LifeQ

Comparison of QSS Solver's LIQSS2 solver with OpenModelica's DASSL solver

1 Introduction

The aim of this document is to compare QSS Solver's (stand-alone application) LIQSS2 solver [4] with OpenModelica's DASSL solver. The actual results and the simulation times are of importance. We will provide insight into determining whether the LIQSS2 solver is adequate to use as LifeQ's solver for its VHM models.

A number of models will be compared. The models considered in this document are the standard models that are found in QSS Solver's test model suite. These models were specifically created by the authors of QSS Solver to showcase the savings in execution times as opposed to traditional ODE solvers.

2 The solvers

All the solvers used have been mentioned in §1 of this document. These solvers include:

- OpenModelica DASSL (OM_DASSL)
- QSS Solver's LIQSS2 solver. (QSS_LI2)

3 Simulation details

For each simulation run, the following will be specified:

- duration of the simulation,
- some key state variables that will be investigated,
- the actual completion time of the simulation, and

- a conclusion of the obtained results.

For the DASSL solver in OpenModelica a tolerance of `1e-6` is used and the number of intervals will be fixed at 10 000 for all simulations.

In QSS Solver and absolute tolerance of `1e-9` and relative tolerance of `1e-6` is used.

An Intel(R) Core(TM) i7-4710MQ CPU @ 2.50GHz computer containing 8 processors and 16GB RAM, using operating system Linux Ubuntu 16.04 was used for the tests.

4 Results

The numerical results and related finding will be considered in this section. The computational times reported are in seconds and are the sum of the `user` time and `sys` time provided by the UNIX `time` command, which is used in conjunction with the executable that is built for each model. The `user` time is defined as the *total number of CPU-seconds that the process used directly (in user mode) in seconds*, while the `sys` time is the *total number of CPU-seconds used by the system on behalf of the process (in kernel mode), in seconds*. The total time `user+sys` indicates the total amount of actual CPU time a simulation uses.

4.1 Advection-Reaction model

This model is the Method of Line discretization of an Advection-Reaction model, which leads to the set of ODEs:

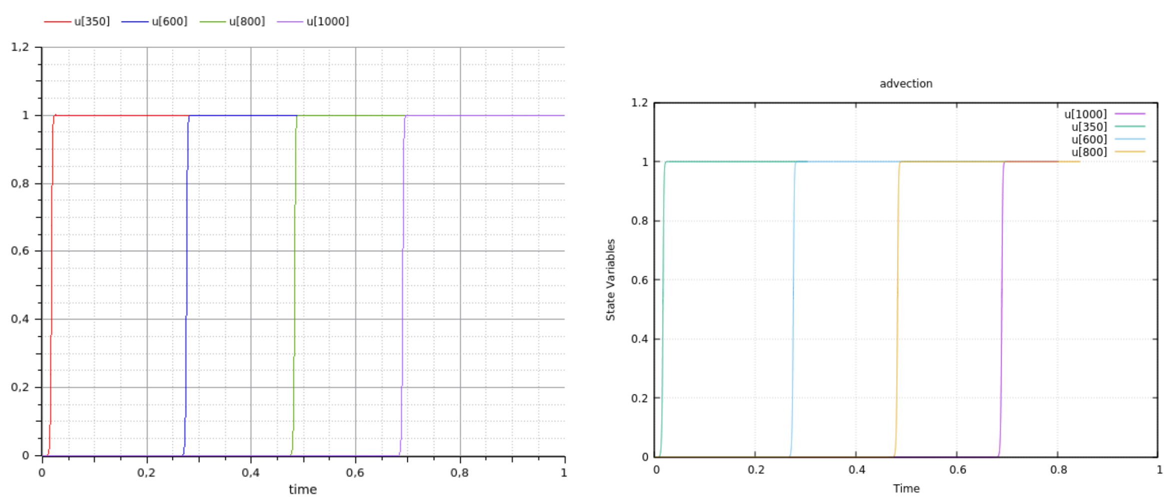
$$\dot{u}_i(t) \quad = \quad N \cdot (u_{i-1}(t) - u_i(t)) - \mu \cdot u_i(t) \cdot (u_i - \alpha) \cdot (u_i(t) - 1)$$

with some initial equations specified.

This model was simulated for 1 second and for a total of 1 000 state variables ($N = 1\,000$).

Solver	Sim time	Comments
OM_DASSL	9.716	Correct simulation results obtained
QSS_LI2	0.492	Correct simulation results obtained

Table 4.1: Simulation results of the Advection-Reaction model.



(a) DASSL (b) LIQSS2

Figure 4.1: Simulation results of the Advection-Reaction model.

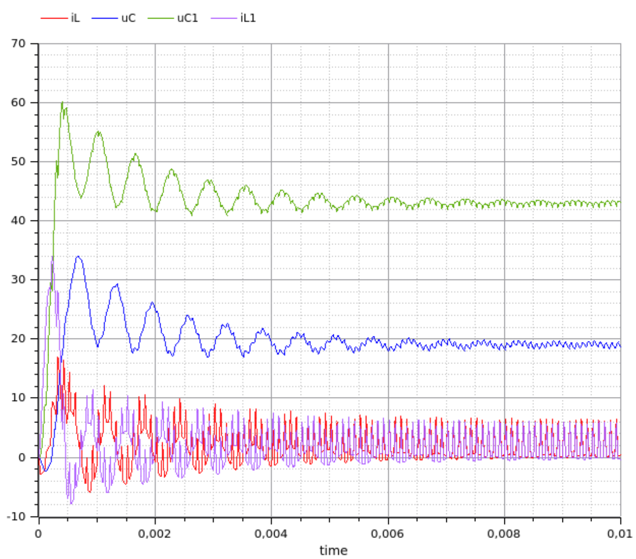
4.2 Buck Converter

A buck converter (step-down converter) is a DC-to-DC power converter which steps down voltage from its input to its output. The buck converter in this case consists of 4 branches.

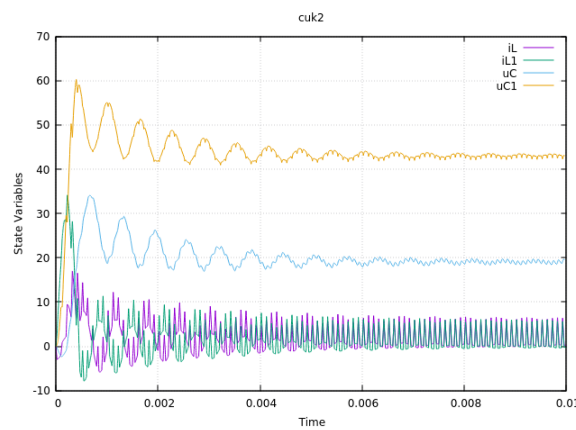
This model was simulated for 0.01 seconds.

Solver	Sim time	Comments
OM_DASSL	0.272	Correct simulation results obtained
QSS_LI2	0.016	Correct simulation results obtained

Table 4.2: Simulation results of the Buck Converter.



(a) DASSL



(b) LIQSS2

Figure 4.2: Simulation results of the Buck Converter.

4.3 Logical Inverter Chain

This model is the represents a chain of m logical inverters:

$$\dot{\omega}_j(t) = U_{op} - \omega_j(t) - \Upsilon \cdot g(\omega_{j-1}(t), \omega_j(t))^2$$

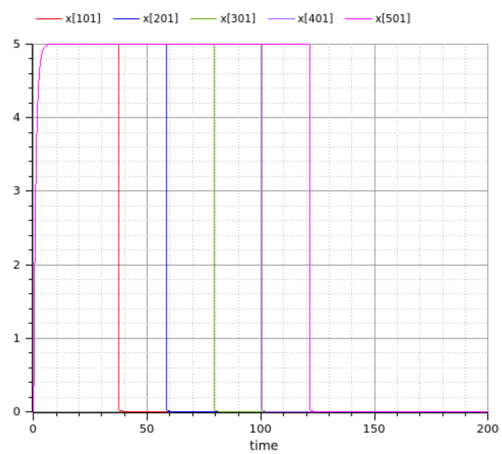
with $j = 1, \ldots, m$ where

$$g(u, v) = (\max(u - U_{th}, 0))^2 - (\max(u - v - U_{th}, 0))^2$$

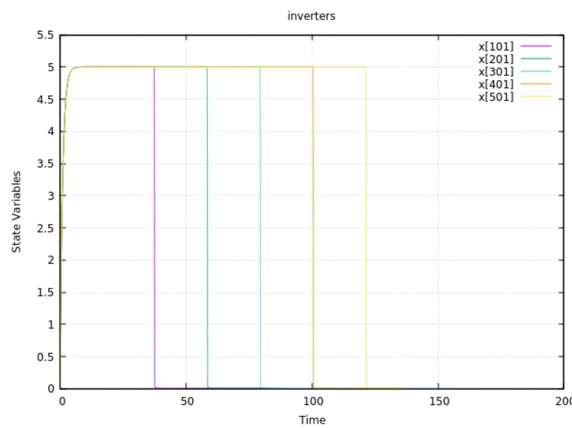
This model was simulated for 200 seconds. In this case we set $m = 501$. Parameter values and initial values have also been set.

Solver	Sim time	Comments
OM_DASSL	500.264	Correct simulation results obtained
QSS_LI2	0.160	Correct simulation results obtained

Table 4.3: Simulation results of the Logical Inverter Chain.



(a) DASSL



(b) LIQSS2

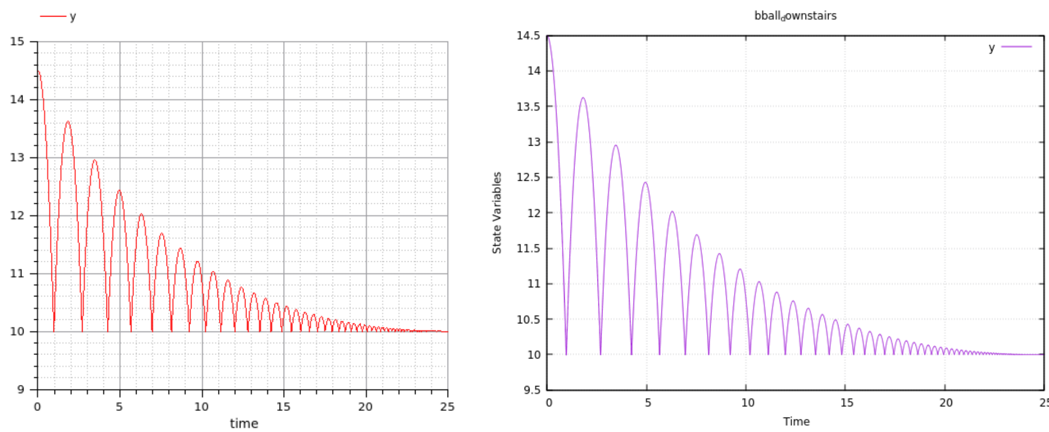
Figure 4.3: Simulation results of the Logical Inverter Chain.

4.4 Bouncing Ball

The bouncing ball is an example of a hybrid system, containing events and conditional expressions. This model was simulated for 25 seconds.

Solver	Sim time	Comments
OM_DASSL	0.020	Correct simulation results obtained
QSS_LI2	0.160	Correct simulation results obtained

Table 4.4: Simulation results of the Bouncing Ball.



(a) DASSL (b) LIQSS2

Figure 4.4: Simulation results of the Bouncing Ball.

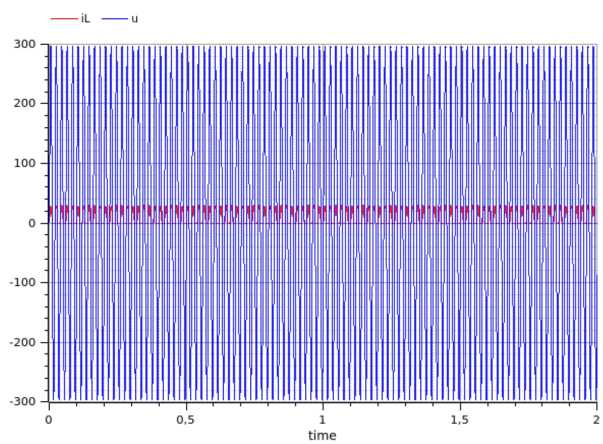
4.5 Rectifier

I have no comments on this model. It is merely a model that is provided by QSS Solver.

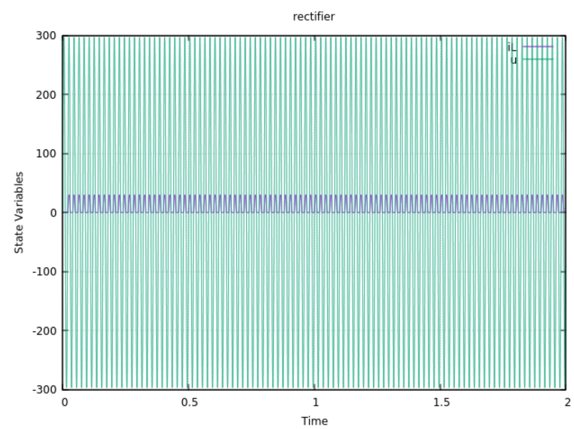
This model was simulated for 2 seconds.

Solver	Sim time	Comments
OM_DASSL	0.100	Correct simulation results obtained
QSS_LI2	1.548	Correct simulation results obtained

Table 4.5: Simulation results of the Rectifier.



(a) DASSL



(b) LIQSS2

Figure 4.5: Simulation results of the Rectifier.

5 Conclusion

The first three models considered were stiff mathematical systems and the LIQSS2 solver was faster in all cases. The last two models were not stiff systems and DASSL was the faster solver in this case. So, this begs the question: "*How dot he models compare when you have a both stiff components in the model as well as non-stuff ODE systems?*"

Lastly, I created a model that combines the advection-reaction model (a stiff mathematical system) with the bouncing ball (not a stiff mathematical system). From the results in the table below and some investigation both models take the largest portion of time simulating the stiff portion (the advection part). The bouncing ball part simulates quickly. The simulation run for this combination model was done for 5 seconds.

Solver	Sim time	Comments
OM_DASSL	10.206	Correct simulation results obtained
QSS_LI2	1.020	Correct simulation results obtained

Table 5.1: Simulation results of the the combination of the advection model and the bouncing ball example.

References

[1] OMCompiler, 2016, *LifeQ OMCompiler submodule repository*, [Online], Cited 15$^{\text{th}}$ March 2016, Available from `https://bitbucket.org/antonpdv/omcompiler`

[2] OpenModelica, 2016, *Open Source Modelica Consortium*, [Online], Cited 15$^{\text{th}}$ March 2016, Available from `https://openmodelica.org/`

[3] Petzold LR, 1982, *A description of DASSL: A differential/algebraic system solver*, Technical Report, Applied Mathematics Division, Sandia National Laboratories, Livermore (CA).

[4] QSS Solver, 2016, *Modeling and simulation tool for continuous and hybrid systems*, [Online], Cited 15$^{\text{th}}$ March 2016, Available from `https://sourceforge.net/projects/qssengine/`