Benchmark Examples

The MATMOL Group (2009)

Inside the `Examples` directory one can find, among others, the Matlab© .m-files for the examples used in Logist et al. (1). The examples simulate the Buckley-Leverett equations for an oil well and a dispersive jacketed tubular reactor. The methods employed are based on (*i*) a method of lines approach (2) and (*ii*) an operator splitting approach (3).

These example codes are available free of charge and on an as is basis. The authors cannot be held liable for any deficiency, fault or inconvenience resulting from their use.

These benchmark problems have been included in two different folders: The first one, `Nonlinear_operators`, contains the flux limiting examples while the second one, `Hybrid`, contains the splitting methods. In order to test the configuration, in each of the folders, files have been added which contain the function calls to all examples: (*i*) `Auto_Pe_1.m`, `Auto_Pe_100.m`, and `Auto_Pe_10000.m`, for the tubular reactor example, and (*ii*) `Auto_eps_01.m`, `Auto_eps_001.m`, and `Auto_eps_0001.m` for the oil well example.

NOTE: The `Nonlinear_operators` folder is divided into two folders:

- `DFR`, which contains the example files for the tubular reactor simulation based on a method of lines approach,
- `Buckley_Leverett`, which contains the example files for the oil well simulation based on a method of lines approach.

while, as explained in the installation manual, the `Hybrid` examples were included in the `Methods` (not in the `Examples`) folder since these techniques are problem dependent. This folder has one subdirectory

- `DFR_SeqMeth`, which contains the example files for for the tubular reactor simulation based on an operator splitting approach, i.e., the sequencing method by (3)

Contents

As indicated in the paper (1), all benchmark example codes have been programmed as Matlab© functions. This approach allows the user to directly choose between several

options and to readily experiment with the codes. Each of the available options is documented at the beginning of the .m-file. To maintain a clean overview over the different example codes, all required subfunctions have each time been implemented in the same file as nested functions. One exception, however, are the files `BC_DynGrid.m`, `Flux_DynGrid.m`, and `dFlux_DynGrid_dx.m` which for the dynamic regridding approach specify the boundary conditions, the fluxes, and their Jacobian, respectively. In addition, one auxiliary file `AddIntStats.m` is added to enable the availability of the integration statistics. Table 1 illustrates the available files and describes their features.

Note on the calling of ODE/DAE solvers in Matlab©

In most of the files the Matlab© ODE/DAE integrators are called in order to return a structure which apart from the solution trajectories also contain the integration statistics (e.g., number of steps, number of function evaluations, . . .). This options has been selected for illustrative reasons.

```
   sol = solver(odefun,[t0 tf],y0...)
Stats = sol.stats;
```

However, in this formulation the evolution of the independent variables is stored at each of the steps taken by the integrator in between the initial point `t0` and the final `tf` in the interval. Hence, whenever fine steps have to be taken for large ODE/DAE systems (which originate, e.g., from a PDE discretisation) the amount of information to be stored increases rapidly. When this amount of data approaches the size, or exceeds the amount of the available memory, parts of the data have to be stored in the virtual memory (i.e., the hard disk). Access to this information requires each time read and write operations which tremendously slow down the computation. Evaluating the dependent variables at certain desired points `tspan` is finally done by employing the function:

```
Y = deval(sol,tspan);
```

By consequence, when the number of points at which the dependent variables have to be evaluated, is rather limited, it can be advantageous to call the integrator as follows:

```
[T,Y] = solver(odefun,tspan,y0);
```

which only stores the dependent variables at the points provided in `tspan`. However, it should be noted that in this last case the integration statistics are not accessible.

Dispersive tubular reactor		
File	Integrator	Description
TDFR_SeqMeth_Euler.m	none	Sequencing Method with Euler scheme for reaction part
TDFR_SeqMeth_Transition.m	none	Sequencing Method with transition matrix formulation for reaction part
TDFR_SeqMeth_Explicit.m	ode45	Sequencing Method with explicit ODE solver for reaction part
TDFR_SeqMeth_Implicit.m	ode15s	Sequencing Method with implicit ODE solver for reaction part
TDFR_MatMOL.m	ode15s	MatMOL with low- and high order spatial approximations Explicit algebraic boundary condition formulation (→ DAE formulation)
TDFR_MatMOL_ForcingFunction.m	ode45	MatMOL with low- and high order spatial approximations Forcing function approach for boundary conditions (→ ODE formulation)
TDFR_MatMOL_Elimination.m	ode45	MatMOL with low- and high order spatial approximations Elimination of algebraic boundary conditions (→ ODE formulation)
TDFR_MatMOL_FluxLimiter.m	ode15s	MatMOL with flux limiting functions (e.g., Koren, Minmod, Mc, Smart, Superbee, Van Leer)
TDFR_MatMOL_StaticRegridding.m	ode23s	MatMOL with static regridding
TDFR_MatMOL_DynamicRegridding.m	ode23s	MatMOL with dynamic regridding
Oil well		
File	Integrator	Description
Buckley_Leverett_MatMOL.m	ode15s	MatMOL with low- and high order spatial approximations
Buckley_Leverett_FluxLimiter.m	ode15s	MatMOL with flux limiting functions (e.g., Koren, Minmod, Mc, Smart, Superbee, Van Leer)
Buckley_Leverett_MatMOL_StaticRegridding.m	ode23s	MatMOL with static regridding
Buckley_Leverett_MatMOL_DynamicRegridding.m	ode23s	MatMOL with dynamic regridding

Table 1: Overview and description of the example files

References

[1] F. Logist, P. Saucez, J.F. Van Impe, and A. Vande Wouwer 2009. Simulation of (bio)chemical processes with distributed parameters using Matlab. *(submitted)*

[2] A. Vande Wouwer, P. Saucez, and W.E. Schiesser 2004. Simulation of distributed parameter systems using a Matlab-based method of lines toolbox: Chemical engineering applications, *Industrial and Engineering Chemistry Research*, 43, 3469-3477.

[3] S. Renou, M. Perrier, D. Dochain, and S. Gendron 2003. Solution of the convection-dispersion-reaction equation by a sequencing method. *Computers and Chemical Engineering*, 27, 615-629.