Parameter Identification in One-Dimensional Partial Differential Algebraic Equations

K. Schittkowski[1]

Abstract

In this paper we discuss a couple of situations, where algebraic equations are to be attached to a system of one-dimensional partial differential equations. Besides of models leading directly to algebraic equations because of the underlying practical background, for example in case of steady-state equations, there are many others where the specific mathematical structure requires a certain reformulation leading to time-independent equations. To be able to apply our approach to a large class of real-life problems, we have to take into account flux formulations, constraints, switching points, different integration areas with transition conditions, and coupled ordinary differential algebraic equations (DAEs), for example. The system of partial differential algebraic equations (PDAEs) is discretized by the method of lines leading to a large system of differential algebraic equations which can be solved by any available implicit integration method. Standard difference formulas are applied to discretize first and second partial derivatives, and upwind formulae are used for transport equations. Proceeding from given experimental data, i.e., observation times and measurements, the minimum least squares distance of measured data from a fitting criterion is computed, which depends on the solution of the system of PDAEs. Parameters to be identified can be part of the differential equations, initial, transition, or boundary conditions, coupled DAEs, constraints, fitting criterion, etc. Also the switching points can become optimization variables. The resulting least squares problem is solved by an adapted sequential quadratic programming (SQP) algorithm which retains typical features of a classical Gauss-Newton method by retaining robustness and fast convergence speed of SQP methods. The mathematical structure of the identification problems is outlined in detail, and we present a number of case studies to illustrate the different model classes which can be treated by our approach.

Keywords: parameter estimation, data fitting, least squares optimization,partial differential algebraic equations, method of lines

[1]Department of Computer Science, University of Bayreuth, 95440 Bayreuth, Germany

1 Introduction

Parameter estimation or data fitting, respectively, is applied in practical situations, where a mathematical model is available to simulate and predict the dynamical structure of the system. The idea is compute unknown model parameters by minimizing the distance of a fitting function from experimentally observed data.

In this paper, we consider one-dimensional partial differential algebraic equations (PDAEs) with optional flux functions, coupled ordinary differential algebraic equations, arbitrary fitting criteria, disjoint spatial integration areas, switching times, and dynamical constraints. Despite of the restriction that only one-dimensional spatial variables are considered, the mathematical model is fairly general and broad from the practical point of view. But even if the underlying mathematical formulation requires a two- or three-dimensional simulation, it is often reasonable to estimate first some unknown model parameters by a least squares fit for a simplified one-dimensional variant.

Partial differential algebraic equations (PDAEs) are based on the same model structure as one-dimensional, time-dependent partial differential equations. The only difference is that additional algebraic equations are permitted as in the case of ordinary differential algebraic equations (DAEs). We proceed from the general explicit formulation

$$\begin{aligned} \frac{\partial u_d}{\partial t} &= F_d(p, u, u_x, u_{xx}, x, t) \ , \\ 0 &= F_a(p, u, u_x, u_{xx}, x, t) \ , \end{aligned} \tag{1}$$

where $x \in \mathbb{R}$ is the spatial variable with $x_L \leq x \leq x_R$, and t the time variable with bounds of the form $0 \leq t \leq T$. $p \in \mathbb{R}^n$ is an unknown parameter vector we would like to identify by fitting experimental data.

The state variables are divided into n_d so-called differential variables $u_d = (u_1, \ldots, u_{n_d})^T$ and n_a algebraic variables $u_a = (u_{n_d+1}, \ldots, u_{n_d+n_a})^T$, where the number of algebraic variables is identical to the number of algebraic equations summarized by the vector F_a. The dynamical system (1) is also written in equivalent form

$$\begin{aligned} \frac{\partial u_1}{\partial t} &= F_1(p, u, u_x, u_{xx}, x, t) \ , \\ &\cdots \\ \frac{\partial u_{n_d}}{\partial t} &= F_{n_d}(p, u, u_x, u_{xx}, x, t) \ , \\ 0 &= F_{n_d+1}(p, u, u_x, u_{xx}, x, t) \ , \\ &\cdots \\ 0 &= F_{n_d+n_a}(p, u, u_x, u_{xx}, x, t) \ , \end{aligned} \tag{2}$$

if we consider the individual coefficient functions $F = (F_1, \ldots, F_{n_d+n_a})^T$ and $u = (u_d, u_a) = (u_1, \ldots, u_{n_d+n_a})^T$. We denote the solution of (2) by $u(p, x, t)$, since it depends on the time value t, the space value x, and the actual parameter p.

Initial values and boundary conditions may depend on the parameter vector to be estimated. Since the starting time is assumed to be zero, initial values have the form

$$u(p, x, 0) = u_0(p, x) \tag{3}$$

and are defined for all $x \in [x_L, x_R]$. For both end points x_L and x_R we allow Dirichlet or Neumann boundary values

$$\begin{aligned} u(p, x_L, t) &= u^L(p, t) \ , \\ u(p, x_R, t) &= u^R(p, t) \ , \\ u_x(p, x_L, t) &= \hat{u}^L(p, t) \ , \\ u_x(p, x_R, t) &= \hat{u}^R(p, t) \end{aligned} \tag{4}$$

for $0 < t \leq T$, where T is the final integration time, for example the last experimental time value. The availability of all boundary functions is of course not required. Their particular choice depends on the structure of the PDAE model, for example whether second partial derivatives exist on the right-hand side or not. A *rule-of-thumb* says that the number of partial derivatives should be identical to the number of initial and boundary equations.

However, we must treat initial and boundary conditions with more care. We have to guarantee that at least existing boundary conditions satisfy the algebraic equations, for example

$$\begin{aligned} 0 &= F_a(p, u(p, x_L, t), u_x(p, x_L, t), u_{xx}(p, x_L, t), x_L, t) \ , \\ 0 &= F_a(p, u(p, x_R, t), u_x(p, x_R, t), u_{xx}(p, x_R, t), x_R, t) \ , \end{aligned} \tag{5}$$

where u is the combined vector of all differential and algebraic state variables. If the initial values

$$0 = F_a(p, u(p, x, 0), u_x(p, x, 0), u_{xx}(p, x, 0), x, 0) \tag{6}$$

for the discretized algebraic equations are violated after inserting corresponding approximations for spatial derivatives, the corresponding system of nonlinear equations must be solved numerically. In other words, consistent initial values can be computed automatically, where given data serve as starting parameters for the nonlinear programming algorithm applied.

But even if we succeed in finding consistent initial values for (1) *by hand*, we have to take into account that the spatial derivatives of the dynamical equation (6) are approximated numerically by the method of lines and suitable difference formulae. The corresponding discretized equations of the DAE system are in general not consistent, or, more precisely, are satisfied only within the given discretization accuracy. Thus, we have to assume that the resulting DAE is an index-1-system unless it is guaranteed that consistent initial values for the discretized DAE are available, see for example Caracotsios and Stewart [8] for a similar approach.

A complex practical application is a model for molten carbonate fuel cells with e few second-order temperature equations and about 20 additional transportation equations, see Heidebrecht and Sundmacher [24] or Pesch et el. [34]. Another application from chemical engineering, the simulation model for an acetylene reactor, is given in form of a distributed system and consists of one temperature and nine reaction equations, all of first order, see Birk et al. [5].

Next we proceed from r experimental data sets (t_j, y_j^k), $j = 1, \ldots, l_t$, $k = 1, \ldots, r$, where l_t time values and $l = l_t r$ corresponding measurement values are defined. Then the objective function to be minimized is

$$\sum_{k=1}^{r} \sum_{j=1}^{l_t} (w_j^k (h_k(p, t_j) - y_j^k))^2 \quad . \tag{7}$$

The fitting criteria $h_k(p,t)$ depend now on so-called state variables, i.e., the solution of an implicitly defined system of partial differential algebraic equations. w_j^k are suitable weights by which the influence of measurement y_j^k on the optimal parameter set is influenced.

To indicate that the fitting criteria $h_k(p,t)$ depend on the solution of the dynamical equation at the corresponding fitting point and its derivatives, where k denotes the index of a measurement set, we use the notation

$$h_k(p,t) = \overline{h}_k(p, u(p, x_k, t), u_x(p, x_k, t), u_{xx}(p, x_k, t), t) \quad . \tag{8}$$

Each set of experimental data is assigned a spatial variable value $x_k \in [x_L, x_R]$, $k = 1, \ldots, r$, where r denotes the total number of measurement sets. Some or all of the x_k-values may coincide, if different measurement sets are available at the same local position. Since partial differential equations are discretized by the method of lines, the fitting points x_k are rounded to the nearest line.

By defining now $f_i(p) := w_j^k (h_k(p, t_j) - y_j^k)$ where the index i runs from 1 to $l := r l_t$ in any order, we obtain the constrained least squares problem

$$\begin{array}{lll} & \min \sum_{i=1}^{l} f_i(p)^2 & \\ p \in R^n : & g_j(p) = 0 \ , & j = 1, ..., m_e \ , \\ & g_j(p) \geq 0 \ , & j = m_e + 1, ..., m \ , \\ & p_l \leq p \leq p_u \ . & \end{array} \tag{9}$$

We assume that the parameter vector p is n-dimensional and that the nonlinear functions by which (9) is defined, are continuously differentiable with respect to p, since all efficient numerical algorithms are based more or less on the Gauss-Newton method and require first derivatives. Upper and lower bounds are treated independently from the remaining constraints. The fitting criteria $f_i(p)$ are supposed to depend also on the solution of a system of one-dimensional partial differential equations.

Possible extensions of the general formulation are discussed in Section 2 in more detail to cover a broad domain of possible applications, for example flux functions, coupled ordinary differential algebraic equations, arbitrary fitting criteria, disjoint spatial integration areas, switching times, and dynamical constraints. Section 3 contains a brief summary of the numerical procedures that are useful to discretize the partial differential algebraic equations by the method of lines, to integrate the resulting system of ordinary differential algebraic equations, and to solve the constrained least squares problem efficiently. Only some basic features of the underlying ideas are presented. More details are found in the references and especially in Schittkowski [50]. Some case studies are outlined in Section 4 to illustrate that despite of the fixed model structure as shown above, we are able to solve also many other identification problems by formulating suitable algebraic equations,

1. distributed systems,
2. stationary systems,
3. higher order spatial derivatives,
4. general boundary conditions,

The purpose is to motivate the necessity for investigating algebraic partial differential equations and to show how special model classes are successfully treated as systems of PDAEs after some straightforward transformations.

A subset of 43 PDAEs is listed in Section 5, which are part of a large database of 1,170 data fitting problems with name EASY-FIT and which can be downloaded from the home page of the author,

`http://www.klaus-schittkowski.de/`

The software comes with data fitting software, which is based on the discretization techniques outlined in this paper, and allows repeating the numerical test runs, see Schittkowski [49, 50].

2 Extensions of the Model Structure

Practical application models are often more complex and do not fit precisely into the general model frame given by (1), (3), and (4). Some of them discussed in this section in more detail, require also alternative discretization techniques and specialized solvers. But the goal is to show that all of these additional options can be combined into one unified algorithm leading to the same constrained least squares problem (9).

2.1 Flux Functions

Flux functions facilitate the definition of the right-hand side of a partial differential equation, and to allow the usage of special discretization formulae in situations where standard symmetric formulas break down. An example are hyperbolic equations with non-continuous boundary and initial conditions leading to the propagation of shocks through the integration interval. A typical example for the first case is the Richards equation

$$u_t = \frac{1}{c}\frac{\partial}{\partial x}\left(k(u_x - 1)\right)$$

with

$$\begin{aligned} c &= (n-1)(\theta_s - \theta_r)(1+u^n)^{1/n-2}u^{n-1} \quad , \\ k &= \frac{K_s(1-u^{n-1}(1+u^n)^{1/n-1})^2}{(1+u^n)^{(1-1/n)/2}} \end{aligned}$$

where the formulation of analytic derivatives *by hand* is cumbersome. On the other hand, flux functions are required for example to integrate systems of hyperbolic equations of the form

$$u_t + f_x(u) = 0 \quad ,$$

see Thomas [54] or Renardy and Rogers [41], for which a large variety of different discretization schemes is available.

We suppose that a flux function f depends on p, u, u_x, x, and t, i.e., f is of the form $f(p, u, u_x, x, t)$. It is then possible to attach them and together with their first partial derivatives to the right-hand side of an equation. Again we proceed from a system of n_d differential and n_a algebraic equations in explicit formulation (1), where the state variables consist of n_d differential variables u_d and n_a algebraic variables u_a, $u = (u_d, u_a)^T$. It is supposed that $f(p, u, u_x, x, t)$ and $f_x(p, u, u_x, x, t)$ are of the same dimension, and we get the extended dynamical system

$$\begin{aligned} \frac{\partial u_d}{\partial t} &= F_d(p, f(p, u, u_x, x, t), f_x(p, u, u_x, x, t), u, u_x, u_{xx}, x, t) \ , \\ 0 &= F_a(p, f(p, u, u_x, x, t), f_x(p, u, u_x, x, t), u, u_x, u_{xx}, x, t) \ , \end{aligned} \tag{10}$$

where $x \in \mathbb{R}$ is the spatial variable with $x_L \leq x \leq x_R$, and $0 < t \leq T$. Initial and boundary conditions are the same as in (3) and (4).

2.2 Coupled Ordinary Differential Algebraic Equations

A particular advantage of applying the method of lines is the possibility of coupling additional ordinary differential algebraic equations to the given partial ones easily. We proceed from the general explicit formulation

$$\begin{aligned} \frac{\partial u_d}{\partial t} &= F_d(p, f(p, u, u_x, x, t), f_x(p, u, u_x, x, t), u, u_x, u_{xx}, u, u_x, u_{xx}, v, x, t) \ , \\ 0 &= F_a(p, f(p, u, u_x, x, t), f_x(p, u, u_x, x, t), u, u_x, u_{xx}, u, u_x, u_{xx}, v, x, t) \ , \end{aligned} \tag{11}$$

see (10), where $x \in \mathbb{R}$ is the spatial variable with $x_L \leq x \leq x_R$, $0 < t \leq T$, and where we add the state variable $v \in \mathbb{R}^{n_c}$ of a system of ordinary differential algebraic equations. Initial values and boundary conditions are the same as in the previous section, (3) and (4),

$$u(p, x, 0) = u_0(p, x) \tag{12}$$

to be satisfied for all $x \in (x_L, x_R)$, and

$$\begin{aligned} u(p, x_L, t) &= u^L(p, v, t) \ , \\ u(p, x_R, t) &= u^R(p, v, t) \ , \\ u_x(p, x_L, t) &= \hat{u}^L(p, v, t) \ , \\ u_x(p, x_R, t) &= \hat{u}^R(p, v, t) \end{aligned} \tag{13}$$

for $0 < t \leq T$, where either Dirichlet or Neumann or any mixed boundary conditions must be defined. These boundary conditions may depend on the coupled differential and algebraic

variables, for example, if boundary conditions are given in the form of ordinary differential equations or in implicit form.

By distinguishing between differential and algebraic variables, $v = (v_d, v_a)^T \in \mathbb{R}^{n_c}$, we proceed from a system of DAEs

$$\begin{aligned}
\frac{\partial v_1}{\partial t} &= G_1(p, u(p, x_1, t), u_x(p, x_1, t), u_{xx}(p, x_1, t), v, t) \ , \\
&\dots \\
\frac{\partial v_{n_{dc}}}{\partial t} &= G_{n_{dc}}(p, u(p, x_{n_{dc}}, t), u_x(p, x_{n_{dc}}, t), u_{xx}(p, x_{n_{dc}}, t), v, t) \ , \\
0 &= G_{n_{dc}+1}(p, u(p, x_{n_{dc}+1}, t), u_x(p, x_{n_{dc}+1}, t), u_{xx}(p, x_{n_{dc}+1}, t), v, t) \ , \\
&\dots \\
0 &= G_{n_c}(p, u(p, x_{n_c}, t), u_x(p, x_{n_c}, t), u_{xx}(p, x_{n_c}, t), v, t) \ ,
\end{aligned} \tag{14}$$

where $u(p, x, t)$ is the state variable of the partial differential equation. x_j is any x-coordinate value where the corresponding ordinary differential or algebraic equation is to be coupled to the partial one, $j = 1, \dots, n_c$. Some of these values may coincide. When discretizing the system by the method of lines, they have to be rounded to the nearest neighboring line. Note that the partial and the ordinary differential equations must be integrated simultaneously.

Also initial values

$$v(p, 0) = v_0(p) \tag{15}$$

may depend again on the parameters to be estimated. A solution of the coupled system depends on the spatial variable x, the time variable t, the parameter vector p, and is therefore written in the form $v(p, t)$ and $u(p, x, t)$.

To indicate that also the fitting criteria $h_k(p, t)$ depend on the solution of the differential equation at the corresponding fitting point and its first and second spatial derivatives, we use the notation

$$h_k(p, t) = \overline{h}_k(p, u(p, x_k, t), u_x(p, x_k, t), u_{xx}(p, x_k, t), v(p, t), t) \ , \tag{16}$$

see (8). k denotes the index of a measurement set. Also fitting points x_k are rounded to their nearest line when discretizing the system.

Coupled ordinary differential equations can be used to define a fitting criterion, for example if the flux into or out of a system is investigated. A practical case study is the transdermal application of drugs which leads to four coupled ordinary differential equations, where the corresponding state variables are measured, see Schittkowski [47].

Another reason is that they allow replacing Dirichlet or Neumann boundary conditions by differential equations or general algebraic equations either at a boundary point or even inside of the spatial integration area.

2.3 Integration Areas and Transition Conditions

We further extend the model structure to allow different integration intervals in the x-direction. A possible application is the diffusion of a substrate through different media, where we want to describe the transition from one area to the next by special conditions. Since

these transition conditions may become non-continuous, we need a more general formulation and have to adapt the discretization procedure.

The general model is defined by a system of n_d one-dimensional partial differential equations and n_a algebraic equations in one or more spatial intervals, see also Schittkowski [46]. These intervals are given by the outer boundary values x_L and x_R that define the total integration interval for the space variable x, and optionally some additional internal transition points x_1^a, ..., $x_{m_a-1}^a$. Thus, we get a sequence of $m_a + 1$ boundary and transition points

$$x_0^a = x_L < x_1^a < \cdots < x_{m_a-1}^a < x_{m_a}^a = x_R \ . \tag{17}$$

For each integration interval, we have a system of partial differential equations

$$\begin{aligned} \frac{\partial u_d^i}{\partial t} &= F_d^i(p, f^i(p, u^i, u_x^i, x, t), f_x^i(p, u^i, u_x^i, x, t), u^i, u_x^i, u_{xx}^i, v, x, t) \ , \\ 0 &= F_a^i(p, f^i(p, u^i, u_x^i, x, t), f_x^i(p, u^i, u_x^i, x, t), u^i, u_x^i, u_{xx}^i, v, x, t) \ , \end{aligned} \tag{18}$$

where $x \in \mathbb{R}$ is the spatial variable with $x_{i-1}^a < x < x_i^a$ for $i = 1$, ..., m_a, $t \in \mathbb{R}$ the time variable with $0 < t \leq T$, $v \in \mathbb{R}^{n_c}$ the state variable of the coupled system of ordinary differential algebraic equations, $u^i = (u_d^i, u_a^i)^T \in \mathbb{R}^{n_p}$ the state variables consisting of the partial differential variables $u_d^i \in \mathbb{R}^{n_d}$ and partial algebraic variables $u_a^i \in \mathbb{R}^{n_a}$, and $p \in \mathbb{R}^n$ is the parameter vector to be identified by the data fitting algorithm.

Optionally, the right-hand side may also depend on a flux function $f^i(p, u^i, u_x^i, x, t)$, where we omit for simplicity a possible dependency from coupled ordinary differential equations. A solution depends on the spatial variable x, the time variable t, the parameter vector p, the corresponding integration interval, and is therefore written in the form $v^i(p, t)$ and $u^i(p, x, t)$ for $i = 1$, ..., m_a.

For both boundary points x_L and x_R we allow Dirichlet or Neumann conditions in more general form

$$\begin{aligned} u^1(p, x_L, t) &= u^L(p, v(t), t) \ , \\ u^{n_a}(p, x_R, t) &= u^R(p, v(t), t) \ , \\ u_x^1(p, x_L, t) &= \hat{u}^L(p, u^1(p, x_L, t), v(t), t) \ , \\ u_x^{n_a}(p, x_R, t) &= \hat{u}^R(p, u^{n_a}(p, x_R, t), v(t), t) \ , \end{aligned} \tag{19}$$

$0 < t \leq T$, to indicate that boundary information is also contained in coupled ordinary differential equations and that boundary fluxes may depend also on the state variable at the boundary. Again, we do not require the evaluation of all boundary functions. A user has to adopt them carefully depending on the mathematical structure of the dynamical model.

Transition conditions between different integration areas must be defined in addition. They are allowed at most at transition points and have the form

$$\begin{aligned} u^i(p, x_i^a, t) &= c_i^R(p, u^{i+1}(p, x_i^a, t), v(t), t) \ , \\ u^{i+1}(p, x_i^a, t) &= c_i^L(p, u^i(p, x_i^a, t), v(t), t) \ , \\ u_x^i(p, x_i^a, t) &= \hat{c}_i^R(p, u^{i+1}(p, x_i^a, t), u_x^{i+1}(p, x_i^a, t), v(t), t) \ , \\ u_x^{i+1}(p, x_i^a, t) &= \hat{c}_i^L(p, u^i(p, x_i^a, t), u_x^i(p, x_i^a, t), v(t), t) \end{aligned} \tag{20}$$

with $0 < t \leq t_s$, $i = 1, \ldots, m_a - 1$. A transition condition of the i-th area either in Dirichlet or Neumann form depends on the time variable, the parameters to be estimated, and the solution of the neighboring area. Again, the user may omit any of these functions, if a transition condition does not exist at a given x_i^a-value. More complex implicit boundary and transition conditions can be defined in the form of coupled algebraic equations.

Since the starting time is assumed to be zero, initial values must have the form

$$u^i(p, x, 0) = u_0^i(p, x) \quad , \quad i = 1, \ldots, m_a \tag{21}$$

and are defined for all $x \in \left[x_{i-1}^a, x_i^a\right]$, $i = 1, \ldots, m_a$. If initial values for algebraic variables are not consistent, i.e., do not satisfy the algebraic equations of (18), we have to apply Newton's method for solving the corresponding system of nonlinear equations.

If the partial differential equations are to be coupled to ordinary differential algebraic equations, we proceed from an additional DAE system of the form

$$\dot{v}_j = G_j(p, u^{i_j}(p, x_j, t), u_x^{i_j}(p, x_j, t), u_{xx}^{i_j}(p, x_j, t), v, t) \tag{22}$$

for $j = 1, \ldots, n_{dc}$, and

$$0 = G_j(p, u^{i_j}(p, x_j, t), u_x^{i_j}(p, x_j, t), u_{xx}^{i_j}(p, x_j, t), v, t) \tag{23}$$

for $j = n_{dc} + 1, \ldots, n_c$, see (14). Coupling of ordinary differential equations is possible at arbitrary points within the integration interval and the corresponding area is denoted by the index i_j. The spatial variable value x_j, some or all of which may coincide, belongs to the i_j-th area, $x_j \in [x_{i_j-1}^a, x_{i_j}^a)$ or $x_j \in [x_{m_a-1}^a, x_{m_a}^a]$, respectively, $j = 1, \ldots, n_c$, and is called a coupling point.

Coupling points are rounded to their nearest line when discretizing the system by the method of lines. The right-hand side of the coupling equation may depend on the corresponding solution of the partial differential equation and its first and second derivative subject to the space variable at the coupling point under consideration.

To indicate that the fitting criteria $h_k(p, t)$ also depend on the solution of the differential equation at the corresponding fitting point, where k denotes the index of an experimental data set, we use the notation

$$h_k(p, t) = \overline{h}_k(p, u^{i_k}(p, x_k, t), u_x^{i_k}(p, x_k, t), u_{xx}^{i_k}(p, x_k, t), v(p, t), t) \tag{24}$$

and insert $\overline{h}_k$ into the data fitting function (16). The fitting criteria may depend on solution at a given spatial variable value in an integration interval defined by the index i_k. The spatial variable x_k belongs to the i_k-th integration area,

$$x_k \in \left[x_{i_k-1}^a, x_{i_k}^a\right) \quad , \quad x_k \in \left[x_{m_a-1}^a, x_{m_a}^a\right] \quad ,$$

respectively, $k = 1, \ldots, r$, where r denotes the total number of experimental data sets. Fitting points are rounded to their nearest line when discretizing the system.

In principle, each integration area is treated as an individual boundary value problem and is discretized separately by the method of lines. The transition functions are treated in the same way as Dirichlet or Neumann boundary conditions, respectively.

2.4 Switching Points

There are many practical situations where model equations change during the integration over the time variable, and where corresponding initial values at the switching points must be adopted. A typical example is a pharmacokinetic application with an initial infusion and subsequent application of drug doses by injection. In this case, it is even possible that the solution becomes non-continuous at a switching or break point, respectively. We consider now the same model that was developed step-by-step in previous sections, a system of one-dimensional partial differential algebraic equations with flux functions, coupled ordinary differential algebraic equations, and different integration areas with transition conditions, see (18). We suppose that n_b break or switching points with

$$\tau_0 = 0 < \tau_1 < \ldots < \tau_{n_b} < \tau_{n_b+1} = T \tag{25}$$

are given, where T is the last experimental time value.

For the first time integration interval, the same initial, boundary, and transition values are given as before, see (21), (15), (19), and (20), respectively. For all subsequent intervals, however, the integration subject to the time variable is to be restarted at a switching point and new function values can be provided that may depend now also on the solution of the previous section. Initial values at a switching point are evaluated from

$$\begin{aligned} u^i(p,x,\tau_k) &= b_k^i(p,u_-^i(p,x,\tau_k),v_-(p,\tau_k),x) \ , \\ v(p,\tau_k) &= \tilde{b}(p,v_-(p,\tau_k)) \end{aligned} \tag{26}$$

for $i = 1, \ldots, m_a$ and $k = 1, \ldots, n_b$, where $u_-^i(p,x,\tau_k)$ and $v_-(p,\tau_k)$ denote the solution of the coupled PDAE system in the previous time interval at $t = \tau_k$.

Since the right-hand side of the partial differential equation (18) and also the corresponding boundary and transition functions depend on the time variable, they may change from one interval to the next. Particularly non-continuous transitions at switching points are allowed.

It is possible that break points become variables to be adapted during the optimization process. However, there exists a very dangerous situation when a variable switching point passes or approximates a measurement time value during an optimization run. If both coincide and if there is a non-continuous transition, then the underlying model function is no longer differentiable with respect to the parameters to be optimized. Possible reactions of the least squares algorithm are slow final convergence rates or break downs because of internal numerical difficulties. On the other hand, variable switching points are very valuable when trying to model for example the input feed of chemical or biological processes given by a bang-bang control function or any other one with variable break points.

2.5 Dynamical Constraints

It is often reasonable to define dynamical constraints, where the restriction functions depend on the solution of the partial differential equation and its first and second spatial derivatives

at predetermined time and spatial values, and the solution of coupled ordinary differential algebraic equations,

$$g_j(p) = \overline{g}_j(p, u^{i_j}(p, x_j, t_{k_j}), u_x^{i_j}(p, x_j, t_{k_j}), u_{xx}^{i_j}(p, x_j, t_{k_j}), v(p, t_{k_j}), t_{k_j}) \tag{27}$$

for $j = m_e + 1, \ldots, m_r$. Here the index i_j denotes the corresponding integration area that contains the spatial parameter x_j rounded to its nearest line, and k_j the corresponding experimental time where a restriction is to be formulated. Thus, constraints are evaluated only at certain lines and experimental time values. If they are required also at some intermediate points, one has to increase the number of lines or the number of experimental data with zero weights. Only inequality constraints are considered, since equality constraints should be declared in the form of algebraic equations.

Equation (27) is the discretized form of the dynamical constraints we have in mind. In a more general context, our intention is to limit certain functions depending on the state variable for all time and/or spatial variable values,

$$\overline{g}_j(p, u(p, x, t), u_x(p, x, t), u_{xx}(p, x, t), v(p, t), x, t) \geq 0 \tag{28}$$

or

$$\overline{g}_j(p, u^{i_j}(p, x_j, t), u_x^{i_j}(p, x_j, t), u_{xx}^{i_j}(p, x_j, t), v(p, t), t) \geq 0 \tag{29}$$

or

$$\overline{g}_j(p, u(p, x, t_j), u_x(p, x, t_j), u_{xx}(p, x, t_j), v(p, t_j), x) \geq 0 \quad , \tag{30}$$

respectively, for $j = m_e + 1, \ldots, m_r$, $x \in (x_L, x_R)$, and $t \geq 0$. All these constraints can be mixed with time-independent parameter constraints.

3 Numerical Methods

A widely used idea is to transform one-dimensional partial differential equations into a system of ordinary differential algebraic equations by discretizing the model functions subject to the spatial variable x. This approach is known as the numerical method of lines, see for example Schiesser [42]. For the i-th integration interval of the spatial variable, we define a uniform grid and get a discretization of the whole space interval from x_L to x_R. To approximate the first and second partial derivatives of $u(p, x, t)$ subject to the spatial variable at a given point x_k, $k = 1, \ldots, n_g$, several different alternatives can be implemented, see Schittkowski [48, 50] for more details.

In this section, several difference schemes are considered for approximating first derivatives, see for example Smith [53] for a deeper treatment. Second derivatives are computed either by successive application of a first-order scheme, or directly by a special difference scheme for second derivatives. We assume that the spatial variable x is again discretized as outlined in the previous section, $u_i(t) = u(x_i, t)$ for $i = 1, \ldots, n$, where

$$x_i = x_L + \frac{i-1}{n-1}(x_R - x_L) \quad .$$

In case of algebraic differential equations, boundary conditions have to satisfy the algebraic equations. Consistent initial values are computed internally, where some data must

be given to serve as starting parameters for the nonlinear programming algorithm. Consequently, we allow only index-1-systems unless it is guaranteed, that consistent initial values for the discretized DAE are available.

It is possible that the right-hand side of a PDAE becomes non-continuous subject to integration time. Thus, it is necessary to supply time values and corresponding initial values depending on the solution of the previous interval, where the integration of the DAE is to be restarted with initial tolerances, for example with the initially given stepsize. The integration in the proceeding interval is stopped at the time value given minus a relative error in the order of the machine precision. Break or switching points are either constant or optimization variables to be adapted by the optimization code.

It can be shown that the resulting large system of ordinary differential equations becomes stiff in some situations, when discretization accuracy increases. Thus, the usage of implicit solvers is recommended. Since the Jacobian of the discretized right-hand side has a band structure, it is essential that the selected method is capable to exploit sparsity efficiently.

To simplify the notation, we neglect any dependencies on an optimization parameter p as long as we discuss the discretization methods for systems of PDAEs.

3.1 Second-Order Formulae for First Derivatives

A very simple approach is to approximate first derivatives $u_x(x,t)$ by a second-order formula. Using the discretized values $u_{i-1}(t)$, $u_i(t)$ and $u_{i+1}(t)$ defined on a uniform grid, we get the difference formula

$$u_x(x_i,t) \simeq \frac{1}{2h}\Big(u_{i+1}(t) - u_{i-1}(t)\Big) \tag{31}$$

with $h = 1/(n-1)$ and $i = 2, \ldots, n-1$. The formula must be adapted at the left boundary of the integration area. Using $u_1(t)$, $u_2(t)$, and $u_3(t)$, we obtain

$$u_x(x_1,t) \simeq \frac{1}{2h}\Big(-3u_1(t) + 4u_2(t) - u_3(t)\Big) \ . \tag{32}$$

By a symmetric formula, the derivatives at the right boundary are approximated.

3.2 Fourth-Order Formulae for First Derivatives

To improve the approximation order, we can also use five successive grid points $u_{i-2}(t)$, $u_{i-1}(t)$, $u_i(t)$, $u_{i+1}(t)$, and $u_{i+2}(t)$ for approximating the first derivative $u_x(x_i,t)$. A straightforward analysis shows that a fourth-order formula is given by

$$u_x(x_i,t) \simeq \frac{1}{4!h}\Big(2u_{i-2}(t) - 16u_{i-1}(t) + 16u_{i+1}(t) - 2u_{i+2}(t)\Big) \tag{33}$$

for $i = 3, \ldots, n-2$, see Schiesser [42]. The corresponding approximations at the left boundary point are

$$\begin{aligned} u_x(x_1,t) &\simeq \frac{1}{4!h}\Big(-50u_1(t) + 96u_2(t) - 72u_3(t) + 32u_4(t) - 6u_5(t)\Big) \ , \\ u_x(x_2,t) &\simeq \frac{1}{4!h}\Big(-6u_1(t) - 20u_2(t) + 36u_3(t) - 12u_4(t) + 2u_5(t)\Big) \ , \end{aligned} \tag{34}$$

and we get similar formulae for the right boundary point. All these formulae are of fourth-order and the approximation of second derivatives $u_{xx}(x,t)$ can be computed by recursive application of formulae (33) and (34).

3.3 Fourth-Order Formulae for Second Derivatives

Alternatively, it is also possible to approximate second derivatives directly, for example by the five-point-difference formula

$$u_{xx}(x_i,t) \simeq \frac{1}{4!h^2}\Big(-2u_{i-2}(t) + 32u_{i-1}(t) - 60u_i(t) + 32u_{i+1}(t) - 2u_{i+2}(t)\Big) \tag{35}$$

for $i = 3, \ldots, n-2$. Corresponding approximation at the left boundary is

$$\begin{aligned} u_{xx}(x_2,t) \;\simeq\; & \frac{1}{4!h^2}\Big(20u_1(t) - 30u_2(t) - 8u_3(t) + 28u_4(t) - 12u_5(t) \\ & +2u_6(t)\Big) \;. \end{aligned} \tag{36}$$

Depending on the type of boundary condition, we use either the formulae

$$\begin{aligned} u_{xx}(x_1,t) \;\simeq\; & \frac{1}{4!h^2}\Big(\tfrac{-415}{3}u_1(t) + 192u_2(t) - 72u_3(t) + \tfrac{64}{3}u_4(t) - 3u_5(t) \\ & -100\hat{u}^L(t)\Big) \end{aligned} \tag{37}$$

in the case of a Neumann boundary condition or

$$\begin{aligned} u_{xx}(x_1,t) \;\simeq\; & \frac{1}{4!h^2}\Big(90u^L(t) - 308u_2(t) + 428u_3(t) - 312u_4(t) \\ & +122u_5(t) - 20u_6(t)\Big) \end{aligned} \tag{38}$$

in the case of a Dirichlet boundary condition. Again similar formulae are found vice versa at the right boundary. For a more detailed outline of difference formulae, see Schiesser [42].

3.4 The First-Order Upwind Scheme

Now we suppose that the formulae discussed in the previous sections fail or show an irregular behavior. This situation arises, in particular, if initial shocks at the boundary propagate through the interior of the integration area.

For simplicity, we assume that there is only one scalar first-order equation of the form

$$u_t + f_x(u) = 0 \tag{39}$$

with a flux function $f(u)$ together with suitable initial conditions $u(x,0) = u_0(x)$ and left Dirichlet boundary condition $u(x_L,t) = u^L(t)$. The above formulation is usually derived from conservation or transportation laws.

The above equation is approximated in the form

$$\dot{u}_i + \frac{1}{h}(F_{i+1/2} - F_{i-1/2}) = 0 \quad ,$$

where $F_{i+1/2}$ is supposed to be an integral of $f_x(u)$ at midpoint $x_{i+1/2} = 0.5(x_{i+1} + x_i)$ for $i = 1, \ldots, n-1$. In the following, we will discuss some of the most simple formulae for evaluating $F_{i+1/2}$ by shock-capturing schemes. More of the classical diffusive and dispersive schemes and the one-parameter family of TVD schemes are found in Chakravarthy and Osher [9, 10], or Yang [59], respectively, see also Schittkowski [50].

The interface flux $F_{i+1/2}$ is now defined by

$$F_{i+1/2} = \frac{1}{2}(f_{i+1} + f_i) - \frac{1}{2}|a_{i+1/2}|\triangle_{i+1/2} \quad , \tag{40}$$

where $\triangle_{i+1/2} = u_{i+1} - u_i$. Then a wave speed $a_{i+1/2}$ is computed from

$$a_{i+1/2} = \begin{cases} (f_{i+1} - f_i)/\triangle_{i+1/2} \quad , & \text{if } \triangle_{i+1/2} \neq 0 \quad , \\ \\ (f_i)_u \quad , & \text{otherwise} \quad , \end{cases} \tag{41}$$

also denoted the Roe speed, and can be interpreted as a measure for the *wind* direction. Equation (40) does not satisfy the so-called entropy condition, see Chakravarthy et al. [9] and Yee [60] for details, $|a_{i+1/2}|$ is often replaced by

$$\varphi(a_{i+1/2}) = \max(|a_{i+1/2}|, \delta) \tag{42}$$

with a small positive number δ.

Particular upwind formulae are obtained if it is known that $a_{i+1/2} > 0$ and $a_{i-1/2} > 0$, or, alternatively, if $a_{i+1/2} < 0$ and $a_{i-1/2} < 0$. In the first case, we get $F_{i+1/2} = f_i$ and $F_{i-1/2} = f_{i-1}$, leading to

$$\dot{u}_i + \frac{1}{h}(f_i - f_{i-1}) = 0 \quad ,$$

in the second

$$\dot{u}_i + \frac{1}{h}(f_{i+1} - f_i) = 0 \quad .$$

However, if a differential equation is not given in hyperbolic form (39), we have still the possibility to approximate u_x directly by upwind formulae, i.e., either in the form

$$u_x(x_i, t) \simeq \frac{1}{h}\Big(u_i(t) - u_{i-1}(t)\Big) \tag{43}$$

or

$$u_x(x_i, t) \simeq \frac{1}{h}\Big(u_{i+1}(t) - u_i(t)\Big) \quad , \tag{44}$$

respectively.

3.5 Solving Systems of Ordinary Differential Algebraic Equations

We assume now that the system of differential algebraic equations obtained after a suitable discretization of the PDAE, is explicitly given in the general form

$$\begin{aligned} \dot{y}_1 &= F_1(y,z,t) \ , \quad y_1(0) = y_1^0 \ , \\ &\cdots \\ \dot{y}_{s_1} &= F_{s_1}(y,z,t) \ , \quad y_{s_1}(0) = y_{s_1}^0 \ , \\ 0 &= G_1(y,z,t) \ , \quad z_1(0) = z_1^0 \ , \\ &\cdots \\ 0 &= G_{s_2}(y,z,t) \ , \quad z_{s_2}(0) = z_{s_2}^0 \ . \end{aligned} \tag{45}$$

We are looking for a simultaneous solution given by the differential variables $y(t)$ and the algebraic variables $z(t)$ at time t. Moreover, let y_0 and z_0 be given initial values. Without loss of generality, we assume that the initial time from where the integration is to be started, is zero. s_1 is the number of differential equations or differential state variables, respectively, and s_2 the number of algebraic equations or algebraic state variables of the DAE (45).

We have to consider in more detail the question, how to treat the algebraic equations $G(y,z,t) = 0$, where $G(y,z,t) := (G_1(y,z,t), \ldots, G_{s_2}(y,z,t))^T$. Obviously, we should not expect that the algebraic variables z can be eliminated directly from these equations, unless we know that the matrix $\nabla_z G(y,z,t)$ possesses full rank. Then the above system of s_2 equations and s_2 unknowns z is solvable. We could compute a solution $z(y,t)$ and insert it into the remaining differential equations to get

$$\begin{aligned} \dot{y}_1 &= F_1(y,z(y,t),t) \ , \quad y_1(0) = y_1^0 \ , \\ &\cdots \\ \dot{y}_{s_1} &= F_{s_1}(y,z(y,t),t) \ , \quad y_{s_1}(0) = y_{s_1}^0 \ . \end{aligned} \tag{46}$$

If, however, the computation of $z(y,t)$ is not possible in analytical form or if we want to avoid time-consuming iterative algorithms, there is still another way to handle algebraic equations. Since $G(y,z,t) = 0$ is to be valid for all $t \geq 0$, we differentiate the equation subject to t and get an additional implicit ordinary differential equation

$$\nabla_y G(y,z,t)^T \dot{y} + \nabla_z G(y,z,t)^T \dot{z} + \frac{\partial}{\partial t} G(y,z,t) = 0$$

or an equivalent explicit one

$$\dot{z} = -\nabla_z G(y,z,t)^{-T} (\nabla_y G(y,z,t)^T F(y,z,t) + \frac{\partial}{\partial t} G(y,z,t)) \ . \tag{47}$$

Together with (46) we get a system of ordinary differential equations that can be solved by any available algorithm. The initial values z_0 should be chosen so that the algebraic

constraint (45) is satisfied at $t = 0$. Since we need one differentiation to transform the DAE into an ODE, we say that the differential algebraic equation has index 1.

If coupled algebraic variables violate the algebraic equations after inserting initial conditions (15) and a suitable discretization of the partial derivatives u_x and u_{xx}, Newton's method can be applied to compute consistent initial values. The equations are added to the algebraic partial differential equations and the whole system of nonlinear equations must be solved simultaneously.

I practice, it is possible to apply special implicit solvers which are able to directly take algebraic equations into account. One example is the implicit Runge-Kutta method of Radau-type, confer Hairer and Wanner [22]. The corresponding code is available under the name RADAU5. Another alternative is the usage of the code DASSL of Petzold [35].

3.6 Solving Constrained Least Squares Problems

Finally, the resulting least squares problem (9) can be solved by any of the available standard solvers, see for example Dennis et al. [14, 15], Gill et al. [18], or Schittkowski [44]. The algorithms discussed in these references, are based on the Gauss-Newton method and require first derivatives of the fitting criteria subject to the parameters to be estimated.

The goal is to minimize the sum of squares of distances of a certain model function from experimental measurement values. However, we are not able to exploit this specific structure mathematically. Instead, we write the parameter estimation problem in the form of a least squares problem, where a sum of squared nonlinear functions is to be minimized,

$$\begin{aligned} &\min \ \tfrac{1}{2} \sum_{i=1}^{l} \ f_i(p)^2 \\ &p \in \mathbb{R}^n \ \ . \end{aligned} \tag{48}$$

These problems possess a long history in mathematical programming and are extremely important in practice, particularly in nonlinear data fitting or maximum likelihood estimation. In consequence, a large number of mathematical algorithms is available for solving (48).

To understand their basic features, we introduce the notation

$$F(p) = (f_1(p), \ldots, f_l(p))^T$$

for the objective function vector, and let $f(p) = \frac{1}{2} \sum_{i=1}^{l} \ f_i(p)^2$. Then

$$\nabla f(p) = \nabla F(p) F(p) \tag{49}$$

defines the Jacobian of the objective function with $\nabla F(p) = (\nabla f_1(p), \ldots, \nabla f_l(p))$. If we assume now that all functions f_1, ..., f_l are twice continuously differentiable, we get the Hessian matrix of f

$$\nabla^2 f(p) = \nabla F(p) \nabla F(p)^T + B(p) \ \ , \tag{50}$$

where

$$B(p) = \sum_{i=1}^{l} f_i(p) \nabla^2 f_i(p) \ \ . \tag{51}$$

Proceeding from a given iterate p_k, Newton's method can be applied to (48) to get a search direction $d_k \in \mathbb{R}^n$ by solving the linear system $\nabla^2 f(p_k)d + \nabla f(p_k) = 0$ or, alternatively,

$$\nabla F(p_k)\nabla F(p_k)^T d + B(p_k)d + \nabla F(p_k)F(p_k) = 0 \quad . \tag{52}$$

Assume that $F(p^\star) = 0$ at an optimal solution $p^\star$. Then we neglect matrix $B(p_k)$ and (52) defines the so-called normal equations of the linear least squares problem

$$\begin{aligned} &\min \ \|\nabla F(p_k)^T d + F(p_k)\| \\ &d \in \mathbb{R}^n \quad . \end{aligned} \tag{53}$$

A new iterate is obtained by $p_{k+1} = p_k + \alpha_k d_k$, where d_k is a solution of (53) and where α_k denotes a suitable steplength parameter. It is obvious that a quadratic convergence rate is achieved when starting sufficiently close to an optimal solution. The above calculation of a search direction is known as the Gauss-Newton method and represents the traditional way to solve nonlinear least squares problems, see Björck [6] for more details.

However, the assumptions guaranteeing convergence of Gauss-Newton methods are very strong and cannot be satisfied in real situations. We have to expect difficulties in case of non-zero residuals, rank-deficient Jacobian matrices, non-continuous derivatives, and starting points far away from a solution. Numerous modifications have been proposed in the past, for example the Levenberg-Marquardt algorithm, see Levenberg [27] and Marquardt [32], where the key idea is to replace the Hessian in (52) by a multiple of the identity matrix, say $\lambda_k I$, with a suitable positive factor λ_k. For further reviews see Gill, Murray and Wright [19] or Ramsin and Wedin [39]. Lindström [29] proposed a combination of a Gauss-Newton and a Newton method by using a certain subspace minimization technique.

Another possibility is to apply a well-known sequential quadratic programming (SQP) method, which are widely used in nonlinear programming applications. By transforming the original problem into a general nonlinear programming problem in a special way, typical features of a Gauss-Newton and quasi-Newton least squares method are retained, see Schittkowski [45] for details. A particular advantage is that additional constraints are easily taken into account. The resulting optimization problem is solved by any available SQP code, for example by NLPQL of Schittkowski [44, 52].

4 Case Studies

The efficient implementation of a PDAE model and the integration within an identification environment is time-consuming. Thus, one would like to have the possibility to solve also a wider class of problems by the same code although knowing that special purpose methods are available and often perform much better. We present a few situations, which can be transformed into or considered as a system of one-dimensional PDAEs. Only the underlying model equations are investigated. Identifiers of typical examples to find more details, are shown in brackets. Corresponding data, fitting criteria, references etc. of the corresponding parameter estimation problems are found in Table 1 and can be downloaded from the home page of the author, see Section 5 for details. To simplify the notation, we omit the parameter variable p

4.1 Distributed Systems (ACETYL_T, ACETYL_Z)

Partial differential equations of order one are sometimes called distributed parameter systems, especially in chemical engineering. A typical situation is described by

$$\begin{aligned} u_t &= F_1(u, v, x, t) \ , \\ v_x &= F_2(u, v, x, t) \end{aligned}$$

with initial values $u(x, 0) = u_0(x)$, $v(0, t) = v_0(t)$ are transformed into the PDAEs

$$\begin{aligned} u_t &= F_1(u, v, x, t) \ , \\ 0 &= v_x - F_2(u, v, x, t) \end{aligned}$$

or

$$\begin{aligned} v_x &= F_2(u, v, x, t) \ , \\ 0 &= u_x - F_1(u, v, x, t) \ , \end{aligned}$$

respectively. Distributed systems often arise in chemical engineering, see for example the acetylene reactor discussed by Birk et al. [5] in more detail.

4.2 Stationary Systems (2ND_DIR1, 2ND_DIR2, CAPILL)

If there are no time-derivatives, we get a steady-state or stationary system of the form

$$F(u, u_x, u_{xx}, x, t) = 0 \ ,$$

which is identical to an implicit system of second order ordinary differential equations. By defining a zero or a sufficiently small time integration interval, the solver is stopped immediately after computing consistent initial values. Since we have to define additional boundary conditions, we consider this situation also as a boundary-value problem. The drawback is, that the spatial stepsize is fixed and that the discretization accuracy is at most four when applying the approximation schemes of the previous section.

4.3 Higher Order Spatial Derivatives (ELA_TUBE, UNI_BEAM, KDV)

If we assume that our underlying mathematical model allows the solution of partial differential equations at most of order two, we can nevertheless solve higher order equations by taking algebraic equations into account. A typical example is a fourth order equation, say

$$u_t = F(u, u_{xxxx}, x, t) \ ,$$

transformed into a second-order equation by introducing an additional variable w,

$$\begin{aligned} u_t &= F(u, w_{xx}, x, t) \ , \\ 0 &= w - u_{xx} \ . \end{aligned}$$

Boundary conditions must be selected very carefully to guarantee unique solutions.

4.4 General Boundary Conditions (HEAT_NLD, HEAT_NLC)

For simplicity, we consider a system of partial differential equations without algebraic ones,

$$u_t = f(u, u_x, u_{xx}, x, t) \quad ,$$

where $x \in [x_L, x_R]$. Instead of Dirichlet or Neumann boundary conditions (4), we suppose that there are implicitly defined equations of the form

$$h_L(u(x_L, t), u_x(x_L, t), t) = 0 \quad , \qquad h_R(u(x_R, t), u_x(x_R, t), t) = 0 \quad ,$$

also called conditions of Stefan-Boltzmann type. These equations are added to the partial ones in form of a system of DAEs without specifying additional explicit boundary values. Initial consistent values are computed by Newton's method. Instead of inserting x_L or x_R, any other spatial values could be used. They must be rounded to the nearest line on order to couple the corresponding equation.

5 Software and Test Examples

We offer the possibility to test the PDAE models of the previous section, to try alternative solution and discretization methods, or to change scaling parameters and solution tolerances. The corresponding interactive system is called EASY-FIT, see Schittkowski [49, 50], and can be downloaded through the URL

`http://www.uni-bayreuth.de/departments/math/~kschittkowski/`
`easy_fit_demo.htm`

The discretization schemes, equation solvers and least squares algorithms of Section 3 are implemented, among a number of additional techniques for solving data fitting problems in dynamical systems. EASY-FIT consists of a database for model data, experimental data, and results and of two executable files containing the numerical algorithms

MODFIT	parameter estimation in explicit functions, steady state systems, Laplace transforms, ordinary differential equations, and differential algebraic equations,
PDEFIT	parameter estimation in one-dimensional time-dependent partial differential equations and partial differential algebraic equations.

Moreover, EASY-FIT comes with a set of 1,170 test examples, among which are 43 PDAEs. Nonlinear functions are defined by the modelling language PCOMP, see Dobmann et al. [16] or Schittkowski [51]. A few characteristic data and the application background of these test problems are listed in Table 1. Besides problem name and some figures characterizing problem size, we show some references in the column headed by *ref*, from where further details can be retrieved. Not listed are the number of integration areas, switching times, and structure of the boundary conditions. There are no equality constraints.

Table 1 PDAE Test Examples

name	n	$\tilde{l}$	n_d	n_a	n_c	m	*background*	*ref*
2ND_DIR1	3	20	1	1	0	0	Second order Dirichlet problem	[28]
2ND_DIR2	3	20	1	1	0	0	Second order inhomogeneous Dirichlet problem	[28]
ACCRET_A	1	20	6	4	0	0	Thermal equilibrium curves in Keplerian accretion disks	[38]
ACCRET_F	1	112	3	1	0	0	Thermal equilibrium curves in Keplerian accretion disks	[38]
ACETYL_T	2	180	10	9	0	0	Tubular acetylene reactor, time-dependent formulation	[5]
ACETYL_Z	2	20	10	1	0	9	Tubular acetylene reactor, space-dependent formulation	[5]
BEAM1	3	99	2	2	0	0	Curved beam	[57], [55]
BEAM2	3	90	2	2	0	0	Linked beams	[57] , [55]
BIFURC1	3	180	2	0	2	0	Bifurcation with codimension 2 (Ginzburg-Landau equation)	[1]
BIFURC2	2	300	2	0	2	0	Bifurcation with codimension 2 (Ginzburg-Landau equation), dense observation grid	[1]
BVP_TRIV	1	10	1	1	0	0	Boundary value problem with known solution	[26]
CAPILL	2	13	1	1	0	0	Capillar filled with water under electric charge	
CNT_CUR2	4	80	3	1	0	0	Counter-current separation of fluid phase concentrations with phase equilibrium	[36]
CO_OXYD	3	68	2	0	2	0	CO oxydation on Pt(110)	[3]
CTFLOW	1	40	4	2	0	0	Two incompressible counter-current flows of binary liquid mixture with semi-permeable wall	[33]
CUSP	2	40	3	0	3	0	Threshold-nerve impulse with cusp catastrophe	[61]
ELA_TUBE	3	40	2	1	0	0	Waves propagating in a liquid-filled elastic tube (Korteweg-de Vries-Burgers equation)	[25]
ELDYN_A	3	20	4	2	0	0	Electrodynamic application with algebraic equations	[7]
EW_WAVE	2	48	2	1	0	0	Wave propagation in media with nonlinear steepening and dispersion	[23]
FLAT_MEM	1	20	1	0	2	0	Concentration boundary layer over a flat membrane	

(continued)

name	n	l	n_d	n_a	n_c	m	*background*	*ref*
FLOWDIFF	2	101	2	0	2	0	Flow system with diffusion	
HEAT_A	2	27	2	1	0	0	Heat equation, formulated with algebraic equation	
HEAT_F	2	27	2	1	0	0	Heat equation, formulated with algebraic equation and flux formulation	
HEAT_NLC	2	10	1	0	1	0	Heat equation with nonlinear boundary condition of Stefan-Boltzmann type	[56]
HEAT_NLD	2	10	1	0	1	0	Heat equation with nonlinear boundary condition of Stefan-Boltzmann type	[56]
HEAT_R	2	27	2	1	0	9	Heat equation with dynamical restrictions and algebraic equation	
HEAT_RAD	2	10	2	1	0	0	Heat conduction with radiation and forced convection	[20]
KDV1	2	44	2	1	0	0	Korteweg-de-Vries equation, exact solution with one soliton	[43]
KDVE	1	17	2	1	0	0	Shallow water flow, balancing front sharpening and dispersion to produce solitons	[43]
MC_DIST	4	63	4	2	4	0	Multi-component distillation	[33]
MCFC1	20	20	19	2	0	0	MCFC fuel cell	
MCFC2	3	110	19	2	0	0	MCFC fuel cell	
MCFC3	3	110	19	2	0	0	MCFC fuel cell	
MCMC_0	5	70	19	2	9	0	MCMC fuel cell	
NET_3	10	100	2	1	2	0	Network with three beams and controlled Neumann knot	[17]
PAR_SINA	2	84	2	1	0	0	Parabolic PDE with inhomogeneous sinus-term (with algebraic equation)	[40], [37]
PDAE4	2	6	2	1	0	0	Simple fourth-order PDAE with exact solution	
PLASMA	3	20	4	2	0	0	Space-time movement of ions and electrons	[31]
PRESSURE	3	15	3	1	0	0	Pressure-driven flow in porous media	[30]
SILICON	1	102	2	1	0	0	Diffusion in silicon	
TUNNEL	2	24	2	1	0	0	Tunnel furnace with heating and cooling section	[33]
UNI_BEAM	2	10	5	3	0	0	Thin uniform cantilevered beam	[20]
VIB_BEAM	6	6	3	1	0	0	Boundary control of transverse vibrations of a beam	[4]

6 Conclusions

We show an approach to compute unknown parameters in a dynamical model consisting of partial differential algebraic equations by a least squares data fit. The dynamical equations are discretized by the method of lines leading to large system of DAEs. Ordinary differential equations and least squares problems can be solved by available standard codes.

The intention of the paper is to present a review on some techniques that are routinely used to estimate data in dynamical systems. The model structure is very flexible and covers a broad and realistic domain. Some *real life* applications are addressed, which reflect typical situations in industry and academia. The complexity of practical mathematical models is illustrated and numerical results are included. They show that unknown parameters of realistic dynamical systems can be estimated more or less routinely by available standard algorithms.

References

[1] Argentine M., Coullet P. (1997): *Chaotic nucleation of metastable domains*, Physical Reviews E, Vol. 56, 2359-2362

[2] Ascher U.M., Petzold L.R. (1998): *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations,* SIAM, Philadelphia

[3] Bär M., Hegger R., Kantz H (1999): *Fitting partial differential equations to space-time-dynamics*, Physical Reviews E, Vol. 59, 337-342

[4] Bazeze A., Bruch J.C., Sloss J.M. (1999): *Numerical solution of the optimal boundary control of transverse vibrations of a beam*, Numerical Methods for Partial Differential Equations, Vol. 15, No. 5, 558-568

[5] Birk J., Liepelt M., Schittkowski K., Vogel F. (1999): *Computation of optimal feed rates and operation intervals for turbular reactors*, Journal of Process Control, Vol. 9, 325-336

[6] Björck A. (1990): *Least Squares Methods,* Elsevier

[7] Blom J.G., Zegeling P.A. (1994): *Algorithm 731: A moving grid interface for systems of one-dimensional time-dependent partial differential equations*, ACM Transactions on Mathematical Software, Vol. 20, No. 2, 194-214

[8] Caracotsios M., Stewart W.E. (1995): *Sensitivity analysis of initial-boundary-value problems with mixed PDE's and algebraic equations*, Computers and Chemical Engineering, Vol. 19, 1019-1030

[9] Chakravarthy S.R., Osher S.(1984): *High resolution schemes and the entropy condition,* SIAM Journal on Numerical Analysis, Vol. 21, No. 5, 955-984

[10] Chakravarthy S.R., Osher S. (1985): *Computing with high resolution upwind schemes for hyperbolic equations,* Lectures in Applied Mathematics, Vol. 22, 57-86

[11] Chudej K., Petzet V., Scherdel S., Pesch H.J., Schittkowski K., Heidebrecht P., Sundmacher K. (2003): *Index analysis of a nonlinear PDAE system describing a molten carbonate fuel cell*, to appear: PAMM

[12] Chudej K., Petzet V., Scherdel S., Pesch H.J., Schittkowski K., Heidebrecht P., Sundmacher K. (2003): *Numerical simulation of a 1D model of a molten carbonate fuel cell*, to appear: PAMM

[13] Dennis J.E.jr. (1977): *Nonlinear least squares,* in: The State of the Art in Numerical Analysis, D. Jacobs ed., Academic Press, New York, London

[14] Dennis J.E.jr., Gay D.M., Welsch R.E. (1981): *An adaptive nonlinear least-squares algorithm,* ACM Transactions on Mathematical Software, Vol. 7, No. 3, 348-368

[15] Dennis J.E.jr., Gay D.M., Welsch R.E. (1981): *Algorithm 573: NL2SOL-An adaptive nonlinear least-squares algorithm,* ACM Transactions on Mathematical Software, Vol. 7, No. 3, 369-383

[16] Dobmann M., Liepelt M., Schittkowski K. (1995): *Algorithm 746: PCOMP: A FORTRAN code for automatic differentiation,* ACM Transactions on Mathematical Software, Vol. 21, No. 3, 233-266

[17] Geisler J.(1999): *Dynamische Gebietszerlegung für Optimalsteuerungsprobleme auf vernetzten Gebieten unter Verwendung von Mehrgitterverfahren*, Diploma Thesis, Dept. of Mathematics, University of Bayreuth, Germany

[18] Gill P.E., Murray W. (1978): *Algorithms for the solution of the non-linear least-squares problem,* SIAM Journal on Numerical Analysis, Vol. 15, 977–992

[19] Gill P.E., Murray W., Wright M.H. (1981): *Practical Optimization,* Academic Press

[20] Goodson R.E., Polis M.P. (1978): *Identification of parameters in distributed systems*, in: Distributed Parameter Systems, W.H. Ray, D.G. Lainiotis eds., Marcel Dekker, New York, Basel, 47-134

[21] Gugat M., Leugering G., Schittkowski K., Schmidt E.J.P.G. (2001): *Modelling, stabilization and control of flow in networks of open channels*, in: Online Optimization of Large Scale Systems, M. Groetschel, S.O. Krumke, J. Rambau eds., Springer, 251-270

[22] Hairer E., Wanner G. (1991): *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems,* Springer Series Computational Mathematics, Vol. 14, Springer

[23] Hamdi S., Gottlieb J.J., Hanson J.S. (2001): *Numerical solutions of the equal width wave equation using an adaptive method of lines*, in: Adaptive Methods of Lines, A. Vande Wouwer, Ph. Saucec Ph., W. Schiesser eds., Chapman and Hall/CRC, Boca Raton

[24] Heidebrecht P., Sundmacher K.: *Molten carbonate fuel cell (MCFC) with internal reforming: Model-based analysis of cell dynamics*, Chemical Engineering Sciences, Vol. 58, 1029-1036

[25] Johnson R.S. (1970): *A nonlinear equation incorporating damping and dispersion*, Journal of Fluid Dynamics, Vol. 42, 49-60

[26] Langtangen H.P. (1999): *Computational Partial Differential Equations*, Lecture Notes in Computational Science and Engineering, Vol. 2, Springer, Berlin, Heidelberg

[27] Levenberg K. (1944): *A method for the solution of certain problems in least squares,* Quarterly Applied Mathematics, Vol. 2, 164-168

[28] Lewis R.M., Patera A.T., Peraire J. (2000): *A posteriori finite element bounds for sensitivity derivatives of partial-differential-equation outputs*, Finite Elements in Design, Vol. 34, 271-290

[29] Lindström P. (1982): *A stabilized Gauß-Newton algorithm for unconstrained least squares problems,* Report UMINF-102.82, Institute of Information Processing, University of Umea, Umea, Sweden

[30] Logan J.M. (2001): *Transport Modeling in Hydrochemical Systems*, Interdisciplinary Applied Mathematics, Springer, New York

[31] Lucht W., Debrabant K. (1996): *Models of quasi-linear PDAEs with convection*, Report, Dept. of Mathematics and Computer Science, University of Halle, Germany

[32] Marquardt D. (1963): *An algorithm for least-squares estimation of nonlinear parameters,* SIAM Journal of Applied Mathematics, Vol. 11, 431-441

[33] Molander M. (1990): *Computer aided modelling of distributed parameter process*, Technical Report No. 193, School of Electrical and Computer Engineering, Chalmers University of Technology, Göteborg, Sweden

[34] Pesch H.J., Chudej K., Petzet V., Scherdel S., Schittkowski K., Heidebrecht P., Sundmacher K. (2003): *Numerical simulation of a 1D model of a molten carbonate fuel cell*, PAMM-Proceedings of Applied Mathematics and Mechanics, Vol. 3, 521-522

[35] Petzold L.R. (1982): *A description of DASSL: A differential/algebraic system solver,* in: Proceedings of the 10th IMACS World Congress, Montreal, Canada

[36] Pfeiffer B.-M., Marquardt W. (1996): *Symbolic semi-discretization of partial differential equation systems*, Mathematics and Computers in Simulation, Vol. 42, 617-628

[37] Pfleiderer J., Reiter J. (1991): *Biplicit numerical integration of partial differential equations with the transversal method of lines*, Report No. 279, DFG SPP *Anwendungsbezogene Optimierung und Steuerung*, Technical University, Dept. of Mathematics, Munich

[38] Pin-Gao Gu, E.T. Vishniac, J.K. Cannizo (2000): *Thermal equilibrium curves and turbulent mixing in Keplerian accretion disks*, The Astrophysical Journal, Vol. 534, 380-397

[39] Ramsin H., Wedin P.A. (1977): *A comparison of some algorithms for the nonlinear least squares problem,* Nordisk Tidstr. Informationsbehandlung (BIT), Vol. 17, 72-90

[40] Rektorys K. (1982): *The Method of Discretization in Time and Partial Differential Equations*, Reidel, Dordrecht

[41] Renardy M., Rogers R.C. (1993): *An Introduction to Partial Differential Equations,* Texts in Applied Mathematics, Vol. 13, Springer, Berlin

[42] Schiesser W.E. (1991): *The Numerical Method of Lines,* Academic Press, San Diego

[43] Schiesser W.E. (1994): *Method of lines solution of the Korteweg-de Vries equation*, Computers in Mathematics and Applications, Vol. 28, No. 10-12, 147-154

[44] Schittkowski K. (1985/86): *NLPQL: A FORTRAN subroutine solving constrained nonlinear programming problems,* Annals of Operations Research, Vol. 5, 485-500

[45] Schittkowski K. (1988): *Solving nonlinear least squares problems by a general purpose SQP-method*, in: *Trends in Mathematical Optimization*, K.-H. Hoffmann, J.-B. Hiriart-Urruty, C. Lemarechal, J. Zowe eds., International Series of Numerical Mathematics, Vol. 84, Birkhäuser, pp. 295-309.

[46] Schittkowski K. (1997): *Parameter estimation in partial differential equations,* Optimization Methods and Software, Vol. 7, No. 3-4, 165-210

[47] Schittkowski K. (2000): *Parameter estimation in a mathematical model for substrate diffusion in a metabolically active cutaneous tissue*, Progress in Optimization, X. Yang et al. eds., Kluwer Academic Publishers, 329 - 342

[48] Schittkowski K. (1999): *PDEFIT: A FORTRAN code for parameter estimation in partial differential equations*, Optimization Methods and Software, Vol. 10, 539-582

[49] Schittkowski K. (2001): *EASY-FIT: A software system for data fitting in dynamic systems*, Structural and Multidisciplinary Optimization, Vol. 23, No. 2, 153-169

[50] Schittkowski K. (2002): *Numerical Data Fitting in Dynamical Systems - A Practical Introduction with Applications and Software*, Kluwer Academic Publishers, Dordrecht, Boston, London

[51] Schittkowski K. (2004): *PCOMP: A modeling language for nonlinear programs with automatic differentiation*, in: Modeling Languages in Mathematical Optimization, J. Kallrath ed., Kluwer Academic Publishers, 349-367

[52] Schittkowski K. (2004): *NLPQLP20: A Fortran implementation of a sequential quadratic programming algorithm with distributed and non-monotone line search-User's guide*, Report, Department of Mathematics, University of Bayreuth

[53] Smith G.D. (1985): *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Clarendon Press, Oxford Applied Mathematics and Computing Science Series

[54] Thomas J.W. (1995): *Numerical Partial Differential Equations*, Texts in Applied Mathematics, Vol. 22, Springer, Berlin

[55] Timoshenko S., Goodier J.N. (1970): *Theory of Elasticity*, McGraw Hill, New York

[56] Troeltzsch F. (1999): *Some remarks on second order sufficient optimality conditions for nonlinear elliptic and parabolic control problems*, in: Proceedings of the Workshop 'Stabilität und Sensitivität von Optimierungs- und Steuerungsproblemen', Burg (Spreewald), Germany, 21.-23.4.99

[57] van Kan J.J.I.M., Segal A. (1995): *Numerik partieller Differentialgleichungen für Ingenieure,* Teubner

[58] Wansbrough R.W. (1985):*Modelling chemical reactors*, Chemical Engineering, Vol. 5, 95-102

[59] Yang H.Q., Przekwas A.J. (1992): *A comparative study of advanced shock-capturing schemes applied to Burgers' equation*, Journal of Computational Physics, Vol. 102, 139-159

[60] Yee H.C. (1985): *Construction of a class of symmetric TVD schemes*, Lectures in Applied Mathematics, Vol. 22, 381-395, Springer, Berlin

[61] Zeeman E.C. (1972): *Differential equations for the haertbeat and nerve impulse*, in: Towards a Theoretical Biology, C.H. Waddington ed., Edinburgh University Press, Vol. 4, 8-67