Watermatex 2015

9th IWA Symposium on Systems Analysis and Integrated Assessment

14-17 June 2015, Gold Coast, Australia

Automatic generation of a symbolic Jacobian of non-linear equations

Garneau, C.*, Claeys, F.H.A., Vanrolleghem P.A.***

* modelEAU, Département de génie civil et de génie des eaux, Université Laval, 1060 Avenue de la Médecine, Québec, Qc, G1V 0A6, Canada

** DHI, Guldensporenpark 104 block K, B-9820 Merelbeke, Belgiumv

Keywords: Finite differences; Jacobian matrix; Symbolic derivation

Summary of key findings

- The generation of a symbolic Jacobian provides insight into the sparsity pattern of the matrix and favours the use of sparse matrix tools.
- Benefits of symbolic manipulations increase with increasing complexity of test models.
- Time invested in the generation of a symbolic Jacobian matrix can easily be compensated by reduced computational complexity of individual simulation runs.

Background and relevance

One of the major bottlenecks of water quality and wastewater treatment models simulation is still the computation time required for running a simulation. In the last decade, it has been observed that the complexity of water quality models was highly correlated to the computational power at hand [2], constraining the complexity of future simulation models to the availability of more powerful computers. In parallel, important advances were made in numerical mathematics to provide precise, stable and efficient algorithms to solve large classes of models written as ordinary differential equations (ODE). However, symbolic manipulation has been given limited attention as it required important computer resources. Nevertheless, as the symbolic solution to a problem provides an exact solution, and sometimes to a fraction of the cost of a numerical approximation, it can provide improved performance of numerical solvers by alleviating bottleneck computations. This work examines the potential of coupling a tool generating a symbolic Jacobian matrix to an implicit stiff solver. The symbolic Jacobian is then tested on three cases based on wastewater treatment models of increasing complexity.

Numerical methods and model

The Jacobian matrix consists in the derivatives of state functions $f_i(x)$ with respect to each state variable x_i (eq. 1).

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \quad [1]$$

The Jacobian matrix, evaluated at a point x, represents the best linear approximation of the model near the operating point and is used extensively by stiff solvers and in algorithms such as the Newton-Raphson algorithm. Nowadays, three methods can be used to compute or estimate the Jacobian matrix: symbolic derivation (SD), automatic differentiation (AD) and the approximation by finite differences (FD).

SD consists of applying chain derivation to the state functions with respect to all state variables. Although conceptually simple, this solution has not been applied to large models for a long time because of the important memory requirements. In recent years, however, the memory requirement has been considerably lifted and symbolic derivation has gained more visibility [1].

AD is an algorithmic technique based on two programming paradigms, namely operator overloading and source transformation, which makes heavy use of the chain derivation [4]. In contrast to symbolic

derivation, only the numerical values of the derivatives are computed. Therefore, the model generation is much faster, as no symbolic manipulation is needed, but the computation of the derivatives has been shown to be slower when compared to SD [1].

Finally, the numerical approximation of the Jacobian is routinely calculated by FD through eq. 2. The FD approximation requires n+1 model evaluations to estimate all terms of the Jacobian matrix. Despite its simplicity, the precision of the method depends on the choice of the perturbation Δx and the management of numerical accuracy (round-off error, truncation, etc.).

$$\frac{\partial f_i}{\partial x_j} \approx \frac{f_i(x_j + \Delta x_j) - f_i(x_j)}{\Delta x_j} \quad [2]$$

In the scope of this work, a tool for SD was developed and its performances were compared to those of the FD approximation. The AD was not considered because of the complexity of its implementation.

Three plant-wide models based on the Activated Sludge Model (ASM) were devised to compare the performances of SD and FD. The first model consists in a single activated sludge unit (ASU) using the ASM2d model and an intermittent aeration coupled to a Takács settling tank [6]. The second is the Benchmark Simulation Model (BSM) [5], consisting of five ASU in series using the ASM1 model and a Takács settling tank. The third is a Full Plant model consisting of 18 ASU using the ASM2d model and a Takács settler. The mathematical complexity of the three models increases from 31 state variables for the first to 108 state variables for the second and to 554 state variables for the third.

All calculations were performed in the scope of the simulation software WEST/Tornado [2], which adopts a compiled executable model approach. A Diagonally Implicit Runge-Kutta (DIRK) solver was used to assess the importance of symbolic derivation of the Jacobian. The results presented here will be detailed in an article in preparation [3].

Results and discussion

Symbolic derivation of the Jacobian matrix offers great insights of the matrix structure since all the non-zero elements of the matrix can be identified prior to the simulation of the model starts. Figure 1 shows the sparsity pattern of the Jacobian matrices for the three test models. Each dot corresponds to a non-zero element of the Jacobian. Additionally, it can be seen that the filling ratio diminishes with the increase of model complexity. Therefore, a first numerical conclusion is that matrix computations can and should systematically be performed with sparse matrix tools.

The results in Table 1 suggest that generating the symbolic Jacobian matrix of the largest model can therefore take up to 8 minutes. However, this generation is invested before the evaluation of the first Jacobian. Once done, the calculation of the symbolic Jacobian proved to be between 11.7 and 27.5 times faster than the finite difference alternative, while providing round-off free derivatives. The investment in the generation of the symbolic Jacobian therefore quickly becomes beneficial.

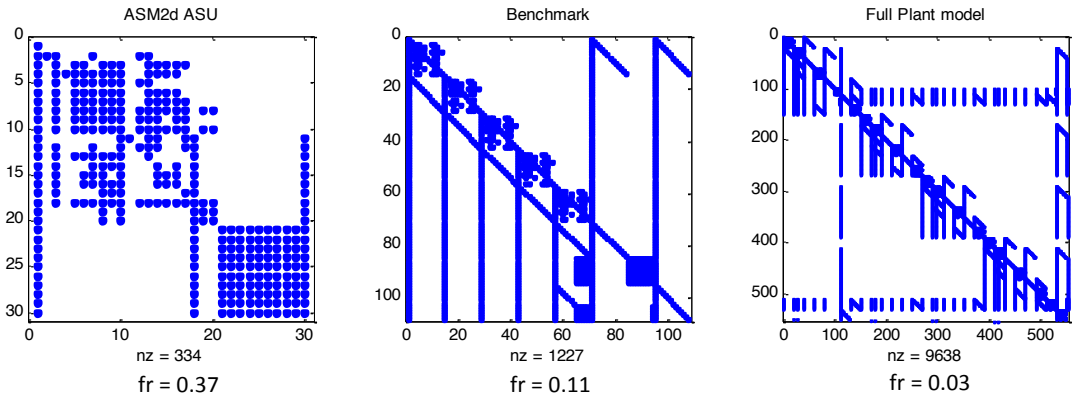

Figure 1 Sparse structure of the three models. The dots represent non-zero (nz) elements and the filling ratio (fr) is the ratio of non-zero elements divided by the total number of elements.

Table 1 Comparison of complexity, compilation times and numerical performances of the FD Jacobian versus the SD Jacobian on the test models and computation time for the three test models under different Jacobian computation schemes with the DIRK stiff solver.

Generation of the Jacobian			
Model	ASM2d_ASU	Benchmark	Full plant
No. of state variables	30	108	554
No. of equations in the model (differential + intermediate algebraic eq.)	418	946	5 694
Time to generate Jacobian (s)	6	18	475
No. of equations in the Jacobian	1 719	5 178	50 193
Compilation time with (without) optimization (s)	10 (<1)	80 (1)	– [1] (30)
Time to compute 1 000 000 optimized (non-optimized) Jacobian matrices (s)	62.7 (76.3)	112 (151)	– [1] (1 550)
Time to compute 1 000 000 Jacobian matrices, finite-differences (s)	732	2616	42 735
Speedup	11.7	23.3	27.5
Simulation results			
Time simulated	60 days, inputs every 15 minutes		
Number of Jacobian evaluations	348 690	240 114	53 580
Simulation time, FD, dense matrix operations (s)	222	984	27 240
Simulation time, FD, sparse matrix operations (s)	234	732	3 120
Simulation time, SD, sparse matrix operations (s)	144	210	680[1]
Speedup observed between FD with sparse matrix operations and SD	1.63	3.48	4.59
Simulated number of days to reach cross-over	10.7	11.3	12.4

[1]The compilation of the generated C-code was not possible under Visual Studio 2008 with speed optimization on because the compiler ran out of memory. Only the non-optimized result is provided.

In the scope of dynamic simulations, matrix manipulations go beyond the computation of the Jacobian and often involve matrix decomposition (e.g. LU or QR decomposition) along with other computations to solve the ODE system of equations. Therefore, the overhead represented by the computation of the Jacobian must also be included in the analysis of the stiff solver's performance.

Table 1 also provides the simulation times of the three test models under different numerical hypotheses and highlights the importance of sparse matrix tools when large matrices are involved. In our tests, the use of sparse matrix tools with the FD Jacobian could already reduce computation times by a factor as high as 9 in the Full plant model. However, the additional use of the symbolic Jacobian was able to further reduce the computation times by a factor of 1.63 to 4.59, with the greatest improvements on the largest and more complex model. Based on these results, we estimated that a simulation horizon of 10.7 to 12.4 days with these models was sufficient to recover the time invested in the generation and compilation of the Jacobian.

Discussion

The simulation of stiff models can be appreciably fastened by providing a symbolic Jacobian matrix. Our work shows impressive results on the DIRK solver, which heavily depends on an accurate Jacobian estimation. Other implicit solvers may require less evaluations of the Jacobian, but can nevertheless benefit from faster and more precise calculations of the Jacobian than the finite difference approximation.

References

[1] Åkesson, J., Braun, W., Lindholm, P., Bachmann, B., 2012. Generation of sparse jacobians for the function mock-up interface 2.0, in: 9th International Modelica Conference. Munich, Germany.
[2] Claeys, F., 2008. A generic software framework for modelling and virtual experimentation with complex environmental systems. Ph.D. thesis, Ghent University, Belgium.
[3] Garneau, C., Claeys, F.H.A., Vanrolleghem, P.A., in prep. Automatic generation of a symbolic Jacobian of non-linear and non-analytical equations.
[4] Griewank, A., Walther, A., 2008. Evaluating derivatives: principles and techniques of algorithmic differentiation, 2nd ed. ed. Society for Industrial and Applied Mathematics, Philadelphia, PA.
[5] Spanjers, H., Vanrolleghem, P.A., Nguyen, K., Vanhooren, H., Patry, G.G., 1998. Towards a simulation-benchmark for evaluating respirometry-based control strategies. Water Sci. Technol. 37, 219–226.
[6] Takács, I., Patry, G.G., Nolasco, D., 1991. A dynamic model of the clarification-thickening process. Water Res. 25, 1263–1271.