Zuse Institute Berlin

Takustr. 7
14195 Berlin
Germany

TOM STREUBEL
CHRISTIAN STROHM
PHILIPP TRUNSCHKE
CAREN TISCHENDORF

Generic Construction and Efficient Evaluation of Network DAEs and Their Derivatives in the Context of Gas Networks

ZIB Report 17-41 (July 2017)

Zuse Institute Berlin
Takustr. 7
14195 Berlin
Germany

Telephone: +49 30-84185-0
Telefax: +49 30-84185-125

E-mail: bibliothek@zib.de
URL: http://www.zib.de

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

Generic Construction and Efficient Evaluation of Network DAEs and Their Derivatives in the Context of Gas Networks

Tom Streubel[1,2], Christian Strohm[2], Philipp Trunschke[1,2], and Caren Tischendorf[2]

[1] Department of Optimization, Zuse Institute Berlin, Germany
[2] Department of Mathematics, Humboldt University of Berlin, Germany

Abstract. We present a concept that provides an efficient description of differential-algebraic equations (DAEs) describing flow networks which provides the DAE function f and their Jacobians in an automatized way such that the sparsity pattern of the Jacobians is determined before their evaluation and previously determined values of f can be exploited. The user only has to provide the network topology and local function descriptions for each network element. The approach uses automatic differentiation (AD) and is adapted to switching element functions via the abs-normal-form (ANF).

Keywords: compressed sparse row format, algorithmic differentiation, abs-normal form, piecewise linear tangent approximation, piecewise smooth

1 Introduction

The dynamic behavior of flow networks is often modeled by differential-algebraic equations, cf. [7]. The network is considered as an oriented graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with a node set $\mathcal{N}$ and a hyper edge set $\mathcal{E}$. A hyper edge $E \in \mathcal{E}$ is a non-empty ordered tuple of nodes from $\mathcal{N}$. Each hyper edge $E_i \in \mathcal{E}$ represents a network element such as a junction, pipe, valve or compressor station. The element model is then given by an *element function* $\tilde{f}_i : \mathbb{R}^{m_i} \times \mathbb{R}^{m_i} \times \mathbb{R} \to \mathbb{R}^{n_i}$ imposing

$$\tilde{f}_i(\dot{x}_i(t), x_i(t), t) = 0. \tag{1}$$

When simulating gas networks, t refers to the time and x usually contains pressures and flows. Depending on the topology some element functions may share some of their variables with others. If, for example, two pipes represented by $E_i, E_j \in \mathcal{E}$ are sharing the same junction, then their pressures associated to that junction are equal.

Further, it is important to mention that $\tilde{f}_i$ may not depend on all components of $\dot{x}_i$. And in the case of static elements $\dot{x}_i$ has even no influence. By taking the union x of all needed x_i, i. e. ignoring redundant variables, and incerting hyperedge functions that describe certain x_i explicitly, we obtain the whole flow network model as

$$f(\dot{x}(t), x(t), t) = \begin{pmatrix} f_1(\dot{x}(t), x(t), t) \\ \vdots \\ f_m(\dot{x}(t), x(t), t) \end{pmatrix} = 0 \tag{2}$$

with $f_j(\dot{x}(t), x(t), t)$ for $j = 1, ..., m \leq |\mathcal{E}|$ being deduced from $\tilde{f}_i(\dot{x}_i(t), x_i(t), t)$ for $i = 1, ..., |\mathcal{E}|$. Notice that the functions f_j may be not smooth at certain points. This is particularly the case when valves and limiting bounds are described (usually by min- and max-evaluations).

Several solvers have been developed to solve DAEs of the form (2), e. g. DASPK from L. Petzold, ode15i (in Matlab) from L.F. Shampine and IDAS from SUNDIALS. Such solvers often run more efficiently and more stable if the user provides not only evaluations of the residual function $f(y, x, t)$ but also evaluations of the partial derivatives $f_y(y, x, t)$ and $f_x(y, x, t)$.

In this paper we present a concept that automatically provides functions f, f_y and f_x. The user has to provide only the network graph $\mathcal{G}$, the element functions $\tilde{f}_i$ and their sparsity patterns. Thereby, the sparsity patterns of f_y and f_x are determined prior to their evaluation. Previously determined values of f, f_y and f_x can be exploited. The presented approach focusses on the use of automatic differentiation [4] but could also use other variants of differentiation. For treating non-smooth functions as min() and max() we use an approach via their abs-normal-form representation, see Section 3.

2 Jacobian representation

We consider the structure of the nonlinear functions $\mathfrak{f}$ to be differentiated for the determination of f_y and f_x. Fixing $x = x_*$, $t = t_*$ and $y = y_*$, $t = t_*$, respectively, we have to differentiate the functions

$$\mathfrak{f}(y) := \begin{bmatrix} f_1(y, x_*, t_*) \\ \vdots \\ f_m(y, x_*, t_*) \end{bmatrix}, \quad \mathfrak{f}(x) := \begin{bmatrix} f_1(y_*, x, t_*) \\ \vdots \\ f_m(y_*, x, t_*) \end{bmatrix} \tag{3}$$

in order to provide f_y and f_x. Since we are interested in an element-wise computation of $\mathfrak{f}'(y)$ and $\mathfrak{f}'(x)$ the CSR format (compressed row format [1]) is a suitable choice to represent f_y and f_x.

3 Treatment of switching elements using the abs-normal-form

In the case of switching elements, we need min/max-evaluations. Consequently, the element functions $\tilde{f}_i$ are only *piecewise differentiable* (PD). In order to treat them, we introduce the following representation of functions.

A function $\mathfrak{f}$ is called in abs-normal-form (ANF, [5]) if there exist twice differentiable functions F and G such that the function value $\mathfrak{f}(x)$ can be computed via

$$z = G(x, |z|), \qquad \mathfrak{f}(x) = F(x, |z|)$$

where $G_w(x, w) \equiv \frac{\partial}{\partial w} G(x, w)$ is of strictly lower triangular form. The vector z represents *switching variables* and is uniquely determined. Moreover it can be evaluated component-wise in a forward fashion, because of the special nilpotent form of G_w. So $z = G(x, |z|)$ may be understood as an explicit evaluation of z for given input x. Notice that all piecewise linear functions have an ANF representation [2].

A first order Taylor expansion of F and G at $(\mathring{x}, \mathring{w}) \in \mathbb{R}^{n+s}$ followed by a subsequent substitution $\mathring{w} = |\mathring{z}|$, $\Delta w \equiv |\mathring{z} + \Delta z| - |z(\mathring{x})|$, where $\mathring{z} \equiv z(\mathring{x}) = G(\mathring{x}, |\mathring{z}|)$ leads to a piecewise linear operator in ANF mapping $\Delta x \equiv x - \mathring{x}$ to $\mathfrak{f}(\mathring{x}) + \Delta\mathfrak{f}$:

$$\begin{pmatrix} \mathring{z} + \Delta z \\ \mathfrak{f}(\mathring{x}) + \Delta\mathfrak{f} \end{pmatrix} = \begin{pmatrix} G(\mathring{x}, |\mathring{z}|) \\ F(\mathring{x}, |\mathring{z}|) \end{pmatrix} + \begin{bmatrix} G_x(\mathring{x}, |\mathring{z}|) & G_w(\mathring{x}, |\mathring{z}|) \\ F_x(\mathring{x}, |\mathring{z}|) & F_w(\mathring{x}, |\mathring{z}|) \end{bmatrix} \cdot \begin{pmatrix} x - \mathring{x} \\ |\mathring{z} + \Delta z| - |\mathring{z}| \end{pmatrix}, \tag{4}$$

that satisfies the approximation property $\mathfrak{f}(x) = \mathfrak{f}(\mathring{x}) + \Delta\mathfrak{f} + \mathcal{O}(\|x - \mathring{x}\|^2)$. The block matrix of the piecewise linear operator (4) can be stored in a CSR fashion as well as the Jacobians in the differentiable case.

For standard DAE solvers we have to provide one suitable representative $\mathfrak{f}'(x)$ for the Bouligand subdifferential $\partial_B \mathfrak{f}(x)$. This can be derived from equation (4)

$$\mathfrak{f}'(x) := J + Y\Sigma(I - L\Sigma)^{-1} Z, \qquad \begin{bmatrix} Z & L \\ J & Y \end{bmatrix} := \begin{bmatrix} G_x(\mathring{x}, |\mathring{z}|) & G_w(\mathring{x}, |\mathring{z}|) \\ F_x(\mathring{x}, |\mathring{z}|) & F_w(\mathring{x}, |\mathring{z}|) \end{bmatrix}$$

using a suitable signature Σ, see [2].

A better way would be to exploit (4) directly in the numerical integration scheme for the differential-algebraic equation. It is demonstrated in [3] for the implicit Trapezoidal method for the integration of ordinary differential equations.

Then, the ANF operators propagated from network structures appear in a more complex form compared to (4):

$$\begin{pmatrix} z^I \\ y^I \\ \hline z^{II} \\ y^{II} \\ \hline \vdots \\ \hline z^K \\ y^K \end{pmatrix} = \begin{pmatrix} c^I \\ b^I \\ \hline c^{II} \\ b^{II} \\ \hline \vdots \\ \hline c^K \\ b^K \end{pmatrix} + \left[\begin{array}{cccccc} \bullet & G_w^I & * & & * & \\ \bullet & F_w^I & * & & * & \\ \hline * & & \bullet & G_w^{II} & * & \\ * & & \bullet & F_w^{II} & * & \\ \hline \vdots & & \vdots & & \vdots & \\ \hline * & & * & & \bullet & G_w^K \\ * & & * & & \bullet & F_w^K \end{array}\right] \cdot \begin{pmatrix} x^I \\ |z^I| \\ x^{II} \\ |z^{II}| \\ \vdots \\ x^K \\ |z^K| \end{pmatrix}. \tag{5}$$

Here $*$ (typically empty or sparse) and $\bullet$ (typically sparse or dense) are submatrices of G_x^j and F_x^j, respectively. The horizontal lines indicate *element blocks*.

4 Network structure preserving representation and implemention

First, for each f_i, we realize the Jacobian evaluations or, if necessary, their ANF representations by a new class, which we call *partial CSR*. Contrary, each ANF representation of $\mathfrak{f}$ is stored in a so-called *complete CSR* class, obtained by merging all the corresponding partial CSRs.

These CSR classes implement slightly modified versions of the CSR format, each comprising a `data`-, `indices`- and `indptr`-array as well as a `shape`-attribute. Further, there is implemented a new attribute `nabs` containing the number of switching variables. In contrast to the classical CSR format, the `indices`-array shall be initialized as a signed array to mark all indices of nonzero entries from G_w and F_w by signs. In doing so we can distinguish coefficients for x from those of the absolute value of the switching vector $|z|$.

The relationship between both classes and their individual attributes are illustrated in Figure 1. Here it becomes clear that the corresponding partial CSRs are collected in a list `partialCSRs` and parsed, as the only argument, to the constructor of complete CSR. On the other hand partial CSR objects are created with the arguments `nnzPerRow`, `ncols` and `nabs`. It is `nnzPerRow` a list containing the numbers of variable dependencies per component of the element function f_i. Further, `ncols` is the amount of variables contributed to the whole DAE system (2).

The partial CSR object proceeds as follows: A local `indptr` is created as the cumulative sum of `nnzPerRow`. Additional informations are derived, such as `nnz` the number on non zero entries of the local CSR and `nrows` the number of rows of the CSR. The `signature` $= \text{sign}(z)$ stores the sign-vector of switching variables and is needed for certain evaluation routines.
The complete CSR proceeds in a different manner: Its `indptr`-array gets aggregated from the `indptr`-arrays of the elemental partial CSR instances. Thereafter,

the lengths of the `indices`- and `data`-array is determined, the arrays can be allocated and local views are provided to the partial CSRs. In this way an arbitrary number of complete CSR instances for any purpose, e. g. all arguments, can be created dynamically.

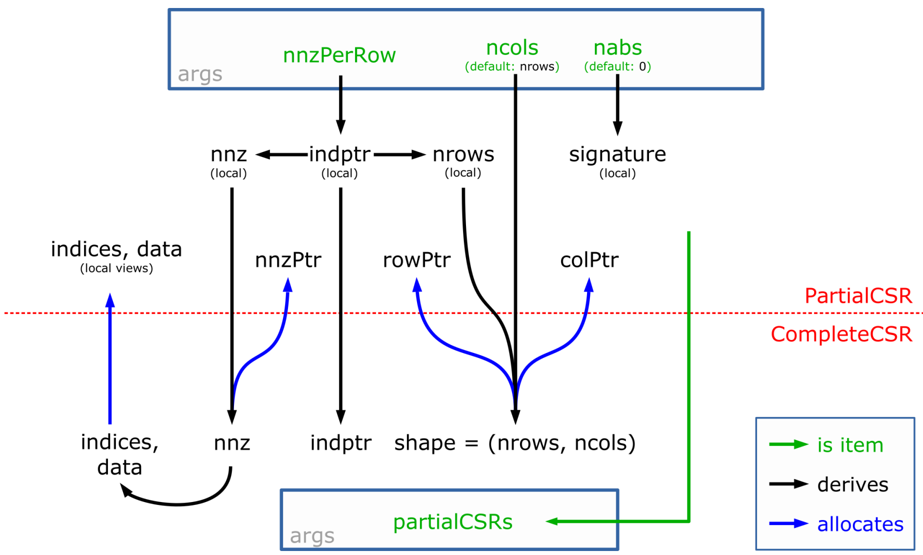

Fig. 1: Schema to manage Jacobians and ANFs of flow networks in CSR format.

5 Jacobians for a gas network example

We tested the GasLib40 instance from the open gas network library [6]. Figure 2 shows the topology of the network and fingerprints of the Jacobians f_y and f_x. The Jacobian f_y is constant. The Jacobians f_x is in ANF representation, due to check valve functionality of two (modified) resistors. Their partial CSRs are displayed as enlarged section on top of Figure 2. The first and third row of the partial CSRs represent the data of G_x and G_w for the determination of the two switching variables.

Acknowledgements This work was supported by the German Federal Ministry of Education and Research (BMBF) within the Research Campus MODAL (fund number 05M14ZAM) and by the Deutsche Forschungsgemeinschaft through the Collaborative Research Centre TRR154 Mathematical Modelling, Simulation and Optimization Using the Example of Gas Networks.

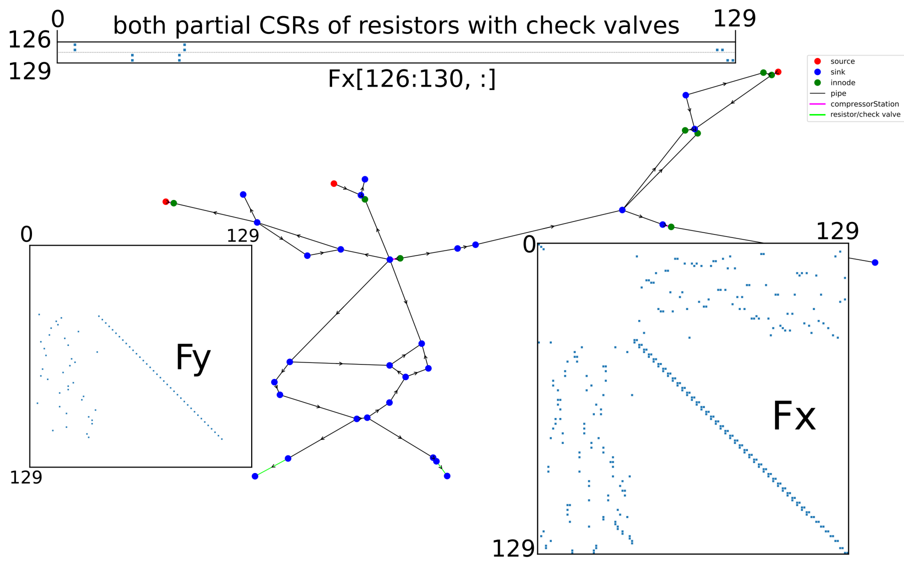

Fig. 2: Modified version of GasLib-40 [6], Fy and Fx are fingerprints of f_y and f_x, respectively. On the top is a zoom of the fingerprints of the two partial CSRs belonging to the switching elements of the nework (two check valve resistors).

References

1. Golub, Gene H. and Van Loan, Charles F.: Matrix computations. JHU Press (2012)
2. Griewank, A. and Bernt, J.-U. and Radons, M. and Streubel, T.: Solving piecewise linear systems in abs-normal form. Linear Algebra and its Applications (2015)
3. Griewank, A. and Hasenfelder, R. and Radons, M. and Streubel, T.: Integrating Lipschitzian Dynamical Systems using Piecewise Algorithmic Differentiation (2017)
4. Griewank, A. and Walther, A.: Evaluating Derivatives. second edition, Society for Industrial and Applied Mathematics (2008),
5. Griewank, A. and Walther A.: First and second order optimality conditions for piecewise smooth objective functions. Optimization Methods and Software (2016),
6. Humpola, J. and Joormann, I. and Oucherif, D. and Pfetsch, M. E. and Schewe L. and Schmidt, M. and Schwarz R.: GasLib – A Library of Gas Network Instances.
7. L. Jansen and C. Tischendorf. A Unified (P)DAE Modeling Approach for Flow Networks. In Schöps, Sebastian and Bartel, Andreas and Günther, Michael and ter Maten, E. Jan W. and Müller, Peter C, editors, Progress in Differential-Algebraic Equations , Differential-Algebraic Equations Forum, 127-151. Springer Berlin Heidelberg (2014).