_{N_2} = y_1^{ic} - u_1 + \frac{T_{SL,AG}^{inv}}{T_{SL,AG}^{inv} - T_{AG}^{inv}}(u_1 - y_0^{ic})$. The solution for the cathode gas preheater has the same form; only the parameters are different. (All labels "N_2" (or AG) in

Table I. Model parameters, control and state variables.

	Parameters (to identify)
α_i, α_j, α_k	coefficients of heat convection
$c_{N_2,0}$, $c_{N_2,1}$	heat capacity of nitrogen as $c_{N_2}(\theta) = c_{N_2,0}(+c_{N_2,1} \cdot \theta)$
$c_{CG,0}$, $c_{CG,1}$	heat capacity of the cathode gas as $c_{CG}(\theta) = c_{CG,0}(+c_{CG,1} \cdot \theta)$
T^{inv}_{AG}	inverse time constant of the anode gas preheater
T^{inv}_{CG}	inverse time constant of the cathode gas preheater
$T^{inv}_{SL,AG}$	inverse time constant of the anode gas supply line
$T^{inv}_{SL,CG}$	inverse time constant of the cathode gas supply line
c	specific heat capacity of the stack module
m	mass of the stack module
c^{inv}_m	$= 1/(c \cdot m)$
	Variables
$y_0 = v_{N_2}$	$\dot{m}^{in}_{N_2} \cdot \theta_{N_2}$ at the preheater outlet, $y_0(T_b) =: y^{ic}_0$
$y_1 = v^{in}_{N_2}$	$\dot{m}^{in}_{N_2} \cdot \theta^{in}_{N_2}$ at the stack inlet, $y_1(T_b) =: y^{ic}_1$
$y_2 = v_{CG}$	$\dot{m}^{in}_{CG} \cdot \theta_{CG}$ at the preheater outlet, $y_2(T_b) =: y^{ic}_2$
$y_3 = v^{in}_{CG}$	$\dot{m}^{in}_{CG} \cdot \theta^{in}_{CG}$ at the stack inlet, $y_3(T_b) =: y^{ic}_3$
θ	temperature of the stack, $\theta(T_b) =: \theta^{ic} = 293.9$K
	Control variables
$\dot{m}^{in}_{N_2}$	mass flow of anode gas (recorded data)
$\dot{m}^{in}_{CG}$	mass flow of cathode gas (recorded data)
θ_A	ambient temperature
θ^d_{AG}	desired temperature of the anode gas (recorded data)
θ^d_{CG}	desired temperature of the cathode gas (recorded data)
$u_1 = v^d_{N_2}$	desired $v_{N_2} = \theta^d_{AG} \cdot \dot{m}^{in}_{N_2}$
$u_2 = v^d_{CG}$	desired $v_{CG} = \theta^d_{CG} \cdot \dot{m}^{in}_{CG}$

Eqs. (13)–(14) should be changed to "CG".) If $T^{inv}_{SL,AG} = T^{inv}_{AG}$, we obtain trivial solutions $y_0(t) = y^{ic}_0$, $y_1 = y^{ic}_1$.

The solution for the temperature can be obtained after substituting into (12) the corresponding expressions for $y_3(t)$ and $y_1(t)$. Using variable separation and variation of the constant delivers the general solution form as

$$\theta(t) = \mathcal{I}_{N_2}(t) + \mathcal{I}_{CG}(t) + k_\theta e^{-c^{inv}_m \cdot k_{lin}(t-t_0)} - \frac{k_{const}}{k_{lin}} \ , \tag{15}$$

where

$$\mathcal{I}_{N_2}(t) := \frac{u_1 \cdot c_{N_2,0}}{k_{lin}} + c_m^{inv} \cdot c_{N_2,0} \cdot \left(- \frac{T_{SL,AG}^{inv}}{(T_{SL,AG}^{inv} - T_{AG}^{inv})(c_m^{inv} k_{lin} - T_{AG}^{inv})} (u_1 - y_0^{ic}) e^{-T_{AG}^{inv}(t-t_0)} + \frac{k_{N_2}}{c_m^{inv} k_{lin} - T_{SL,AG}^{inv}} e^{-T_{SL,AG}^{inv}(t-t_0)} \right) ,$$

$\mathcal{I}_{CG}(t)$ analogously, and $k_\theta = \theta^{ic} - \mathcal{I}_{CG}(t_0) - \mathcal{I}_{N_2}(t_0) + \frac{k_{const}}{k_{lin}}$. If $T_{SL,AG}^{inv} = T_{AG}^{inv}$ and $T_{SL,CG}^{inv} = T_{CG}^{inv}$, that is, if we have trivial solutions for the preheaters, the corresponding solution for the temperature is also trivial:

$$\theta(t) = -\frac{\tilde{k}_{const}}{k_{lin}} + \left(\theta^{ic} + \frac{\tilde{k}_{const}}{k_{lin}} \right) \cdot e^{-c_m^{inv} \cdot k_{lin} \cdot (t-t_0)} , \tag{16}$$

where $\tilde{k}_{const} = -\theta_A \cdot k_a - (c_{CG,0} \cdot y_3^{ic} + c_{N_2,0} \cdot y_1^{ic})$. This version of the solution is also useful if we assume that the preheaters are piecewise constant in time and employ the recorded sensor data instead of the dynamic model given by Eqs. (8)–(11).

In principle, we can use the same approach to obtain closed-form solutions for temperature models of higher dimensions with constant heat capacities. If the preheaters are assumed to be constant, the resulting linear system has only constant coefficients so that well-known techniques from the calculus can be employed. However, the expressions become very complicated. The numerical results from, for example, (Auer at al., 2015), suggest that there is almost no gain in using them even in the one dimensional case MPC1 from the point of view of the computing time for identification. One of their advantages is higher accuracy: better model parameter sets can be identified with their help. In (Auer at al., 2015a), we were able to identify the parameter set shown in Table II for the main parameters of interest of MPC1 with the root mean square measure $e = 3.5259$K. As a comparison, the best parameter set obtained for the non-simplified model of dimension one so far has the measure $e = 2.1641$K. Another advantage of MCP1 is the reduced computing time for pure simulation with known parameter sets. Our general conclusion about the model MPC1 is that it can be used efficiently in the context of an online simulation or control over relatively short time intervals. In Section 4, we will further explore the properties of this model by studying its sensitivity to parameters and its ability to account for parameter uncertainty.

Table II. The best obtained parameter sets for MPC1 and MPL1.

Model	c	$\alpha_i = \alpha_j = \alpha_k$	$c_{N_2,0}$	$c_{N_2,1}$	$c_{CG,0}$	$c_{CG,1}$	e
MPC1(F1.a&F2.a)	$4.53503 \cdot 10^6$	$1.49548 \cdot 10^{-3}$	$1.96452 \cdot 10^6$	–	$1.52826 \cdot 10^7$	–	3.5K
MPL1(F1.b&F2.a)	$3.38579 \cdot 10^4$	$3.60235 \cdot 10^{-5}$	$5.52725 \cdot 10^6$	-11221.8	$-8.88817 \cdot 10^4$	483.868	0.5K

3.3. Simplified Model MPL1: Linear Heat Capacities

The simplified model MPL1 relies on linear polynomial approximations for the heat capacities. Assuming that the preheater states are not constant inside the time interval (of width 1s) where other control variables are constant, we obtain a Riccati equation for the temperature which does not seem to have an analytical solution. For piecewise constant preheater states, we can derive a closed-form expression similarly to (Auer et al., 2014) as shown in the following.

If the preheater states are not constant, Eqs. (8)–(11) stay the same as for MPC1 from the previous section. The equation for the temperature θ turns into

$$\dot{\theta} = -c_m^{inv} \cdot (k_{const}(t) + k_{lin}(t)\theta + k_{sq}\theta^2) \tag{17}$$

where $k_{const}(t)$, $k_{lin}(t)$ depend on time and θ^2 appears on the right side of the equation with the constant coefficient k_{sq}:

$$\begin{aligned} k_{const}(t) &= -\theta_A k_a - (y_3(t)c_{CG,0} + y_1(t)c_{N_2,0}) \ \ (k_a \ \text{ as in Section 3.2}) \ , \\ k_{lin}(t) &= k_+ \dot{m}_{CG}^{in} \cdot c_{CG,0} + \dot{m}_{N_2}^{in} \cdot c_{N_2,0} - (y_3(t)c_{CG,1} + y_1(t)c_{N_2,1}) \ , \\ k_{sq} &= \dot{m}_{CG}^{in} \cdot c_{CG,1} + \dot{m}_{N_2}^{in} \cdot c_{N_2,1} \ . \end{aligned}$$

Now we have two additional parameters $c_{CG,1}$, $c_{N_2,1}$ as compared to MPC1, resulting from the fact that we represent heat capacities of gases as $c_{N_2}(\theta) = c_{N_2,0} + c_{N_2,1} \cdot \theta$, $c_{CG}(\theta) = c_{CG,0} + c_{CG,1} \cdot \theta$. Equation (17) is a Riccati equation. Since k_{sq} does not depend on time in our case, it can be transformed into a system of the following linear ODEs with non-constant coefficients:

$$\begin{pmatrix} y_4 \\ y_5 \end{pmatrix}' = \begin{pmatrix} 0 & 1 \\ -(c_m^{inv})^2 k_{const}(t) k_{sq} & -c_m^{inv} k_{lin}(t) \end{pmatrix} \cdot \begin{pmatrix} y_4 \\ y_5 \end{pmatrix} \ , \tag{18}$$

where the temperature θ equals $\dfrac{y_5}{c_m^{inv} k_{sq} y_4}$. The matrix in Eq. (18) does not necessarily commute with itself for given points of time t, s so that finding a closed-form expression in this case seems impossible. Therefore, we assume now that the preheaters behave as constant functions inside intervals of width 1s (piecewise constant on the whole) similarly to control variables. The values of y_1 and y_3 can be measured with the sampling frequency of 1s between T_b and T_e. The closed-form solution for the temperature equation with – now constant – coefficients k_{const}, k_{lin} and $y_1(t) = y_1^{meas}$, $y_3(t) = y_3^{meas}$ has different branches in dependence on $D = k_{lin}^2 - 4 \cdot k_{const} \cdot k_{sq}$:

$$\begin{aligned} D > 0: \ \theta(t) &= \frac{1}{k_{sq}} \left(\frac{\sqrt{D}}{1 - e^{-c_m^{inv}(t-t_0)\sqrt{D}} \cdot \left(1 - \dfrac{2\sqrt{D}}{2k_{sq}\theta^{ic} + k_{lin} + \sqrt{D}}\right)} - \frac{k_{lin} + \sqrt{D}}{2} \right) \ , \\ D < 0: \ \theta(t) &= \frac{\sqrt{-D}\tan\left(-\frac{\sqrt{-D}}{2} c_m^{inv}(t-t_0) + \theta^c\right) - k_{lin}}{2k_{sq}} \ , \ \theta^c = \operatorname{atan}\left(\frac{2k_{sq}\theta^{ic} + k_{lin}}{\sqrt{-D}}\right) \ , \\ D = 0: \ \theta(t) &= \frac{2\theta^{ic} + k_{lin}/k_{sq}}{2 + c_m^{inv}(t-t_0)(2k_{sq}\theta^{ic} + k_{lin})} - \frac{k_{lin}}{2k_{sq}} \ . \end{aligned} \tag{19}$$

From the definitions for k_{const}, k_{lin}, k_{sq} and the data, it is likely (but not certain) that D is positive. For the set of parameters obtained in MATLAB (F1.b&F2.a), we checked that to be so. However, the measure e for this parameter set and MPL1 is not satisfactory. If we take this set as starting values in UniVerMeC and use the model variant F1.b&F2.a, we obtain a much better parameter set (cf. Table II). Note that the measure e for this set is even better than that achieved by using the non-simplified model (Auer at al., 2015a). A drawback is that there is a branch change for the solution $\theta(t)$ in this case (from $D > 0$ to $D < 0$). However, this fact does not make any difference for the pure simulation. The difficulties would occur during stages in the verification and validation process requiring derivatives of θ (e.g. sensitivity analysis, uncertainty quantification with Taylor models). More information about that is in the next section. Our general conclusion is that MPL1 is a very promising simplification (both as F1.a and F1.b), which is able to handle a wide range of operating conditions in the same way as the non-simplified model. Moreover, it is faster and possesses a closed-form solution.

4. Uncertainty Quantification and Sensitivity Analysis for MPC1 and MPL1

In this section, we analyse the performance of the proposed simplified models MPC1 and MPL1 with respect to their ability to take into account uncertainty in model parameters during the simulation stage. That is, we assume that the parameters of the both models have already been identified as reported in (Auer at al., 2015a). We rely on the parameter values given by Table II which we obtained by using the interior point optimizer IPOPT (Wächter and Biegler, 1991) inside the framework UniVerMeC (Kiel, 2014). In this sense, the results of the parameter identification stage for these SOFC temperature models are not verified and have verification degree of C3 (cf. Section 2.3).

One possibility to quantify the uncertainty during the simulation stage is to use methods with result verification. This has an additional advantage: the produced simulation data are proven to be correct wrt. the considered mathematical model (i.e., do not contain any numerical errors). If there is no uncertainty in parameters, the simulation results – now nonetheless (tight) intervals containing the true solution – can be intersected with the sets given by Condition (4). If they stay inside the bounds of (4), the parameters obtained in the non-verified way mentioned above are validated. Both of the sets from Table II are valid in this sense.

To actually quantify the uncertainty, we consider variants F1.a and F1.b for MPC1 and MPL1 and three kinds of arithmetics with result verification (F2.b-F2.c). These are the interval arithmetic as implemented by C-XSC (Hofschuster et al., 2008), affine arithmetic implemented by Yalaa (Kiel, 2014), and Taylor model arithmetic from RiOT (Eble, 2007). The term "arithmetic" means that we need only basic operations such as addition along with several elementary functions such as the exponential (e^x). However, the combination F1.b&F2.b requires a verified solver for initial value problems, for which purpose we employ VNODE-LP (Nedialkov, 2002). At the moment, there are no other verified IVP solvers available in UniVerMeC where we implement the models and conduct computations during all stages of modeling and simulation cycle for SOFC temperature. That is, the combination F1.b&F2.c is possible in principle but not currently implemented inside UniVerMeC.

We have not mentioned variant F1.c so far, the reason being that it was found to be too slow (Pusch, 2013; Auer et al., 2014) even in its non-verified form F2.a. Additionally, it is not interesting from the point of view of the simulation stage since it is the same as for F1.b. The real advantage concerns the stage of parameter identification which is not in our focus in this paper.

The question of MPC1 and MPL1 sensitivity to parameters is also an interesting one. The answer allows us not only to apportion the output uncertainty to different inputs, but also to choose variables with dependency tracking (e.g., in affine or Taylor model arithmetic) efficiently. Therefore, this section is structured as follows. First, we outline the principles of the framework UniVerMeC in short. After that, we carry out sensitivity analysis for MPC1 and MPC2 using floating point arithmetic and algorithmic ("exact") differentiation as implemented in FADBAD++ (Stauning, 1997). Finally, we consider uncertainty in heat capacities of gases and propagate it through the system, which allows us to assess the usefulness of the considered models for that purpose.

4.1. UniVerMeC — An Interoperable Framework

Although originally developed for distance computations, the framework UniVerMeC works very well for the stages of parameter identification and simulation during the modeling and simulation cycle for SOFC temperature. Its major advantage consists in decoupling a computerized model from the arithmetic it works with. Usually, users have to decide which arithmetic their implementations are based on. The default choice is the floating-point arithmetic, although it has a number of disadvantages (for example, we cannot guarantee that the computed result is actually correct). Through its layered structure and, in particular, its adapter concept, UniVerMEC allows us to implement a mathematical model largely without the need to think about the kind of arithmetic we want to use. This helps to choose arithmetics according to the actual goal (e.g., offline verification as opposed to fast online simulation/control) and even combine verified and non-verified techniques in a meaningful way (e.g., floating-point parameter identification followed by interval simulation). Currently, UniVerMeC does not implement any stochastic methods or arithmetics. However, it can be extended in this way, which would provide even more possibilities for interoperable work.

UniVerMeC's layered structure is relaxed, that is, a layer can be skipped. The bottom layer *core* provides access to floating point, interval, and affine arithmetic as well as to Taylor models, which share a common interface but rely an different adapters. The next layer, *function*, allows us to represent scalar and vector-valued functions uniformly if their mathematical sense is supplied according to the formalization from (Kiel, 2014). This concept helps to evaluate a function with all arithmetics supported at the *core* layer. We introduce abstractions for derivatives, slopes, Taylor coefficients or contractors at this level, a list which advanced users can extend if necessary. The third layer is responsible for defining models in the framework, for example, the IVPs given in Section 3. It merges the relevant abstractions provided at the previous two layers into one entity. Specialized data structures for higher-level algorithms are at the fourth level, for example, those for special types of search space decomposition used in optimisation. Actual algorithms are implemented at the topmost level. UniVerMeC offers its own global optimisation algorithm *GlobOpt* based on that described in (Hansen and Walster, 2004). Additionally, external software such as IVP solvers or further optimisers can be interfaced at this level. For example, we interface the interior-point tool IPOPT for non-verified optimisation.

4.2. Sensitivity Analysis

Since UniVerMeC already implements the possibility to obtain derivatives for the models, it is easy to study different forms of MPC1 and MPL1 from this point of view. We chose to rely on floating point arithmetic in this case although we could have also used intervals or affine forms. The reason is that a number characterising the first derivative of the temperature $\theta(t)$ or the objective function J wrt. a certain parameter p gives us enough information to apportion uncertainty appropriately.

First, we consider the SOFC temperature $\theta(t)$ as modeled by MPL1 (cf. Eqs. (17) and (19)) in some detail. Since our simplifications concerned the heat capacities, we compute $\frac{\partial\theta(t_k)}{\partial c_{N_2,0}}$, $\frac{\partial\theta(t_k)}{\partial c_{N_2,1}}$, $\frac{\partial\theta(t_k)}{\partial c_{CG,0}}$, and $\frac{\partial\theta(t_k)}{\partial c_{CG,1}}$ for each point of time $t_k \in \{T_b, T_b+1, \ldots, T_e\}$. In Figure 1, the results are shown for F1.a (that is, Eq. (19)) on the left and F1.b (that is, Formula (6) in floating point with f defined by the right side of Eq. (17)) on the right. The Figure demonstrates that, out of the four mentioned parameters, both model variants are most sensitive to $c_{CG,1}$. On the whole, variant F1.b is much less sensitive to changes in parameters.

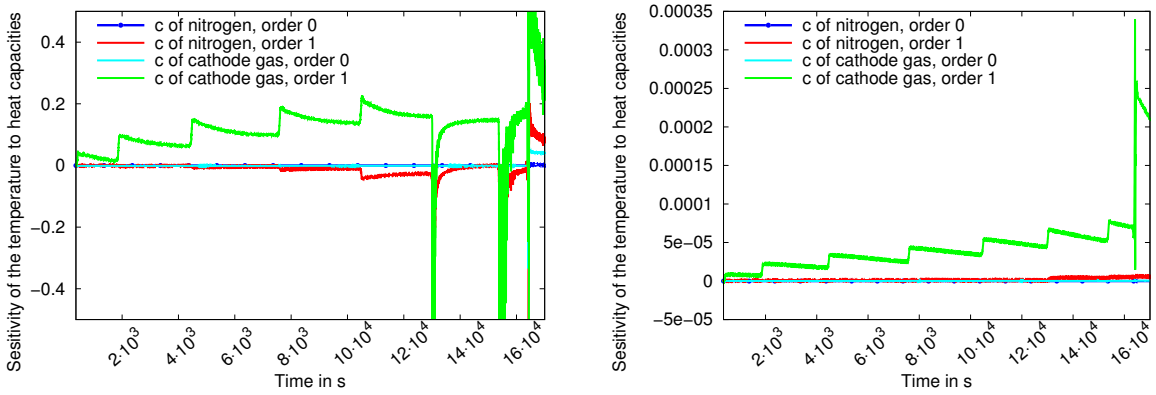

Figure 1. Sensitivity of the modeled SOFC temperature to the heat capacity of nitrogen and cathode gas for the model MPL1: exact solution on the left (F1.a), the Euler approximation on the right (F1.b).

Out of all possible parameters, MPL1 is most sensitive to α_i (equal to α_j and α_k in our setting). Instead of computing $\frac{\partial\theta(t_k)}{\partial p}$ for all relevant points of time, we now consider $\frac{\partial J}{\partial p}$ because, according to Formula 3, it essentially summarizes such plots as in Figure 1 in one number. The results for both MPL1 and MPC1 (as F1.a and F1.b) are given in Table III. Aside from demonstrating extreme sensitivity to α_i an all cases (which was also witnessed in experiment), it lets us observe that both variants of MPC1 are almost equally sensitive to changes in parameters, whereas there is a big difference for the variants of MPL1. The variant MPL1 as F1.b is the least sensitive one and, therefore, can be efficiently employed for parameter identification, the claim supported by the fact that the best parameter set so far was computed using this option. In Figure 2, we can see that MPL1 describes the measured data in the best way (both as F1.a and F1.b).

Table III. Sensitivities of the objective function J (cf. Eq. (3)) for the models MPC1 and MPL1 under F2.a.

J	$\left\lvert\frac{\partial J}{\partial c}\right\rvert$	$\left\lvert\frac{\partial J}{\partial \alpha_i}\right\rvert$	$\left\lvert\frac{\partial J}{\partial c_{N_2,0}}\right\rvert$	$\left\lvert\frac{\partial J}{\partial c_{N_2,1}}\right\rvert$	$\left\lvert\frac{\partial J}{\partial c_{CG,0}}\right\rvert$	$\left\lvert\frac{\partial J}{\partial c_{CG,1}}\right\rvert$
F1.a, MPL1	$2.93594 \cdot 10^{1}$	$3.8117 \cdot 10^{10}$	$4.9631 \cdot 10^{-2}$	$2.63897 \cdot 10^{1}$	1.46662	$1.26157 \cdot 10^{2}$
F1.b, MPL1	$4.76907 \cdot 10^{-3}$	$6.87924 \cdot 10^{5}$	$5.36099 \cdot 10^{-5}$	$2.20879 \cdot 10^{-2}$	$1.3089 \cdot 10^{-3}$	$5.25226 \cdot 10^{-1}$
F1.a, MPC1	$3.09309 \cdot 10^{-1}$	$2.44704 \cdot 10^{8}$	$8.48586 \cdot 10^{-2}$	–	$1.36277 \cdot 10^{-1}$	–
F1.b, MPC1	$3.09307 \cdot 10^{-1}$	$2.44754 \cdot 10^{8}$	$8.48573 \cdot 10^{-2}$	–	$1.36288 \cdot 10^{-1}$	–

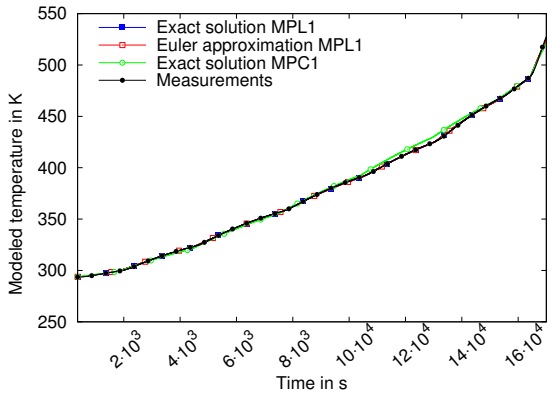

Figure 2. SOFC temperature modeled by MPL1 and MPC1 using F2.a in comparison to the measured data.

4.3. Uncertainty Quantification

We consider uncertainty of different magnitudes in parameters $c_{N_2,0}$ and $c_{CG,0}$ and try to propagate it through the models MPL1 and MPC1 (both as F1.a) using interval, affine, and Taylor model arithmetics. The variant F1.b as a pure approximation by Formula (6) is not feasible in this context because the deviation from the true solution would be too large. That is, to be accurate, we need to solve the corresponding simplified system (Eqs. (8)–(12) or Eq. (17)) numerically in this case. For comparison, we do this under uncertainty using the interval IVP solver VNODE-LP. Additionally, we measure the wall times needed for each relevant simulation. Our goal is to establish an uncertainty level in these parameters which would allow us enclose the measured data in a meaningful way, that is, not entirely because of overestimation effects.

The MPL1 solution from Eq. (19) takes preheaters into account only as measurements, whereas the exact solution in Eqs. (13),(14),(16) explicitly describes them. Besides, the solution in Eq. (19) has three different branches. To be able to enclose all possible states, it is necessary to compute the convex hull of enclosures of all three branches each time when the upper bound of D is not strictly less than zero or the lower bound of D is not strictly larger than zero. This fact makes MPL1 as F1.a more prone to overestimation than other variants, which is reflected in Figure 3, left. Only the uncertainty of $\pm 10^{-5}\%$ of the nominal parameter values of $c_{N_2,0}$ and $c_{CG,0}$ (cf. Table II) can be considered in interval arithmetic meaningfully, for both variants F.a and F.b (the

Table IV. Computing times for temperature simulations with exact solutions to models MPC1 and MPL1 under uncertainty $\pm U\%$ in $c_{N_2,0}$ and $c_{CG,0}$ using interval, affine, and Taylor model arithmetics. The last column shows times for the corresponding variant without the exact solution, obtained using interval-based solver VNODE-LP.

	Intervals	Affine forms	Taylor models	VNODE-LP
MPC1	0.34 s ($U = 10^{-4}$)	112.09 s ($U = 1$)	207 s ($U = 10$)	$>$ 1 day ($U = 10^{-4}$)
MPL1	0.12 s ($U = 10^{-5}$)	23.34 s ($U = 50^{-5}$)	120.5* s ($U = 10^{-3}$)	228.42 s ($U = 10^{-5}$)

latter with VNODE-LP). Note that the nominal values for these parameters are of orders 10^6, 10^4 for the considered parameter set. The uncertainty of $\pm 10^{-4}\%$ leads to too large overestimation in the output $\theta(t)$, so that these enclosures are not useful. Affine forms (with a restart each 5 steps so that the dependencies are lost and the overall simulation is faster) can handle a slightly larger uncertainty of $\pm 50^{-5}\%$. Taylor models perform better with respect to uncertainty ($\pm 10^{-3}\%$) but have a different problem with the solution as in Eq. (19). The library RIOT we employ for Taylor model computations does not implement the elementary function *atan*, so that the branch containing it cannot be used and we have to stop computations.

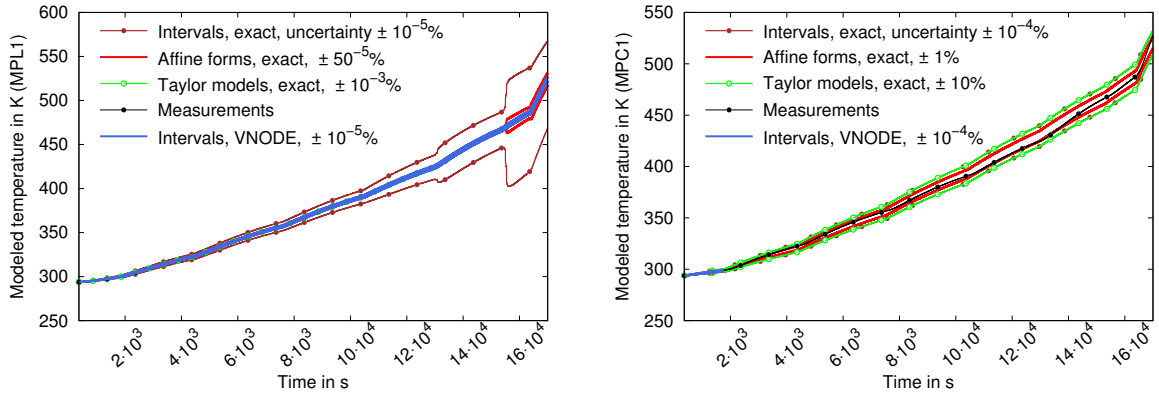

Figure 3. Upper and lower bounds of the temperature with uncertainty in $c_{N_2,0}$ and $c_{CG,0}$: MPL1 on the left and MPC1 on the right.

The model MPC1 behaves better with respect to uncertainty propagation (cf. Figure 3, right). Although interval arithmetic cannot handle more uncertainty than $\pm 10^{-4}\%$ in a meaningful way, affine forms propagate $\pm 1\%$ and $\pm 10\%$, respectively, despite large nominal values of $c_{N_2,0}$, $c_{CG,0}$ for the considered parameter set. The general form of the trajectory is preserved, and the measurement data are enclosed.

In Table IV, the computing times are shown. As a benchmark, we recorded the time necessary for 10 runs of each variant. The table gives the corresponding wall times divided by 10. Not surprisingly, the variant F1.b for both models is slow, extremely so for MPC1 (cf. the last column of the Table). Note that the speed in this case depends on the parameter set, since VNODE-LP implements

a dynamic stepsize control. That is, the simulation can be faster for a different parameter set; however, since the set from Table II had the closest fit for MPC1, we decided not to change it. The simulation reached $t_k = 1761$ s before we terminated it. As opposed to it, VNODE-LP could handle the variant F1.b for MPL1 in an acceptable time. However, the variant F1.a is still faster (especially for MPL1 because it does not have to handle the preheaters explicitly). Besides, we are able to use affine forms and Taylor models here, which achieve acceptable computing times. Note that the time for MPL1 with Taylor models is provided only as a reference for a different parameter set for which it does not become necessary to change the solution branch (so that we do not have to compute *atan* and the simulation is not terminated).

5. Conclusions

In this paper, we analysed two simplified versions of the one dimensional SOFC temperature model from the point of view of their performance during the simulation stage of the modeling and simulation process. The simplification concerned the heat capacities of gases, which were assumed to be constant (MPC1) or linear (MPL1) in temperature, respectively, as opposed to the normal version of the model operating with polynomials of order two. For MPC1 and MPL1, we were able to provide closed form expressions for the solutions (F1.a). Additionally, we analysed the performance of MPC1 and MPL1 when the obtained IVPs were solved numerically (F1.b). Having identified good parameter sets for both models, we performed a sensitivity analysis, which confirmed the experimentally observed fact that both models were very sensitive to changes in coefficients of heat convection. Since the simplifications concerned the heat capacities of the anode and cathode gases, we considered bounded uncertainties in them and propagated them through the system using interval, affine, and Taylor model based methods, which, in addition to the mentioned uncertainty quantification, allowed us to rule out the possibility of numerical errors during simulation.

The models MPC1 and MPL1 are now analysed almost completely from the point of view of simulation. The MPL1 is able to describe the measured data for wide ranges of operating conditions in the best way if no uncertainty in parameters is considered. The variant relying on the closed form expression is the fastest in all considered arithmetics, but less suitable for uncertainty propagation than the corresponding MPC1 one. Using the latter, we are able to process uncertainty of up to the order $\pm 10\%$ of the (large) nominal values of parameters under Taylor model arithmetic without much overestimation, whereas the former one is good for uncertainties up to $\pm 10^{-3}\%$. The problem is that the solution to MPL1 has different branches, which might lead to a large overestimation in "undecided" situations where a convex hull of all possibilities needs to be considered to ensure a verified result. Note that the nominal values are of orders up to 10^7 in the parameter sets with the smallest root mean square measures. The numerical variants of both models are actually too slow to propagate the uncertainty in real time simulations, especially for MPC1. However, they are less sensitive to changes in parameters, which makes them attractive at the (offline) identification stage of the process. An overall conclusion is that MPC1 can be efficiently used over shorter time intervals (especially with uncertainty, in which case it produces enclosures covering the data). MPL1 can compete with the usual non-simplified model from the point of view of data fit over long time intervals and is faster both as the closed-form and the numerical variant.

In our future work, we plan to explore the possibilities to speed up the parameter identification stage using the GPU. (Note that it would not be possible to employ the GPU for the classical simulation stage in our context since it is sequential in its nature — we need initial values as simulation results from the previous step to be able to proceed.) In (Auer et al., 2012; Auer et al., 2014), we have already applied that idea to the verified parameter identification for the non-simplified SOFC temperature model, achieving a significant speed-up. However, we were not able to verify the optimum because of overestimation, and the obtained candidate parameter sets were not very satisfactory with respect to the data fit. We hope that exact solutions to simplified models will help to verify the optimum, so that an overall verification degree of C1 can be achieved (in an acceptable time owing to the GPU). Moreover, it might be possible to employ the model variant F1.c which was dismissed until now because of very high computing times. This goal cannot be reached very easily for a technical reason: implementations of verified methods (e.g., interval analysis, etc.) on the GPU are still very rudimentary in comparison to those available for the CPU.

Acknowledgements

Thanks to A. Rauh and L. Senkel from the University of Rostock for providing measurement data for the SOFC test rig (available in Rostock) along with invaluable advice about the models.

References

AIAA. Guide for the Verification and Validation of Computational Fluid Dynamics Simulations. American Institute of Aeronautics and Astronautics (AIAA G-077-1998), 1998

Auer, E., S. Kiel, T. Pusch, and W. Luther. A Flexible Environment for Accurate Simulation, Optimization, and Verification of SOFC Models, in *Proc. of ASCE-ICVRAM-ISUMA, July 13–16, 2014*. Liverpool, UK, 2014.

Auer, E., S. Kiel, and A. Rauh. Verified Parameter Identification for Solid Oxide Fuel Cells, in *Proceeding of REC 2012*, pages 41–55, 2012.

Auer, E. and W. Luther. Numerical Verification Assessment in Computational Biomechanics, in *Proc. of Dagstuhl Seminar 08021: Numerical validation in current hardware architectures — From embedded systems to high-end computational grids*, Lecture Notes in Computer Science, pages 145–160, 2009.

Auer, E., L. Senkel, S. Kiel, and A. Rauh. Performance of Simplified Interval Models for Simulation and Control of SOFC, in Y. Tsompanakis et al., editors, *Proceedings of the Conference on Soft Computing Technology in Civil, Structural and Environmental Engineering*, Civil-Comp Press, Stirlingshire, UK, 2015.

Auer, E., L. Senkel, S. Kiel, and A. Rauh. Control-Oriented Models for SO Fuel Cells From the Angle of V&V: Analysis, Simplification Possibilities, Performance. *Special Issue of Proceedings of the Conference on Soft Computing Technology in Civil, Structural and Environmental Engineering*, 2015a, submitted.

Berz, M. Modern Map Methods for Charged Particle Optics. *Nucl. Instrum. Methods*, 363(1–2):100–104, 1995.

Bove, R. and S. Ubertini, editors. *Modeling Solid Oxide Fuel Cells.* Springer, Berlin, 2008.

Eble, I. Über Taylor-Modelle. PhD thesis, Universität Karlsruhe, 2007.

de Figueiredo, L. H. and J. Stolfi. Affine Arithmetic: Concepts and Applications. *Numerical Algorithms*, 34 (1–4):147–158, 2004.

Hansen, E. and G.W. Walster. *Global Optimization Using Interval Analysis.* Marcel Dekker, New York, 2004.

Henninger, H., S. Reese, A. Anderson, and J. Weiss. Validation of Computational Models in Biomechanics, in *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Eng. in Med.*, 224:801–812, 2010.

Hofschuster, W., W. Krämer, and M. Neher. C-XSC and Closely Related Software Packages, in Cuyt A. M., W. Krämer, W. Luther, and P. W. Markstein, editors, *Numerical Validation in Current Hardware Architectures LNCS 5492*:68–102, 2008.

Huang, B., Y. Qi, and A. Murshed. *Dynamic Modeling and Predictive Control in Solid Oxide Fuel Cells*. Wiley, 2012.

Kiel, S. UniVerMec – *A Framework for Development, Assessment, Interoperable Use of Verfied Techniques; with Applications in Distance Computation, Global Optimization, and Comparison Systematics*. PhD thesis, University of Duisburg-Essen, 2014.

Kiel, S., E. Auer, and A. Rauh. Uses of GPU Powered Interval Optimization for Parameter Identificationin the Context of SO Fuel Cells, in *Procedings of NOLCOS 2013*, pages 558–563, 2013.

Lohner, R. On the Ubiquity of the Wrapping Effect in the Computation of Error Bounds, in U. Kulisch, R. Lohner, A. Facius (Editors), *Perspectives on Enclosure Methods*, pages 201–218, Springer-Verlag, 2001.

Moore, R. E., R. B. Kearfott, and M. J. Cloud. Introduction to Interval Analysis. Society for Industrial and Applied Mathematics, Philadelphia, 2009.

Nedialkov, N. S. The Design and Implementation of an Object-Oriented Validated ODE Solver. Kluwer Academic Publishers, 2002.

Oberkampf, W. L., T. G. Trucano, and C. Hirsch: Verification, Validation, and Predictive Capability in Computational Engineering and Physics. Technical Report SAND2003-3769, Sandia National Laboratories, 2003.

Pusch, T. Umsetzung einer verifizierten Parameteridentifikation mit GUI Realisierung für Festoxidbrennstoffzellen, Master's thesis, Universität Duisburg-Essen, November 2013.

Rauh, A., L. Senkel, and H. Aschemann. Interval-Based Sliding Mode Control Design for Solid Oxide Fuel Cells with State and Actuator Constraints, in *IEEE Transactions on Industrial Electronics*, 2015.

Rauh, A., L. Senkel, E. Auer, and H. Aschemann. Interval Methods for the Implementation of Real-Time Capable Robust Controllers for Solid Oxide Fuel Cell Systems, *Mathematics in Computer Science*, 8(3–4):525–542, 2014.

Rauh, A., L. Senkel, J. Kersten, and H. Aschemann. Reliable Control of High-Temperature Fuel Cell Systems using Interval-Based Sliding Mode Techniques, *IMA Journal of Mathematical Control and Information*, 2014.

Saltelli, A., M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. Global Sensitivity Analysis: The Primer. Wiley, 2008

Stauning, O. Automatic Validation of Numerical Solutions. PhD thesis, September 1997.

Wächter, A. and L.T. Biegler. On the Implementation of an Interior-point Filter Line-search Algorithm for Large-scale Nonlinear Programming, *Math. Program.*, 106(1):25–57, 2006.