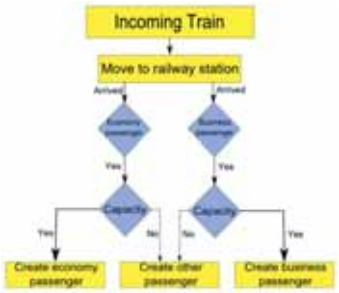
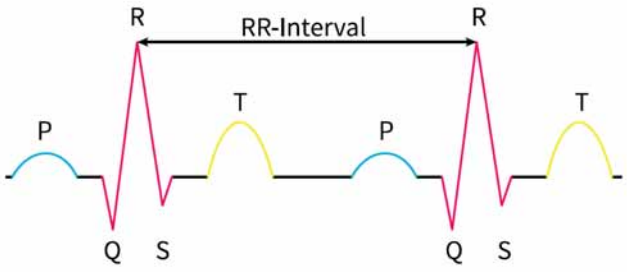
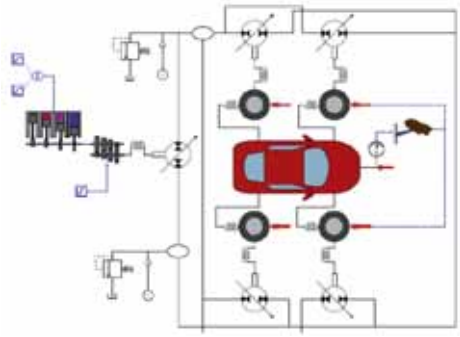
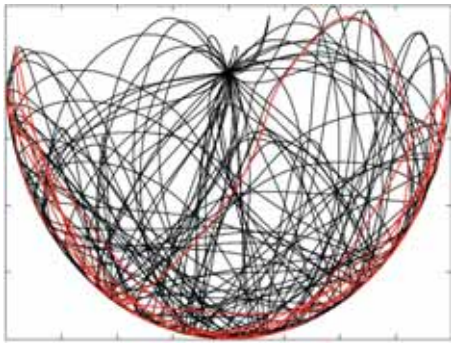
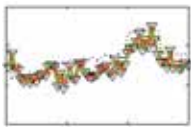
SNE SIMULATION NOTES EUROPE

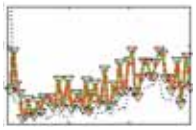


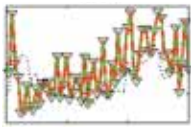


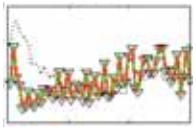












Volume 24 No.3-4 Dec. 2014
doi: 10.11128/sne.24.34.1025

Print ISSN 2305-9974
Online ISSN 2306-0271

Journal on Developments and Trends in Modelling and Simulation

Membership Journal for Simulation Societies and Groups in EUROSIM

ARGESIM

EUROSIM 2016

9th EUROSIM Congress on Modelling and Simulation

City of Oulu, Finland, September 12 – 16, 2016

EUROSIM Congresses are the most important modelling and simulation events in Europe. For EUROSIM 2016, we are soliciting original submissions describing novel research and developments in the following (and related) areas of interest: Continuous, discrete (event) and hybrid modelling, simulation, identification and optimization approaches. Two basic contribution motivations are expected: M&S Methods and Technologies and M&S Applications. Contributions from both technical and non-technical areas are welcome.

Congress Topics The EUROSIM 2016 Congress will include invited talks, parallel, special and poster sessions, exhibition and versatile technical and social tours. The Congress topics of interest include, but are not limited to:

- Intelligent Systems and Applications
- Hybrid and Soft Computing
- Data & Semantic Mining
- Neural Networks, Fuzzy Systems & Evolutionary Computation
- Image, Speech & Signal Processing
- Systems Intelligence and Intelligence Systems
- Autonomous Systems
- Energy and Power Systems
- Mining and Metal Industry
- Forest Industry
- Buildings and Construction
- Communication Systems
- Circuits, Sensors and Devices
- Security Modelling and Simulation
- Bioinformatics, Medicine, Pharmacy and Bioengineering
- Water and Wastewater Treatment, Sludge Management and Biogas Production
- Condition monitoring, Mechatronics and maintenance
- Automotive applications
- e-Science and e-Systems
- Industry, Business, Management, Human Factors and Social Issues
- Virtual Reality, Visualization, Computer Art and Games
- Internet Modelling, Semantic Web and Ontologies
- Computational Finance & Economics
- Simulation Methodologies and Tools
- Parallel and Distributed Architectures and Systems
- Operations Research
- Discrete Event Systems
- Manufacturing and Workflows
- Adaptive Dynamic Programming and Reinforcement Learning
- Mobile/Ad hoc wireless networks, mobicast, sensor placement, target tracking
- Control of Intelligent Systems
- Robotics, Cybernetics, Control Engineering, & Manufacturing
- Transport, Logistics, Harbour, Shipping and Marine Simulation

Congress Venue / Social Events The Congress will be held in the City of Oulu, Capital of Northern Scandinavia. The main venue and the exhibition site is the Oulu City Theatre in the city centre. Pre and Post Congress Tours include Arctic Circle, Santa Claus visits and hiking on the unique routes in Oulanka National Park.

Congress Team: The Congress is organised by SIMS - Scandinavian Simulation Society, FinSim - Finnish Simulation Forum, Finnish Society of Automation, and University of Oulu.

Esko Juuso EUROSIM President, Erik Dahlquist SIMS President, Kauko Leiviskä EUROSIM 2016 Chair

Info: www.eurosim.info, office@automaatioseura.fi

Editorial

Dear Readers – *This third issue of SNE Volume 24 comes with a different structure. While up to now contributions were grouped depending on the type of note (from technical note to benchmark note), this issue is grouped into the four thematic parts traffic, biology, mechatronics, and numerics, and each part consist of different types of contributions notes. For instance, the mechatronics part and the traffic part combine technical notes with short notes and benchmark notes. On the other hand side, the contributions are of different origin, because this issue also is based on the new submission strategy which invites individual submissions and post-conference publications from EUROSIM societies' conferences. In SNE 24(3-4), the post-conference publications have been submitted from SIMS Conference 2013 (SIMS – Scandinavian Simulation Society), and from ASIM's (German Simulation Society) Symposium Simulation Technique 2014. But this combination of different contributions types and different origin again demonstrates the fact, that modelling and simulation is a common interdisciplinary denominator in science and engineering.*

I would like to thank all authors for their contributions, and the organizers of the EUROSIM conferences for co-operation in post-conference publication, and the ARGESIM SNE staff for helping to manage the SNE administration and the improved SNE layout and extended templates for submissions (now also tex).

Felix Breitenecker, SNE Editor-in-Chief, eic@sne-journal.org; felix.breitenecker@tuwien.ac.at

Contents SNE 24(3-4)

SNE doi: 10.11128/sne.24.34.1025

Reader's Info

Simulation Notes Europe publishes peer r eviewed *Technical Notes*, *Short Notes* and *Overview Notes* on developments and trends in modelling and simulation in various areas and in application and theory, with main topics being simulation aspects and interdisciplinarity.

Individual submissof scientific papers are welcome, as well as post-conference publications of contribtions from conferences of **EUROSIM** societies.

Furthermore **SNE** documents the **ARGESIM Benchmarks** on *Modelling Approaches and Simulation Implementations* with publication of definitions, solutions and discussions (*Benchmark Notes*). Special *Educational Notes* present the use of modelling and simulation in and for education and for e-learning.

SNE is the official membership journal of **EUROSIM**, the Federation of European Simulation Societies. A News Section in **SNE** provides information for **EUROSIM** Simulation Societies and Simulation Groups.

SNE is published in a printed version (Print ISSN 2305-9974) and in an online version (Online ISSN 2306-0271). With **Online SNE** the publisher **ARGESIM** follows the **Open Access** strategy, allowing download of published contributions for free. Since 2012 **Online SNE** contributions are identified by a DOI (Digital Object Identifier) assigned to the publisher **ARGESIM** (DOI prefix 10.1112 8). **Print SNE**, high-resolution **Online SNE**, full **SNE Archive**, and source codes of the *Benchmark Notes* are available for members of **EUROSIM** societies.

SNE Print ISSN 2305-9974, SNE Online ISSN 2306-0271

SNE Issue 24(3-4) Dec. 2014 doi: 10.11128/sne.22.34.1025

→ *www.sne-journal.org*

office@sne-journal.org, eic@sne-journal

SNE Editorial Office, c/o ARGESIM / DWH, Neustiftgasse 57-59, 1070 Vienna, Austria

SNE Editorial Board

Author's Info

Authors are invited to submit contributions which have not been published and have not being considered for publication elsewhere to the **SNE** Editorial Office. Furthermore, SNE invites organizers of EUROSIM conferences to provide post-conference publication for the authors of their conference (with peer review).

SNE distinguishes different types of contributions (*Notes*):

- *Overview Note* – State-of-the-Art report in a specific area, up to 14 pages, only upon invitation
- *Technical Note* – scientific publication on specific topic in modelling and simulation, 6 – 8 (10) pages
- *Education Note* – modelling and simulation in / for education and e-learning; max. 6 pages
- *Short Note* – recent development on specific topic, max. 4 p.
- *Software Note* – specific implementation with scientific analysis, max 4 pages
- *Benchmark Note* – Solution to an ARGEIM Benchmark; basic solution 2 pages, extended and commented solution 4 pages, comparative solutions 4-8 pages

Further info and templates (doc, tex) at **SNE's** website.

SNE Contact & Info

→ *www.sne-journal.org*
✉ *office@sne-journal.org, eic@sne-journal.og*
✉ SNE Editorial Office, ARGESIM / dwh Simulation Services, Neustiftgasse 57-59, 1070 Vienna, Austria

SNE SIMULATION NOTES EUROPE

ISSN SNE Print ISSN 2305-9974, SNE Online ISSN 2306-0271

WEB: → *www.sne-journal.org*, DOI prefix 10.11128/sne

Scope: Technical Notes, Short Notes and Overview Notes on developments and trends in modelling and simulation in various areas and in application and theory; benchmarks and benchmark documentations of ARGESIM Benchmarks on modelling approaches and simulation implementations; modelling and simulation in and for education, simulation-based e-learning; society information and membership information for EUROSIM members (Federation of European Simulation Societies and Groups).

Editor-in-Chief: Felix Breitenecker, Vienna Univ. of Technology, Inst. f. Analysis and Scientific Computing, Div., Math. Modelling and Simulation, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria; ✉ *Felix.Breitenecker@tuwien.ac.at,* ✉ *eic@sne-journal.org*

Layout / Administration: J. Tanzler, F. Preyser, T. Wobruba; C. Wytrzens, R. Leskovar et al.; Math. Modelling and Simulation Group, Vienna Univ. of Technology, Wiedner Haupstrstasse 8-10, 1040 Vienna , ✉ *office@sne-journal.org*

Print SNE: Grafisches Zentrum, TU Vienna, Wiedner Hauptstrasse 8-10, 1040, Vienna, Austria

Online SNE: ARGESIM /ASIM, c.o. dwh Simulation Services, Neustiftgasse 57-59, 1070 Vienna, Austria

Publisher: ARGESIM ARBEITSGEMEINSCHAFT SIMULATION NEWS - WORKING COMMITTEE SIMULATION NEWS, Neustiftgasse 57-59, 1070 Vienna, Austria; → *www.argesim.org,* ✉ *info@argesim.org* on behalf of ASIM(→ *www.asim-gi.org* and EUROSIM → *www.eurosim.info*

Model-based Parallelization of Discrete Traffic Simulation Models

Oliver Ullrich[1*], Daniel Lückerath[1], Ewald Speckenmeyer[2]

[1] National Science Foundation's Industry-University Cooperative Research Center, School of Computing and Information Sciences, Florida International University, ECS 243C, 11200 SW 8th St, Miami FL-33199; *oullrich@fiu.edu

[2] Institut für Informatik, Universität zu Köln, Albertus-Magnus-Platz, 50923 Köln, Germany

Simulation Notes Europe SNE 24(3-4), 2014, 115 - 122
DOI: 10.11128/sne.24.tn.10251
Received: September 28, 2014 (Selected ASIM SST 2014 Postconf. Publ.); Accepted: November 3, 2014;

Abstract. To re-establish regular operations in a tram traffic network after a large disturbance, e.g. resulting from vehicle breakdown or station closure, the viability of several rescheduling and rerouting strategies has to be evaluated prior to their implementation. Here, a multi-modal traffic simulation system can help to enhance the decision quality. Such a system obviously faces tight time constraints, so simulation data has to be acquired fast.

In this paper we propose a method for the parallel execution of discrete traffic simulation models, which would accelerate data generation in comparison to a sequential model. To assess this method's dynamic behavior in real-world applications, some experiments conducted on a software system modeling schedule based tram traffic are presented.

After giving an introduction to the scope and aim, we show some background on the parallelization of discrete simulation models. The main part of the paper begins with the proposal of a method to parallelize the execution of simulation models with problem specific properties. Some estimations of the method's efficiency are shared, followed by several experiments to highlight its dynamic behavior in real-world applications.

The paper ends with a short summary and some thoughts on further research.

Introduction

When severe disturbances occur in tram networks, e.g. originating from broken down trams, closed stations, or other blocked resources, traffic operators have to apply rescheduling and rerouting strategies (see [10] and [12]) to reestablish regular operations. To be effective, these strategies are inevitably multimodal: trams are rescheduled to compensate for cancellations, regional and local buses are rerouted to relieve the tram network, some transit operators even co-operate with taxi companies (see [26]). To evaluate the applicability of a given rescheduling or rerouting strategy prior to its implementation in the realworld system, a multi-modal simulation software is needed. Operators obviously face tight time constraints for their decisions, so simulation data has to be acquired fast.

In this paper we propose a method for the parallel execution of discrete traffic simulation models. We do not aim for a general approach, which would be equally well applicable for all discrete models, but for a method that utilizes some specific properties of a subclass of models, including traffic simulation models. The traffic planners' laptop or desktop computers constitute the target platform of the resulting simulation tools; the method thus should utilize their capacity for small scale parallel processing. To employ the available resources effectively, the method applies a dynamic and adaptive load balancing scheme. From the model's point of view, the mechanics of parallelization and load balancing are transparent, so that these internals can be changed without compromising the subsequent use of the model. A sequential model of tram traffic is already in place (see [11]); the accompanying representation of bus traffic is only partly implemented yet (see [24]).

This paper continues with a presentation of some background on general methods of parallel simulation and their customization in practical applications (section 1). We then present an approach on parallel execution of simulation models that share some characteristics like being spatially explicit, having mostly local dependencies, and having computational load generated by transient entities moving through a network (section 2), followed by some efficiency estimations (section 3). Based on an implementation of the approach, some experiments are conducted, focusing on the parallel computation of artificial loads, and the parallel simulation of tram traffic (section 4). The paper closes with a short summary of the lessons learned and some thoughts on future work (section 5).

1 Background

A discrete simulation model comprises of entities which communicate with each other via simulation events or messages, and which change their states at discrete points in model time. The main task when parallelizing a discrete simulation model is to avoid the occurrence of parallelization artifacts, i.e. to assure that parallel execution yields the same results as sequential execution. This is not a trivial task because a model usually contains both spatial dependencies (which are often local, i.e. between neighboring entities) and temporal dependencies (which are also often local, between close points in model time).

Several approaches to the parallelization of discrete simulation models are known: The simplest approach is the concurrent execution of several sequential experiments (see [15]). Outsourcing of maintenance functions (e.g. random number generation or database access) to secondary processors is also relatively simple but usually does not scale very well. A special case of this is the parallel administration of central data structures, usually applied to the future event list (for an overview of suitable data structures, see [18]). Time based parallelization techniques, i.e. computing different simulation time intervals in parallel, are scaling very well in principal but are only suited for specific models (a few applications are known, as described in [6] or [7]).

A well examined, and also well scaling technique is model based (also called spatial) parallelization, which can be found in both literature (see [5], pp. 39) and real-world applications (e.g. see [1], [13] or [19]).

1.1 Model based parallelization

Model based parallelization utilizes the model's inherent parallelism, i.e. that many state changes can be executed independently from each other. To achieve this, the model is decomposed into partial models, which are assigned to the involved processors. With this method, entities of different partial models communicate via messages sent over the network or a common cache memory.

Obviously, careful synchronization of the model's execution is necessary. The local causality constraint (see [5], pp. 52) demands that each entity executes its concerning simulation events in a non-decreasing order regarding their scheduled time of occurrence. Non-adherence to this constraint may result in causality errors which invalidate the simulation results. Two categories of synchronization methods are known: Conservative synchronization methods prevent the out-of-order execution of simulation events by technical measures, thus guaranteeing adherence to the local causality constraint. Important conservative synchronization mechanisms include synchronization via null messages (described in [2] and [3]), deadlock detection and recovery (described in [4]), and synchronized execution (see [17] and [21]). Optimistic synchronization methods execute the partial models as fast as possible, and thereby allow violations of the local causality constraint. The methods detect and subsequently repair these violations by rejecting and re-computing the invalid regions of the simulation run. The best known optimistic approach is the aptly named “time warp”, first described in [8].

1.2 Load balancing for parallel discrete simulation systems

Resulting from the typical dependencies in simulation models, the simulation's execution speed is generally dependent on the processor which advances slowest in simulation time. It is therefore necessary to incorporate a load balancing system into the simulation engine. This system does not aim at high utilization of the processors' capacity alone, but also has to consider an uniform advance in simulation time. Discrete simulation systems therefore often apply special load balancing schemes. Those methods can be characterized as dynamic, static, adaptive, non-adaptive, local, centered, or hierarchical (see [14], pp. 12).

A dynamic load balancing method continuously considers imbalances which develop from shifts in the model's computational load, and re-assigns partial models to appropriate processors while executing the simulation run. In contrast, static methods estimate the load and assign partial models to processors in a preprocessing step before the start of the simulation run, and thus don't consider dynamic changes in the model's activity. Adaptive methods consider fluctuations in the available processor power originating from the demand of dynamic processes belonging to third parties. In inhomogeneous computer networks adaptive methods also consider the dissimilar performance power of the respective processors. A non-adaptive approach ignores those fluctuations. In local methods, the processors only exchange data within limited neighborhoods and act on this local information alone, while centered methods utilize a marked controller process to which the other processors report. Hierarchical methods usually organize communication in a tree topology.

A load balancing method viable for traffic simulation models on a PC or laptop computer should be both dynamic and adaptive, and thus has to consider both the varying computational load of the model, and the changing availability of resources on a nonexclusively used machine. Such a dynamic and adaptive method requires a load measure which considers both the available processing power and the engaged processors' advance in simulation time. A centered method is usually easier to implement and quite adequate for a PC system with only eight to sixteen processor cores; if a method is targeted at a massively parallel system it should avoid a potential bottleneck by utilizing a hierarchical or local scheme.

2 An Approach to the Parallelization of Discrete Traffic Simulation Models

To be viable for the proposed method, a model has to comply to some prerequisites: It has to be spatially explicit and representable as a sparsely populated graph; dependencies have to be typically local, so that neighborhood relationships can therefore be exploited; the model's activity, and thus the computational load, has to be produced by transient entities which steadily move through the model, so the load typically shifts slowly and is caused by several simulation steps.

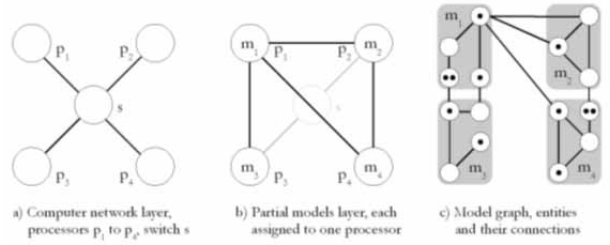


Figure 1: Three layers of abstraction.

Many traffic simulation models comply with these prerequisites (see e.g. [11], [16], and [24]). The proposed parallelization method is flexible regarding simulation paradigms: the implemented models can be event based, process based, agent based, or be based on the activity scanning approach. The method builds upon three layers of abstraction (see Figure 1): On the computer network layer processors and processor cores form a star-shaped graph, connected by a local area network or a shared cache memory; the partial models are assigned to these processors, and connected by the communication occurring during the simulation run; the model graph builds up on that and consists of model nodes, which represent entities, and their connecting edges. Transient entities (here represented by tokens) map the dynamically changing model activity; they move through the model graph along its edges.

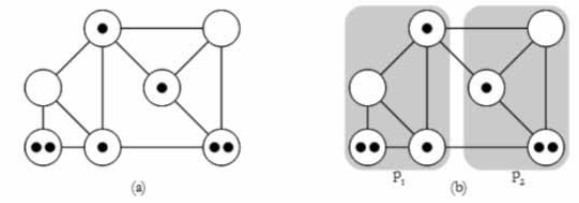


Figure 2: Static load balancing.

Before the start of a simulation run, the model is partitioned, and the resulting partial models are assigned to the participating processors (see Figure 2). A heuristic method (see [9]) is applied in this static load balancing step to reduce the number of edges between partitions, and thus to reduce the communication load. During the simulation run, tokens move from node to node and thus generate load imbalances, which have to be handled dynamically (see Figure 3).

To accomplish this, a processor p_i which is overcharged, but has a neighbor p_j which is not fully loaded, selects a number of model nodes and shifts them over to p_j (see Figure 4). This adjustment is done iteratively in the course of several simulation steps, until a stable state is reached.

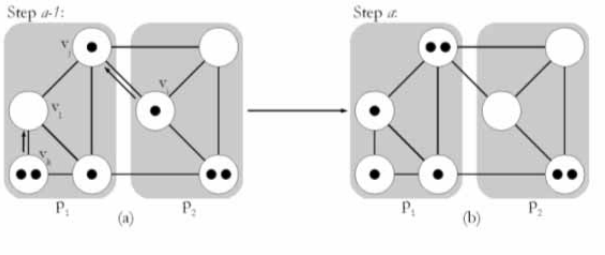


Figure 3: Dynamic load shifting.

The method uses a dynamically calculated load measure, considering both the changes in the model's activity and the time elapsing while computing, and therefore is dynamic and adaptive. It also exploits knowledge of regional dependencies to keep down communication load; existing neighborhood relations are not affected by the mechanism. Load balancing is carried out when all processors have entered the synchronization barrier (see [5], pp. 65-96), and are thus done with processing step t, but did not yet start with processing step $t+1$. The method prefers to shift model nodes from a slow processor p_s to a fast processor p_f in a way that iteratively further reduces the communication load during the simulation run. To accomplish this, it classifies each model node v in the set V_{ps} of all nodes hosted by processor p_s in one of four priority classes:

4. All $v_i \in V_{ps}$:
3. each node v_i in 4 which has an edge to a node $v_j \in V_{\acute{p}}$, with any $\acute{p} \neq p_s$;
2. each node v_i in 3 which has an edge to a node $v_j \in V_{pf}$, which is hosted by a processor p_f which is not operating at full capacity; and
1. each node v_i in 2 with a greater number of edges to nodes hosted by processor p_f than to nodes administrated by p_s.

The method prefers to shift nodes from class 1, followed by class 2 and 3. Nodes which are only members of class 4 are not shifted

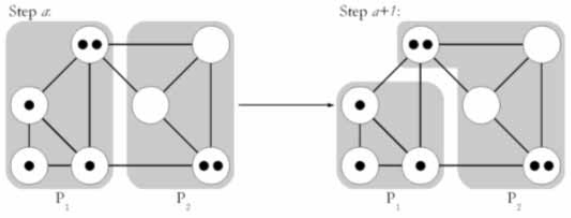


Figure 4: Dynamic load balancing.

Each simulation step consists of three phases: The computation of the model, the duration of which is dependent on the partial model's activity, and which the load balancing method tries to distribute equally over all processors; the synchronization phase, in which processors already finished with computing wait for the others to finish their tasks; and the communication and load balancing phase, in which the method shifts model nodes from fully loaded processors to underloaded ones. The load balancing step itself consists of three phases (see Figure 5): load measurement, load assessment, and load shifting. For a more detailed description of the method see [22], pp. 61-76.

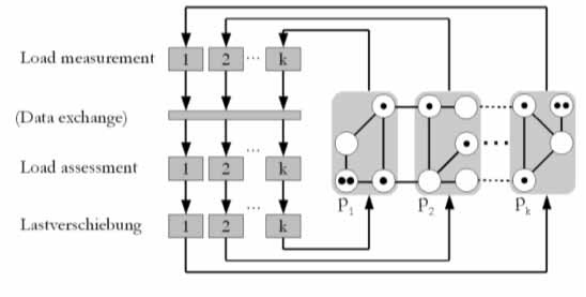


Figure 5: Phase of the load balancing process.

3 Scalability and Efficiency

As described, each simulation step i consists of three phases: computation of the model, synchronization, and communication, whose computational complexity for each processor p is denoted by $t_m(p,i)$, $t_s(p,i)$, and $t_c(p,i)$ (see Figure 6). These values can be estimated, which in turn yields estimations for the scalability and efficiency of the proposed method.

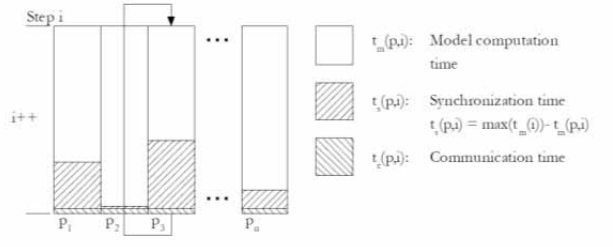


Figure 6: Phase of the simulation step.

The combined complexity $t_g(i)$ of each simulation step i in a system with k processors for a bad case (load balancing mechanism is switched off and computational loads for each processor p in each step i are drawn randomly, for a detailed description see [22], pp. 76-84) is shown in equation 1:

$$t_g(i) = \underbrace{c_m * \left(1 - \frac{1}{k+1}\right)}_{t_m(p,i)+t\_s(p,i)} + \underbrace{6 * (k-1) + c_n * \frac{k+1}{k\text{^}2}}_{t_c(p,i)} \quad (1)$$

Here, c_m denotes the computational load generated by the model, and c_n denotes the number of transient entities to be moved between model nodes.

The average time complexity $t_g(i)$ of a single step i (load balancing mechanism is switched on, computational loads are shifting smoothly through the model) is shown in equation 2:

$$t_g(i) = \underbrace{\frac{c_m}{k}}_{t_m(p,i)} + \underbrace{6 * (k-1) + c_n * \frac{k+1}{k^2} + 2 * c_l * (k-1)}_{t_c(p,i)} \quad (2)$$

Here, c_l denotes the number of resident entities which have to be transferred in the context of load balancing. The scalability of the method is thus mainly dependent on the values c_m and c_n, and in the average case also on c_l. These values are all properties of the model, and are thus not influenced by the method itself. In the average case, with the partial model's loads shifting in a benign way, the method shows linear scaling. The manifesting scaling factor for an individual model is directly dependent on the ratio of its computation load $t_m(p,i)$ to its communication load $t_c(p,i)$. Or, to put it simple: Bigger models scale better.

The expected efficiency of the method for a model with unfavorable arbitrary load imbalances can be shown to be always greater than 0.5 (see equation 3).

$$E\left(t_{busy}(k)\right) > E\left(t_{idle(k)}\right) \quad \forall k > 1 \quad (3)$$

For a detailed analysis of computational complexity, scalability and efficiency, see [22], pp. 76-86.

4 Experiments

The proposed method was implemented as a C++ framework (described in [22], chapter 4), and is utilized in two different scenarios. To keep influences of a complex real-world model with often irregular properties at a minimum, the framework is first applied to the computation of artificial loads moving through a randomly generated graph. This is followed by a real-world application in the simulation of timetable based tram traffic. Experiments are run using notebook computers with 6 gigabytes memory and an Intel Core i7-740QM processor with four cores running at 1.73 gigahertz. The turbo boost technology for accelerating single thread applications is turned off for the experiments. For experiments with more than four processor cores, several notebooks are connected by a 100 megabit ethernet switch.

4.1 Computation of artificial loads

The first set of experiments is conducted on the parallel computation of artificial load generated by token movements on a randomly generated graph. Tokens stay at a node for a certain number of simulation steps, and then move over an edge to a randomly picked neighboring node. During each simulation step, these tokens generate computational load. This is generated by executing the Sieve of Eratosthenes algorithm (see [20], pg. 85) to identify prime numbers up to an upper bound. This upper bound q_{max} is set to the sum of the base load of the node v_i and the token's weights: $q_{max} = l_{base} + l_{token} * |T_i|$.

To generate the graph, n nodes with a base load of $l_{base} = 10.000$ are generated. For each node v, two nodes $u_1 \neq v$ and $u_2 \neq u_1 \neq v$ not yet connected to v are chosen randomly. Then, two edges $v \leftrightarrow u_1$ and $v \leftrightarrow u_2$ are added to connect v to those nodes. A token is generated for every fifth node v_i with $i \bmod 5 = 0$. Each of those tokens has a weight of $l_{token} = 10.000$, a maximum retention period of $t_{max} = 100$ simulation steps, and a current retention time t_t drawn from a uniform distribution between 0 and t_{max}. This value t_t is decreased by one during each simulation step. When it reaches zero, an edge outgoing from its current host node is selected randomly; the token is then moved over this edge and is reinitialized by its new host with $t_t = t_{max}$. In the conducted experiments, each simulation run consists of 500 steps, the load balancing scheme is active.

#Proc.	Runtime (sec)	Speedup	Marg. utility
1	2,010.0	1.00	1.00
2	1,010.3	1.99	0.99
3	682.9	2.95	0.96
4	516.5	3.89	0.94
5	421.0	4.77	0.88
6	372.1	5.40	0.63
7	340.9	5.90	0.49
8	325.8	6.17	0.27

Table 1: Runtime and speedup for the computation of artificial loads.

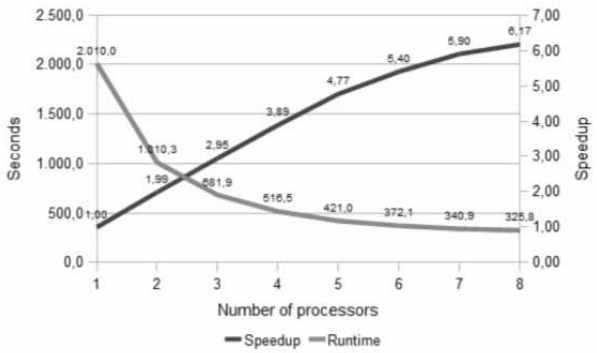


Figure 7: Runtime and speedup for the computation of artificial loads.

We begin by running a mid size instance of $n = 400$ nodes on up to eight processor cores. Average runtime, speedup values, and marginal utility are shown in Table 1 and Figure 7. A second series of asymptotic experiments begins with a graph consisting of $n = 100$ nodes and 200 edges on a single processor core, going up to $n = 800$ nodes on eight processors. Average runtime, scaling factor and marginal utility are shown in Table 2 and Figure 8.

#Proc.	Graph		Scalability		
	|V|	|E|	Runtime (sec)	Scaling factor	Marg. utility
1	100	200	495.8	1.00	1.00
2	200	400	499.1	1.99	0.99
3	300	600	523.4	2.84	0.86
4	400	800	521.4	3.80	0.96
5	500	1,000	521.7	4.75	0.95
6	600	1,200	550.5	5.40	0.65
7	700	1,400	551.8	6.29	0.89
8	800	1,600	568.4	6.98	0.69

Table 2: Runtime and scaling factor for the computation of artificial loads.

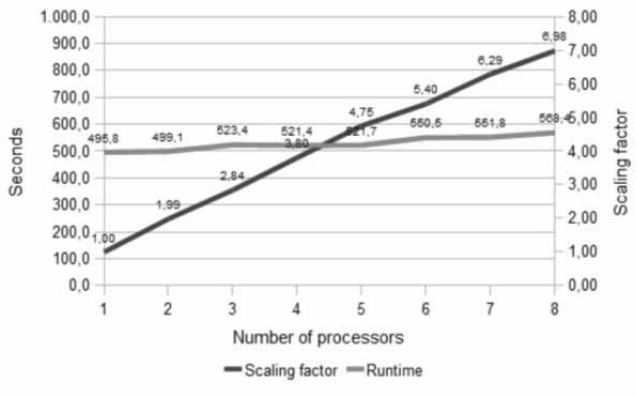


Figure 8: Runtime and scaling factor for the computation of artificial loads.

4.2 Simulation of time-table based tram traffic

The proposed method was then utilized to parallelize a sequential simulation engine of time-table based tram traffic (described in [11]). The resulting software tools were applied to the KVB network of Cologne, Germany (see [23]), and the TAM Tramway network of Montpellier, France (see [25]).

The described experiments are conducted on a model of Cologne's network. It consists of 528 platforms and 58 track switches connected via 584 tracks. These tracks cover a total length of 407.4 kilometers, resulting in an average track length of 697.6 meters. 15 lines with 182 line routes are served by 178 vehicles which execute 2,814 trips per operational day. The simulation engine was run on up to eight processors of the described type, under the parameter set described in section 4.1. Average runtime, speedup and marginal utility are shown in Table 3 and Figure 9.

4.3 Results and discussion

The simulation models are executed 10 times per measuring point. For the computation of artificial loads (see Figure 7 and Table 1), the method yields a high gain in speedup for up to five processors (speedup 4.72), which flattens when more processors are added. With partial models getting smaller, the ratio of synchronization and communication time to model computation time rises, so efficiency is declining. For the asymptotic experiments (see Figure 8 and Table 2) a linear regression yields a function $s(k) = 0.86 * k + 0.27$ for scaling, and $T(k) = 10.16 * k + 483.3$ for run time. Under the described conditions the method thus shows a linear scalability with a scaling factor of around $0.86 * k$.

The simulation of time-table based tram traffic shows mixed results (see Figure 9 and Table 3): For up to four processor cores - based on a single parallel computer - the speedup rises to 2.83, and then caves in to 1.89 when the fifth processor - connected via LAN - is added. The reason for this behavior is the significantly higher communication cost between LAN connected computers in relation to communication between parallel processor cores. The ratio of high communication cost to a relatively low computation cost for the distributed partial models forbids an effective execution on LAN connected computers. A linear regression for the first four measuring points yield a function of $z(k) = 0.62 * k + 0.5$ for speedup, and $T(k) = -55.03 * k + 289.5$ for run time. The last four points yield functions of $z(k) = 0.12 * k + 1.3$ and $T(k) = -7.71 * k + 177.3$. The model instance is therefore large enough to be efficiently run on a parallel computer, but too small to be executed expediently on LAN connected computers. For a more in-depth discussion of the experiments see [22], chapters 5 and 6.

#Proc.	Runtime (sec)	Speedup	Marg. utility
1	263.8	1.00	1.00
2	144.5	1.82	0.82
3	106.4	2.44	0.62
4	93.1	2.83	0.39
5	139.8	1.89	-0.94
6	130.6	2.02	0.13
7	121.0	2.12	0.10
8	117.3	2.25	0.13

Table 3: Runtime and speedup for the simulation of time-table based tram traffic.

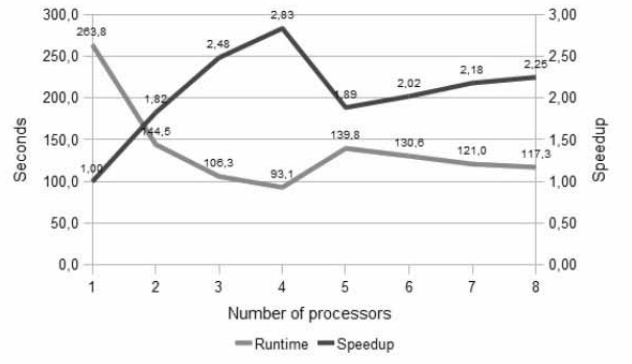


Figure 9: Runtime and speedup for the simulation of time-table based tram traffic.

5 Summary and Further Research

We presented an approach to the parallel execution oftraffic simulation models, which includes a dynamicand adaptive load balancing scheme. Some thoughtson scalability and efficiency were shared: even underadverse circumstances the efficiency does not get lower than 0.5. Experiments conducted on the computation of artificial loads yield a speedup of 3.89 on four processors and 6.17 on eight processors. Parallel execution of a tram traffic model shows a speedup of 2.83 on four processor cores. A higher speedup might be reached when executing a larger model. In a next step, the multimodal model will be extended by a representation of bus transit.

Acknowledgements

This material is partially based upon work supported by the National Science Foundation under grants I/UCRC IIP-1338922, AIR IIP-1237818, SBIR IIP- 1330943, III-Large IIS-1213026, MRI CNS-0821345, MRI CNS-1126619, CREST HRD-0833093, I/UCRC IIP-0829576, MRI CNS-0959985, FRP IIP-1230661, and U.S Department of Transportation under a 2013 TIGER grant.

This contribution is a post-conference publication from ASIM SST 2014 (22nd Symposium Simulationstechnik, Berlin, September 3-5, 2014, HTW Berlin). The original contribution was published in Tagungsband ASIM 2014, 22. Symposium Simulationstechnik, J. Wittmann and Ch. Deatcu (Eds.), ASIM Mitteilung 151, ARGESIM Report 43, ISBN 978-3-901608-44-5, ARGESIM Verlag, Wien, 2014.

References

[1] Avril H, Tropper C. The Dynamic Load Balancing of Clustered Time Warp for Logic Simulation. In: *Proceedings of the tenth workshop on Parallel and distributed simulation*; 1996 May; P. 20-27

[2] Bryant RE. Simulation of Packet Communication Architecture Computer Systems. Computer Science Laboratory. Cambridge, Massachusetts, Massachusetts Institute of Technology, 1977. Report

[3] Chandy KM, Misra J. Distributed Simulation: A Case Study in Design and Verification of Distributed Programs. *IEEE Transactions on Software Engineering. 1979;* SE- 5(5): 440-452.

[4] Chandy KM, Misra J. Asynchronous distributed simulation via a sequence of parallel computations. *Communications of the ACM*. 1981; 24(4): 198-205

[5] Fujimoto RM. *Parallel and Distributed Simulation*. New York: John Wiley & Sons, 2000

[6] Greenberg AG, Lubachevsky BD, Mitrani I. Algorithms for Unboundedly Parallel Simulations. *ACM Transactions on Computer Systems*: 1991; 9(3): 201-221

[7] Heidelberger P, Stone H. Parallel Trace-Driven Cache Simulation by Time Partitioning. In: *Proceedings of the 1990 Winter Simulation Conference*. 1990; 734-737

[8] Jefferson DR. Virtual Time. *ACM Transactions on Programming Languages and Systems.* 1985; 7(3): 404-425

[9] Kernighan BW, Lin S. An Efficient Heuristic Procedure for Partitioning Graphs. *Bell Syst. Tech Journal*.1970; 49(2): 291-307

[10] Lückerath D. Thoughts on restauration of regular tram operation. In: *Proceedings of Sommertreffen Verkehrssimulation 2012*, AM 143, ARGESIM/ASIM Pub., TU Vienna, pp. 4-6, 2012.

[11] Lückerath D, Ullrich O, Speckenmeyer E. Modeling time table based tram traffic. *Simulation Notes Europe.* 2012; 22(2): 61-68

[12] Lückerath D, Ullrich O, Speckenmeyer E. Applicability of rescheduling strategies in tram networks. In: *Proceedings of ASIMWorkshop STS/GMMS 2013*. ARGESIM Report 41, AM 145, ARGESIM/ASIM Pub., TU Vienna, 2013.

[13] Meisgen F. Dynamic Load Balancing for Simulations of Biological Aging. *International Journal of Modern Physics C*. 1997; 8(3): 575-582

[14] Meisgen F. *Dynamische Lastausgleichsverfahren in heterogenen Netzwerken*. Aachen: Shaker Verlag, 1998.

[15] Merz M, Bröcker E.: Einsatz von Open Source Frameworks zur Parallelisierung von Dymola Simulationen. In: Proc. ASIM/GI Workshop STS/GMMS, Ed.: W. Commerell, ISBN 978-3-9810998-3-6, Ulm, März 04-05, S. 283-289, 2010.

[16] Nagel K, Schreckenberg M. A cellular automaton model for freeway traffic. *Journalde Physique I*. 1992; 2(12): 2221-2229

[17] Nicol DM. The cost of conservative synchronization in parallel discrete event simulations. *Journal of the Association of Computing Machinery*. 1993; 40(2): 304-333

[18] Rönngren R, Ayani R. A Comparative Study of Parallel and Sequential Priority Queue Algorithms. *ACM Transactions on Modeling and Computer Simulation*. 1997; 7(2): 157-209

[19] Schlagenhaft R. Dynamischer Lastausgleich optimistisch synchronisierter, verteilter Simulation. In: *Proc. ASIM-Workshop VSPP*. 1999

[20] Sedgewick R. *Algorithms in C++, Vol. 1.* Boston: Addison-Wesley, 1998.

[21] Steinman J. SPEEDES: *Synchronous parallel environment for emulation and discrete event simulation.* Advances in Parallel and Distributed Simulation, SCS Simulation Series.1991; Vol. 23: 95-103

[22] Ullrich O. *Modellbasierte Parallelisierung von Anwendungen zur Verkehrssimulation - Ein dynamischer und adaptiver Ansatz.* Dissertation, Univ. Köln, 2014.

[23] Ullrich O, Lückerath D, Franz S, Speckenmeyer E. Simulation and optimization of Cologne's tram schedule. *Simulation Notes Europe* . 2012; 22(2): 69-76

[24] Ullrich O, Proff I, Lückerath D, Kuckertz P, Speckenmeyer E. Agent-based modelling and simulation of individual traffic as an environment for bus schedule simulation. In: *mobil.TUM 2013*. Proceedings of mobil.TUM 2013. 2013 Jun

[25] Ullrich O, Lückerath D, Speckenmeyer E. A robust schedule for Montpellier's Tramway network. In: *ASIM 2014*.Proceedings of ASIM 2014 - 22nd Symposium on Simulation Technique; 2014 Sept, Berlin

[26] Zeng AZ, Durach CF, Fang Y. Collaboration decisions on disruption recovery service in urban public tram systems.*Transportation Research Part E*.2012; 48(3): 578-590

Agent-based Simulation of the Railway Connection from and to the Vienna International Airport

Matthias Obermair[2], Barbara Glock[1*]

[1]dwh GmbH, Simulation Services, Neustiftgasse 57-59, 1070 Vienna, Austria; [*]*barbara.glock@dwh.at*
[2]Vienna Univ. of Technology, Inst. of Analysis and Scientific Computing, Karlsplatz 13, 1040 Vienna, Austria;

Simulation Notes Europe SNE 24(3-4), 2014, 123 - 126
DOI: 10.11128/sne.24.sn.10253
Received: March 25, 2014; Revised September 10, 2014;
Accepted: October 20, 2014;

Abstract. This agent-based simulation deals with the railway connection of the Vienna International Airport and was created using the simulation program AnyLogic. With this program a transportation company should be able to estimate easily which timetable allows to transport a maximum of passengers without the use of too many trains.

Introduction

The Vienna International Airport is located in the centre of Europe. This makes the airport an important platform for international air travelling especially in Central and Eastern Europe. In 2013 around 22 million passengers were transported. There are more than 15 million people who need to be transported to the airport. Additionally the airport staff uses the different transportation possibilities as well [1].

This simulation investigates and simulates the railway connection of the airport using the Java-based simulation program AnyLogic.

Depending on various parameters that mostly can be changed by the user, this model offers a simplified presentation of a passenger movement at the airport between gates and railway station. Moreover the trains of the Viennese S-Bahn and the City Airport Train (CAT) are simulated. These trains are created according to the original timetable, but the amount of trains can be increased by the user, which leads to a shorter time interval of the trains during rush hours. Furthermore the expenses, income and profit of the railway companies are calculated as well.

1 The Model

There are nine types of agents representing different types of passengers and different types of trains:

- Incoming economy/business travelers
- Outgoing economy/business travelers
- Passengers who do not use a train for airport transportation
- Incoming S-Bahn/CAT trains
- Outgoing S-Bahn/CAT trains

This amount of different agents is necessary because all of them have different goals and features. They will be described in later on in much more detail.

The parameters that are variable are:

- Percentage of train travelling passengers
- Percentage of S-Bahn travelling passengers
- Percentage of passengers who only have hand luggage
- Walking speed of passengers
- Capacity of the trains
- Ticket price
- Working expenses

Passenger parameters are separated into economy and business, train parameters into S-Bahn and CAT. There is also the possibility to add an arbitrary amount of additional trains for each train provider during a time interval which can be inserted too.

The environment in which the agents act and interact is a simplified illustration of the airport including its railway station and rails, check-in, baggage claim area and the main exit as shown in Figure 1. The passenger representing agents are moving in this environment. The other agents, who represent the trains are moving along the railways that are attached to the railway station.

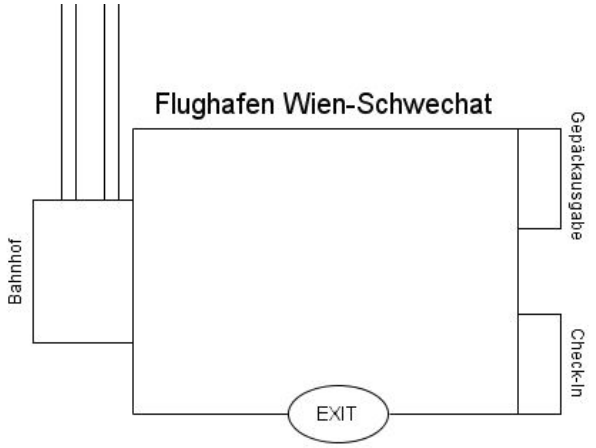


Figure 1: Simplified model of the Vienna International Airport used in the simulation. 'Bahnhof' stands for railway station and 'Gepäckausgabe' means baggage claim area.

In addition there are two graphics that illustrate the expenses, income and profit of the railway companies. Another diagram illustrates proportionately how passengers arrive at or leave from the airport.

2 The Agents

To illustrate the decision making and rules of the agents, their possible activities are illustrated in the following.

2.1 Incoming passengers

In Figure 2 the process of an incoming passenger is shown. An incoming agent is created at the baggage claim area of the model according to the actual flight plan. Afterwards it is decided probabilistically (according to the modulations the user made before the simulation started) if he is travelling only with hand luggage, if the passenger plans to take the train and in that case, if he is heading to the S-Bahn or the CAT.

Depending on these parameters the passenger is either waiting for his luggage at the baggage claim area, which would take half an hour on average, or is heading to the airport right away. If the passenger does not intend to go by train he/she is leaving the airport trough the EXIT. If the agent wants to use the S-Bahn, the simulation calculates his arrival time at the railway station and checks if there is a train connection. In that case the passenger starts to walk to the railway station.

As soon as an S-Bahn representing agent arrives at the railway station the passenger gets the message that he is now able to enter the S-Bahn, which he is doing in the following as long as there is still place for the passenger inside the train. Otherwise he has to wait for the next S-Bahn. If there is no train connection, the passenger has to look for another way of transportation and is heading for the EXIT. Afterwards the agent is removed from the model.

CAT travelling passengers follow the same rules except that for them the CAT timetable is checked for a connection and they are waiting for an CAT train.

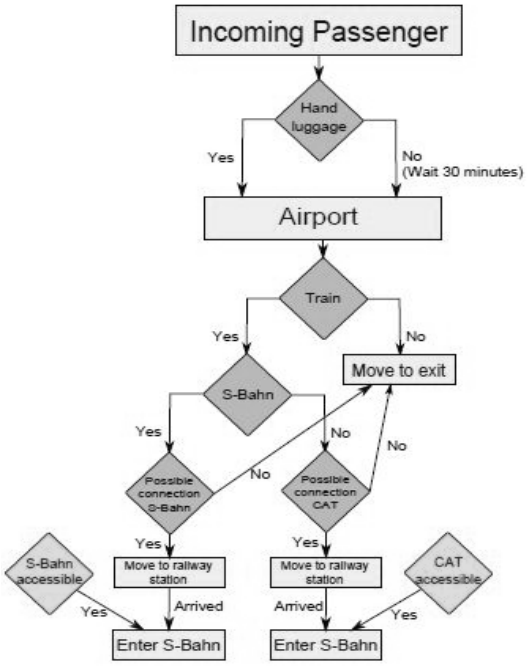


Figure 2: Decision making of a passenger who just exited the plane.

2.2 Outgoing Passengers

For an outgoing agent it is assumed that he is going by train. Otherwise he would have been constructed as a passenger who does not use a train for airport transportation in the first place. So an agent of the type outgoing passenger is either coming by S-Bahn or CAT, as shown in Figure 3. He is created at the railway station as soon as the best possible connection of his transportation type to get his flight reaches it.

Once the agent left the train he enters the airport right away and gets the order to move to the check-in area. As soon as he arrived there he is waiting for his flight to take off and then is removed.

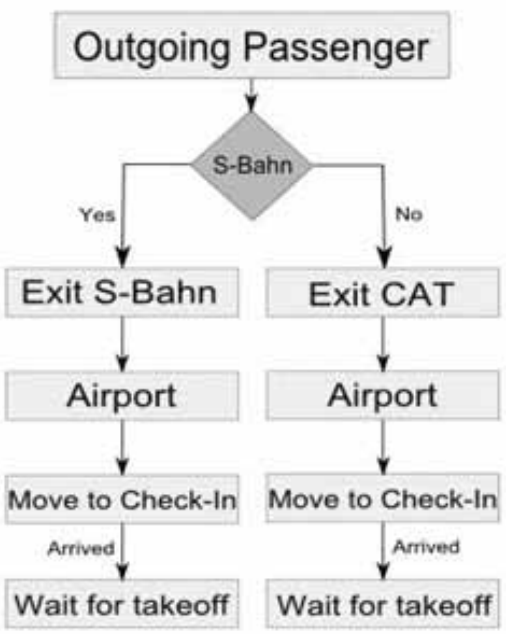


Figure 3: Decision making of a passenger who heads to the airport using train.

2.3 Passengers who do not use a train for airport transportation

This class is representing travelers who are not using the railway system and have therefore no direct input on the transportation companies profits. They are used to complete the amount of airplane users.

Therefore the decision making of these agent is very simple. They are created at the main exit and afterwards sent to the check-in area, where they are waiting for their flight to takeoff to be removed from the simulation afterwards.So this agent type is more or less a simplification of the outgoing passenger classes.

2.4 Incoming trains

Every incoming train is created, according to the (eventually modified) timetable [2] & [3], at the upper end of the rails in the model. Then the agent gets the order to move along the rails straight to the airports railway station, as shown in Figure 4. Once he arrived, two functions calculate for which outgoing passengers (economy and business) this train is the ideal connection to reach their plane.

All these passengers are now created and behave as described in chapter 2.2 as long as the capacity of the train is not exceeded. Afterwards the train is removed from the simulation.

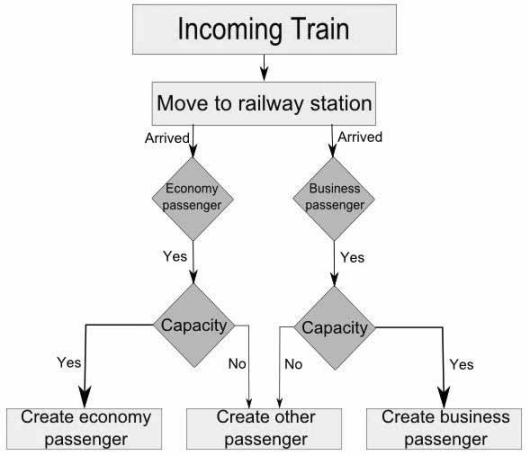


Figure 4: Decision making of an incoming train.

2.5 Outgoing trains

Every outgoing train is created according to the timetable [2] & [3], directly at the airports railway station.

Once created he sends a message to the incoming passengers waiting at the railway station for his train type (so either S-Bahn or CAT), that they are now able to enter the train. As long as the capacity of the train is not exceeded these travelers are removed from the simulation like described above in chapter 2.1. After the train waited for 3 minutes in the railway station, he is sent to the edge of the visualization along the railways. Once reaching border the agent is removed from the simulation.

3 Results

Since this model has various parameters that can be changed by the user, there is also a high amount different scenarios. For that reason there are only a few interesting scenarios presented in the following.

In the tables S-Bahn is abbreviated with SB, economy travelers with EC and business travelers with BU. In both scenarios the ticket prices are set with 4,4 € for travelling with the S-Bahn and 12 € for the CAT.

3.1 Scenario 1: Additional trains at peak hours

In Scenario 1 most parameters remain unchanged during the different runs. This scenario analyzes if the S-Bahn could increase its profit and/or the amount of transported passengers.

Table 1 shows, that S-Bahn profits decrease in all runs with additional trains. So with the given working expenses and ticket prices it is not profitable to extend the timetable at the chosen intervals.

But if the amount of transported passengers is considered, it is obvious, that especially in the night hours the amount of trans ported passengers increases significantly. It might be possible that the usage of trains in the night hours could be profitable with an increased ticket price and/or reduced working expanses.

The profit and passenger chances of the CAT are uninfluenced and within the probability variation.

	Run 1	Run 2	Run 3	Run 4
Train user EC	70%	70%	70%	70%
SB user EC	80%	80%	80%	80%
Only hand luggage EC	25%	25%	25%	25%
Train user BU	35%	35%	35%	35%
SB user BU	45%	45%	45%	45%
Only hand luggage BU	75%	75%	75%	75%
Working expenses per SB	1250	1250	1250	1250
Capacity SB	500	500	500	500
Capacity CAT	500	500	500	500
Additional SB amount	0	3	2	2
Add. SB time	-	14-18 h	6-10 h	1-6 h
Profit SB in €	19.074,56	12.131,60	15.357,60	16.169,68
Profit CAT in €	58.262,80	58.423,20	57.962,40	57.928,80
Passengers SB	27.062	27.241	27.354	27.537
Passeng. CAT	13.855	13.869	13.830	13.827

Table 1: Scenario 1. Most parameters remain unchanged during the different run. The only differences are extensions of the S-Bahn timetable during different intervals.

Analyzing the results of Table 2, it is obvious that even the use of 16 more trains still extends the com pany's profit. B ut it h as to be co nsidered th at in Ru n 3 the amount of tra nsported passengers is t he highest although no f urther trains ha ve bee n used. Therefore an increase of the capacity could be the best solution of this scenario if t he working e xpenses c ould be re duced. Especially if it is not possible to use 16 trains additionally Run 3 is definitely the best option to choose, since 8 more trains with less capacity in Run 2 offer less profit.

The big loss of the CAT c ompany is a conse quence of the big reduction of its users in this scenario.

3.2 Scenario 2: Handling of increased use of the S-Bahn

In the second scenario a different initial position is taken. It is supposed, t hat nearly all (90%) of the travelers are usi ng the S-Bahn. So this scena rio analyzes the different opportunities of the S-Bahn operator to handle such an increased demand.

	Run 1	Run 2	Run 3	Run 4
Train user EC	90%	90%	90%	90%
SB user EC	90%	90%	90%	90%
Only hand luggage EC	25%	25%	25%	25%
Train user BU	90%	90%	90%	90%
SB user BU	90%	90%	90%	90%
Only hand luggage BU	75%	75%	75%	75%
Working expenses per SB	1250	1250	1900	1250
Working expenses per CAT	1500	1500	1500	1500
Capacity SB	500	500	750	500
Capacity CAT	500	500	500	500
Additional SB amount	0	8	0	16
Additional SB time	-	6-23 h	-	4-24
Profit SB in €	51.055,52	59.294,72	60.893,12	62.990,88
Profit CAT in €	-19.694,4	-19.557,6	-18.477,6	-19.000,8
Passengers SB	34.331	40.749	48.385	45.850
Passengers CAT	7.359	7.370	7.460	7.417

Table 2: Scenario 2. Most parameters remain unchanged during the different run. The only differences are extensions of the S-Bahn timetable during different intervals in run 2 and 4 and higher working expenses in run 3.

References

[1] www.viennaairport.com/unternehmen/flughafen_wien_ag/facts__figures_fwag_gruppe, 24.8.2014.

[2] www.viennaairport.com/passagiere/anreise__parken/s-bahn, 25.08.2014.

[3] www.viennaairport.com/passagiere/anreise__parken/city_airport_train_cat, 25.08.2014.

An Agent-based Approach to ARGESIM Benchmark C16 'Restaurant Business Dynamics' based on NetLogo

Julian Ruths, Günter Schneckenreither*

Dept. Mathematical Modelling and Simualtion, Vienna Univ. of Technology,
Wiedner Haupstraße 8-10, 1040 Vienna, Austria; *guenter.schneckenreither@tuwien.ac.at*

Simulation Notes Europe SNE 24(3-4), 2014, 127 - 130
DOI: 10.11128/sne.24.bn16.10254
Received: January 20, 2014; Revised: October 11, 2014;
Accepted: October 20, 2014;

Abstract. In this comparison the business dynamic of restaurants is modeled. Restaurants open in an area inhabited by people, who visit those restaurants from time to time. Restaurants may close or a new restaurant may open, depending on their business performances. For this comparison, we chose an agent-based approach, using the freely available agent-based programming language and integrated modeling environment NetLogo. We will give a brief overview of the model assumptions, the reasons that led us to the decision of an agent-based approach, as well as a short introduction of NetLogo and its advantages regarding the implementation of the model. We will discuss specific aspects of the implementation itself and the results of several tasks that had to be performed.

Introduction

30 restaurants are initially evenly distributed in a rectangular area. There are 3000 people living in this area, each belonging to one of 5 cities and placed around those cities within a given radius. Every day a changing number of people (depending on their state of 'hunger') visit a restaurant within a given radius and pay the restaurant a defined amount of money for their meals. At the end of each week, restaurants calculate their profit. Depending on the profit the location then may close or a new restaurant might open somewhere else in the area.

1 Model

Although there is no precise definition of what an 'agent' is, general agreements can be found. One of these agreements is stated in [1]:

- Indentifiable: every agent is a discrete individual with behavior rules and attributes that can vary with every agent.
- Situated: agents live in an environment and are able to interact with it as well as other agents.
- Autonomous and self-directed: agents function independently.
- Goal-oriented: agents may have goals they want to achieve.
- Adaptive: agents may be able to learn and adapt their behaviour.

An agent-based model therefore conists of agents living in an environment, interacting with each other or the environment, based on behaviuor rules. Having inspected the model descriptions, an agent-based approach turned out to be applicable to this problem.

1.1 Simulator

We developed the model using NetLogo. NetLogo is a free and open source software and a dialect of the programming language Logo [2].

The NetLogo world is made up of 4 basic elements [3]: *Turtles*, which are agents and inhabit the world; Patches, which make up the environment of the world; *Links*, which can be used to signal relations between turtles; *Observer*, which doesn't have a location and is watching everything.

Turtles are distinguishable by a unique ID and NetLogo allows different types of turtles, which is implemented by defining different turtle-breeds. Each turtle-breed has its own properties and is defined by the user.

NetLogo has implemented a functional-language style, which makes most usages of loops omittable. One can simply address members of a certain breed that fulfil given criteria and go through each one of them without having to know how many there actually are. Since the number of restaurants throughout the course of the simulation can vary extensively, this allows for a simple access without having to first count all the existing restaurants and inserting a loop to go through them all. Very intuitive and readable syntax as well as an easy-to-handle interface made NetLogo a perfect candidate to work with on this comparison.

1.2 Model implementation with NetLogo

Space. The rectangular area is easily defined using the environment settings provided by the NetLogo interface. At startup the cities are placed according to the given coordinates. Every city then creates the number of people belonging to the city. They are placed around the city with the angle being uniformly distributed and the radius being triangularly distributed towards the city center and a maximum of maxR (see cities below). This represents the fact that the people tend to live closer to the citycenters.

Time. Every tick (time-step) represents one day. After seven ticks, a week has passed and restaurants need to calculate their profit.

Persons. Each turtle of this group belongs to the turtle-breed person. Besides their predefined attributes, every person has a *dining intervall* and a list of *destinations.* The dining interval ranges from 0 to 8 and indicates how many days have to go by until the next meal. At the end of every day, this value is reduced by 1. If the value at the start of the day is zero, the person visits a restaurant within dining range and a new dining interval is randomly calculated (one day is added to the new value, since it will be reduced by 1 again at the end of the day). The destination list contains the IDs of all restaurants within dining range of the person. In an earlier version of the model, this attribute was not included as will be explained further down.

Restaurants. Each turtle of this group belongs to the turtle-breed restaurant. Besides their predefined attributes, every restaurant has an *income*, a *profit* and an attribute titled *new*.

Every time a person visits the restaurant, the income is increased by the value of the dining cost. The profit is calculated at the end of every week using a given taxrate and running costs. The value *new* indicates whether the restaurant exists since the beginning of the simulation and was only introduced for the completion of task c.

Cities. Each turtle of this group belongs to the turtle-breed city. Besides their predefined attributes, every city has a population percentage and a radius maxR. At startup every city will be placed at its given location and create a number of persons according to the population percentage. These people will then be distributed around the city area as was described in the Introduction.

Cells. Each turtle of this group belongs to the turtle-breed cell. Besides their predefined attributes, every cell has a population density and a location coefficient. The population density represents the number of people living on that cell and since people don't move, it is calculated once at startup via the following command:

```
ask cells [
set peopledensity count persons with [ abs
   (xcor - cellx) <= 10 and abs (ycor - celly)
    <= 10 ] ]
```

The location coefficient is calculated with the given formula at startup and and again at every time-step where the event of opening a restaurant occurs.

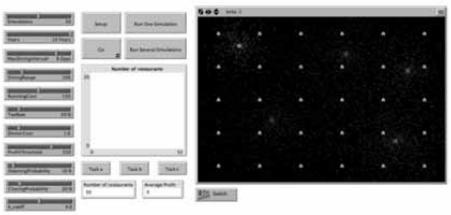


Figure 1: The user-interface of the model allows easy changes to several parameters.

User-Interface. The interface provided by NetLogo, allows easy manipulation of parameters using global variables. In the finished model-interface the user can vary all parameters such as MaxDiningIntervall, runningCost, taxrate etc. A seperate window plots the number of restaurants over time and another one shows the representation of the two-dimensional world with persons, restaurants and cities all visible and accessible via mouse click.

1.3 Specific Implementation

The intuitive syntax of NetLogo allows for quick and easy programming. What follows is an example of a piece of code that we initially used to simulate the process of hungry people choosing a random restaurant within range and the restaurant then increasing its income:

```
ask persons with [diningintervall = 0] [
   ask one-of restaurants in-radius DiningRange
   [ set income income + 1 ] ]
```

Obviously the code doesn't differ much from the in-text formulation, which is one of the advantages of NetLogo. One has to be careful though, as to not overlook more effective ways of programming. The first simulations we ran, used the above stated way and turned out to be much too time consuming. The reason for this is easily found: With around 400 to 500 people being hungry every day, the command 'in-radius', which is a quite complex command in itself, needs to be executed 400 to 500 times as well. It became obvious then, that, since people never change their location after the startup, the restaurants within dining range only change if there is a closing, or an opening nearby. As a solution we introduced the previously mentioned destination list. This simple change resulted in a massive lowering of computation time.

1.4 Flow of Information

A question that we had to ask ourselves in the process of modelling, was 'Who is the acting agent?' At first glance, the answer seems obvious. People look for restaurants nearby; people visit these restaurants randomly and pay money for their consumed meals. Restaurants may close down, but that is only a result of the behaviour of the person.

The conclusion, that the person is the acting agent is therefore quickly drawn. In this particular model however it is more effective to look at the restaurants as the acting agents and reversing the flow of information. Until now, each person was looking for restaurants nearby. Even with the implemented destination list and the update of this list only when a change in restaurants occurs, this results in 3000 executions of the in-radius command. Reversing the information flow, results in restaurants telling people once that they are located within dining range and another time - upon closing - that they are not existing anymore.

```
let nummer who
ask persons in-radius DiningRange [
    set destination remove nummer destination ]
die
```

The simulation of 4 weeks using the first implementation, took about 90 seconds. With all the changes that were proposed in this subsection as well as subsection 1.3, implemented in the final version, allows a cycle of 10 simulations - each set to run for 5 years - to be completed in about 60 seconds.

2 Simulation Tasks

2.1 Task a - Time domain analysis

Because of the high amount of randomness, we had to perform the simulations multiple times and averaged the data to get meaningful results. As demanded, 50 simulations were run and the limit value of the number of restaurants after the 5th year was calculated. This was done by inserting an outer loop and storing the number of restaurants after 5 years in a list. We then averaged the data in this list, using standard methods of NetLogo.

Mean	5.66
Variance	1.98
Deviation	1.41

Table 1. Meanvalue, variance and standard deviation of the number of restaurants after 5 years.

The deviation of the overall number of restaurants is quite high in respect to the mean value. Paying attention to the number of restaurants during a simulation run reveals a great number of openings and closings, due to the fact that restaurants have no memory of previous weeks. This results in abrupt closings of restaurants that just had a bad week.

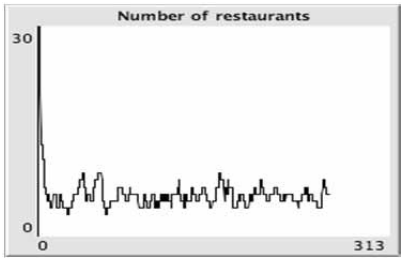


Figure 2: Number of restaurants plotted over the course of 260 weeks.

Due to this fact, the number of restaurants after 5 years can vary a great deal, since a steady state of the system can hardly be achieved.

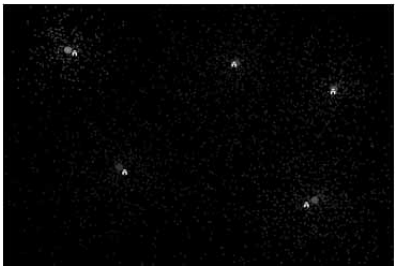


Figure 3: In-simulation view of the deployment of restaurants around the cities.

2.2 Task b – Maximisation of tax income

The relationship between taxrate and taxincome is quite obvious. The lower the taxrate, the more restaurants are able to survive. Therefore a higher number of restaurants pay taxes, but the amount they have to pay is low. A high taxrate results in fewer restaurants paying a very high amount of taxes.

The question of maximising taxincome via variation of the taxrate is strongly dependend on the looked upon time period. Wanting to maximise taxincome over to course of 5 years results in completely different taxrates, than maximising over the course of 6 months, or even 2 weeks. We assumed a longterm wish of steady tax income and ran a cycle consisting of 10 simulations – each lasting 5 years – and averaged the tax income. We then increased the tax rate by 1% starting with 0% and ending with 100%.

The results show a maximum taxincome of 193658,688 currency units at a taxrate of 32% as seen in Figure 4. The high fluctuation in the area of 25% to 50% may be smoothed out when increasing the iterations per taxrate.

2.3 Task c – Maximisation of new restaurants' revenue

The 'best' location for opening a new restaurant is only dependent on the coefficient k. This represents the importance, that the location of other restaurants play in the evaluation of a possible new location. $k = 0$ resulting in restaurants opening where the peopledensity is highest and, with gradually increasing values of k, paying more attention to the restaurants close by. Several variations of the parameter k showed that a maximal revenue of 9151 curreny units, can be found at $k = 1$.

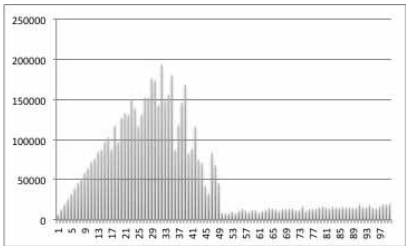


Figure 4: The average tax income after 5 years with varying tax rates.

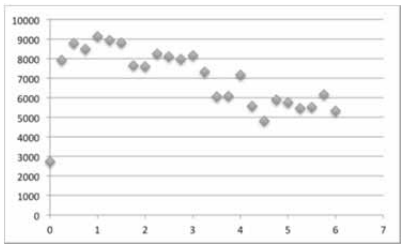


Figure 5: Average revenue of restaurants opened after startup.

3 Conclusion

Agent-based modelling with NetLogo is a very intuitive and straight forward method. But one of the problems with agent-based modelling & simulation is that it often involves a very high number of computations due to the fact that a system is made up of thousands of agents, all acting and interacting with each other. This can result in very time-consuming simulation runs. Therefore considerate thought should go into asking how these interactions should be executed. A simple inversion of the information flow proved to be very effective in this particular case, respective in an efficient simulation.

Model sources

NetLogo is an Opene Source simulator, available from web. For this C16 benchmark approach, Netlogo model files, parametrization files, and a short file description (and a link for NetLogo download) can be downloaded (zip format) by EUROSIM societies' members from SNE website, or are availably from the corresponding author.

References

[1] Macal, CM, North MJ. Tutorial on agent-based modelling and simulation. *Journal of Simulation*. 2010; 4(3), 151-162.

[2] Wilensky, U. *NetLogo*. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling Northwestern University. Evanston, IL. 1999.

[3] Izquierdo, LR. *NetLogo 4.0–Quick Guide*. 2007

Simulation of Fluid Dynamics in a Network of Blood Vessels with 1D FEM

Antonia Lichtenegger[2], Bernhard Hametner[1,2*], Siegfried Wassertheurer[1,2]

[1]AIT Austrian Institute of Technology, Department of Health & Environment, Vienna, TechGate Vienna, Donau-City-Straße 1, 1220 Vienna , Austria; **bernhard.hametner@ait.ac.at*

[2]Vienna University of Technology, Inst. of Analysis and Scientific Computing, Wiedner Haupstraße 8-10, 1040 Vienna, Austria

Simulation Notes Europe SNE 24(3-4), 2014, 131 - 136
DOI: 10.11128/sne.24.tn.10255
Received: May 30, 2014; Revised August 16, 2014;
Accepted: September 22, 2014;

Abstract. The aim of this paper is to simulate the blood-stream through a network of blood vessels with a Finite Element Method in one dimension. A one dimensional system of partial differential equations is used. This system can be written in hyperbolic conservation form with the state variables cross-sectional area, the flow, the velocity and the pressure. To solve the system of partial differential equations, numerically correct boundary conditions have to be considered. For the input, a pressure function is used. To simulate the load downstream and the compliance of the arterial segments, a Windkessel model consisting of three elements is used. By simulating bifurcations the considered abstract vascular network can be build up. For that a nonlinear system of equations is set up and solved. The partial differential equation system cannot be solved analytically. Hence, to solve it a numerical Finite Element Method is used. In this context, a Taylor Galerkin method of second order with basic functions of first order is used. The model is implemented by using the mathematical software MATLAB. To verify the model, several simulations are done, using an abstract arterial tree built up by thirteen central arterial segments. In all simulations, the parameters of the Windkessel model and the parameters of the arterial segments are based on experiments and on physiological values. In all tests, physiologically realistic results are obtained. It can be concluded that the application of a one dimensional Finite Element Method approach along with the particular implementation presented can describe the effects in a system of human arteries in a realistic way.

Introduction

Cardiovascular diseases are the most common cause of death in the modern society. To improve the diagnosis and further on the therapy of such disease, more often dynamic models for the heart circulation system are used. In these models the main factors which must be considered are accurateness, computing time and identifiability of the parameters. Therefore one dimensional models, which have in fact a hight efficiency, come to the centre of attention.

1 Methods & Models

1.1 The model

A one dimensional model of the following form is used

$$\begin{aligned} &\frac{dA}{dt}+\frac{dQ}{dz}=0 \\ &\frac{dQ}{dt}+\alpha\frac{d}{dz}\frac{Q^2}{A}+\frac{A}{\rho}\frac{dP}{dz}+K_R\frac{Q}{A}=0 \end{aligned} \tag{1}$$

In this system of partial differential equations the calculated parameters will be the area of the arterial segments A, the pressure $P$and the flow Q.

The other occurring values are set to be constant and are namely the blood density ρ, the Coriolis coefficient α, which defines the velocity profile, and a friction parameter which is given through

$$K_R = 8\pi\nu \tag{2}$$

where ν is the viscosity of blood. In equation (1), one has two equations for the three parameters.

To find an unique solution for this problem another third equations must be specified through equation (3).

$$P = \beta \frac{\sqrt{A} - \sqrt{A_0}}{A_0} \tag{3}$$

This equation gives a connection between the pressure and the area of a vessel. Additionally the parameters A_0, $\beta = E_0 h_0 \sqrt{(\pi)}$are set to be constant. A_0Is the initial value for the area, $E_0$is the Young modulus and $h_0$is the wall thickness of the arterial segments. The system of differential equations can also be written in a matrix conservative form:

$$\frac{dU}{dt} + \frac{dF}{dz}(U) = B(U), U = [A, Q] F(U) = \\ = \begin{pmatrix} Q \\ \alpha \frac{Q^2}{A} + \frac{\beta}{3\rho A_0} A^{3/2} \end{pmatrix} B(U) = \begin{pmatrix} 0 \\ -K_R \frac{Q}{A} \end{pmatrix} \tag{4}$$

By setting $\alpha = 1$, a flat velocity profile is supposed, which is reasonable for blood stream simulation [1,4,5].

1.2 Boundary conditions

Boundary conditions have to be set to get an unique solution. Here one distinguish between the proximal, namely the inflow and the distal, namely the outflow boundary conditions.

Figure 1 shows a pressure curve which which will be used as the input function for the proximal boundary condition. The pressure of the left ventricle in the heart is approximated by these two sine functions.

For the outflow a Windkessel model consisting of three elements is used. In this case the flow and the area must fulfil the following relation, given trough a ordinary differential equation.

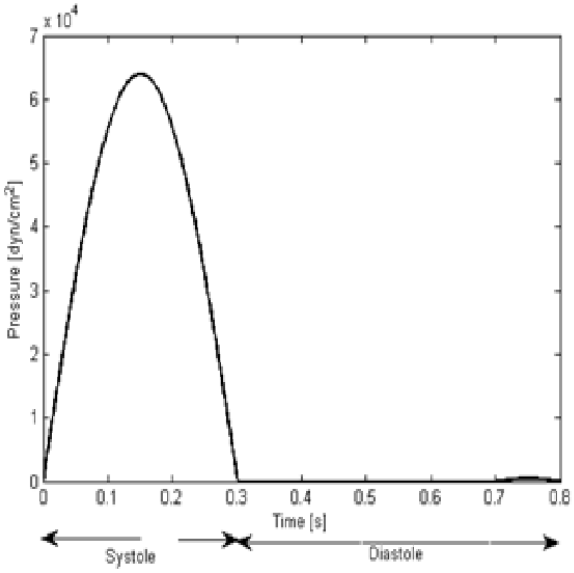


Figure 1: Approximated Pressure Input function for the model.

$$Q(1 + \frac{R_0}{R_1}) + CR_0 \frac{dQ}{dt} = \frac{P}{R_1} + C\frac{dP}{dt} \tag{5}$$

The Windkessel parameters are the resistances R_0 and R_1 the and the compliance C of the arterial segment [3].

1.3 Bifurcations

A bifurcation is the branching of an artery in two. There are two conditions which has to be fulfilled. Suppose one has an arterial segment (A) which is split up in two arterial segments namely (B) and (C). First of all the condition

$$P^{(A)} = P^{(B)} = P^{(C)} \tag{6}$$

must be fulfilled, meaning that the pressure is is the same at all three boundaries and secondly that

$$Q^{(A)} = Q^{(B)} + Q^{(C)} \tag{7}$$

meaning that the flow of the first arterial segment is spilt up in the the flow of the following arterial segments [2].

1.4 The arterial network

The atrial tree consists of 13 central arterial segments.

The rest of the arterial segments will be modelled by choosing the Windkessel parameters in a correct way. Figure 2 shows the abstract arterial tree. The values which will be used are shown later in Table 1 and Table 2 and are taken from physiological data.

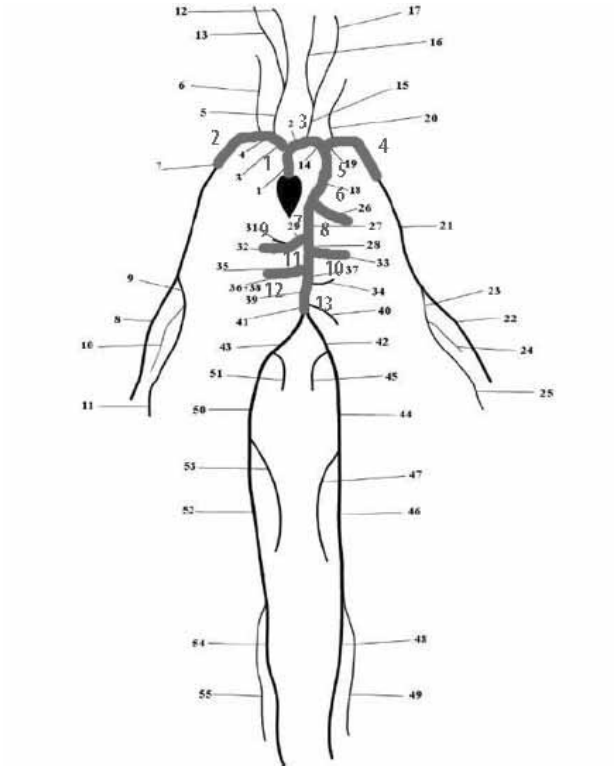


Figure 2: Abstract arterial tree, consisting of the 13 central arteries.

1.5 Taylor-Galerkin method

The system of partial differential equations cannot be solved analytically, that is why the numerical Finite Element Method is used. The basic idea of this method is to do a discretization in time and space.

Lets start with the discretization in time, to do so one uses a Talyor approximation of second order. After this, the discretization in space has to be done. For that the domain $\Omega = [0, L]$, which is namely the arterial segment, is decomposed into N equally spaced elements $[z_i, z_{i+1}]$ and the solution U is discretisized by basic functions of first order through equation (9).

$$U_h = \Sigma_{i=1}^{N} U_i \phi_i \tag{8}$$

To be more precise for $\phi_i(z)$ first order basic functions or hat functions will be used. Now the weak formulation of the problem has to be calculated. To do so, the term, gained from the second order Taylor approximation, has to be multiplied by a test function and integrated over the domain. After inserting the discretisized solution and discretisized test function the result has the following form:

$$\begin{aligned}(U_h^{n+1}, \phi_h) = (U_h^n, \phi_h) + \Delta t \Big(F^n(U_h^n) + \\ \frac{\Delta t}{2} H^n(U_h^n) B^n(U_h^n), \frac{d\phi_h}{dz}\Big) \frac{-\Delta t^2}{2} \\ (B_U^n(U_h^n) \frac{dF^n}{dz}(U_h^n), \phi_h) \frac{-\Delta t^2}{2} (H^n(U_h^n) \frac{dF^n}{dz}(U_h^n), \frac{d\phi_h}{dz}) \\ + \Delta t (B^n(U_h^n) + \frac{\Delta t}{2} B_U^n(U_h^n) B^n(U_h^n), \phi_h)\end{aligned} \tag{9}$$

Equation (9) is the final result and in the implementation the integral terms of the left and the right side will be calculated. This will be done by a numerical quadrature. The implementation is done in MATLAB although some parts of the code are written in C to get more efficient results [7].

1.6 Discretization of boundary conditions

To get unique results boundary conditions must be defined. Before one can do so, the characteristic values have to be calculated. For this system of differential equations one gets the characteristic variables

$$W_{1,2} = \frac{Q}{A} \pm 4\left(\sqrt{\frac{\beta}{2\rho A_0}} A^{1/4} - c_0\right). \tag{10}$$

Because the system of differential equations is a hyperbolic one, only one boundary condition on each side of the arty has to be set. [2]

Proximal boundary conditions

An pressure input function should be used as an input. To do so one manipulate the definition of the inflow characteristic variable to

$$W_1 = W_2^0 + 8\sqrt{\frac{1}{2\rho} p(t) + \frac{\beta}{2\rho\sqrt{A_0}}} - c_0, \tag{11}$$

so that the pressure $p(t)$ can be used as input directly [7].

Distal boundary conditions

For the distal boundary conditions the Windkessel model is used. The new value of A is found by solving the non-linear equation (12)

$$R_0\left[\frac{Q_R}{A_R}+4c_1(A_R)\right]A-4\mathrm{R}_0c_1(A)A- \\ -P(A)+P_C^n=0, \tag{12}$$

where P_C is calculated through (14)

$$P_C^n=P_C^{n-1}+\frac{\Delta t}{C}(Q_R-\frac{P_C^{n-1}}{R_1})P_C^{n-1} \\ =P(A_R)-Q_RR_0. \tag{13}$$

To solve this non-linear equation a Newton method with start value $A=A_R$ is used. A_R and Q_R are the values at the end of the arterial segment. The new value of Q is then found through equation (14). [7]

$$Q=\frac{P(A)-P_C}{R_0} \tag{14}$$

1.7 Boundary conditions at the bifurcations

When one artery is split up into two others, the two conditions from equations (6) and (7) has to be fulfilled. This is done by solving a system of six non-linear equations given through (15). [7]

$$\begin{aligned}
&Q^{(A)}=Q^{(B)}+Q^{(C)}\\
&W_1^{(A)}=\frac{Q^{(A)}}{A^{(A)}}+4(\sqrt{\frac{\beta^{(A)}}{2\rho A_0^{(A)}}}(A^{(A)})^{1/4}-c_0^{(A)})\\
&W_2^{(B)}=\frac{Q^{(B)}}{A^{(B)}}-4(\sqrt{\frac{\beta^{(B)}}{2\rho A_0^{(B)}}}(A^{(B)})^{1/4}-c_0^{(B)})\\
&W_2^{(C)}=\frac{Q^{(C)}}{A^{(C)}}-4(\sqrt{\frac{\beta^{(C)}}{2\rho A_0^{(C)}}}(A^{(C)})^{1/4}-c_0^{(C)})\\
&\beta^{(A)}\frac{\sqrt{A^{(A)}}-\sqrt{A_0^{(A)}}}{A_0^{(A)}}\\
&=\beta^{(B)}\frac{\sqrt{A^{(B)}}-\sqrt{A_0^{(B)}}}{A_0^{(B)}}\beta^{(A)}\frac{\sqrt{A^{(A)}}-\sqrt{A_0^{(A)}}}{A_0^{(A)}}\\
&=\beta^{(C)}\frac{\sqrt{A^{(C)}}-\sqrt{A_0^{(C)}}}{A_0^{(C)}}
\end{aligned} \tag{15}$$

2 Implementation

In this section, a part of the code is presented and described. This part realizes the bifurcations in the arterial tree. As a starting point one has a matrix (`Tree`) consisting of the numbers of the parent and the child arteries.

First of all, out of this matrix the connection are read out.

```
Place_1=Tree(:,1);
Place_2=Tree(:,2);
Place_3=Tree(:,3);

bif=@(ub)[
ub(2)/ub(1)+4*(sqrt(beta(Place_1)
/(2*rho*A0(Place_1)))*(ub(1))^(1/4)-
c0(Place_1))-W1L(1,1,Place_1);...

ub(4)/ub(3)-
4*(sqrt(beta(Place_2)/(2*rho*A0(Place_2)))*(u
b(3))^(1/4)-c0(Place_2))-W20(1,1,Place_2);...

ub(6)/ub(5)-
4*(sqrt(beta(Place_3)/(2*rho*A0(Place_3)))*(u
b(5))^(1/4)-c0(Place_3))-W20(1,1,Place_3);...

ub(4)+ub(6)-ub(2);...

beta(Place_2)*(sqrt(ub(3))-
sqrt(A0(Place_2)))/A0(Place_2)-
beta(Place_1)*(sqrt(ub(1))-
sqrt(A0(Place_1)))/A0(Place_1);...

beta(Place_3)*(sqrt(ub(5))-
sqrt(A0(Stelle_3)))/A0(Place_3)-
beta(Place_1)*(sqrt(ub(1))-
sqrt(A0(Place_1)))/A0(Place_1)];
```

After setting up the six nonlinear equations (`bif`) taken from equation (16), the system can be solved with the MATLAB function `fsolve` and the new boundary values can be set.

```
[Ub(:)]=fsolve(bif,Ub);

uAL(Place_1)=(Ub(1));
uQL(Place_1)=(Ub(2));
uA1(Place_2)=(Ub(3));
uQ1(Place_2)=(Ub(4));
uA1(Place_3)=(Ub(5));
uQ1(Place_3)=(Ub(6));
```

3 Results

The parameters which where used for the simulation are shown in Table 1 and Table 2. The unit for the radius and length are cm, for the resistances is $\frac{dyns}{cm^5}$, for the compliance it is $\frac{cm}{dyn}10^{-5}$and for the Young modulus is $\frac{dyns}{cm^2}10^6$.

Parameters	Values	Unit
ρ	1	g/cm^3
h_0	0.15	cm
β	$E_0 * h_0 * \sqrt{(\pi)}$	dyn/cm
K_R	$8\pi v$	$Poise$
v	0.035	$Poise$
dt	$2 * 10^{-5}$	sec
$time$	0.8	sec

Table 1: Values of the parameters used for the simulation .

Artery	Child	r	l	R0	R1	C	E0
1	2 \ 3	1.4	4	-	-	-	4
2	-\ -	0.6	3	50	1000	8	1.8
3	4 \ 5	1.3	2	-	-	-	3.8
4	-\ -	0.4	6	60	1200	8	1.2
5	6 \ 7	1	4	-	-	-	3.5
6	-\ -	0.3	5	70	1400	9	0.9
7	8 \ 9	0.9	3	-	-	-	3.2
8	10 \ 11	0.8	5	-	-	-	2.9
9	-\ -	0.3	6	90	1600	9	0.9
10	-\ -	0.2	7	90	1600	9	0.7
11	12 \ 13	0.6	2	-	-	-	2.7
12	-\ -	0.4	6	100	1800	8.2	1.2
13	-\ -	0.5	2	100	1800	8.2	2.4

Table 2: Artery parameter used for the simulation of the arterial tree .

For every arterial segment one gets the pressure, the flow, the velocity and the cross-section area as an output.

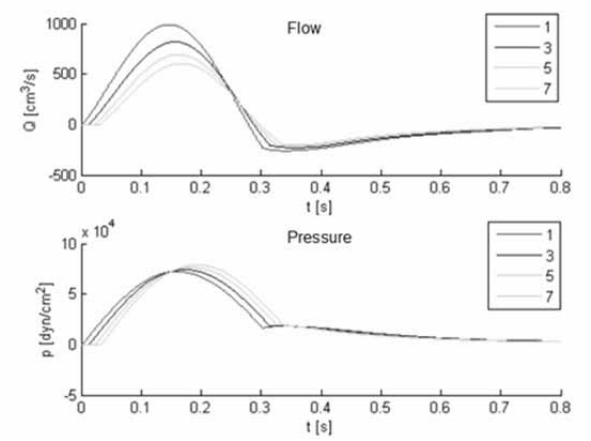


Figure 4: Flow and pressure results for the first, the third, the fifth and the seventh arterial segment. The values are read out in the middle of the arterial segments.

First of all the results from a way through the arterial tree, from Figure 2, is presented. The way goes from the first to the third, then to the fifth and finally in the seventh arterial segment. Figure 4 shows the results for the pressure and the flow in these arterial segments taken at the midpoint of the arterial segments.

Two effects can be seen: First that the pressure level rises and secondly the flow is decreasing from arterial segment to arterial segment.

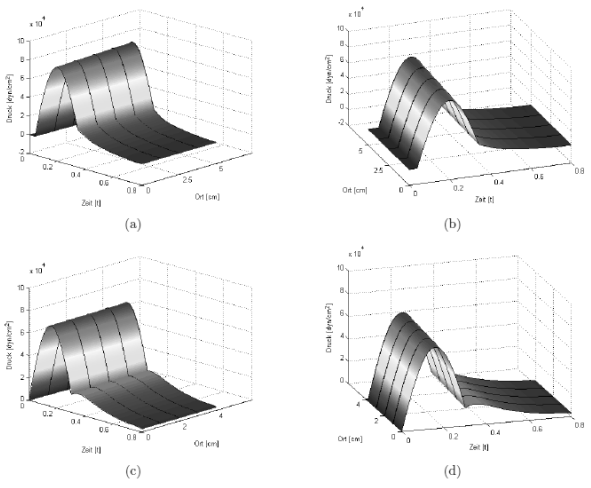


Figure 5: Pressure-position-time plot of the eight arterial segment (a)-(b) and the first arterial segment (c)-(d) in two different perspectives.

Figure 5 shows the pressure-position-time plot of the eight arterial segment from two different perspectives (a) and (b) and the the pressure-position-time plot of the first arterial segment from two different perspectives (d) and (c). Here one can observe again the influence of the Windkessel model and also the time dependent delay in place can be seen in Figure 5 (b) and (d).

4 Conclusion and Outlook

In comparison to already accomplished models and physiologically data the gained results with this model are very similar. That is why it can be concluded that the application of a one dimensional Finite Element Method approach along with the particular implementation presented can describe the effects in a system of human arteries in a realistic way.

A field of application for this model is the early diagnosis of cardiovascular diseases. With measurements gained from healthy patients, the model can be parameterized. The calculations from this model can be compared with measurements from patients with cardiovascular diseases in order to conclude about abnormal changes in the cardiovascular system [6].

References

[1] Quarteroni A, Formaggia L, *Mathematical modelling and numerical simulation of the cardiovascular system*, Handbook of numerical analysis 12 (2004), 3-127

[2] Alastruey AJ. *Numerical modelling of pulse wave propagation in the cardiovascular system: development, validation and clinical application,* PhD thesis, University of London, 2006.

[3] Alastruey AJ, Parker KH. Arterial Pulse wave haemodynamics. *11th International Conference on Pressure Surges*, Lisbon, Portugal, 24th-26th October 2012; 401-442

[4] Roth CJ. *1d pulse wave propagation in the human vascular system*, Master's thesis, Imperial College of Science, Technology and Medicine, London, 2012.

[5] Lamponi D. *One dimensional and multiscale models for blood flow circulation*, PhD thesis, Section de mathematique pour l'obtention du grade de docteur es sciences par laurea in fisica, Universita degli Studi di Bologna, 2004

[6] Willement M, Lacroix V, Marchandise E. Validation of a 1d patient-specific model of the arterial hemodynamics in bypassed lower-limbs: Simulations against in vivo measurements. *Medical engineering & physics*. 2013; 35(11): 1573-1583

[7] Sherwin S, Formaggia L, Peiro J, Frank V. Computational modelling of 1d blood flow with variable mechanical properties and its application to the simulation of wave propagation in the human arterial system. *International Journal for Numerical Methods in Fluids*. 2003; 43(6-7): 673-700

Comparison of Two Filtering Methods for Heart Rate Variability Analysis

Matthias Hörtenhuber[1], Martin Bachler[1,2*], Siegfried Wassertheurer[2], Christopher Mayer[2]

[1] Inst. of Analysis and Scientific Computing, Vienna University of Technology, Wiedner Haupstraße 8-10, 1040 Vienna, Austria; * *martin.bachler@tuwien.ac.at*

[2] AIT Austrian Institute of Technology, Health & Environment Department, Biomedical Systems, Donau-City-Str. 1, 1220 Vienna, Austria

Simulation Notes Europe SNE 24(3-4), 2014, 137 - 142
DOI: 10.11128/sne.24.tn.10257
Received: June 15, 2014; Revised August 30, 2014;
Accepted: October 2, 2014;

Abstract. Heart diseases are amongst the most common causes of death in the industrialized world. Since the cardiological system is very complex and hard to capture in its entirety, researchers are looking for indicators of its health. A promising one is the heart rate variability (HRV), i.e., the variation of the time intervals between two heartbeats. It reflects physiological processes, which influence the rhythm of the heart. An approach by researchers is a visualization tool, the Poincaré plot, to analyse HRV. Numerous data models exist in order to automatically quantify Poincaré plots.

To extract as much information as possible from Poincaré plots, it has to be filtered from artefacts and outliers before applying the data models.

The goal of this work is to test the influence of two different filtering methods on the Poincaré plot quantification methods.

A test case was constructed were a database with healthy heart rates and one with pathological heart rates were filtered with the two methods. Thereafter two Poincaré plot measures were evaluated using the filtered data sets. Afterwards the differences between these data sets were statistically examined.

It can be concluded that the fully automated filtering via clustering shows no large drawbacks compared to the traditional method of ECG annotation based filtering for HRV-analysis via Poincaré plots.

Introduction

According to a report by the European Society of Cardiology, heart failure is a leading cause of death in the EU and is on the rise due to an increasing age of the population [1]. Since an early diagnosis of heart conditions leads to more successful treatments, researchers look for markers of heart diseases [2].

More than 30 years ago HRV was introduced as such a method [3]. HRV is the variation of the time interval between consecutive heartbeats. It highly depends on the extrinsic regulation of the heart rate, i.e., the time interval between two beats, and reflects changes in the balance of the different regulatory systems, including the autonomous nervous system [3].

Studies show a connection between the balance of the autonomic nervous system measured with HRV and cardiovascular diseases [4].

In studies of HRV, both time- and frequency-domain measures are typically used by practitioners and researchers [3]. Since the influences of the generation of beats are also non-linear [5], a visualization tool originating in chaos theory, the Poincaré plot and models to quantify it, have become popular tools to analyse HRV in the last 20 years [6].

The data basis of most HRV-measures, including the Poincaré plot used in this work, consists of so called RR-Intervals. These are the time distances between two consecutive R-peaks in an ECG Signal, as shown in Figure 1.

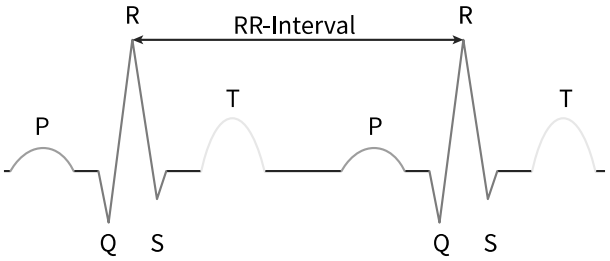


Figure 1: An ECG signal of two heart beats and the corresponding RR-interval.

Since hardly any ECG recording is ever without any artefacts or outliers (due to movement or ectopic beats, i.e., irregular heart beats) the signal has to be filtered to improve the information density of the Poincaré plots and the corresponding HRV-measures [7].

In most cases, this is done based on ECG annotations, where each heartbeat in the ECG signal is labelled, e.g. as a normal beat or as a premature beat. These annotations are either created manually by a physician, automatically by a computer, or semi-automatically, where the computer output is reviewed manually. All beats not labelled as normal are filtered out.Since the creation of annotations is a very time consuming task, which can only be done by trained personal, fully automatic methods should be considered. One of these approaches is to find artifices and outliers via clustering algorithms and filter these out. The question we examine in this work is, what the impact of the two filter methods is on HRV analysis via Poincaré plots.

1 Methods & Models

1.1 Poincaré plot

As mentioned beforehand Poincaré plots are a visualization tool for heart rate data. They are constructed as follows.

Given a data set of N RR-intervals $\{RR_1, \dots, RR_N\}$ a Poincaré plot is defined as the following mapping:

$$R \to R \times R$$
$$\{RR_1, \dots, RR_N\} \mapsto$$
$$\{(RR_1, RR_2), (RR_2, RR_3), \dots, (RR_{N-1}, RR_N)\}$$

A typical Poincaré plot of a non-pathological, 2 hour-long, unfiltered heart rate recording is shown in Figure 2.

1.2 Filtering via cluster algorithms

Clustering is defined as the grouping of data points based on similarities between these points [8]. Which similarity is measured depends on the used data, in our case we cluster based on the distances of the points in the Poincaré plot.

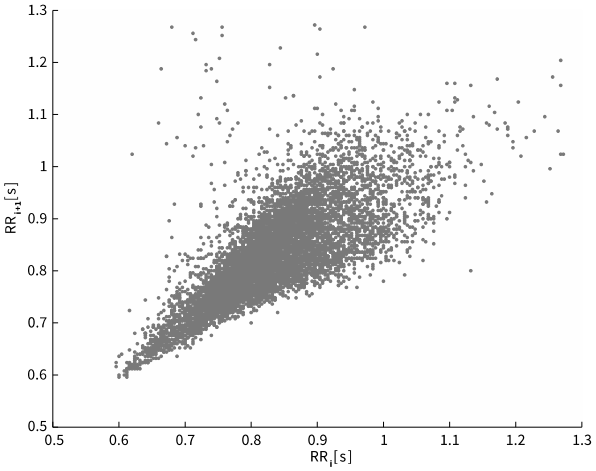


Figure 2: Unfiltered Poincare plot of a non-pathological 2 hour-long heart rate recording.

After considering different algorithms (k-means, single linkage and mean-shift), we chose the DBSCAN-Algorithm, because it does not require an a-priori number of clusters and shows a high robustness against noise. DBSCAN stands for Density-Based Spatial Clustering of Applications with Noise and was proposed by Ester et al. in [9].

The algorithm needs the parameters ε and *MinPts* as inputs, where ε is a neighbourhood threshold and *MinPts* is the minimum number of points a cluster consists of. DBSCAN distinguishes three types of data points:

- *Core points:* These have *MinPts* or more different points in their ε-environment.
- *Density reachable points:* These have at least one other data point in their ε-environment, but less than *MinPts*.
- *Noise:* These are neither core points nor density reachable points.

The following pseudo code describes the algorithm:

```
function=DBSCAN(D,ε , MinPts)
for (all unvisited points P in dataset D)
   mark P as visited
   N=getNeighboringPoints(P,ε)
   if(sizeof(N) < MinPts)
     mark P as noise
   else
     C = next cluster
     add P to cluster C
     for (P' in N)
       if(P' is not a member of any cluster)
         recursiveExpandCluster(P',C,ε ,
   MinPts)
       end
     end
    end
end
end

function=recursiveExpandCluster(P,C,ε
,MinPts)
  add P to cluster C
  if(P is not visited)
    mark P as visited
    N = getNeighbors(P,ε)
    if(sizeof(N) >= MinPts)
      for{ P' in N}
        if(P' is not member of any cluster)
          recursiveExpandClus-
ter(P',C,ε,MinPts)
        end
      end
    end
  end
end
```

One of the difficulties lies in the choice of ε. If it is too small, the algorithm overclusters, i.e., it separates visibly connected clusters, or it underclusters if ε is too large, i.e., it merges visibly unconnected clusters.

Therefore, we applied a refinement of DBSCAN, the Ensemble-DBSCAN (EDBSCAN) proposed by Xia et al. in [10].

This algorithm runs DBSCAN r-times iterating ε equidistantly from ε_{min} to ε_{max}, with:

$$\varepsilon_{min} := D_4^{mean} - \frac{D_4^{mean} - D_4^{min}}{8},$$
$$\varepsilon_{max} := D_4^{mean} + \frac{D_4^{max} - D_4^{mean}}{8}$$

The variable D_4 stands for the set of distances between the data points and their fourth nearest neighbour, D_4^{mean} is its mean value and D_4^{min} and D_4^{max} are the minimal and maximal value of the set. The result of every iteration is saved in the co-association matrix A, by adding 1 to each entry $A_{i,j}$, if the i-th and the j-th data point are in the same cluster and 0 otherwise. After all iterations the co-association matrix is normalized via element-wise division by r.

The final clusters are then constructed by using a voting method, described in the following pseudo code:

```
assign first data point to first cluster
for (all other points of D)
  calculate A_max := max_{j=1,...,i-1} A_{i,j}
  if(A_max < 0.5)
    assign current point to a new cluster
  else
    assign current point to cluster
ofD(k),
    whereA_{i,k} = A_max
  end
end
```

Afterwards, clusters with less data points than a given threshold are considered as noise. The threshold for this categorization is set so that clusters consisting of presumably non-pathological extrasystoles are ignored. Therefore, a number of 10 extrasystoles per hour is used as a threshold, based on the findings in [11], [12].

The sinusoidal beat cluster was then chosen as the one closest to the mean value of $\{RR_1, \ldots, RR_N\}$. To reduce assignment errors a correction by adding a small shift of 0.01 seconds to both coordinates of the mean was implemented.

This can be justified by the following reasons. First, most of the errors occur because of arrhythmias with a shorter RR-interval length. Therefore, these beats move the mean closer to zero, away from the actual sinus beats. Second, no case was observed were the mean value was above the sinusoidal cluster, which would be the case for very atypical heart rates of a high amount of single slow beats in connection with a very fast sinus beat. Third, the small shift of 0.01 seconds only slightly alters the mean value, but enough to prohibit most of the incorrect assignments.

1.3 Deleting points of a Poincaré plot

Since Poincaré plots represent the relation between two consecutive beats, the filtering can not be done by deleting one beat interval in $RR_x := \{RR_1, \ldots, RR_{N-1}\}$ and the same in $RR_y := \{RR_2, \ldots, RR_N\}$, but the preceding one in RR_x and the following one in RR_y has to be deleted as well, as shown in Figure 3 [7].

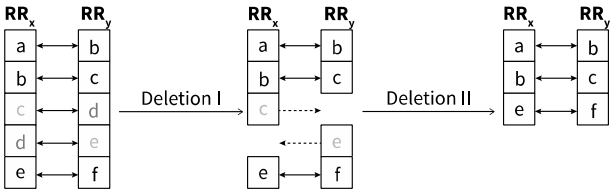


Figure 3: Deletion of the incorrect RR-interval "d" (Deletion I) and the corresponding counterparts "c" in RR_x and "e" in RR_y (Deletion II).

1.4 Poincaré plot measures

Different methods exist to automatically quantify Poincaré plots. In this work the following were used.

Ellipse Fitting Method.

The commonly used method to quantify Poincaré plots is the ellipse fitting method [13]. For this method an ellipse is fitted to the Poincaré plot, as shown in figure 4. The center of the ellipse is the mean value of the RR-intervals, the length of the major axis (*SD2*) is the standard deviation in the direction of the line of identity and the length of the minor axis (*SD1*) is the standard deviation perpendicular to the line of identity. The values *SD1*, *SD2* and their ratio are used as Poincaré plot measurements.

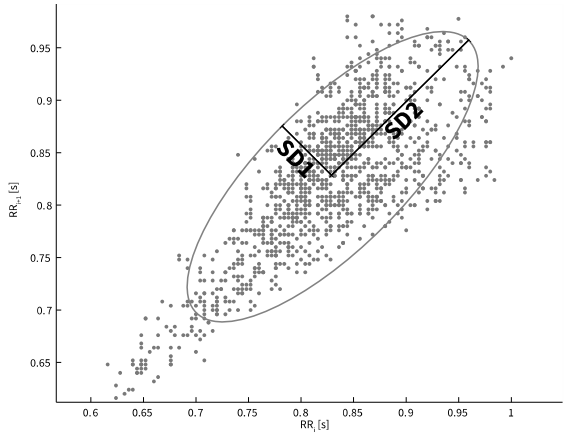


Figure 4: The ellipse fitting method to measure Poincaré plots.

Longitudinal-Transversal Measure.

Another method to quantify Poincare plots was proposed by Toichi et al. in [14]. It consists of the measurement of the Poincaré plot's maximal extension in the direction of the line of identity (L) and perpendicular to it (T), as shown in Figure 5. Their ratio is used as an additional measure.

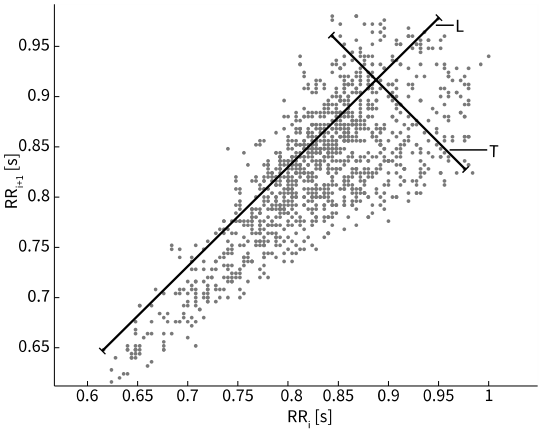


Figure 5: Longitudinal (L) and transversal (T) measurements of a Poincaré plot.

2 Data and Tests

2.1 Data

All data used to test the indices were taken from Physionet.org [15], a free-access, on-line archive of physiological signals. Physionet.org guarantees that all data have been fully anonymised, and may be used without further institutional review board approval.

To create a control group, the *Normal Sinus Rhythm RR Interval Database* was used. It consists of 54 ECG recordings, each one approximately 24 hours long. It contains semi-automatically annotated heart rate data of subjects with normal sinus rhythm (30 men, aged 28.5 to 76, and 24 women, aged 58 to 73) digitized at a sample frequency of 128 *Hz* [15].

For pathological heart rate data the *Massachusetts Institute of Technology (MIT) - Boston's Beth Israel Hospital (BIH) Arrhythmia Database* was used. It contains 48 half-hour recordings, sampled with a frequency of 360 *Hz*, from 47 subjects (25 men aged 32 to 89 years and 22 women aged 23 to 89 years) [16]. It consists of a set of randomly chosen recordings and 25 recordings especially chosen to include examples of uncommon but clinically important arrhythmias recorded at the BIH Arrhythmia Laboratory [16], all annotated manually.

2.2 Test procedure

For both data sets 1500 data points were taken from the middle of all recordings. These were filtered either according to the semi-automatic annotations as provided by Physionet, i.e., excluding all beats, which are not labeled as normal, or filtered via clustering, where only the sinusoidal cluster, i.e., the cluster consisting of the regular beats, is retained for analysis. The first 1000 data points were taken from these filtered points, in order to create a baseline, because after the filtering the recordings had different amount of data points. Afterwards their Poincaré plot measures were calculated.

To test if the measures show significant differences between the non-pathological and the pathological data set, a Wilcoxon rank sum test was applied to calculate the p-value, as recommended in [17], since most of the results were not normally distributed.

A test outcome was declared significant for $\rho < 0.05$ and very significant for $\rho < 0.01$.

3 Results

The results of the Wilcoxon rank sum test are shown in Table 1, with significant differences are marked with * and highly significant differences with **.

	Annotation Filtering	Cluster Filtering
SD1	0.8789	<0.01**
SD2	0.0149*	0.0564
SD1/SD2	0.0288*	<0.01**
L	0.0187 *	0. 1290
T	0. 5953	0. 1962
L/T	0. 3210	<0.01**

Table 1: The p-values of differences between pathological and non-pathological data sets of different Poincaré plot measures, either filtered via clustering or via annotations.

The ellipse fitting measures *SD2*, and *SD1/SD2* have significant differences for pathological and non-pathological heart rate data filtered with annotations. Of the longitudinal-transversal measures only *L* shows significant differences for the same data sets.

Data filtered via clustering have very significant differences for the two ellipse fitting measures *SD1* and *SD1/SD2*. These data sets have also very significantly different longitudinal-transversal measure *L/T*.

The following tables show the parameters of distribution for data filtered via annotations, see Table 2, and based on clustering, see Table 3. The first value is the median value of the measure for all recordings in this data set and the following two values describe the central range, i.e. they are the 2.5th and 97.5th percentiles.

	Non-Pathological Data	Pathological Data
SD1	0.0288, (0.0113, 1.6299)	0.0290, (0.0113, 0.2172)
SD2	0.0865, (0.0236, 1.5946)	0.0589, (0.0204, 0.3944)
SD1/SD2	0.3554, (0.1358, 1.2175)	0.5618, (0.1696, 1.5005)
L	0.6933, (0.1991 32.8089)	0.5003, (0.2609, 1.7208)
T	0.5359, (0.0710, 64.5699)	0.4639, (0.1034 2.2881)
L/T	1.2444, (0.5082, 4.7220)	0.9605, (0.6488, 3.5580)

Table 2. Parameters of distributions (median, 2.5th and 97.5th percentile) of data filtered via annotations.

	Non-Pathological Data	Pathological Data
SD1	0.0184, (0.0078, 0.1313)	0.0249, (0.0113, 0.1729)
SD2	0.0743, (0.0230, 0.1856)	0.0544, (0.0188, 0.1592)
SD1/SD2	0.2527, (0.1089, 0.8478)	0.5354, (0.1925, 1.6819)
L	0.4558, (0.1428, 1.0422)	0.3564, (0.1319, 0.9899)
T	0.1933, (0.0599, 0.7737)	0.2157, (0.0661, 0.9868)
L/T	2.0893 (1.2270, 4.7136)	1.3403, (0.6908, 3.1703)

Table 3. Parameters of distributions (median, 2.5th and 97.5th percentile) of data filtered via clustering.

4 Discussion

No instances were found in literature, where Poincaré plots were filtered via clustering. If mentioned at all the filtering was done either manually or semi-automatically.

A visual comparison of the data sets showed that both filtering methods could not find every outlier, but data filtered via annotations had a higher rate of unfiltered outliers, which also had a greater distance to the rest of the points, compared to the unfiltered outliers after cluster based filtering.

Therefore, Table 1 could be interpreted as an indicator for a higher sensitivity of the ellipse fitting method to outliers, which also reduce the differences between pathological and non-pathological data sets.

Comparing Table 2 and 3 one sees, that data filtered via annotations has larger 95th percentiles for *SD1*, *SD2*, *L* and *T*. This is also due to a higher number of outliers and their larger distances for these data.

The Ellipse fitting measures indicate in Table 2 and Table 3 wider but shorter Poincaré plots for pathological heart rate data filtered via annotations. The same is true for the longitudinal measures in Table 3. This behaviour is in accordance to the traditional interpretation of Poincaré plot shapes [18].

5 Conclusion

The fully automated filtering via clustering shows no drawbacks compared to the traditional method of ECG annotation based filtering for HRV analysis via Poincaré plots. This is done with a largely reduced effort in contrast to the annotation based filtering method.

References

[1] Nichols M, Townsend N, Luengo-Fernandez R, Leal J. *European cardiovascular disease statistics.* European Heart Network; 2012.

[2] Bundkirchen A, Schwinger RHG. Epidemiology and economic burden of chronic heart failure. *European Heart Journal Supplements*. 2004; 6(suppl D), D57–D60.

[3] Acharya UR, Joseph KP, Kannathal N, Lim CM, Suri JS, Heart rate variability: a review. *Med Bio Eng Comput*. 2006; 44(12). 1031–1051.

[4] Guidelines -- Heart rate variability. *European Heart Journal.*1996; 17: 354–381.

[5] Michaels DC, Chialvo DR, Matyas EP, Jalife J, Chaotic activity in a mathematical model of the vagally driven sinoatrial node. *Circ. Res.* 1989; 65(5): 1350–1360.

[6] Thong, T., Geometric measures of Poincare plots for the detection of small sympathovagal shifts. *Conf Proc IEEE Eng Med Biol Soc*. 2007; 2007: 4641–4644.

[7] Piskorski J, Guzik P. Filtering poincare plots. *Computational Methods in Science and Technology*. 2005; 39–48.

[8] Rodriguez A, Laio A. Clustering by fast search and find of density peaks. *Science*. 2014; 344(6191): 1492–1496.

[9] Ester M, Kriegel HP, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD*; 1996, 96, 226–231.

[10] Xia L, Jing J, An Ensemble Density-based Clustering Method. *ISKE*; Oct. 2007.

[11] Orth-Gomer K, Hogstedt C, Bodin L, Söderholm B, Frequency of extrasystoles in healthy male employees. *Br Heart J*. 1986; 55(3): 259–264.

[12] Bjerregaard, P., Premature beats in healthy subjects 40-79 years of age. *Eur. Heart J.* 1982; 3(6): 493–503.

[13] Brennan M, Palaniswami M. Do existing measures of Poincare plot geometry reflect nonlinear features of heart rate variability?. *IEEE Trans Biomed Eng*. 2001; 48(11): 1342–1347.

[14] Toichi M, Sugiura T, Murai T, Sengoku A. A new method of assessing cardiac autonomic function and its comparison with spectral analysis and coefficient of variation of R-R interval. *J. Auton. Nerv. Syst.* 1997; 62(1): 79–84.

[15] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, Ivanov PC, Mark RG, Mietus JE, Moody GB, C. K. Peng K, Stanley HE. PhysioBank, PhysioToolkit, and PhysioNet : Components of a New Research Resource for Complex Physiologic Signals. *Circulation*. 2000; 101(23): e215–e220.

[16] Moody GB, Mark RG. The impact of the MIT-BIH arrhythmia database. *IEEE Eng Med Biol Mag. 2001;* 20(3): 45–50.

[17] Die Auswahl statistischer Tests und Maße. 1999; 50(3):157–164.

[18] Esperer HD, Esperer C, Cohen RJ. Cardiac arrhythmias imprint specific signatures on Lorenz plots. *Ann Noninvasive Electrocardiol*. 2008; 13(1): 44–60, Jan. 2008.

Computational Aspects of Models for Minimizing the Effects of Ectopic Beats on Heart Rate Variability

Martin Frank[1,2], Martin Bachler[1,2*], Siegfried Wassertheurer[2], Christopher Mayer[2]

[1] Institute of Analysis and Scientific Computing, Vienna University of Technology, Wiedner Haupstraße 8-10, 1040 Vienna, Austria; * *martin.bachler@tuwien.ac.at*

[2] AIT Austrian Institute of Technology GmbH, Health & Environment Department, Biomedical Systems, Donau-City-Straße 1, 1220 Vienna, Austria

Simulation Notes Europe SNE 24(3-4), 2014, 143 - 148
DOI: 10.11128/sne.24.tn.10258
Received: June 10, 2014; Revised August 15, 2014;
Accepted: October 10, 2014;

Abstract. Several studies have highlighted the need of reliable markers to determine cardiac health. One of the most promising markers is the heart rate variability (HRV). However, artifacts, such as ectopic beats, have to be corrected before HRV parameters can be reliably used. This paper reviews several recently established models that are used for ectopic beat correction, with an emphasis on execution time and memory requirements. In general, physiologic models require far more execution time, while memory demand is compareable to simpler algorithms. Moreover, physiologic models result in better corrections, compared to simpler methods. Therefore, we conclude that physiologic models are prefferable since the execution time is low enough to use all models in an online approach.

Introduction

The latest reports of the "American Heart Association" and the "European Heart Network" highlight that the leading cause of death are heart diseases in the United States [1], and diseases of the heart and circulatory system in Europe [2]. Since early action is the key of surviving heart failure, reliable markers have to be established [1]. The so called heart rate variability (HRV) is one of the best suited markers for the relationship between the autonomic nervous system and cardiovascular mortality [3]. HRV parameters are immensely influenced by the presence of ectopic beats, which have to be corrected before performing HRV analysis [3][7].

In the last few years several different processing and correction methods have emerged. However, no comprehensive comparative study is available yet. The approaches may be categorized in (1) interpolation and removal [8]-[10], (2) filtering [6],[11][14] and (3) model-based approaches [4][15]-19].

The goal of this work is to provide a comparative review of different model-based approaches for the correction of ectopic beats. Thus, different test cases are defined to perform an objective comparison using artifically corrupted error-free RR interval time series. In this work we focus on the computational aspects, i.e. the execution time and the required memory, of the models under investigation. A comprehensive evaluation of the medical parameters and inplications of ectopic beat correction on HRV measures can be found elsewhere.

1 Methods & Models

In total, seven model-based approaches were compared. All these algorithms were entirely self-implemented and tested in Matlab version 2007b (The Mathworks Inc., Natick, US) based on literature.

1.1 Ectopic Beat Correction models

The first correction model is called "buffer with combination rules" (BUFFER), a fast online correction algorithm that was introduced by Rand et al. [18]. It uses several combination and/or splitting rules of neighbouring RR intervals in the following pre-defined order, as given in Figure 1.

Correction	Description
Split	Missed heartbeat; divide IBI into two equal intervals
Split 3	Two missed heartbeats; Split IBI into three equal intervals
Combine	False trigger; combine two IBIs into one
Combine 2 / Split 2	Replace two IBIs with their average
Combine 3 / Split 2	Get two new IBIs as average of three
Combine 2 / Split 3	Get three new IBIs as average of two
Combine 3 / Split 3	Replace three IBI values with their average
Uncorrected	Could not apply any rule, but IBI appears faulty

Figure 1: BUFFER algorithm by Rand et al. [18].

Before applying the correction, its usefulness is determined by comparing the RR interval under consideration to statistical parameters of the output buffer containing *n* preceding correct RR intervals (five RR intervals were used, as suggested in the original work). In addtion, the new RR interval must fit to those of the intervals in the input buffer covering *m* unprocessed RR intervals accounting for at least 6s. The buffer is also able to detect ectopic beats on its own.

'Gross positioning of beats' (GP-IIA) was suggested by Mateo et al. as coarse positioning of beats [4]. This method also performs a classification of all beats. By means of Lagrange's interpolation formula the derivative of the instantaneous heart rate is estimated at the kth beat by

$$|\dot{r_k}| = 2\left|\frac{t_{k-1} - 2t_k + t_{k+1}}{(t_{k-1} - t_k)(t_{k-1} - t_{k+1})(t_k - t_{k+1})}\right| < U, \quad (1)$$

where U is a predefined threshold that is calculated by

$$\mathrm{U} = 4.3 \cdot \mathrm{std}(r_k), \quad (2)$$

and limited by 0.5. Based on six different test cases the beat type is determined. Single ectopic beats are simply shifted to an intermediate position between the previous and following normal sinus beat. False-positives (FPs) are deleted and false-negatives (FNs) are corrected by insertion of an intermediate evenly spaced beat. Consecutive ectopic beats are corrected by insertion of multiple evenly spaced beats.

In contrast to the former mentioned algorithms, the 'integral pulse frequency modulation' (IPFM) model relies on a physiological relationship. This model is able to predict the autonomous nervous system (ANS) activity on the sinoatrial (SA) node by simulating the series of cardiac events as firings of the SA node [20]. The IPFM model integrates the input signal until a beat is generated and is then reset to zero [4]. Hence, the k^{th} beat can be interpreted as the integration of the instantaneous heart rate over the actual RR interval.

The index of the k^{th} beat can be calculated by

$$\mathrm{k} = \int_0^{t_k} \frac{1 + m(t)}{T}\, dt, \quad (3)$$

where t_k denotes the occurrence time of the k^{th} beat, $m(t)$ is the modulating part of the heart rate and T is the mean RR interval length.

The IPFM model with s-parameter (IPFM-S) was introduced by Mateo et al. [4]. According to the IPFM model, the integrator is reset too early if an ectopic beat occurs, resulting in a lower integration value than expected (denoted as s). An indexing function of the beat occurrence times is introduced, which is split into a forward and a backward function. The forward function is based on the normal sinus beats prior to the ectopic beat and the backward function is based on the normal sinus beats afterwards. Both functions are extrapolated to the neighbouring beat until they overlay. The vertical difference of the two indexing functions is the s-parameter:

$$\hat{\mathrm{s}} = \frac{1}{t(k_e) - t(k_e - 1 + s)} \cdot \cdot \int_{t(k_e - 1 + s)}^{t(k_e)} (\hat{x}^f(t) - \hat{x}^b(t))dt, \quad (4)$$

where k_e denotes the index of the ectopic beat, $\hat{x}^f$ is the forward indexing function, and $\hat{x}^b$is the backward indexing function.

The IPFM model with δ-parameter (IPFM-D) was suggested by Solem et al., since calculation of the s-parameter may be rather time consuming [15]. The δ-parameter corresponds to the time shift of the beat occurrence times followed by an ectopic beat and is related to the IPFM model by

$$\int_0^{t_k} 1 + m(\tau)\, d\tau = kT_0 + \delta. \quad (5)$$

Different δ - parameters may be used, dependent on how many beats prior to the ectopic beat are involved in the calculation (δ1, δ2 and δ3). Solem et al. mentioned that the consideration of more than one prior beat does not enhance the correction ability [15]. Hence, the δ1-parameter was used in this work. Its performance is nearly identical to that of the s-parameter, despite of a reduction in computation time.

The IPFM model with cost function (IPFM-C) was introduced by Brennan et al. [16]. It was designed to only account for single premature ventricular contractions (PVCs).

Basically, the cost function $C(t_e)$ calculates the quadratic deviation from the mean of the impulse height of the integration function in dependence of the beat occurrence time points:

$$C(t_e) = \sum_{S_k \in \psi} (S_k - E[\psi])^2, \quad (6)$$

where S_k denotes the height of the integration value at each beat occurrence time at the reset point, and can be calculated as follows:

$$S_k = \frac{1}{2\pi f_c} \sum_{j=k-M}^{k+M+1} Si(2\pi f_c(t_{k+1} - t_j) - Si(2\pi f_c(t_k - t_j)), \quad (7)$$

where Si is the sinc function and f_c is calculated as the reciprocal of the doubled mean RR interval length $\bar{I}$:

$$f_c = \frac{1}{2 \cdot \bar{I}} \quad (8)$$

Further, $\bar{I}$ is the threshold for the integrate-to-threshold process (the integration function). The mean integration height $E[\psi]$ may be approximated by the following equation:

$$E[\psi] \cong E[S_k] = \frac{1}{2M+2} \sum_{k=e-M-1}^{e+M} S_k. \quad (9)$$

Thereby, just the $\pm M$ adjacent impulse heights ψ, with respect to t_e, are considered. The new beat insertion time corresponds to the lowest costs.

'Trend predict correction' (TPC-HT) was developed by Wen et al. as a method that corrects ectopic beats based on trend correlation of the heart timing signal [17]. Each predicted RR interval is composed of two parts, the trend and the turbulence:

$$RRI_{pred} = RRI_{trend} + RRI_{turb}. \quad (10)$$

The trend is simply calculated as the weighted mean of the n previous NN intervals:

$$RRI_{trend} = \sum_{t=t_{k-n}}^{t_{k-1}} w(t) \cdot RR(t), \quad (11)$$

where $w(t)$ are time dependent exponential weights as described by Citi et al. [19]. The turbulence can be seen as the slope of the previous NN intervals and is approximated by the following calculation:

$$RRI_{turb} = I[t_k] \cdot E[t_k], \quad (12)$$

where $I[t_k]$ is just the sign of the turbulence and $E[t_k]$ is the quantity. $I[t_k]$ can be judged by the signs of the slopes at t_{k-1} and t_{k-2}:

$$I[t_k] = \frac{k_1 \cdot k_2 \cdot (k_1 + k_2)}{|k_1 \cdot k_2 \cdot (k_1 + k_2)|}. \quad (13)$$

The slopes are determined by the following calculation (a tiny value is added to avoid zero, not shown):

$$k_1 = \frac{y(t_{k-1}) - y(t_{k-2})}{y(t_{k-1}) + y(t_{k-2})} \; and \quad (14)$$

$$k_2 = \frac{y(t_{k-2}) - y(t_{k-3})}{y(t_{k-2}) + y(t_{k-3})}. \quad (15)$$

The quantity of the turbulence $E[t_k]$ is calculated by consideration of the two slopes k_1 and k_2, RRI_{trend} and the standard deviation of the previous RR intervals SD_{RRI}:

$$E[t_k] = RR_{trend} \cdot \frac{\sqrt{|k_1 \cdot k_2}}{\frac{a+b}{SD_{RRI}}}. \quad (16)$$

The two coefficients a and b are not specified byWen et al. [17] and thus were approximated by comparison of several values in different magnitudes.

'Point process with history dependent inverse Gaussian distribution' (PPHDIG) is an approach based on a physiologically motivated model, as the IPFM model. Citi et al. mentioned that the Gaussian random walk model with drift is an elementary, stochastic integrate-and-fire model that is able to reflect afferences to the SA node [19]. These excitatory inputs are responsible for the basal cardiac rhythm and the influence of the autonomic nervous system through the sympathetic and para-sympathetic inputs. They used a history-dependent, time-varying model based on the inverse Gaussian probability distribution of the waiting time until the next beat occurs. The probability of the length of the next RR interval, $\tau - u_k$, is described at any beat event u_k by the probability density function (PDF):

$$f\left(\tau - u_k \middle| \mu(H_k, \theta(t)), \lambda(\theta(t))\right) = \\ = \sqrt{\frac{\lambda(\theta(t))}{2\pi(\tau - u_k)^3}} \cdot e^{-\frac{1}{2}\frac{\lambda(\theta(t))\left(\tau - u_k - \mu(H_k, \theta(t))\right)^3}{(\tau - u_k)\mu^2}} \quad (17)$$

H_k is the history vector and contains the P previous RR intervals (standard parameter $P = 5$). Further, λ denotes the shape parameter and μ the mean of the inverse Gaussian distribution.

Both depend on the time varying parameters $\theta(t) = \theta_1(t), \ldots, \theta_{P+1}(t)$, whereby $\lambda(\theta(t))$ is simply $\theta_{P+1}(t)$. The history dependent mean is a regression of the past P RR intervals with time-varying weights:

$$\mu(H_k, \theta(t)) = \sum_{i=1}^{P} \theta_i(t) w_{k-i}. \quad (18)$$

The unknown time-varying parameter set $\theta(t)$ is estimated by a local maximum likelihood method. At each time t, the parameter vector that maximizes the local log likelihood in a given observation interval $U_{m:n}$ is obtained. m denotes the index of the first beat in this interval and n the index of the last beat.

$$L(\theta(t)|U_{m:n}) = \sum_{k=m+P}^{n-1} \omega(t-u_{k+1}) \cdot \log[f(u_{k+1} - u_k|\ \mu(H_k,\theta(t)),\lambda(\theta(t)))], \quad (19)$$

where $\omega(\tau) = e^{-\alpha\tau}$ is an exponential weighting function for the local likelihood.

Logarithmic probabilities of the following beat types are calculated [19]: Extra beat, missed beat, misplaced beat, two misplaced beats and resetting beat. The beat is only then classified as normal, if none of a set of hypotheses holds. Erroneous beats are corrected by deletion, insertion or shifting of beats. Before acceptance of a correction, an improvement check is performed. This ensures that the new RR interval time series is always more reliable than the original one.

2 Population and Tests

2.1 Study population

The used dataset contains 151 recordings, obtained from 17 women, aged 24 to 84, 109 men, aged 30 to 84, and from 25 unknown subjects. All known subjects, except of 22 patients, where just the ECG is described, suffer from at least one of the following heart diseases: Myocardial infarction, coronary artery disease, resting angina, effort angina, mixed angina or 1-, 2-, or 3-vessel disease. Only 5-min excerpts of ectopic free regions were used. All tests were performed 10 times with independently corrupted signals (N = 1510). The data is available via physionet [21], an online free-access-database of physiological datasets, and is composed of the European ST-T Database [22], the MIT-BIH Arrhythmia Database [23], and the QT Database [24].

2.2 Test cases

Two test cases were designed to determine the computational performance of each correction approach. In more detail, we monitored the computation time and peak memory using the built-in profiler function of Matlab. Further, we determined the correction ability of each method, whereby detailed results can be found elsewhere.

Test 1 contains only RR time series with single ectopic beats at a moderate density (about one to five ectopic beats with or without compensatory pause per 5-min signal).

Test 2 includes various ectopic beat types (PVC, premature atrial contraction (PAC), PVC couplets and triplets, bi- and trigeminy, sustained and non-sustained ventricular tachycardia).

3 Results

3.1 Test 1: Weakly corrupted RR interval time series

We could observe that physiologic models required far more computation time than simpler ones. The mean execution time is highest for PPHDIG (30 ms), followed by IPFM-C (26 ms, see Figure 2). In contrast, TPC-HT shows the lowest computation time (0.5 ms). However, we could not see this trend in peak memory. It is highest for IPFM-C ($9.7 \cdot 10^5$ kB), whereas all other algorithms result in a rather similar memory usage, ranging from $0.8 \cdot 10^5$ kB to $2.6 \cdot 10^5$ kB (see Figure 3).

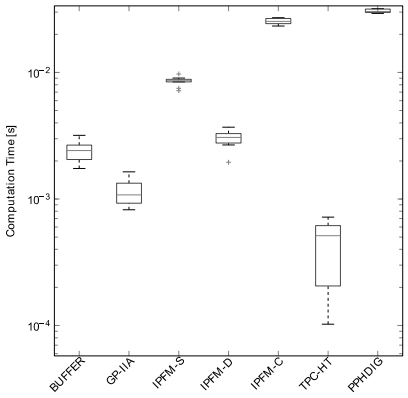


Figure 2: Box plot of computation time of test 1.

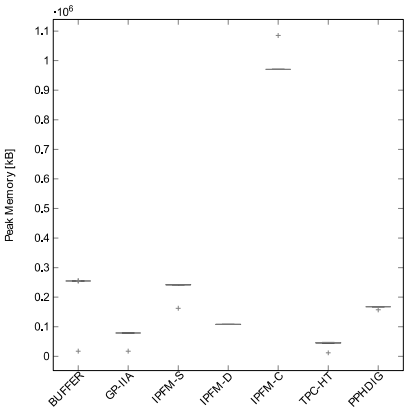


Figure 3: Box plot of peak memory of test 1.

Further, we could demonstrate that all presented models result in a remarkable reduction of the errors in the HRV (data not shown). Correction of single ectopic beats is best achieved by IPFM-S, IPFM-D, TPC-HT and PPHDIG, illustrating the better correction ability of physiologically based models.

3.2 Test 2: Strongly corrupted RR interval time series

Figure 4 illustrates that increasing the amount of artifacts results in a longer median computation time, with respect to test 1. It increased for PPHDIG from 30 to 285 ms, for IPFM-C from 26 to 222 ms and is lowest for GP-IIA (2 ms). On the contrary, peak memory nearly stayed constant. IPFM-C required $9.7 \cdot 10^5$ kB, whereas all other algorithms still resulted in a rather similar memory demand, ranging from $0.8 \cdot 10^5$ kB to $2.6 \cdot 10^5$ kB (see Figure 5).

Similarly to test 1, the correction ability decreased, especially for physiologically motivated approaches.

4 Discussion

Our findings clearly demonstrate the effictiveness of all algorithms to correct both, weakly and strongly corrupted RR interval time series in less than 300 ms for one 5 min ECG signal. Therefore, all of the presented algorithms may also be used in an online fashion. However, attention has to be paid that some algorithms, like the PPHDIG model, require an input of 50 s ECG-signal before the actual correction starts. Thus, some of the models induce a time lag, while still be able to perform an online correction.

The complexity of the different algorithms is well reflected by the median computation time. The PPHDIG model and all IPFM models require more computation time, since they rely on a physiologic relationship. Increasing the error density in test 2 results in a nearly 10-fold rise in computation time for PPHDIG and IPFM-C. This result is not only caused by the higher error density, but also by the fact that both methods were not designed to deal with more complicated artifacts. Therefore, physiologic models just perform best when correcting those specific artifacts they were actually designed for.

Interesstingly, we could not observe any relationship between the complexity of an algorithm and its required peak memory. Most models require a rather similar amount of peak memory, except of IPFM model with cost function.

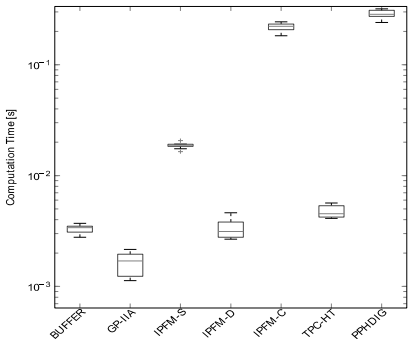


Figure 4: Box plot of computation time of test 2.

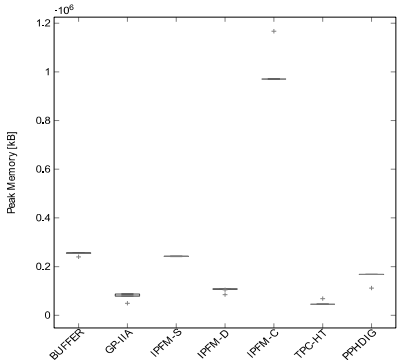


Figure 5: Box plot of peak memory of test 2.

The reason is that the calculation of sinc functions is very memory consuming. Therefore we used a look-up table to improve the computation. Although we could decrease the computation time and the peak memory, the values were still much higher than those of the other algorithms. The original design of this algorithm to correct just single PVCs seems to be responsible for the high computational effort when correcting other types of artifacts.

As stated by Solem et al. IPFM model with δ-parameter requires much less computation time and peak memory [15]. However, the correction performance is compareable only for single ectopic beats, since the range for IPFM-D is much higher when correcting multiple ectopic beats (data not shown).

4.1 Limitations

Though all methods under investigation were implemented to the best of our knowledge and belief, one cannot completely rule out the possibility of programming errors. However, as our results do not show any unexpected outliers, we consider the chances for this scenario negligible. Further, the memory measurement of the MATLAB profiler is an undocumented feature and thus not officialy supported by the Mathworks Company.

5 Conclusion

Physiologically motivated models are best suited to correct single ectopic beats at a compareable memory demand as simpler algorithms. Considering the still very low computation time (below 300 ms per 5 min ECG-signal), these models are also capable to be used in an online fashion.

References

[1] Roger VL, et al. *Heart disease and stroke statistics 2012 update a report from the American heart association.* Circulation, 125(1):e2–e220, 2012.

[2] Nichols M, Townsend N, Luengo-Fernandez R, Leal J, Gray A, Scarborough P, Rayner M. (2012). *European cardiovascular disease statistics 2012.* European Heart Network, Brussels, European Society of Cardiology, Sophia Antipolis, P104.

[3] Task Force of ESC and NASPE. *Heart rate variability: standards of measurement, physiological interpretation and clinical use.* Task Force of the European Society of Cardiology and the North American Society of Pacing and Electrophysiology. Circulation. Mar 1996; 93(5):1043–1065

[4] Mateo J and Laguna P. *Analysis of heart rate variability in the presence of ectopic beats using the heart timing signal.* IEEE Trans Biomed Eng. Mar 2003; 50(3):334–343

[5] Thuraisingham RA. *Preprocessing RR interval time series for heart rate variability analysis and estimates of standard deviation of RR intervals.* Comput Methods Programs Biomed. Jul 2006; 83(1):78–82

[6] Kumaravel N, Santhi C. Nonlinear filters for preprocessing heart rate variability signals. *International Journal of Computer Science and Network Security*. 2010; 10:250–254

[7] Colak OH. Preprocessing effects in time--frequency distributions and spectral analysis of heart rate variability. *Digital Signal Processing*, Elsevier, 2009; 19: 731-739

[8] Peltola MA. Role of editing of R-R intervals in the analysis of heart rate variability. *Front Physiol*. 2012; 3:148

[9] Lippman N, Stein KM, Lerman BB. Nonlinear predictive interpolation. A new method for the correction of ectopic beats for heart rate variability analysis. *J. Electrocardiol*.1993; 26(Suppl.:14–19)

[10] Peltola MA. Role of editing of R-R intervals in the analysis of heart rate variability. *Front Physiol*. 2012; 3:148

[11] Mietus JE. *Time domain measures: from variance to pnnx.* http://physionet.org/events/hrv-2006/mietus-1.pdf, 2006. [Online; Accessed: 03/2014].

[12] McNames J, Thong T, Aboy M. Impulse rejection filter for artifact removal in spectral analysis of biomedical signals. *Conf Proc IEEE Eng Med Biol Soc.* 2004;1:145–148

[13] Begum S, Islam MS, Ahmed MU, Funk P. K-nn based interpolation to handle artifacts for heart rate variability analysis. In Signal Processing and Information Technology (ISSPIT), *2011 IEEE International Symposium on*. 2011; P. 387–392

[14] Keenan DB. Detection and correction of ectopic beats for hrv analysis applying discrete wavelet transforms. *Int. J. Inf. Technol*. 2005; 2:54–60

[15] Solem K, Laguna P, Sornmo L. An efficient method for handling ectopic beats using the heart timing signal. *IEEE Trans Biomed Eng*. Jan 2006; 53(1):13–20

[16] Brennan M, Palaniswami M, Kamen P. A new model-based ectopic beat correction algorithm for heart rate variability. In Engineering in Medicine and Biology Society, 2001. *Proceedings of the 23rd Annual International Conference of the IEEE*, volume 1, pages 567–570 vol.1, 2001.

[17] Wen F, He FT. *An efficient method of addressing ectopic beats: new insight into data preprocessing of heart rate variability analysis.* J Zhejiang Univ Sci B. Dec 2011; 12(12):976–982

[18] Rand J, Hoover A, Fishel S, Moss J, Pappas J, Muth E. Real-time correction of heart interbeat intervals. *IEEE Trans Biomed Eng*. May 2007; 54(5):946–950

[19] Citi L, Brown EN, Barbieri R. A real-time automated point-process method for the detection and correction of erroneous and ectopic heartbeats. *IEEE Trans Biomed Eng*. Oct 2012; 59(10):2828–2837

[20] Ward S, Heneghan C, Nolan P. An integrate-and-fire based model of PP and PR variability in the human electrocardiogram. In Neural Engineering, 2003. Conference Proceedings. *First International IEEE EMBS Conference* on, pages 297–300, 2003.

[21] Goldberger AL, Amaral LAN, Glass L, Hausdorff JM, . Ivanov PCh, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. Physiobank, Physiotoolkit, and Physionet: components of a new research resource for complex physiologic signals. *Circulation*. 2000; 101(23):e215–e220

[22] A. Taddei, G. Distante, M. Emdin, P. Pisani, G. B. Moody, C. Zeelenberg, and C. Marchesi. The European ST-T database: standard for evaluating systems for the analysis of ST-T changes in ambulatory electrocardiography. *Eur. Heart J.*, 13(9):1164–1172, Sep 1992.

[23] Moody GB, Mark RG. The impact of the MIT-BIH arrhythmia database. *Engineering in Medicine and Biology Magazine*, IEEE. May 2001; 20(3):45–50

[24] Laguna P, Mark RG, Goldberg A, Moody GB. A database for evaluation of algorithms for measurement of QT and other waveform intervals in the ECG. *Computers in Cardiology* 1997, P. 673–676. IEEE, 1997.

Tool-Independent Distributed Simulations using Transmission Line Elements and the Functional Mock-up Interface

Robert Braun*, Petter Krus

Division of Fluid and Mechatronic Systems, Linköping University, SE-58183 Linköping, Sweden;
*robert.braun@liu.se

Simulation Notes Europe SNE 24(3-4), 2014, 149 - 154
DOI: 10.11128/sne.24.tn.10259
Received: Jan.10, 2014 (Selected SIMS 2013 Postconf. Publ.);
Accepted: June 15, 2014;

Abstract. This paper describes how models from different simulation tools can be connected and simulated on different processors by using the Functional Mockup Interface (FMI) and the transmission line element method (TLM). Interconnectivity between programs makes it possible to model each part of a complex system with the best suited tool, which will shorten the modelling time and increase the accuracy of the results. Because the system will be naturally partitioned, it is possible to identify weak links and replace them with transmission line elements, thereby introducing a controlled time delay. This makes the different parts of the system naturally independent, making it possible to simulate large aggregated system models with good performance on multi-core processors. The proposed method is demonstrated on an example model. A suggestion of an XML extension to the FMI standard for describing TLM ports is also presented.

Introduction

If different parts of a simulation model can be run on different processor cores, execution time can be considerably shortened. By using the transmission line element method (TLM) with independent distributed solvers, it is possible to achieve natural parallelism. This paper investigates the possibilities the combine this with the Functional Mock-up Interface (FMI), a standardised interface for connecting different simulation environments.

Using TLM and FMI together makes it possible to run distributed simulations with sub-models from different tools, which can fully exploit the benefits of multi-core processors.

First, the backgrounds of FMI, TLM, and the Hopsan simulation environment are explained. Then the implementations of import and export routines of FMI are presented. Finally, the validity of the method is confirmed by experiments on an example model.

1 Functional Mockup Interface

The Functional Mock-up Interface (FMI) is an open standardised interface for connecting simulation environments in a variety of ways. It is developed by the MODELISAR consortium, initiated by Dassault Systems [1]. There are four areas where FMI can be used:

- FMI for model exchange
- FMI for co-simulation
- FMI for applications
- FMI for PLM

The basic concept is to create a Functional Mockup Unit (FMU) from a model in one tool and then import it and use it in a target environment. An FMU consists of a compressed ZIP file with the FMU file extension. It contains an XML description of the contents and the simulation code as a set of C functions, either as binary files and/or compilable source code, which can be used by the host program. It can also contain documentation and a graphical icon. The use of a plain C interface makes FMUs compiler independent. They do, however, still depend on platform and architecture. With FMI for co-simulation the solver is included in the FMU, as opposed to FMI for model exchange, where the solver must be provided by host program.

This paper focuses only on model exchange, mainly because it is supported by a larger number of vendors. In the upcoming FMI 2.0 standard, the difference between co-simulation and model exchange will be reduced [2]. It is, however, not yet released and is not used in this paper.

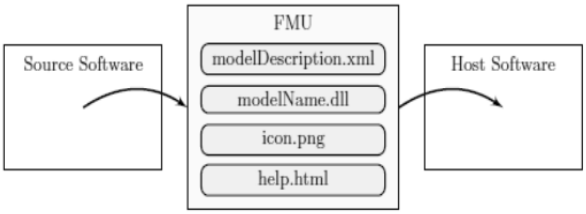


Figure 1: With the FMI standard, Functional Mockup Units (FMUs) can be used to exchange models among simulation tools.

2 Transmission Line Element Connections

The transmission line element method (TLM) is a method for partitioning models by introducing physically motivated time delays. It is related to the method of characteristics [3] and to transmission line modelling [4]. In physical systems, information propagation is always delayed by capacitances.

The concept with transmission line elements is to replace these capacitances in the model with transmission line elements, modelled as characteristic impedances. This method makes it possible to maintain accurate wave propagation, which is not possible by using only pure time delays.

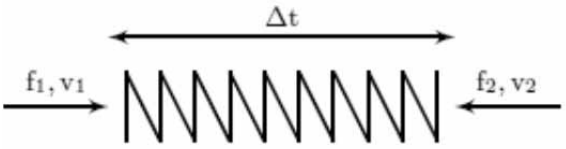


Figure 2: An example of a transmission line element is the linear mechanical spring.

An example of a TLM element is the linear mechanical spring, as shown in Figure 2. As can be seen it is subjected to two forces, f_1 and f_2, from the left and the right side, respectively. According to equation (1) each force is defined by a function of the velocity at the same side and the delayed force and delayed velocity from the other side. This implies that one end is always independent of the opposite end at the same point of time.

This enables the use of distributed solvers, where each sub-component in the model solves its own equations. This approach is very suitable for co-simulation and parallel execution [5].

$$\begin{aligned} f_1(t) &= F\big(v_1(t), f_2(t-\Delta t), v_2(t-\Delta t)\big) \\ f_2(t) &= G\big(v_2(t), f_1(t-\Delta t), v_1(t-\Delta t)\big) \end{aligned} \quad (1)$$

When communicating between different programs using TLM connections, it is important thatthe variables in each connection are clearly specified. This can be done manually by the user when importing the model to the host environment, although this can be quite cumbersome. An alternative solution would be to include this information in the XML specification in the FMU. A TLM connection is defined as four variables; intensity, flow, wave variable, and characteristic impedance. Some additional variables may also be necessary depending on the physical domain, such as position and equivalent inertia for mechanical connections.

Even though similarities between different domains exist, it is unfortunately not possible to use a general definition; the set of variables will always need to be hard-coded depending on the physical type of the connection. For example, an incompressible fluid needs only one flow variable, while a compressible fluid might need variables for both mass flow and volume flow. It is also necessary to provide information of whether the FMU is a resistive component (Q-type) or a transmission line connection (C-type). In order to transfer this information, the following addition to the XML description is suggested, see Figure 3

```
<tlmConnections>
  <tlmConnection type="q" domain="hydraulic">
    <q>q1</q>
    <p>p1</p>
    <c>c1</c>
    <Zc>Zc1</Zc>
  </tlmConnection>
  <tlmConnection type="c" domain="mechanic">
    <F>F2</F>
    <x>x2</x>
    <v>v2</v>
    <me>M2</me>
    <c>c2</c>
    <Zc>Zc2</Zc>
  </tlmConnection>
</tlmConnections>
```

Figure 3: An addition to the FMU XML description, describing TLM ports, is suggested.

3 Importing Functional Mock-up Units

All experiments in this paper are conducted in Hopsan, a distributed simulation environment developed at Linköping University [6]. It is based on the transmission line element method and uses distributed solvers. Components in Hopsan are quite similar to FMUs; they consist of a pre-compiled shared library file and an XML description file. The library file, however, is linked against Hopsan from where it inherits classes and can thus not be used standalone.

The process of importing an FMU to *Hopsan* is implemented in the following way, see Figure 5. First the files are extracted to a temporary directory. Then the XML is parsed, including TLM specifications if present. Then the source file for the component library (`fmuLib.cc`) and the Hopsan component (`fmuName.hpp`) are generated and compiled to a shared library. A simple solver that uses the forward Euler method is also included, in order to be able to solve the equations.

This would not be required with FMI for co-simulation, where the solver is included in the FMU. Finally, the XML description (`fmuName.xml`) is generated. The component can then be loaded from *Hopsan* and will then be available from the component library.

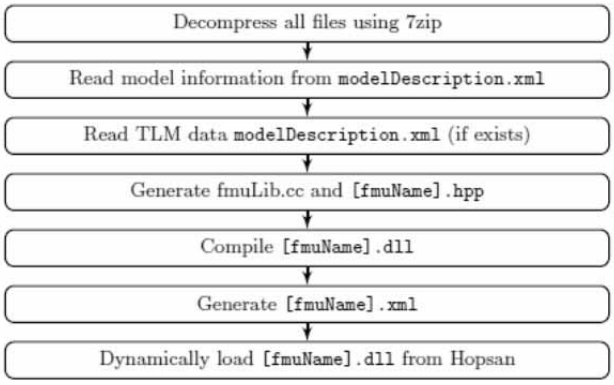


Figure 4: FMUs are imported to Hopsan by compiling a wrapper library.

Hopsan is an object-oriented simulation environment where everything is pre-compiled. Components are objects that can be connected by node objects [7].

During the simulation, each component solves itself independently. Therefore no compilation prior to simulation is required. The main simulation uses fixed time steps for communication between components and nodes. It is, however, up to each component to decide how to perform its calculations. Thus, it is possible to use variable time steps inside a certain component. Theoretically, it is possible to use variable time steps also for the whole model, but previous experiments show that this is generally not worth the effort [8].

4 Exporting Models to Functional Mock-Up Units

Hopsan does not contain any equations or numerical solvers. The natural method would thus be to export an FMU for co-simulation. For compatibility with other software, however, the model exchange interface is used. The exported FMU does, however, actually solve itself and is basically working as an FMU for co-simulation.

According to the FMI specifications, only one shared library file is allowed. It is thus not possible to link against a pre-compiled Hopsan library file; the simulation core must be compiled into the FMU library. The model file is also included as a string variable in a header file and is loaded during initialisation.

Another factor which must be considered is that the FMI standard requires a plain C interface to ensure cross-compiler compatibility. Because the Hopsan simulation core is written in C++, a wrapper file is used to allow access to all required functions without using C++ features, such as classes and objects.

The export process begins with parsing the model and generating the XML description, together with the TLM extension if required. All source code files are then generated, including the model header file and the wrapper files. These files are then compiled along with the source code from the Hopsan simulation core and the required FMI source code. Finally, the binary and the XML description are compressed using 7zip. It is important to use the 'deflate' compression method, to ensure compatibility.

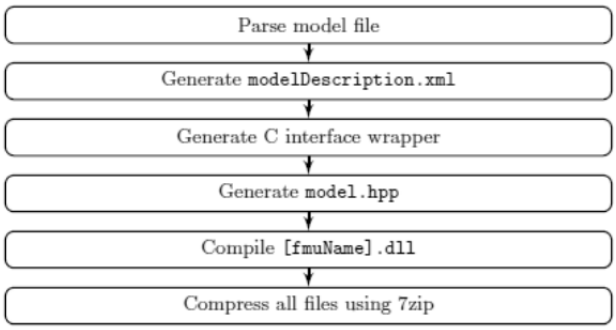


Figure 5: FMUs are exported from Hopsan by compiling the simulation core with a C interface using a wrapper.

FMUs exported from Hopsan are successfully validated by importing them into OpenModelica, Dymola, FMU SDK, and also back into Hopsan itself. They are verified with FMU Checker version 1.0.2. No errors or warnings are reported.

5 Example Simulation

In order to demonstrate the proposed method, an example model is created, see Figure 6. It consists of a four-wheel vehicle with an engine, a mechanical gearbox, and a hydraulic transmission. The hydraulic system is modelled using built-in pre-compiled C++ components in Hopsan.

The engine is modelled as a PI-controlled torque source with velocity feedback. As a demonstration, it is exported from Hopsan to an FMU and then imported back into Hopsan. The brake component is created in the same way. Models for the vehicle, the wheels, and the gearbox are all equation-based models created in OpenModelica, an open-source Modelica-based simulation tool [9]. They are then exported to Hopsan as FMUs. The vehicle consists of a linear inertia with a drag coefficient parameter. Wheels are modelled as rotating inertias with ports for drive shaft, brakes, and attachments in the vehicle. The gearbox is modelled as a rotating inertia with changeable gear ratio.

All components are connected through transmission line elements, representing the shafts and mechanical connections. This means that stiffness is replaced by characteristic impedances and a time delay, which results in a decoupled system with good wave propagation accuracy. Pre-defined components in Hopsan are used for this purpose.

In order to verify the functionality of the model, a simple drive cycle is simulated. The vehicle is first accelerated to 50 km/h and then to 70 km/h. It is then slowed down to 30 km/h and finally comes to a stop, see Figure 7.

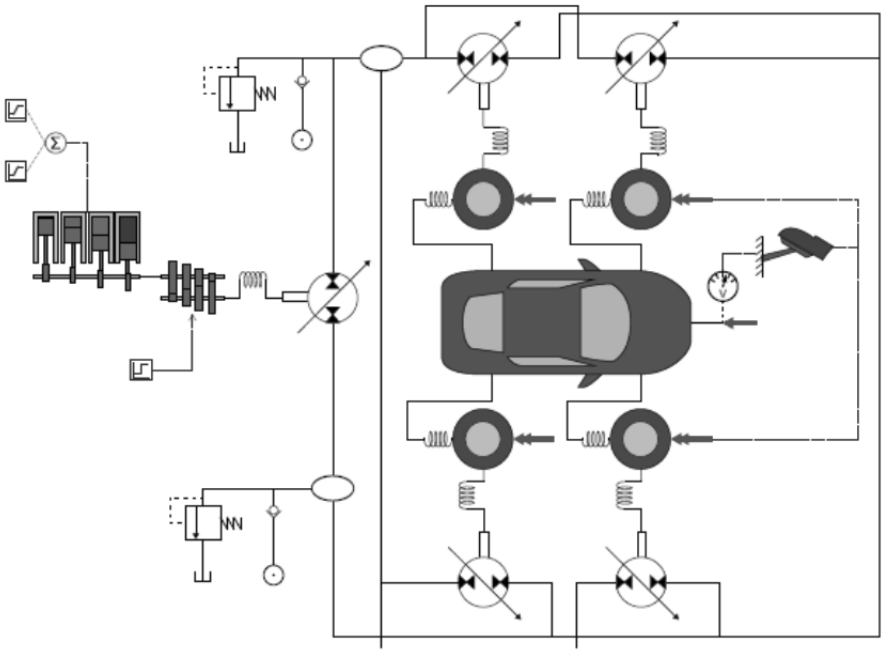


Figure 6: An example model that describes a four-wheel vehicle with a simple hydraulic transmission is used to verify the proposed method.

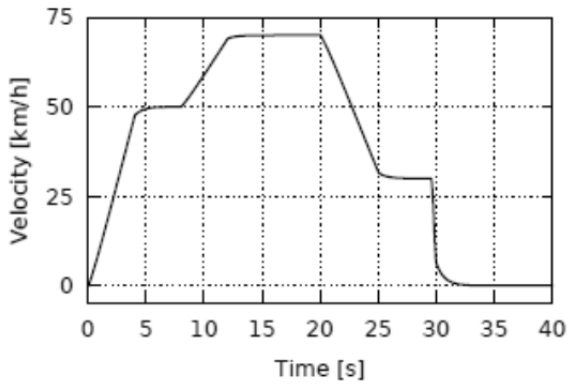


Figure 7: A simple drive cyxle is used to verify the functionality of the example model.

Load balancing is an essential aspect in parallel programming. If the work is not equally distributed over the threads, the speed-up will be limited by the slowest thread. In Hopsan this is solved by an automatic algorithm that measures the simulation time for each component over a few time steps before the actual simulation. This information is in turn used to distribute the components evenly over the simulation threads [5]. The average measured time per iteration for each sub-component type in the example model is shown in Table 1. Note that these measurements are made on different sub-models and can thus not be used to compare the performance of different simulation tools.

Sub-model	Time/iteration
Wheel (OpenModelica, FMU)	1.192 μs
Gearbox (OpenModelica, FMU)	1.083 μs
Vehicle (OpenModelica, FMU)	0.574 μs
Brake (Hopsan, FMU)	0.476 μs
Engine (Hopsan, FMU)	0.189 μs
Relief Valve (Hopsan, built-in)	0.121 μs
Pump (Hopsan, built-in)	0.115 μs
Volume (Hopsan, built-in)	0.024 μs

Table 1: The simulation time for each sub-component ismeasured before the simulation, to achieve good load balancing.

The resulting distribution is shown in Table 2. Components of C-type are generally much faster than those of Q-type. In this case they only required 5.8% of the total time. For this reason, only the threads for Q-type components are analysed.

	Thread 1	Thread 2	Thread 3	Thread 4
	Wheel 1	Wheel 2	Wheel 3	Wheel 4
	Gearbox	Vehicle	Pump 1	Pump 2
			Relief Valve 1	Relief Valve 2
			Pump 3	Pump 4
			Pump 5	Check Valve 1
				Check Valve 2
Total time	2.275 μs	1.765 μs	1.644 μs	1.606 μs

Table 2: Sub-components are automatically distributed over the simulation threads.

As can be seen, a decent although not perfect load balancing is achieved. There are also overhead time costs from time measurements and thread synchronisation. Simulation time is, however, still more than twice as fast with four threads compared to with one thread. See Table 3 for simulation times for 10,000 time steps with different numbers of processors. The time reduction from parallel simulation will increase when larger models are used. Theoretical maximum of speed-up is limited by the number of processor cores [5].

Threads	Simulation time
1	3307 ms
2	2091 ms
3	1466 ms

Table 3: Parallel execution reduces simulation time.

6 Conclusions

This paper shows that it is possible to combine the FMI standard with the transmission line element method. This makes it possible to simulate large aggregated models, consisting of submodels from different modelling tools, in parallel on multi-core processors. Simulation time can then be significantly reduced. An interesting continuation could be real-time applications, where simulation performance is a critical aspect. Other possible future work could be to investigate higher level modelling methods for describing aggregated FMI models.

Experiments were performed with FMI for model exchange using a simple solver. FMI for co-simulation would be more suitable, but is so for not supported by many simulation tools. Such difficulties will be easier to overcome with the FMI 2.0 standard, where co-simulation and model exchange will be harmonised.

Acknowledgement. This work was supported by ProViking research school and the Swedish Foundation for Strategic Research (SSF).

This contribution is a post-conference publication from SIMS 2013 Conference (54th SIMS Conference, Bergen University College, Norway, October 16-18, 2013). The contribution was originally published in the Proceedings of SIMS 2013, to be found http://www.scansims.org/ sims2013/SIMS2013.pdf.

References

[1] Blochwitz T, et al. The functional mockup interface for tool independent exchange of simulation models. In: *Modelica'2011 Conference*, Proceedings of the 8th International Modelica Conference; 2011 March; P.20-22

[2] Blochwitz T, et al. Functional mockup interface 2.0: The standard for tool independent exchange of simulation models. In: *9th International Modelica Conference*; 2012; Munich

[3] Auslander DM. Distributed system simulation with bilateral delay-line models. *J. Fluids Eng*. 1968; 90(2): 195–200. doi: 10.1115/1.3605079

[4] Johns PB, O'Brian MA. Use of the transmission line modelling (T.L.M) method to solve nonlinear lumped networks. *Radio And Electronic Engineer*. 1980; 50(1/2): 59–70. doi: 10.1049/ree.1980.0006

[5] Braun R, Nordin P, Eriksson B, Krus P. High Performance System Simulation Using Multiple Processor Cores. In: Sairiala H, Koskinen KT, editors. SICFP'11. *The Twelfth Scandinavian International Conference On Fluid Power*; 2011 May; Tampere

[6] Axin M, Braun R, Dell'Amico A, Eriksson B, Nordin P, Pettersson K, Staack I, Krus P. Next Generation Simulation Software Using Transmission Line Elements. In: Johnston N and Plummer AR, editors. *Fluid Power and Motion* Control; 2010 October; Bath, England. 265-276

[7] Eriksson B, Nordin P, Krus P. Hopsan NG, A C++ Implementation Using The TLM Simulation Technique. SIMS2010. In: *Proceedings of The 51st Conference On Simulation And Modelling;* 2010; Oulu, Finland

[8] Jansson A, Krus P, Palmberg J-O. Variable time step size applied to simulation of fluid power systems using transmission line elements. In: *Fifth Bath International Fluid Power Workshop*; 1992; Bath, England

[9] OpenModelica website [Internet]. [cited 2013 July]. Available from: https://www.openmodelica.org/

Methodology for Modeling, Parameter Estimation, and Validation of Powertrain Torsional Vibration

Neda Nickmehr*, Lars Eriksson, Jan Åslund

Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden;
*neda.nickmehr@isy.liu.se

Simulation Notes Europe SNE 24(3-4), 2014, 155 - 160
DOI: 10.11128/sne.24.tn.10261
Received: Jan.10, 2014 (Selected SIMS 2013 Postconf. Publ.);
Accepted: June 30, 2014;

Abstract. A vehicular powertrain is a lightly damped dynamic system that transfers the engine torque to the driving wheels through a number of inertias and elastic elements. Therefore, it is prone to vibrate and emit noise when disturbances are applied. Providing a methodology, for powertrain vibration modeling and simulation, is one of the key steps in various research topics in the field of automobile engineering. Verification of the engine crankshaft torsion and vibration model, as a subsystem of the powertrain, is proposed in this paper. This is achieved by constructing a rotational multi-body system in MATLAB and utilizing nonlinear least squares method for estimation of the model parameters. The simulated engine angular velocity is compared to the measured data, from a car, which shows a good agreement.

Introduction

There are different applications in automotive industry where powertrain vibration modeling is needed. Two of the more important cases are as follows:

- Passenger comfort is important for the customers, and consequently the car manufacturers. Powertrain dynamics is one of the main sources of noise, vibration, and harshness (NVH) inside the cars, and reducing its torsional vibration, to an acceptable level, is desirable. A typical powertrain NVH spectrum for a passenger car contains a considerable range of frequencies, from 2 Hz to 5000 Hz [1]. To achieve suitable ride quality, there is a need for better understanding of the dynamics, and a good model is a valuable instrument.
- To distinguish the effects of different excitation sources on the angular velocities of the powertrain parts, is not a trivial task. This causes difficulties for any type of miss-behavior detection. Examples of possible input disturbances are, combustion variation such as, misfire, cold start and cylinder variations; crankshaft torsional vibrations; road roughness; underinflated tires, etc. By using an appropriate powertrain model, and studying the distinct disturbances influences on the simulated outputs, such as angular velocities, it is possible to gain understanding and detection of undesirable modes. Examples of such applications of a model are misfire and underinflated tires, which are investigated by considering the simulated flywheel and driving wheel velocity signals, respectively [2-3].

Powertrain vibration can be modeled by torsional elastic elements. Basic models are linear lumped spring-massdamper systems which can be extended by adding details for the dynamics of the various parts [4]. Rabeih developed a 14-degrees-of-freedom lumped parameter model, from the engine to the driving wheels, for torsional vibration analysis of the powertrain with a four-cylinder engine and manual transmission [5]. The proposed model was good enough for simulating free, steady, and transient situations.

However, no experimental evaluations were done. Crowther used a 6-degrees-of-freedom model to perform numerical simulations for transient vibrations [6]. A powertrain test rig, that was used for measuring torsional vibration, was also presented. Furthermore, powertrain nonlinear torsional phenomena such as, gear backlash and a multi-stage nonlinear clutch, have been studied by Couderc et al., where an experimental test rig was also developed to verify the simulation results [7].

In all the mentioned works, the system parameters are approximate to the passenger cars and no details are provided. In contrast to the previous mentioned studies, the contribution of this paper contains the following three elements. Firstly, building a slightly different model (using dampings between all the inertias). Secondly, focusing on the parameter estimation procedure for the nonlinear engine block model by considering idling condition and using the measurements from a car. Thirdly, performing sensitivity analysis to study the behaviour of the system output with respect to changing of different parameters. This will provide the answer for the question, how important are different parameters in a model.

There is an enormous literature about parameter estimation methods and system identifcation for linear and nonlinear systems, e.g., [8-9]. The procedure for estimating unknown states and parameters of dynamical systems, from noisy measurements, consists of three main tasks [8]:

1. Three basic entities:
   a. Measurements of the inputs and the outputs.
   b. A model structure.
   c. A rule by which candidate models can be assessed using the measurements
2. Model validation which is done using separate data from the estimation data.
3. The system identification loop.

In this work, in-cylinder pressures, and the engine angular velocity, are measured for different working cycles at the idling condition, step 1.(a). Then, the engine-block torsional vibration model is constructed with unknown parameters which have physical interpretations, namely, spring, mass, and damper components, step 1.(b) in the above description. This is called gray-box model. Further, the nonlinear least squares is used as the rule to estimate the model unknown parameters, step 1.(c), by utilizing the experimental data for one working cycle, from step 1.(a). This is named estimation cycle. Finally, the estimated model is validated by simulating the system, at another cycle, and then comparing the engine angular velocity with the validation data, which is step 2. It is possible that the obtained model is not able to pass validation test, then it is necessary to go back and revise different steps of the procedure, which is step 3.

1 Engine-block Model

The proposed powertrain model is a nonlinear lumped parameter system consisting of rotating masses, friction, damping, and stiffness elements, which is shown in Figure 1. There exists different subsystems, i.e., engine block, transmission, final drive, and the wheels. The total friction in each subsystem is modeled by a number of dampings, which are connected to the ground for all the rotating masses. The modular structure of the model makes it flexible enough to add (remove) components with respect to different vehicle configurations such as front-wheel, rear-wheel, or four-wheel drive cars. In order to achieve better identification performance, the suggestion of this work is to disengage the clutch, i.e. idling condition, so that there is no connection between the flywheel and the transmission. Thus the model is simplified to the engine-block and consequently the delivered torque at the flywheel is zero, since, the engine friction will cancel out the produced torque by the cylinders.

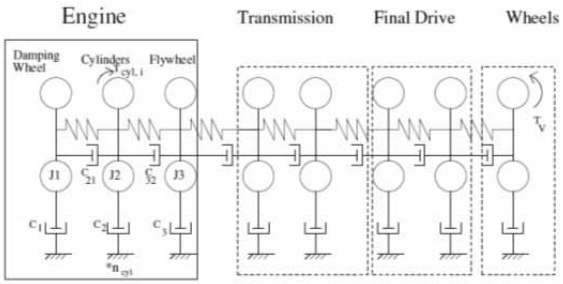


Figure 1: Powertrain model including engine block.

1.1 Mathematical modeling

According to Newton second law, the equations of motion for six different rotating masses, J_i, in the engine block which is a multiple-degrees-of-freedom system, are given by:

$$J_i\ddot{\theta}_i = C_{i+1,i}(\dot{\theta}_{i+1} - \dot{\theta}_i) - C_i\dot{\theta}_i + \\ + K_{i+1,i}(\theta_{i+1} - \theta_i), \text{ for } i = 1, \quad (1)$$

Damping wheel

$$J_i\ddot{\theta}_i = C_{i+1,i}(\dot{\theta}_{i+1} - \dot{\theta}_i) - C_i\dot{\theta}_i + \\ +K_{i+1,i}(\theta_{i+1} - \theta_i) - C_{i,i-1}(\dot{\theta}_i - \dot{\theta}_{i-1}) - \\ -K_{i,i-1}(\theta_i - \theta_{i-1}) + T_{cyl,i} \quad \text{for } i = 2,3,4,5, \quad (2)$$

Four cylinders

$$J_i\ddot{\theta}_i = -C_i\dot{\theta}_i - C_{i,i-1}(\dot{\theta}_i - \dot{\theta}_{i-1}) - K_{i,i-1}(\theta_i - \theta_{i-1}) \quad (3)$$

for $i = 6$, Flywheel

There θ_i, $\dot{\theta}_i$, and, $\ddot{\theta}_i$are angular position, angular velocity, and angular acceleration of the inertia at position i, respectively. Furthermore, C_i is the friction coefficient for element i, and $C_{i,i-1}$ and $K_{i,i-1}$ are the damping and stiffness coefficients, respectively, between two following inertias. The resulting torque, $T_{cyl,i}$, from cylinder i, which imposes the nonlinearity on the system, is calculated as a function of compression pressure force, $F_{c,i}$, and the piston mass force, $F_{p,i}$:

$$T_{cyl,i}(\tilde{\theta}_i) = \left(r\sin(\tilde{\theta}_i) + \frac{r^2 \sin(2\tilde{\theta}_i)}{2\sqrt{l^2 - r^2\sin^2(2\tilde{\theta}_i)}} \right) \times \left(F_{c,i}(\tilde{\theta}_i) + F_{p,i}(\tilde{\theta}_i) \right) \quad (4)$$

where $\tilde{\theta}_i = \theta_i + \delta\theta_i$, and $\delta\theta_i$ is the offset for each cylinder according to ignition order, here, 1-3-4-2. Also, r and l are crankshaft radius and connecting rod length, which are the geometrical characteristics of the cylinder model. More details on the equations and the model for the cylinder pressures and reciprocating torques can be found in Eriksson *et al*. [2].

1.2 State-space equations

The equations (1)-(3) can be transferred to the state-space form by performing two steps. First, utilizing the experimental data of the in-cylinder pressures, at idling, to obtain the delivered torque, $T_{cyl,i}(\tilde{\theta}_i)$, of cylinder i at different angular positions [2]. Second, defining new variables for angular velocities, $\dot{\theta}_i$, of each rotating mass, J_i. Therefore, there will be totally 12 states, including 6 angular positions, θ_i, and 6 angular velocities, $\dot{\theta}_i$. Then, the engine-block mathematical model in vector form, as a nonlinear ordinary differential equation, (ODE), can be written as follows:

$$\dot{\boldsymbol{x}} = \mathrm{f}(\boldsymbol{x}, \boldsymbol{p}), \quad y = \boldsymbol{x}(2) \quad (5)$$

where $\boldsymbol{x}$ is the state vector, y is the model output, which is damping wheel angular velocity $\dot{\theta}_2$, and $\boldsymbol{p}$ is the vector of unknown parameters. The complete powertrain model contains 42 unknown parameters. With the aid of symmetry and the proposed method of using idling conditions for parameters identification, the number of parameters, to be estimated in the engine-block, is reduced to 10. In Figure 1, 8 parameters are noted. The 2 remaining are damping between two cylinders, C_{xx}, and the piston reciprocating mass, m.

The vector of parameters, $\boldsymbol{p}$, can be categorized in two groups, i.e., 1) inertias and piston reciprocating mass, 2) friction coefficients and dampings. The crankshaft system is very stiff, therefore the stiffness coefficients are limited to high values and are hence not estimated.

2 Engine-block Parameter Estimation

One of the most well-known parameter estimation procedures, in nonlinear systems, is to write the estimation problem as a nonlinear optimization problem, where the goal is to minimize the difference between the predicted output, from the model, and the observations. The nonlinear least squares method, which is applied in this paper, is a commonly used optimization formulation. It searches the parameters values which minimize the squared errors.

2.1 Nonlinear least squares (NLS) formulation of the problem

A nonlinear regression model is described by an equation of the form:

$$z_i = g(\boldsymbol{u}_i, \boldsymbol{p}) + \varepsilon_i \quad (6)$$

where z_i denotes the observations, $\boldsymbol{p}$ is the set of unknown parameters that are to be estimated, $\boldsymbol{u}_i$ is a $1 \times k$ vector of known values, and g is a nonlinear function [10]. The least squares estimate is computed by minimizing the following objective function:

$$V_N(\boldsymbol{p}) = \sum_{i=1}^{i=N} (z_i - g(\boldsymbol{u}_i, \boldsymbol{p}))^2 = \sum_{i=1}^{i=N} \varepsilon_i(\boldsymbol{p})^2 \quad (7)$$

where, for the engine block estimation problem, $\varepsilon_i(\boldsymbol{p})$ is the discrepancy between the simulated and the measured damping wheel angular velocity. In general, the solution of the above mentioned nonlinear minimization problem, is not available analytically, thus numerical nonlinear optimization algorithms are used [10].

2.2 Forward sensitivity analysis (FSA)

Sensitivity analysis is a procedure for studying the influence of a parameter value perturbation on the model behaviour. Forward sensitivity analysis is a local method [11] which can be performed by introducing the following initial value problem with n states and m parameters $\boldsymbol{p}$:

$$\dot{\boldsymbol{x}} = \mathrm{f}(t, \boldsymbol{x}, \boldsymbol{p}), \quad y = \boldsymbol{x}(2) \tag{8}$$

and the augmented system with sensitivity equations is given as:

$$\dot{\boldsymbol{w}} = \begin{pmatrix} \mathrm{f}(t, \boldsymbol{x}, \boldsymbol{p}) \\ \boldsymbol{P}_i' \end{pmatrix}, \qquad \boldsymbol{w} = \begin{pmatrix} \boldsymbol{x} \\ \boldsymbol{P}_i \end{pmatrix}, \\ \boldsymbol{P}_i' = \frac{\partial \mathrm{f}}{\partial \boldsymbol{x}} \boldsymbol{P}_i + \frac{\partial \mathrm{f}}{\partial \boldsymbol{p}}, \boldsymbol{P}(0) = 0 \tag{9}$$

where $\boldsymbol{P} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{p}}$ is an $n \times m$ matrix function and $\boldsymbol{P}_i$ points to column i in matrix $\boldsymbol{P}$ which corresponds to one parameter in $\boldsymbol{p}$.

In this paper, FSA is to investigate the engine-block behaviour for small changes of the parameters near estimated values found by NLS.

3 Results and Discussion

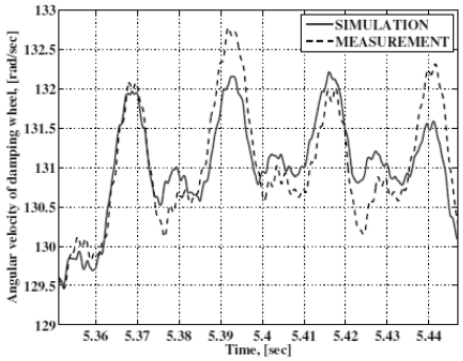
3.1 Estimation results using numerical solver

The `lsoptim` is a package, which is written in-MATLAB by Eriksson [12], for solving non-linear unconstrained least squares optimization problems. The solver uses the Levenberg Marquardt method, as the optimization numerical algorithm. It is second order and thus has good local convergence properties. The iterative optimization, which is applied in this solver, is based on three stages:

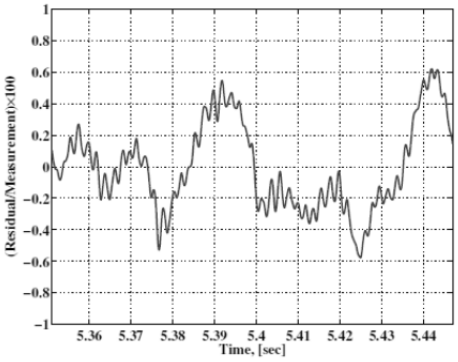
- Start: providing an initial guess for parameters which are to be estimated.
- Iteration: simulation of the ODE in (5) to find the damping wheel angular velocity, which is one of the states in the state vector, $\boldsymbol{x}$. Later, the residual, $\varepsilon_i(\boldsymbol{p})$, can be determined by subtracting the simulated value of this state from the measured data. The residual is used in each iteration to define the direction for updating the set of parameters.
- Stop: The search algorithm will be stopped by a criterion based on the difference between the values of the objective function in two following iterations. In other words, the final iteration will not improve the objective function more than a certain degree. It is worth to mention that lsoptim only converges to a local minimum (which might be the global minimum as well).

In Figure 2(a), the simulated and the measured damping wheel angular velocities, are compared by applying the estimated parameters provided by the numerical solver.

High resolution measurements, for a 4-cylinder car, with angular resolution of 0.5 degree, are used. It is seen the model is not only able to capture the main frequency from the engine and follow the trend, but also, it resembles the measurement data at higher frequencies. The ratio of the residual value, $\varepsilon_i(\boldsymbol{p})$, over the real data, in each time sample i, is shown in Figure 2(b). It provides how far the model result is from the reality, which shows the maximum value of ~ 0.6%, when the engine mean angular velocity is approximately 130 rad/sec.



(a): Damping wheel angular velocity, measurement vs. simulation.

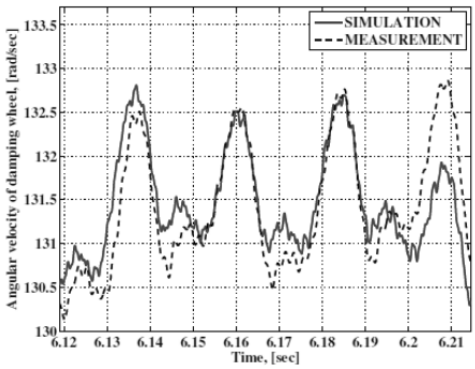


(b): The ratio of the residual value $\varepsilon_i(\boldsymbol{p})$ over the real data, in each time sample i.

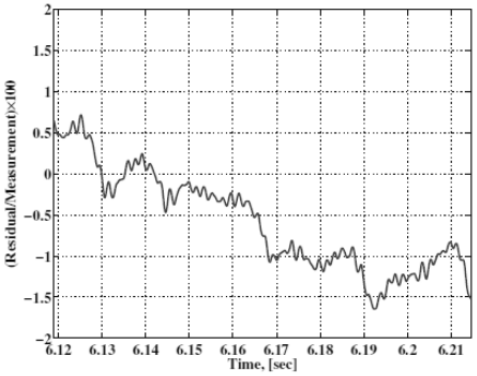
Figure 2: Comparison between simulation and empirical data for *estimation cycle*.

3.2 Model validation

According to the Introduction, the second step in the procedure of the estimation, is model validation, in which the model output is compared with validation data. In other words, for the engine model, the damping wheel angular velocity observations, at validation cycle, is plotted against the simulated results. Therefore, it is determined whether the estimated model accurately captures the system dynamics or not.



(a): Damping wheel angular velocity, measurement vs. simulation.



(b): The ratio of the residual value $\varepsilon_i(\boldsymbol{p})$ over the real data, in each time sample i.

Figure 3: Comparison between simulation and empirical data for *validatioin cycle*.

Figure 3(a) shows the measured versus simulated data plot for validation data, and Figure 3(b) shows the ratio of the residual value over the real data, in each time sample. It is seen that there exists a good agreement between the validation data and the output from the model, which is a good evaluation for the estimated model.

3.3 Results of sensitivity analysis

Here, the effects of the 10 parameters perturbations, mentioned in Section 1.2, are considered. The primary system equations, given in (8), were written in Mathematica and then the sensitivity equations were computed. The final ODE system (9) has been solved in MATLAB.

Figure 4 shows the sensitivities of the damping wheel angular velocity, (rad/sec), to the three friction coefficients, $C_1 - C_3$, and to the three damping coefficients, C_{21}, C_{32}, and C_{xx}.

Looking at the values of all the plots on the vertical axis (rad/sec), it is understood that the sensitivities to friction coefficients are in the same level and significantly higher than the sensitivities to the damping coefficients. Possible interpretation is that, friction coefficients are important to follow the trend and changing them has a high influence on increasing or decreasing the engine velocity (rad/sec). However, from the basic vibration knowledge, it is known that the damping coefficients only control the amount of oscillations and have nothing to do with the mean value of the angular velocity. There also exist interpretations for two different shapes of the sensitivities, described as follows:

- The reducing shape for the sensitivities of the engine angular velocity, (rad/sec), to the three friction coefficients, $C_1 - C_3$, is the result of friction growth. In other words, by increasing friction coefficient, the system is slowed down, and the slope, $\dot{\omega}$, can be obtained by the relation $\Delta C\omega \approx -(\sum J)\dot{\omega}$, in which ω is the current angular velocity of the system and $\sum J$ is the sum of the inertias in the model. The reduction rate, which is found from this relation, is the same as the one seen in the sensitivity plot. The slope for C_2 is four times of the other coefficients, namely, C_1 and C_3. The reason is that C_2 is repeated for the four cylinders. Therefore, by perturbation of C_2, all the four cylinders frictions will be perturbed.
- The damping coefficients will not alter the mean value of the output, and furthermore, it is seen from the plots that after a while, the amplitude of the oscillations becomes lower since the damping coefficients are perturbed.

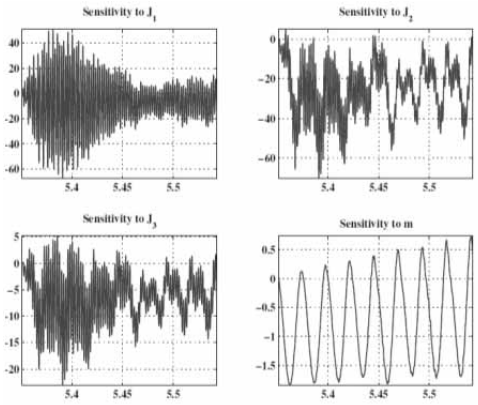


Figure 4: Sensitivities of the damping wheel angular velocity to 3 friction coefficients, $C_1 - C_3$, and 3 damping coefficients, C_{21}, C_{32}, and C_{xx}.

Figure 5 contains the sensitivities of the damping wheel angular velocity, (rad/sec), to the three inertias, i.e., damping wheel J_1, each cylinder J_2, flywheel J_3, and to the piston reciprocating mass, m. The inertias are important for catching the amplitudes of the main frequencies in the system. This is exactly what is seen from the shape of the sensitivities to $J_1 - J_3$. The oscillatory motion of the reciprocating mass of the piston adds an oscillatory torque to the pressure torque. This causes oscillations in the angular velocity of the engine as well, which is also presented in the sensitivity plot.

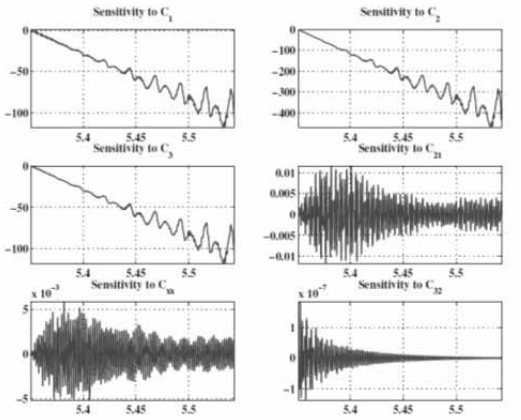


Figure 5: Sensitivities of the engine angular velocity to 3 inertias, i.e., damping wheel J_1, each cylinder J_2, flywheel J_3, and to the piston reciprocating mass m.

4 Conclusion

A powertrain model, suitable for studying torsional vibrations, was presented. The model is in modular form, and thus flexible enough, to be adapted for various powertrains structures, i.e., front-wheel, rear-wheel, and four-wheel drive systems with different number of cylinders. In order to get better parameter estimation performance, for the engine-block, the idling situation was considered and the clutch was disengaged. Therefore, the number of parameters to be estimated decreased to 10. The results from the model for damping wheel angular velocity, using the estimated parameters, validated versus the validation measured data, which showed significant similarity. This proves the ability of the model to resemble the main behaviours of the real system.

Furthermore, the performed sensitivity analysis was helpful to understand the importance of different parameters in the model, besides how the model responses to perturbation of these parameters.

Acknowledgement. Financial support for this research was provided jointly by CADICS, a Linnaeus research environment for Control, Autonomy, and Decision-making in Complex Systems, and Linköping University, Sweden.

This contribution is a post-conference publication from SIMS 2013 Conference (54th SIMS Conference, Bergen University College, Norway, October 16-18, 2013). The contribution was originally published in the Proceedings of SIMS 2013, to be found http://www.scansims. org/ sims2013/SIMS2013.pdf.

References

[1] Rahnejat H. *Multi-body dynamics vehicle, machines, and mechanisms*. Published by Society of Automotive Engineering, Inc.; 1998

[2] Eriksson D, Eriksson L, Frisk E, Krysander M. Flywheel angular velocity model for misfire simulation. In: *7th IFAC international symposium on advances in automotive control (AAC).* 2013*;* Tokyo, Japan

[3] Johansson R. *Modeling of the engine and driveline related disturbances on the wheel speed in passenger cars* [Master Thesis]. [Department of Electrical Engenieering, (SE)]. Linköping University; 2012

[4] Wu J-S Chen C-H. Torsional vibration analysis of gear-branched systems by finite element method. *Journal of sound and vibration.* 2001; 240(1): 159-182. doi: 10.1006/jsvi.2000.3197

[5] Rabeih E. *Torsional vibration analysis of automotive driveline* [Dissertation]. [Mechanical Engineering Department, (UK)]. University of Leeds; 1997

[6] Crowther A. Powertrain vibration: Modeling, simulation and testing. *FISITA world automotive congress*; 2004; Barcelona

[7] Couderc P, Callenaere J, Der Hagopian J, Ferraris G. Vehicle driveline dynamic behaviour: experiment and simulation. *Journal of sound and vibration.* 1998*;* 218(1): 133-157. doi: 10.1006/jsvi.1998.1808

[8] Ljung L. *System identification: Theory for the user*, 2nd ed. Prentice Hall Information and System Sciences Series; 1999; 605 p.

[9] Schön, T. *Estimation of nonlinear dynamic systems- theory and applications*. [Dissertation]. [Department of Electrical Engineering, (SE)]. Linköping Universit; 2006

[10] Heij C, Boer P, Franses P, Kloek T, Dijk H. (2004). Non-Linear Methods. In: *Econometric methods with applications in business and economics*. Oxford University Press; 2004, 814 p.

[11] Cacuci D. *Sensitivity and uncertainty analysis.* Chapman & Hall/CRC; 2003

[12] Eriksson L. *Minimal manual to lsoptim*. Linköping University, Sweden; 2004

State Estimation in a CO_2 Capture Plant

Samoja A. Jayarathna[1*], Bernt Lie[1], Morton C. Melaaen[1,2]

[1]Telemark University College, Kjølnesring 56, P.O. Box 203, N-3901, Porsgrunn, Norway; *morten.c.melaaen@hit.no
[2]Tel-Tek, Kjølnes ring 30, N-3918, Porsgrunn, Norway

Simulation Notes Europe SNE 24(3-4), 2014, 161 - 166
DOI: 10.11128/sne.24.tn.10262
Received: Jan.10, 2014 (Selected SIMS 2013 Postconf. Publ.);
Accepted: June 10, 2014;

Abstract. Post combustion CO_2 capturing holds an important position in the area of carbon capture and sequestration (CCS). Research in this area range from experimental work to modeling work. Dynamic models are interesting since these describe the plant operation during variations, up-stream or down-stream, and due to their use-fulness in control design. To take full advantage of state space models in control design, it is necessary to have on-line knowledge of all states, also states that are not measured directly. Techniques for state estimation, such as Kalman filter based methods, thus form key technology for advancing control solutions. But state estimation is also of interest in its own right for making available on-line knowledge of states. In this study, a dynamic model of an amine based CO_2 capture plant is used as a basis for a state estimator. A high order version of the model is used to represent the "real" plant. A reduced order model of the plant is then used for state estimation, and the Ensemble Kalman filter is used.

Introduction

Power generation via fossil fuel-fired power plants is known to be the largest single source of CO_2 emission in the world [1]. The development of capture technologies targeting such sources therefore is important for achieving the goals in CO_2 emission reduction. Post-combustion capture, pre-combustion capture and oxy-fuel combustion are the three main technologies available at present [2], and much research is done with the prospect of developing those techniques further.

Post combustion capture is still the best known technique, possibly due to the large number of existing power plants, and the promising developments that are available. CO_2 capture by amine absorption and stripping is currently considered to be the most feasible option for the removal of carbon dioxide from the power plants' exhaust gases [3].

Modelling work related to CO_2 capture technologies plays an important role with respect to the design, control and optimization of the capture process. Steady state models are important for design and optimization purposes, and dynamic simulation models are important for control applications. Several dynamic models for simulating the amine based CO_2 absorption plants are presented in literature [4]-[6].

A model consisting of a set of first order differential equations to represent the system is referred to as a state space model. To take full advantage of state space models in control design, it is necessary to have on-line knowledge of all states, also states that are not measured directly. Techniques for state estimation, such as Kalman filter based methods, thus form a key technology for advancing control solutions. But state estimation is also of interest in its own right for making available on-line knowledge of states.

The Kalman filter based methods vary from basic Kalman filter (**KF**) to its extensions and generalizations such as the *Extended Kalman Filter* and *Unscented Kalman Filter*. The basic Kalman filter is applicable for linear dynamic systems, while the extensions and generalizations of the method are there to be used with the nonlinear dynamic systems [7]. The *Ensemble Kalman Filter* (**EnKF**) is another alternative to the traditional Kalman filter for better handling of nonlinear models with large number of states [8].

Use of traditional KF methods for models with high-dimensional state vectors is computationally difficult as an error covariance matrix for the model states needs to be stored and propagated in time. The Extended Kalman filter uses a linearized equation for the error covariance evolution when the model dynamics are nonlinear. This linearization can result in unbounded linear instabilities for the error evolution [8]. These two problems can be solved to a great extent by using the Ensemble Kalman filter.

In this study, a dynamic model of an amine based CO_2 capture plant is used as a basis for a state estimator. The model as developed and published by Jayarathna et al. ([6], [9] - [12]) is used in this study as the plant model. A high order version of the model is used to represent the 'real system' due to the absence of real plant data. A reduced order model of the plant is then used for state estimation, and the Ensemble Kalman filter is used.

1 Theory

The EnKF algorithm is presented in details with the derivation by Evensen [13]. Initialization of the estimator is done by providing values for the initial ensemble. When the number of simulations in an ensemble is N, the values of the initial ensemble are given according to the eq. 1.

$$x_{0|0} \sim \mathcal{N}(\bar{x}_0, P_0) \tag{1}$$

Having the initial ensembles available, the state estimator runs trough a propagation step and a measurement update step at each time step. The propagation step consists of three consecutive steps, the ensemble propagation, the estimated state-output propagation and the covariance calculation. For each simulation $i\,(i = 1, 2, \dots, N)$, the ensemble propagation step is given by eqs. 2 and 3.

$$x^i_{k|k-1} = f_{k-1}\left(x^i_{k-1|k-1}, u_{k-1}, w^i_{k-1}\right) \tag{2}$$

$$y^i_{k|k-1} = h_{k-1}\left(x^i_{k|k-1}, v^i_{k-1}\right) \tag{3}$$

Here w^i_{k-1} and v^i_{k-1} are the model disturbances $(w \sim \mathcal{N}(w_0, W))$ and the measurement noise $(v \sim \mathcal{N}(0, V))$.

The propagation of the estimated state and output is given by eqs. 4 and 5.

$$\hat{x}_{k|k-1} = \frac{\sum_{i=1}^{N} x^i_{k|k-1}}{N} \tag{4}$$

$$\hat{y}_{k|k-1} = \frac{\sum_{i=1}^{N} y^i_{k|k-1}}{N} \tag{5}$$

The covariance calculation is done according to eqs. 6 and 7.

$$e^i_{x,k|k-1} = \left(x^i_{k|k-1} - \hat{x}_{k|k-1}\right) \tag{6}$$

$$P_{k|k-1} = \frac{\sum_{i=1}^{N}\left(e^i_{x,k|k-1}\right)\left(e^i_{x,k|k-1}\right)^T}{N-1} \tag{7}$$

After completing the propagation step, the state estimator updates the predictions using the available measurements. The measurement update step consists of two consecutive stages in the used algorithm, Kalman gain calculation and state-out-covariance update. The Kalman gain calculations is performed according to the eqs. 8-11.

$$e^i_{y,k|k-1} = \left(y^i_{k|k-1} - \hat{y}_{k|k-1}\right) \tag{8}$$

$$P_y = \frac{\sum_{i=1}^{N}\left(e^i_{y,k|k-1}\right)\left(e^i_{y,k|k-1}\right)^T}{N-1} \tag{9}$$

$$P_{xy} = \frac{\sum_{i=1}^{N}\left(e^i_{x,k|k-1}\right)\left(e^i_{y,k|k-1}\right)^T}{N-1} \tag{10}$$

$$K_k = P_{xy} P_y^{-1} \tag{11}$$

The state-out-covariance update step is given by eqs. 12-14.

$$x^i_{k|k} = x^i_{k|k-1} + K_k\left(\left(y_k + v^i_k\right) - y^i_{k|k-1}\right) \tag{12}$$

$$\hat{x}_{k|k} = \frac{\sum_{i=1}^{N} x^i_{k|k}}{N} \tag{13}$$

$$P_{k|k} = P_{k|k-1} - K_k P_y K_k^T \tag{14}$$

Here $y_k = g(x_k)$.

Updated values of the states $(x^i_{k|k})$ are then taken as the initial values to run the estimator for the next time step. The averaged values $(\hat{x}^i_{k|k})$ are the estimates for the time step. The covariance matrix $(P_{k|k})$ can be used to get an idea about the uncertainty of the predictions.

2 Implementation

In the implementation of the state estimator, the number of ensembles is taken to be 60 ($N = 60$). The Parallel Computing Toolbox in MATLAB is used to work with 12 threads at the same time. This way, 12 simulations are performed simultaneously to complete each ensemble, and the time required for completing the simulations for an ensemble is reduced by five times. Initial state values (x_0) are taken from a simulation with sufficiently long simulation time, thus steady state is assumed. The covariance matrix of the initial states (P_0) is given according to the magnitude of the x_0 values (as a rule of thumb, the values of the diagonal matrix (P_0) are taken as a fraction of the initial state values).

A higher order model, i.e. with a higher number of control volumes than in the state estimator, is used to represent the "real" system due to the absence of real plant data. According to the assumptions made during the model development an infinite number of CVs should be used to represent the columns in the real system. But a finite number of CVs has to be used in practice.

Use of a finite number of control volumes introduces diffusion into the column models. Diffusion is a phenomenon that occur in the absorption and stripping columns. Therefore, it is acceptable to have a finite number of control volumes in the model that represents the real plant.

Several compositions and temperatures are taken as the measurements; we assume a total of 26 measurements. Measured compositions are the composition of the cleaned gas and the gas leaving the stripping column (can be measured by gas chromatography). The amount of dissolved CO_2 (total CO_2) and MEA (total MEA) in the solvent streams leaving the absorption tower, stripping tower and the buffer tank are also taken as measurements.

Mass flow rate of the amine solutions leaving the absorber, stripper and the buffer tank without the mass flow rate of the dissolved CO_2 are also included in the measurements. Temperatures of the liquid and vapor leaving the absorption tower, temperatures of the liquid and vapor streams leaving the stripping column, temperatures of the liquid streams leaving the heat exchangers and the temperature of the buffer tank are taken as the temperature measurements. Measurement noise is assumed to be white noise.

3 KF Predictions

An analysis related to the number of control volumes used in the tower models, given in Figure 1, showed that the execution time increases quadratically with the increasing number of control volumes. Therefore the number of control volumes to be used in the state estimator is chosen to be around 50 control volumes.

When the sensitivity of the model predictions to the number of control volumes used in the tower discretization is considered, from Figure 1 it is noticeable that the model predictions improves with the increasing number of control volumes up to about 100 CVs in towers. The predicted values remains with very little variations for higher number of control volumes than 100, but as can be seen from Figure 1 the execution time for the simulations increases very much for higher number of CVs than 100.

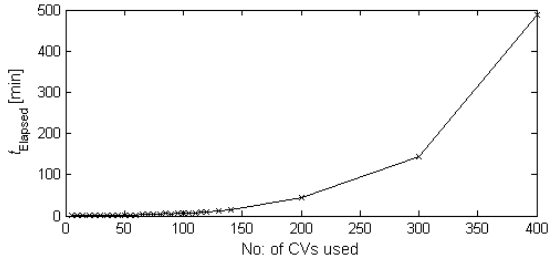


Figure 1: Execution time of the simulations with the number of control volumes used in the tower descretizations.

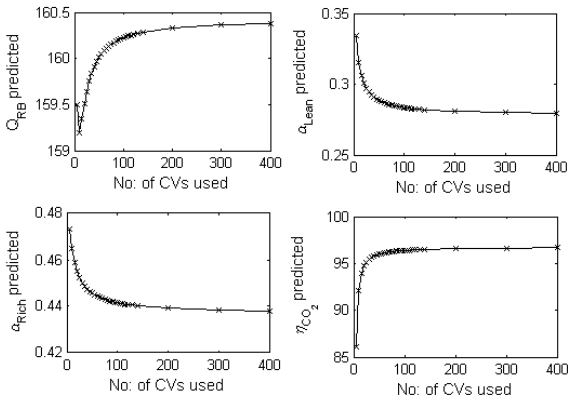


Figure 2: Dependancy of the model predictions on the number of control volumes in the tower descretizations. Q_{RB} predicted is the predicted re-boiler duty, α_{Lean} predicted is the predicted lean loading value, α_{Rich} predicted is the predicted rich loading value and η_{CO_2} predicted is the predicted CO_2 removal efficiency.

Therefore, 100 CVs is chosen as a rough limit for the maximum number of CVs that are used in the towers for the model that is used for representing the real system. The dependency of the model predictions with the number of control volumes used is shown in Figure 2.

Several different cases are studied to analyze the estimates of the EnKF to different scale of model errors. The model error was increased by increasing the difference between the number of control volumes used in the model that represents the real plant and the model used in the state estimator. One hour of the plant operation is simulated. Information about the cases performed for analyzing the sensitivity of the state estimator for the model error are given in the Table 1. Selected states are presented in Figures 3 - 8 for the four cases of Table 1 to analyse the quality of the estimates. Figures 3 - 5 show three measured states and Figures 6 - 8 show three unmeasured states.

Case no:	CVs in Model	CVs in Real System	Ratio	Computation time
1	15	15	1	2 h
2	50	50	1	20 h
3	40	50	1.25	6 h
4	40	100	2.5	6 h

Table 1: Details of the cases used for analyzing the sensitivity to the model error.

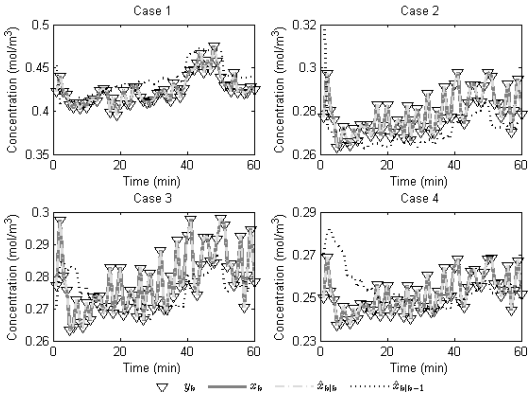


Figure 3: CO_2 concentration in the cleaned gas leaving the absorption tower. y_k: measurement, x_k: real state, $\hat{x}_{k|k}$: a posteriori state, $\hat{x}_{k|k-1}$: a priori state.

Prediction of CO_2 concentration in the cleaned gas leaving the absorption tower (Figure 3), which is a measured state, appear to be a good match with the real state in all four cases. Further, it can be seen that the measurement noise of this state is very small, and that should be the reason for this high quality prediction in all the cases. The other two measured states presented, the temperature of the rich amine steam leaving the absorber (Figure 4) and the temperature of the rich amine leaving the cross heat exchanger (Figure 5), are predicted with larger errors. The pattern of the predictions appears to follow the pattern of the real states, and the error of the prediction appears to be increasing with the model error.

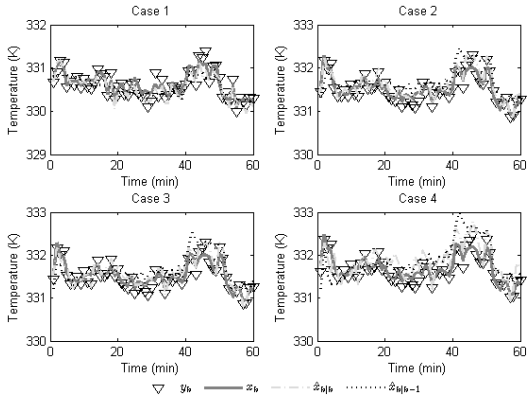


Figure 4: Temperature of the amine streams leaving theabsorption column. y_k: measurement, x_k: real state, $\hat{x}_{k|k}$: a posteriori state, $\hat{x}_{k|k-1}$: a priori state.

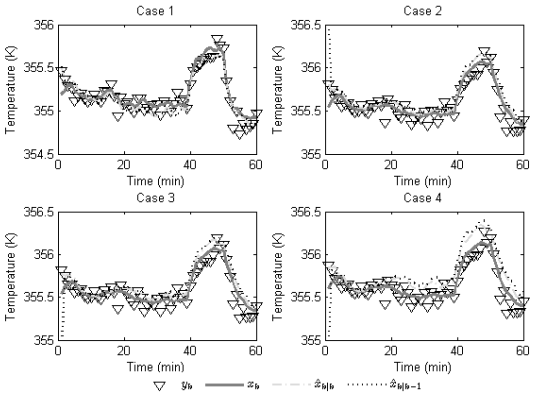


Figure 5: Temperature of the rich amine stream leaving the cross heat exchanger. y_k: measurement, x_k: real state, $\hat{x}_{k|k}$: a posteriori state, $\hat{x}_{k|k-1}$: a priori state.

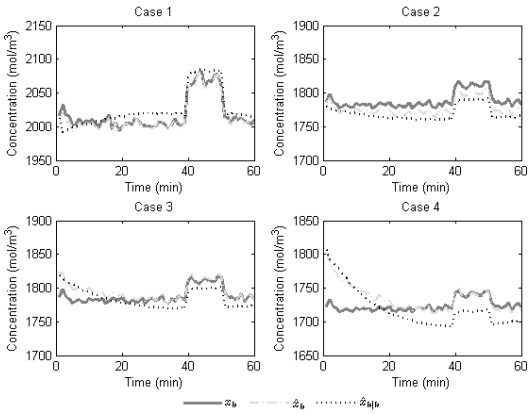


Figure 6: Total CO_2 concentration in the amine stream at 1/3 of the packing height of the absorption tower. x_k: real state, $\hat{x}_{k|k}$: a posteriori state, $\hat{x}_{k|k-1}$: a priori state.

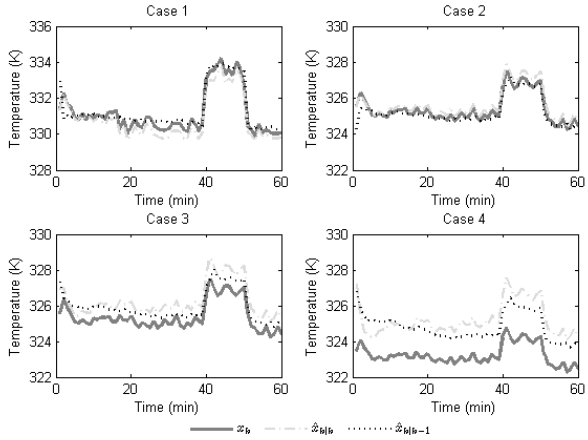


Figure 7: Liquid phase temperature at 1/3rd of the absorber packing height. x_k: real state, $\hat{x}_{k|k}$: a posteriori state, $\hat{x}_{k|k-1}$: a priori state.

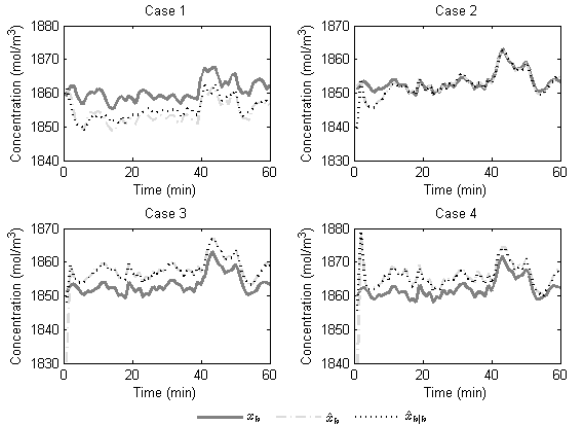


Figure 8: Total CO_2 concentration at 1/3rd of packing height of the stripping tower. x_k: real state, $\hat{x}_{k|k}$: a posteriori state, $\hat{x}_{k|k-1}$: a priori state.

Estimates of the states that are not measured are also in good accordance with the pattern of the real states (Figures 6 - 8). Similar to the observation with the measured states, it can be seen that the error in the estimated values are increasing with the increase in the model error.

When the computation time needed for each of the cases is considered, it can be seen that there will always be a time delay before the estimates are available.

4 Conclusions and Recommendations

A state estimator for making on-line predictions of the states of a CO_2 capture plant is developed. The Ensemble Kalman Filter algorithm is used in the state estimator, due to the suitability of the algorithm to handle nonlinear models with a large number of states. A simple model (with small number of control volumes) is used in the estimator while a high order model (with large number of control volumes) is used to represent the real system. Absence of real plant data is the reason for using a high order model as the real system. According to the assumptions of the model development, an infinite number of control volumes should be used in the tower discretization to represent the real system. In practice a finite number of control volumes is used, thought. Tower discretization into finite number of control volumes introduces diffusion into the system, which is also there in reality. Therefore, use of a finite number of control volumes in tower discretization is justified.

Sensitivity of the estimates to the model error is analyzed by performing simulations with increasing model error (by making the difference between the model and the system to increase). Some predictions seem very good and some predictions seems to be poorer. The observation from the results of the sensitivity analysis, in general, is that the deviation of the estimates from the real states increases with the increasing model error.

According to the simulation time needed for each of the cases considered in the sensitivity analysis, it can be seen that the estimates are available always with a time delay. This high computational time of the estimator is a problem for making available the timely estimates of the unmeasured states.

The use of iterations with the flash calculations in the model can be a reason for the computation time of the estimator. Making table look-up values available to be used instead of performing flash calculations, will be an option to speed up the estimator.

When the tendency to increase the error in the estimates with the increase of model error and the time delay of the estimates are considered, it is concluded that some restructuring of the model is required before the estimator is ready for on-line use.

Acknowledgements

This contribution is a post-conference publication from SIMS 2013 Conference (54th SIMS Conference, Bergen University College, Norway, October 16-18, 2013). The contribution was originally published in the Proceedings of SIMS 2013, to be found http://www. scansims.org/sims2013/SIMS2013.pdf.

References

[1] Freund P. Making deep reductions in CO_2 emissions from coal-fired power plant using capture and storage of CO_2. In: Proceedings of the Institution of Mechanical Engineers Part A. *J. Power Energy.* 2003; 217 (1): 1-7. doi: 10.1243/095765003321148637

[2] Cormos AM, Gáspár J, Padurean A, Cormos CC. Techno-economical analysis of carbon dioxide absorption in mono-ethanolamine by mathematical modelling and simulation, In: Pierucci S, Buzzi Ferraris G, editors. ESCAPE20. Proceedings of 20th European Symposium on Computer Aided Process Engineering; 2010; Ischia, Italy. 81-1 - 81-6.

[3] Plaza JM, Wagener DV, Rochelle GT. Modelling CO_2 capture with aqueous monoethanolamine. *Energy Procedia*. 2009; 1(1): 1171-1178. doi: 10.1016/j.egypro.2009.01.154

[4] Biliyok C, Lawal A, Wang M, Seibert F. Dynamic modelling, validation and analysis of post-combustion chemical absorption CO_2 capture plant. *International Journal of Greenhouse Gas Control*. 2012 July; 9: 428-445. doi: 10.1016/j.ijggc.2012.05.001

[5] Lawal A, Wang M, Stephenson P, Koumpouras G, Yeung H. Dynamic modelling and analysis of post-combustion CO_2 chemical absorption process for coal-fired power plants. *Fuel*. 2010 October; 89(10): 2791-2801. doi: 10.1016/j.fuel.2010.05.030

[6] Jayarathna SA, Lie B, Melaaen MC. Amine based CO_2 capture plant: Dynamic modeling and simulations. *International Journal of Greenhouse Gas Control*. 2013 May; 14: 282 - 290. doi: 10.1016/j.ijggc.2013.01.028

[7] Simon D. *Optimal state estimation: Kalman, H Infinity, and Nonlinear Approaches*. Hoboken: John Wiley & Sons, USA, 2006. 552 p.

[8] Evensen G. The ensemble Kalman Filter for combined state and parameter estimation. *Control Systems, IEEE*. 2009 June; 29(3): 83 - 104. doi: 10.1109/MCS.2009.932223

[9] Jayarathna SA, Lie B, Melaaen MC. NEQ rate based modeling of an absorption column for post combustion CO2 capturing. *Energy Procedia*. 2011; 4: 1797 - 1804. doi: 10.1016/j.egypro.2011.02.056

[10] Jayarathna SA, Lie B, Melaaen MC. Dynamic modelling of the absorber of a post-combustion CO_2 capture plant: Modelling and simulations. *Computers and Chemical Engineering*. 2013 June; 53: 178 - 189. doi: 10.1016/j.compchemeng.2013.03.002

[11] Jayarathna SA, Weerasooriya A, Lie B, Melaaen MC. Dynamic Operations of the Stripping Column of a CO_2 Capture Plant. In*: SIMS53. Proceedings of 53rd SIMS conference on Simulation and Modelling*; 2012 October; Reykjavik.

[12] Jayarathna SA, Lie B, Melaaen MC. Development of a dynamic model of a post combustion CO_2 capture process. In: *GHGT11. Proceedings of 11th International Conference on Greenhouse Gas Technologies.* 2012 Novmeber; Kyoto, Japan.

[13] Evensen G. *Data Assimilation*, 2nd edition. USA: Springer; 2009. 307 p.

A Numerical Simulation of a Boiling Front Moving through Porous Medium

Larus Thorvaldsson*, Halldor Palsson

University of Iceland, Sæmundargötu 2, 101 Reykjavík; *lth31@hi.is

Simulation Notes Europe SNE 24(3-4), 2014, 167 - 172
DOI: 10.11128/sne.24.tn.102 63
Received: Jan.10, 2014 (Selected SIMS 2013 Postconf. Publ.);
Accepted: June 15, 2014;

Abstract. A boiling front of water moving through porous medium is simulated using a numerical finite volume method (FVM) which is then compared to an analytical solution. The FVM is constructed on top of the OpenFOAM framework, which is a highly customizable set of C++ libraries and tools for the solution of problems in continuum mechanics. The solution is stabilized using Fréchet derivatives which enables the use of appropriately sized time steps. The physical properties of the water are determined from the IAPWS-IF97 thermodynamic formulation. The results show that the numerical model is able to simulate the overall behavior of the boiling front sufficiently well. This increases the confidence in the numerical solutions outside the limits of the analytical solution.

Introduction

Numerical simulation of hydrothermal systems has played an important role in their modeling for the past decades. For researchers it has been used to test competing hypothesis in these complex environments, where data is often scarce, and in industry numerical simulation has become standard practice in the planning and management of the development of geothermal fields [9].

The earliest efforts to apply numerical models to geothermal reservoirs were made in the early 1970's, but the usefulness of numerical modelling did not begin to gain acceptance by the geothermal industry until after the 1980 Code Comparison Study [14]. Since that study was performed, the experiences gained in carrying out site-specific studies as well as generic reservoir modelling studies have led to a constant improvement in the capabilities of numerical reservoir models.

Numerical modeling of hydrothermal systems is often defined by which components of the system are taken into account. Traditionally they have been divided into hydrological (H), thermal (T), mechanical (M) and chemical (C) components. Those components are coupled together in a way that is inherently multiscale in nature, such that their temporal and spatial scales vary be several orders of magnitude [4]. Because of the complex nature of those couplings, models involving all four components are rare.

The equations that describe hydrothermal systems are relatively complex but they can nevertheless be solved analytically for a highly idealized set of initial and boundary conditions. Such cases usually only involve one of the four (HTMC) components, where the Theis problem is an example thereof [15]. Some analytical solutions also exists for two components, such as the description of a boiling front moving through a porous medium [13] and the advance of a diffused salt water wedge in a confined aquifer [2]. These analytical solutions are very important in validating numerical models that are supposed to handle more complicated problems.

The current generation of numerical simulators is in most cases able to account for multiphase, multicomponent flow. The most versatile ones are software packages such as Finite Element Heat and Mass Transfer (FEHM) [6] and the Transport of Unsaturated Groundwater. and Heat (TOUGH) family of codes [12]. These solvers have been applied to a wide variety of problems, such as CO2 sequestration, geothermal studies and other environmental issues [4].

Other solvers are more specialized, such as the Complex Systems Modeling Platform (CSMP++) [8] and Fully Implicit Seafloor Hydrothermal Event Simulator (FISHES) [7]. They have been developed specifically to allow simulation of high-temperature multiphase flow of NaCl-H2O fluids. Other codes such as FALCON [10], developed at Idaho National Labs have

focused on the tightly coupled process of fluid rock interaction.

In this paper the applicability of using a free and open source package named OpenFOAM for modeling hydrothermal systems is examined. Open- FOAM is a highly customizable set of C++ libraries and tools for the solution of problems in continuum mechanics. It is also gaining considerable popularity in academic research and among industrial users, both as a research platform and a black-box CFD and structural analysis solver [5].

The object orientation and operator overloading of C++ has enabled the developers of OpenFOAM to build a framework for computational fluid dynamics that enables modelers to work at a very high level of abstraction [17]. This makes it possible to manipulate the set of partialdifferential equations that describe the problem and customize the solver itself for each class of cases that need to be solved. This will enable researchers with sufficient knowledge about the relevant dynamics of each problem to construct efficient and accurate solvers for it. This is the main motivation for using OpenFOAM, rather than currently existing models.

As well as showing how easily an equation of state can be implemented in the previously existing framework, this paper also shows how the underlying equations can be modified in an easy way. This is demonstrated by linearizing the partial differential equations describing the hydrology of the problem using Fréchet derivatives. In this manner it is possible to stabilize the solution for superheated steam, which makes it possible to treat the pressure equation as steady state, allowing for a more computationally efficient solution.

1 Methods and Materials

1.1 Governing equations

For mass conservation the continuity equation must be satisfied

$$\frac{\partial \phi \rho}{\partial t} + \nabla \cdot (\phi \vec{u}) = 0 \tag{1}$$

where ϕ is porosity and ρ is density. In this equation $\vec{u}$ denotes the superficial velocity, which is a hypothetical velocity calculated as if the fluid were the only thin present in a given cross sectional area. For pressure-velocity coupling, Darcy's law can be applied

$$\vec{u} = -\frac{\kappa}{\mu}(\nabla p - \rho \vec{g}) \tag{2}$$

where κ is permeability, μ is viscosity and $\vec{(g)}$ is gravitational acceleration.
This gives following equation for groundwater flow

$$\frac{\partial \phi \rho}{\partial t} - \nabla \cdot \left(\frac{\rho \kappa}{\mu} \nabla p \right) + \nabla \cdot \left(\frac{\rho^2 \kappa}{\mu} \vec{g} \right) = 0 \tag{3}$$

The energy equation includes both effects from the fluid and the soil, and is given as

$$\frac{\partial}{\partial t} (\phi \rho h + (1-\phi) \rho_s c_s T) + \nabla (\rho \vec{u} h) = \nabla (\Gamma \nabla T) \tag{4}$$

where ϕ is porosity, ρ_s is the density of rock, $c_s$is the heat capacity of soil and Γ is the combined conductivity of fluid and soil.If the properties of the soil are assumed to be constant the laplacian of temperature can be broken up in terms of enthalpy and pressure. In this case, conduction effects in the water are also neglected, which gives the following equation for energy

$$\begin{aligned} \phi \frac{\partial \rho h}{\partial t} + (1-\phi) \rho_s c_s \left(\frac{\partial T}{\partial h} \frac{\partial h}{\partial t} + \frac{\partial T}{\partial p} \frac{\partial p}{\partial t} \right) \\ + \nabla \cdot (\rho \vec{u} h) = 0 \end{aligned} \tag{5}$$

Since the density is a strong function of pressure in the liquid-vapor phase, it can be accounted for in the time derivative by using a first order Taylor expansion. If it is also assumed that the porosity is not a function of time, the Taylor expansion of the time derivative becomes

$$\frac{\partial \phi \rho}{\partial t} = \phi \frac{\partial \rho}{\partial t} + \phi \frac{\partial}{\partial t} \frac{\partial \rho}{\partial p} \bigg|_{p=p_0} (p - p_0) \tag{6}$$

The stability of the pressure equation can be increased by linearizing the Laplacian term. This is possible by applying a Fréchet derivative operator on the term, such that

$$\begin{aligned} \frac{\delta F}{\delta p} =& \frac{\delta}{\delta p} \nabla \cdot \left(\frac{\rho \kappa}{\mu} \nabla p \right) \\ =& \nabla \cdot \left(\frac{\kappa}{\mu} \frac{\partial \rho}{\partial p} \nabla p \right) + \nabla \left(\frac{\kappa}{\mu} \rho \nabla \right) \end{aligned} \tag{7}$$

this can then be applied to a first order Taylor expansion of the function

$$\begin{aligned} p =& p_0 + \frac{\delta F}{\delta p} \bigg|_{p=p0} (p - p_0) \\ =& \nabla \cdot \left(\frac{\kappa}{\mu} \rho \nabla p \right) + (p - p_0) \nabla \cdot \left(\frac{\kappa}{\mu} \frac{\partial \rho}{\partial p} \nabla p_0 \right) \end{aligned} \tag{8}$$

Having applied both the Taylor expansion of the time derivative and the linearization of the laplacian term, the pressure equation finally becomes

$$\phi\frac{\partial\rho}{\partial t}+\phi\frac{\partial}{\partial t}\frac{\partial\rho}{\partial p}\bigg|_{p=p_0}(p-p_0)-\nabla\cdot\left(\frac{\kappa}{\mu}\rho\nabla p\right)- \\ -(p-p_0)\nabla\cdot\left(\frac{\kappa}{\mu}\frac{\partial\rho}{\partial p}\nabla p_0\right)+\nabla\cdot\left(\frac{\rho^2\kappa}{mu}\vec{g}\right)=0 \tag{9}$$

The equation of state is implemented from the IAPWS-IF97 thermodynamic formulation [3]. The primary variables for the equation of state are taken to be pressure and enthalpy, since pressure-temperature formulation has been shown to have more difficulties close to the critical point [4]. Given those two state variables the algorithm returns the steam quality x, the density ρ, the temperature T and the partial derivatives of all those variables both with respect to pressure and enthalpy. Those values are then used in the system equation, which makes it possible to solve for each timestep.

1.2 Analytical solution

The problem which Pruess derived an analytical solution to [13] concerns cold water injection into a reservoir initially filled with superheated steam.If the reservoir is assumed to be infinite and the initial conditions to be uniform, this problem can be shown only to depend on the similarity variable $\eta = r^2/t$, where r is the radial distance from the well, and t is the time. Therefore the boiling front must occur at a fixed value $\eta = \eta_f$ and by extension, the front temperature and pressure must be time-independent.

Pruess gives the solution in terms of pressure at the front p_f and the boiling fraction $b = q_v f/q_l$, where q_l is the mass flow rate of liquid water into the vapor zone and $q_v f$ is the mass flow rate of vapor at the boiling front. For the vapor flow Pruess applies mass balance, Darcy's law and the real gas law, which gives the pressure at the front in the following terms [13]

$$p_f^2 = p_0^2 - \frac{ZR_sT_0\mu_v q_{vf}}{2\pi\kappa H}exp\left(\frac{r_f^2}{4\alpha t}\right)Ei\left(\frac{-r_f^2}{4\alpha t}\right) \tag{10}$$

where Ei is the exponential integral and H is the depth of the reservoir. The crompressibility factor Z is evaluated at the initial steam temperature T_0 and the front pressure p_f. The viscosity μ_v is also evaluated at T_0 but an average pressure between the initial and front pressure is used, such that

$$\bar{p} = \frac{1}{2}(p_f + p_0). \tag{11}$$

The diffusivity parameter α in equation 10 is given as

$$\alpha = \frac{\bar{p}\kappa}{\phi\mu_v}. \tag{12}$$

Using mass conservation it is possible to derive a relation for the location of the front in terms of the similarity variable $\eta_f = r_f^2/t$. Looking at the total liquid mass present in the injection plume gives the following

$$M_l(t) = (q_l - q_v f)t = \pi r_f^2 H\phi\bar{\rho}_l \tag{13}$$

$$\Rightarrow \eta_f = \frac{r_f^2}{t} = \frac{(1-b)q_l}{\pi H\phi\bar{\rho}_l} \tag{14}$$

where $\bar{\rho}_l$ is the average liquid density in the single phase liquid plume. Neglecting small pressure effects it is given as follows

$$\bar{\rho}_l = \omega\rho_l(T_{inj}) + (1-\omega)\rho_l(T_f) \tag{15}$$

where ω is the retardation factor which is the ratio of cold volume to total swept volume. The relation for the retardation factor is given by [1] as follows

$$\omega = \frac{\phi c_{l,h}\rho_l, h}{(1-\phi)\rho_s c_s + \phi c_{l,h}\rho_{l,h}} \tag{16}$$

There $c_{l,h}$ and $\rho_{l,h}$ are respectively the heat capacity and density of the liquid water that has been heated by the superheated steam. Pruess then applies heat balance at the front, which is derived from the fact that temperature and pressure at the front must be related by the vapor pressure relationship for water

$$p_f = p_{sat}(T_f) = p_{sat}\left(T_0 - \frac{h_{vl}\phi\bar{\rho}_l}{(1-\phi)\rho_s c_s(b^{-1}-1)})\right). \tag{17}$$

This set of non-linear equations can be solved in various ways, Pruess uses a Newton-Raphson method, while this paper applies a trust region dogleg method [11]. In order for the solution method to function robustly, the front pressure (pf) has to be scaled, such that it is on the same order of magnitude as the boiling fraction b. This is done by defining the non-dimensional pressure in the following way

$$p_{nd} = \frac{p_f - p_0}{p_{max} - p_0} \tag{18}$$

There p_{max} is an estimation of the maximum pressure in the reservoir.

The physical properties are retrieved from a C++ implementation of the IAPWS-IF97 thermodynamic formulation [IAPWS, 2007]. The primary variables are taken to be pressure and temperature which return saturation pressure $p_{sat}(T)$, density ρ, heat capacity c, viscosity μ_v and the compressibility factor Z.

1.3 Discretization and boundary conditions

The numerical solution is calculated on a axisymmetrical grid with logarithmically spaced cells, with the boundaries of the domain at r_0 and r_N. The logarithmic spacing is constructed by making an equally spaced vector which then gives a logarithmically spaced r_i vector

$$r_i = 10^{x_i}, i = 0, ..., N. \tag{19}$$

$$x_i = log_{10}(r_0) + \frac{i}{N}(log_{10}(r_N) - log_{10}(r_0)) \quad i = 0, ..., N \tag{20}$$

In the numerical solution, the injection well has to have

param	value
κ	$5 \cdot 10^{-14} m^2$
H	$200m$
r_0	$1m$
r_N	$2500m$
ρ_s	$2600kg/m^3$
c_s	$920J/kg/K$
Γ	$5W/m/K$
h_{vl}	$2260kJ/kg$
q_l	$27.8kg/s$
ϕ	8%
p_0	$600kPa$

Table 1: Numerical values for problem parameters.

some finite radius, r_0. If the injection rate is given as the mass flow rate q_l the appropriate Neumann pressure boundary condition can be found from Darcy's law, such that

$$r_i = 10^{x_i}, i = 0, ..., N. \tag{21}$$

$$r_i = 10^{x_i}, i = 0, ..., N. \tag{22}$$

$$r_i = 10^{x_i}, i = 0, ..., N. \tag{23}$$

$$r_i = 10^{x_i}, i = 0, ..., N. \tag{24}$$

In order to be able to compare the numerical solution to the analytical one, the outer radius is made sufficiently large, as to approximate boundary conditions at infinity. A Dirichlet boundary condition for pressure is applied there with $p = p_0$. Dirichlet boundary conditions are assumed at both sides for the enthalpy, where a temperature of $T_{inj} = 302.55K$ at the well gives $h_{inj} = 1.258 \cdot 10^2 kJ/kg$, At the boundary the temperature is $T_0 = 513.15K$ which corresponds to enthalpy of $h_0 = 2.937 \cdot 10^3 kJ/kg$.

2 Results

In order to validate the OpenFOAM model, a case where cold water was injected into a reservoir with superheated steam, was set up and then compared to the analytical solution. The parameters which were chosen for the problem are given in Table 1.

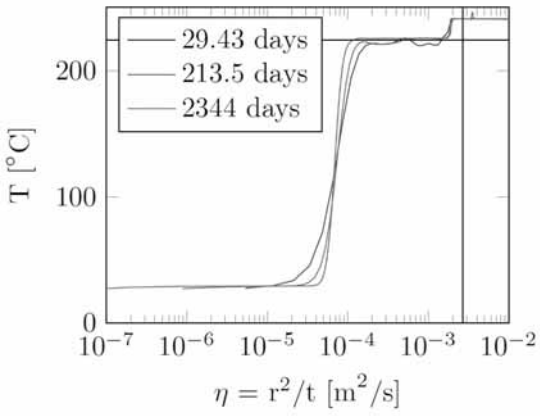


Figure 1: Temperature as a function of the similarity variable $\eta = r^2/t$, along with the analytical solution for the boiling front.

Figure 1 shows the temperature of the reservoir as a function of the similarity variable $\eta = r^2/t$. The figure shows the temperature profile after approximately 29.5 days, 213.5 days and 2344 days. The cross shows the location of the analytical solution as given by equations 10 and 17.

The numerical solution confirms that the location of the front in terms of the similarity variable $\eta_f = r_f^2/t$ is in fact constant. Even though that the time differs by a factor of 10, η_f stays approximately the same. Even though that conduction effects are not taken into account in the numerical solution, it shows some clear signs of numerical diffusion at the interface between hot

and cold single phase liquid. How ever it seems to get sharper as time progresses, probably due to the fact the the width of the interface stays constant in terms of r.

The analytical solution gives the location of the front at $\eta_f = 2.663 \cdot 10^{-3} m2/s$ with temperature of $T_f = 224.1°C$. When compared to the numerical solution at at time $t = 2344$ days the temperature of the front is approximately $T_f = 225.6°C$, which gives a $\Delta T_f = 1.5°C$. For the location of the front there is slightly more difference, where the numerical solution at the same time gives $\eta_f = 1.953 \cdot 10^{-3} m2/s$ which results in a difference of $\eta_f = 7.1 \cdot 10^{-4} m2/s$. This could be due to the fact that in the analytical case, the injection well is infinitely small and the reservoir infinitely large. Numerical constraints result in a finite value for both those quantities, which might result in some small inaccuracies when comparing them to the analytical solution.

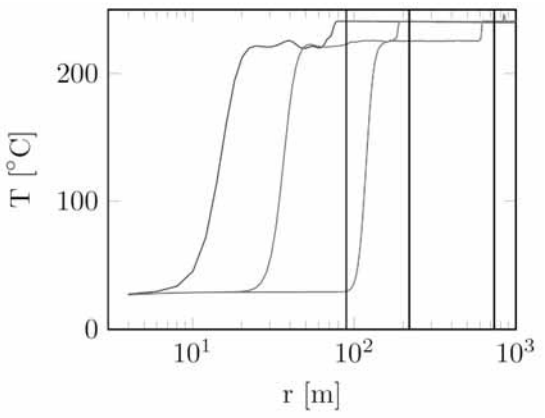


Figure 2: The temperature of the reservoir as a function of distance from the injection well. The temperature profile is given at three different times, after 29.43 days, 213.5 days and 2344 days.

Figure 2 gives a more physical representation of the solution by plotting it as a function of radial distance from the well. In that case it can clearly be seen how the front propagates in time. The vertical lines show the location of the analytical solution for each respective time and seem to show a rather good agreement between the analytical solution and the numerical one.

Figure 3 gives again a more physical representation of Figure 1, but now in terms of time. The phase change from superheated steam to liquid water occurs after 12, 23 and 58 days for the respective distances from the well. After that the hot water slowly cools down as the injection into the well is continued and finally reaches the injection temperature of $29.4°C$.

3 Discussion

This paper has described the applicability of the OpenFOAM framework to take on problems involving phase change in hydrothermal systems. In particular the results show that the numerical model is able to simulate the overall behavior of the benchmark problem of a boiling front, propagating through porous medium, sufficiently well. This is confirmed by comparing the numerical solution $(\eta_f, T_f)_{num} = (1.953 \cdot 10^{-3} m^2/s, 225.6°C)$ to an analytical solution $(\eta_f, T_f)_{an} = (2.663 \cdot 10^{-3} m^2/s, 224.1°C)$, which seem to be in rather good agreement.

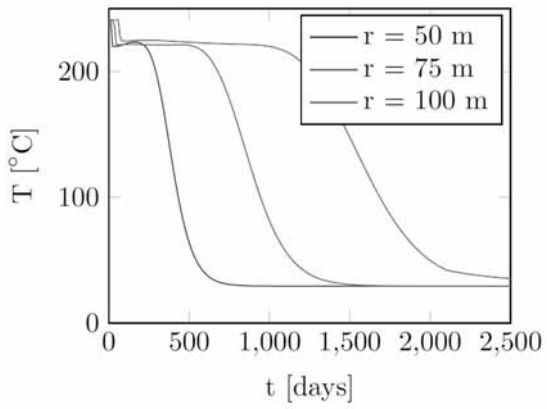


Figure 3: The temperature of the reservoir as a function of time for three different distances from the well.

However both the numerical and analytical solution provided in this paper seems to differ from the solution provided in [13] which uses the same parameters. One explanation for that difference could be that the papers use different thermodynamic formulations for the physical properties. While this paper uses IAPWSIF97 for the physical properties of steam and water, [13] uses the IFC-67 standard.

The ability of the OpenFOAM solver to handle phase change in porous media has also been demonstrated. Phase change can often result in numerical instabilities as displayed by [16]. These instabilities however do not seem to pose a problem in this paper. This is largely due to the fact that by applying Fréchet derivatives a more stable solution can be reached in the two phase region, and in the single phase region an unconditionally stable solution is made possible.

4 Conclusion

This paper has described the applicability of the OpenFOAM framework to take on problems involving phase change in hydrothermal systems. A benchmark case involving a boiling front moving through porous medium, which there exists an analytical solution of, was used to validate the results. The numerical model was also able to simulate the overall behavior of the boiling front sufficiently well. This increases our confidence in the numerical solver outside the limits of the analytical solution, and encourages continued usage of the OpenFOAM framework in geothermal reservoir modeling.

5 Acknowdgements

The authors would like to thank Eimskip University Fund, Landsvirkjun Energy Research Fund and Geothermal Research Group (GEORG) for their financial support of the project. This contribution is a post-conference publication from SIMS 2013 Conference (54th SIMS Conference, Bergen University College, Norway, October 16-18, 2013). The contribution was originally published in the Proceedings of SIMS 2013, to be found http://www. scansims.org/sims2013/SIMS2013.pdf.

References

[1] Bodvarsson G. Thermal problems in the siting of reinjection wells. *Geothermics*. 1972; 1(2): 63–66

[2] Henry HR.Effects of dispersion on salt encroachment in coastal aquifers.*Sea water in coastal aquifers*. 1964: 70-84

[3] IAPWS (2007). Revised Release on the IAPWS Industrial Formulation 1997 for the Thermodynamic Properties of Water and Steam. Technical Report August 2007, Lucerne, Switzerland

[4] Ingebritsen S, Geiger S, Hurwitz S, Driesner T. Numerical simulation of magmatic hydrothermal systems. *Reviews of Geophysics*. 2010; 48(1). doi: 10.1029/2009RG000287

[5] Jasak H, Jemcov A, Tukovic Z. OpenFOAM : A C ++ Library for Complex Physics Simulations. In *International Workshop on Coupled Methods in Numerical Dynamics*. 2007 volume m; 1-20

[6] Keating GN, Geissman JW, Zyvoloski GA. Multiphase modeling of contact metamorphic systems and application to transitional geomagnetic fields. *Earth and Planetary Science Letters*. 2002; 198(3-4):429-448

[7] Lewis KC, Lowell RP. Numerical modeling of two-phase flow in the NaCl-H2O system: Introduction of a numerical method and benchmarking. *Journal of Geophysical Research: Solid Earth*. 2009; 114(B5). doi: 10.1029/2008JB006029

[8] Matthäi SK, Geiger S, Roberts SG, Paluszny A, Belayneh M, Burri A, Mezentsev A, Lu H, Coumou D, Driesner T, Heinrich CA. Numerical simulation of multiphase fluid flow in structurally complex reservoirs. *Geological Society, London, Special Publications*. 2007; 292(1):405-429

[9] O'Sullivan MJ. State of the art of geothermal reservoir simulation. *Geothermics*. 2001; 30(4):395–429

[10] Podgorney R, Huang HC, Lu C, Gaston D, Permann C, Guo L, Andrs D. FALCON: A Physics Based, Massively Parallel, Fully Coupled, Finite Element Model for Simultaneously Solving Multiphase Fluid Flow, Heat Transport, and Rock Deformation for Geothermal Reservoir Simulation. 2011

[11] Powell MJD. A FORTRAN subroutine for solving systems of nonlinear algebraic equations. Technical report, Atomic Energy Research Establishment, Harwell (England). 1968

[12] Pruess K. TOUGH2 — A General Purpose Numerical Simulator for Multiphase Fluid and Heat Flow. Technical Report May, Lawrence Berkeley Laboratory, Berkeley, California. 1991

[13] Pruess K. Celatis R. An analytical solution for heat transfer at a boiling front moving through a porous medium.*International Journal of Heat and Mass Transfer* 1987; 30(12):2595–2602

[14] Stanford Geothermal Program. Proceedings of the Special Panel on Geothermal Model Intercomparison Study. In *Report SGP-TR-42*. 1980 Stanford, CA

[15] Theis, C. The relation between the lowering of the piezometric surface and the rate and duration of discharge of a well using groundwater storage. *Transactions - American Geophysical Union*. 1935; 16:519–524

[16] Thorvaldsson L, Palsson H. A numerical analysis on flow in hydrothermal systems. In *Stanford Geothermal Workshop* 2013.

[17] Weller HG, Tabor G, Jasak H, Fureby C. (1998). A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*. 1998; 12(6).

Comparison of Programmed MATLAB Implementation and Graphically Modelled Simulink Implementation for ARGESIM Benchmark C11 'SCARA Robot'

Tamara Vobruba[1,*], Claudia Wytrzens[1], Andrea Kainz[1,2], Irene Hafner[3]

[1]Department of Analysis and Scientific Computing, Vienna University of Technology, Wiedner Haupstraße 8-10, 1040 Vienna, Austria; *tamara.vobruba@tuwien.ac.at

[2] AIT Austrian Institute of Technology, Tech Gate Vienna, Donau-City-Straße 1, 1220 Vienna, Austria

[3] dwh GmbH, Neustiftgasse 57-59, 1070 Vienna, Austria

Simulation Notes Europe SNE 24(3-4), 2014, 173 - 178
DOI: 10.11128/sne.24.bn11.10265
Received: February 15, 2014; Revised: October 3, 2014;
Accepted: October 20, 2014;

Abstract. This approach to ARGESIM Benchmark C11 'SCARA Robot' compares a programmed MATLAB implementation with a graphically modelled Simulink implementation, esp. with respect to implicit models and state event descriptions.

The mechanical system results in an implicit second order differential equation. Therefore, different ways of solving the equation using MATLAB and Simulink are investigated. In this context, the question of using an explicit or an implicit model description arises. Based on the best fitting model description, a point-to-point motion of the robot controlled by a single axis PD-control is simulated. Furthermore, a collision avoidance maneuver of a defined obstacle is implemented in MATLAB as well as in Simulink. Finally, the two models are compared and the advantages and disadvantages of each model are pointed out.

Introduction

ARGESIM Benchmark C 11 [1] deals with motion of a three axis SCARA (Selective Compliance Assembly Robot Arm) robot. The equation of motion leads to an implicit second order system of differential equations for the joint vector q, where q_1 and q_2 are the joint angles and q_3 is the joint distance.

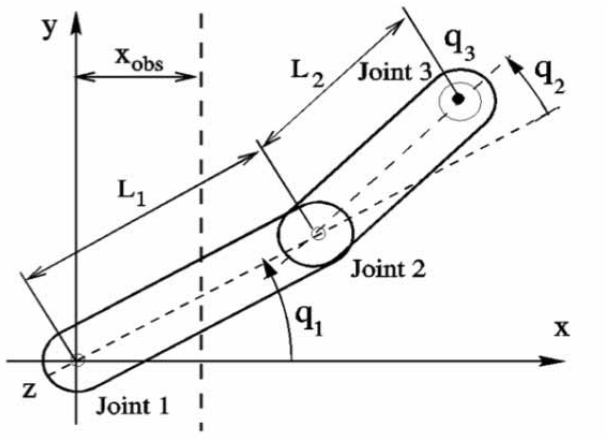


Figure 1: Schematic representation of the SCARA robot. [1]

The observed mechanical system is fully defined. The balance of power and the constraints result in a system of second order differential equations using variational calculus methods [3]. So the equation of motion can be written in a compact way as

$$\boldsymbol{M\ddot{q} = b}. \tag{1}$$

Here the Mass matrix M is a block-diagonal matrix as described in formula (2).

$$\begin{pmatrix} ma_{11} & ma_{12} & 0 \\ ma_{21} & ma_{22} & 0 \\ 0 & 0 & ma_{33} \end{pmatrix} \cdot \begin{pmatrix} \ddot{q}_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}. \tag{2}$$

The elements of the Mass matrix $\boldsymbol{M}$ and the right-hand side $\boldsymbol{b}$ are determined as follows:

$$ma_{11} = \theta_1 + 2\theta_2 \cos(q_2) + \theta_3 \quad (3)$$
$$ma_{12} = ma_{21} = \theta_2 \cos(q_2) + \theta_3 \quad (4)$$
$$ma_{22} = \theta_3 \quad (5)$$
$$ma_{33} = m_{3L} + \theta_{3mot} u_3^2 \quad (6)$$

$$b_1 = T_1 + \theta_2(2\dot{q}_1\dot{q}_2 + \dot{q}_2^2)\sin(q_2) \quad (7)$$
$$b_2 = T_2 - \theta_2 \dot{q}_1^2 \sin(q_2) \quad (8)$$
$$b_3 = T_3 - m_{3L} g\,, \quad (9)$$

where θ_i are the moments of inertia and T_i are the joint forces, whose calculation is based on the assumption that the two links are rods with homogenous mass distribution m_1 and m_2 along the length L_1 and L_2 of the rods.

The capabilities of MATLAB and Simulink to handle this system are explored. For all computations, MATLAB R2013b is used running on a Mac OS X Version 10.9.5.

1 Modelling Method

There are different tools in MATLAB and Simulink to solve systems of differential equations and therefore different methods of describing the system.

1.1 MATLAB

At first, the different methods of solving the model-equations numerically in MATLAB were investigated. The ODE-solver provided by MATLAB can only deal with first order differential equations [2], thus the three-dimensional second order system has to be transformed into a six-dimensional first order system.

$$\begin{pmatrix} ma_{11} & ma_{12} & 0 & 0 & 0 & 0 \\ ma_{21} & ma_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & ma_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix}. \quad (10)$$

The ODE-solver ode45 is able to deal with implicit and explicit differential equations [2]. In order to get an explicit model description, the mass matrix in system (2) is inverted. To prove the regularity of the matrix the determinant has to be examined. Per definition of each of the factors and addends it is a fact that

$$ma_{33} = m_{3L} + \theta_{3mot} u_3^2 \neq 0 \quad (11)$$
$$det \begin{pmatrix} ma_{11} & ma_{12} \\ ma_{21} & ma_{22} \end{pmatrix} = \theta_1\theta_3 - \theta_2^2 \cos(q_2) \neq 0. \quad (12)$$

So it is possible to invert the mass matrix in order to get an explicit system. The inversion of the matrix is calculated analytically and the explicit system is implemented in MATLAB.

The average runtimes of the explicit and the implicit model were compared and as a result – as illustrated in Table 1 - the explicit model is a bit faster and so it is used for the additional tasks.

System	runtimes
implicit (1st order)	4.7382s
explicit (1st order)	3.9431s

Table 1: Runtimes of the implicit and explicit model in MATLAB.

1.2 Simulink

In Simulink it is possible to implement second order differential equation – without transforming it into a first order differential equation – by using the 'Integrator'-block twice [2]. For this comparison the ode-solver ode45 is used in Simulink as well.

So the three dimensional square mass matrix in system (2) is inverted analytically and the resulting explicit second order system can be implemented in Simulink. It is notable that the Simulink model is a lot faster than both models in MATLAB.

System	runtime
explicit (2nd order)	0.2093s

Table 2: Runtime of the Simulink model.

2 Point-to-point Motion

Furthermore, the implementation of a point-to-point motion by the SCARA robot, controlled by a single axis PD-control, was performed. Therefore three more first order differential equations are added to the system of differential equations, which describe the electrical relationship of the armature of the robot servo motor,

$$\dot{I}_i = \frac{U_i - k_{T_i} u_i - R_i I_i}{L_i}, \qquad i = 1,2,3 \tag{13}$$

$$I_i \in [-I_{imax}, I_{imax}], \qquad i = 1,2,3 \tag{14}$$

where k_{T_i} is the motor constant, u_i is the gear ratio, R_i is the resistance and L_i is the inductance. The single-axis PD-control is involved to control the point-to-point motion of the robot by the control of the voltage U_i and represented by the equations

$$U_i = P_i(\hat{q}_i - q_i) - D_i \dot{q}_i, \qquad i = 1,2,3 \tag{15}$$

$$U_i \in [-U_{imax}, U_{imax}], \qquad i = 1,2,3. \tag{16}$$

Here $\hat{q}_i$ stands for the target point of the SCARA robot motion and therefore P_i describes the proportional gains and D_i derivative gains. The values of both of them are given for each controller.

2.1 Implementation in MATLAB

To implement the point-to-point performance of the SCARA robot in MATLAB the six-dimensional first order system – as described in Section 1.1 – has to be transferred into a nine-dimensional first order system by adding the three first order differential equations as given in formula (13).

Furthermore, in order to limit the values of $U_i$and I_i such that their values are located in the intervals given above, their numerical values are compared to the limits of the intervals by using logical expressions in MATLAB in the way that the expression

$$\begin{aligned}(U_i < -U_{imax}) \cdot (-U_{imax}) + (U_i > U_{imax}) \\ \cdot U_{imax} + (-U_{imax} \leq U_i \wedge U_i \\ \geq U_{imax}) \cdot U_i\end{aligned} \tag{17}$$

provides the right value for U_i, which is either $-U_{imax}$ if U_i is not located in the considered interval and less than $-U_{imax}$ or U_{imax} if U_i is larger than U_{imax} or U_i if it is situated in the interval.

In addition to this an event function is included to stop the integration at the target position using an accuracy of $acc = 0.001$ of reaching the target value.

2.2 Implementation in Simulink

The point-to-point motion of the robot is implemented in Simulink by adding the PD-control and the differential equation of the electrical current as given in formula (13) more or less as a system according to q which is running parallel to the system which describes q and its derivations. This parallel running system is shown in Figure 2.

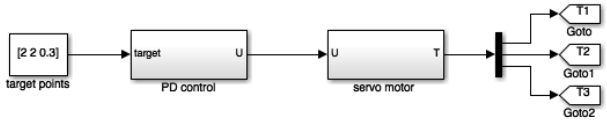


Figure 2: PD-control and servo motor subsystems for the implementation of the point-to-point motion in Simulink.

The outputs of the system are the values T_i, which are needed to calculate the right-hand side b_i of the dynamical equations.

Compared to the implementation in MATLAB, to control that the values of electrical voltage and current are located inside the intervals a 'Saturation'- block is used, which proves to be a much simpler way.

Furthermore, for stopping the integration at the target position the same accuracy as in MATLB $acc = 0.001$ of reaching the target value is used. This is realised in Simulink by using a "Stop"- block.

2.3 Results

The point-to-point motion is – as already mentioned – implemented in MATLAB as well as in Simulink. Figure 3 shows the point-to-point motion of both models with initial value $q = [0,0,0]$ and target point $\hat{q} = [2,2,0.3]$.

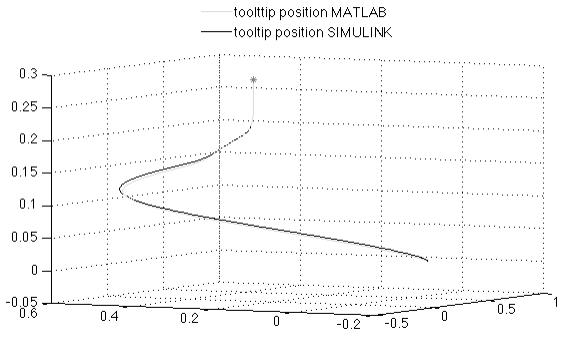


Figure 3: Comparison of the implementation of the point-to-point motion in MATLAB and Simulink.

Figure shows that the tooltip of the MATLAB model is following the same movement as the tooltip of the Simulink model. Both models are reching the target point, which is marked as a red dot in Figure 3. Furthermore, by comparing the simulation times (MATLAB: $t = 1.5719$, Simulink: $t = 1.5719$), which both models need to reach the target position, one can see that there is no significant difference. But if comparing the runtime one can observe that the Simulink simulation with about 0.2093 seconds is way faster than the runtime needed by MATLAB, which is about 3.9431 seconds. So for the simple point-to-point motion the Simulink model in terms of the simulation time and the runtime seems to be the better option as there is no difference in the simulation time but the runtime is a lot faster.

3 Obstacle Avoidance

In addition the model description is extended to simulate an avoidance manoeuvre of a given obstacle. The idea is to slow down the movement of the tooltip in the xy-plane, if the tooltip crosses a critical distance to the obstacle until the height of the obstacle is exceeded. So the given condition for slowdown is:

If $(x_{tip} - x_{obs}) \leq d_{crit} \wedge (q_3 \leq h_{obs})$
decelerate $\dot{q}_1, \dot{q}_2$ until $q_3 \geq h_{obs}$.

Furthermore the permissible interval for the armature voltage U_i is widened to

$U_i \in [-U_{imax}^{max}, U_{imax}^{max}],\ i = 1,2,3.$

3.1 Implementation in MATLAB

The implementation of the obstacle avoidance in MATLAB is realised by extending the point-to-point motion model by using a two-dimensional event function, which stops the integration just when the above described condition is fulfilled. The first component of the event function detects, if the distance between the tooltip and the obstacle in the *xy*-plane is critical. The second component detects, whether the tooltip position exceeds the height of the obstacle in z-direction. As in the point-to-point motion model, the conditions are written in a logical structure.

```
function [ value , isterminal , direction ]
= event_function1 (~ , w)
       value =[1 - double (( abs ( L1 * cos
       ( w (3))+ L2 * cos (w (3)+ w (4)) -
       xobs )< dcrit )
       && (w (5) < hobs )); 1- double (w (5)
       > hobs )];
isterminal =[1;1];
direction =[ -1; -1];

end
```

Using a nested function, the right hand side of the differential equation is state-depending. So detecting the first event, the velocities $\dot{q}_1$ and $\dot{q}_2$ in the right hand side of the differential equation are decelerated dramatically by multiplying them with the factor $dec = 0.00005$. Additionally the widened interval for U_i is used here as well.

By using a while-loop this procedure is repeated using the final values of the last step before the detection of the event as new initial values for the integration until the second event occurs, so until the height of the obstacle is crossed. After this event the point-to-point motion model from task b can be used, since no obstacle is present anymore.

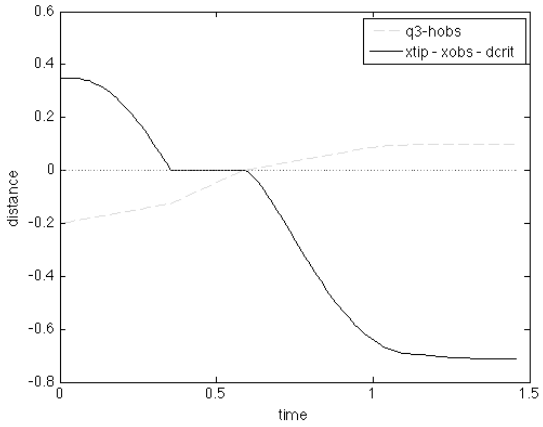


Figure 4: Distance between the joint vector and the obstacle in MATLAB.

In Figure 4 one can see that the motion of the tooltip has to be nearly stopped if it has a critical distance to the obstacle until the z-position of the tooltip exceeds the height of the obstacle and afterwards the point-to-point model is used. Here the initial values have to be set to zero, except the positions q_1, q_2 and q_3.

So the values for the velocities $\dot{q}_i$ and for the armature current $\dot{I}_i$, are set to zero. The reason for this is that otherwise the values of the last integration step of the extended collision avoidance model do not fulfil the initial value problem of the point-to-point motion model.

3.2 Implementation in SIMULINK

In Simulink, the point-to-point model is extended by distinguishing the different states as described above using a 'Switch'-block. This block switches between the values for $\dot{q}_1$, $\dot{q}_2$ and the values $\dot{q}_1 \cdot dec$ and $\dot{q}_1 \cdot dec$, where $dec = 0.00005$. Depending on the state this values are forwarded to the 'Integrator'-block and the right hand side of the differential equation.

The condition $(x_{tip} - x_{obs}) \leq d_{crit} \wedge (q_3 \leq h_{obs})$ is written in a logical structure using the "fcn"-block and a logical operator-block. The same condition is used in a second 'Switch'-block to distinguish between the permissible intervals for the armature voltage U_i. In Simulink it is not necessary to adjust the initial conditions after the height of the obstacle is exceeded.

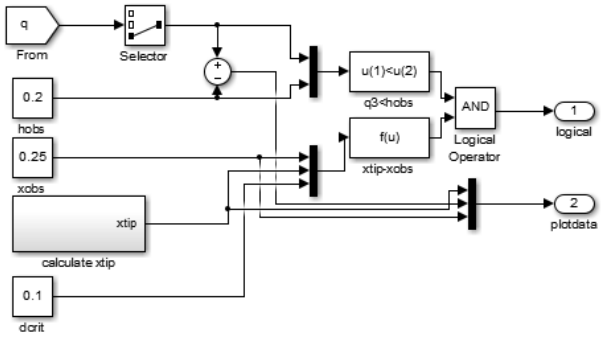


Figure 5: Condition for state – switch in Simulink.

In Figure 6 one can see a similar behaviour as in Figure 4 of the simulation in MATLAB. If the distance between the tooltip and the obstacle is critical the movement in the *xy*-plane is decelerated dramatically by multiplying by dec, until the height of the obstacle is crossed.

3.3 Results

There are some differences in the simulated results, so the movement of the tooltip differs, which can be observed in Figure 7. Until the first event occurs both simulations show very similar behaviour. But after the obstacle is passed the movements differ a lot.

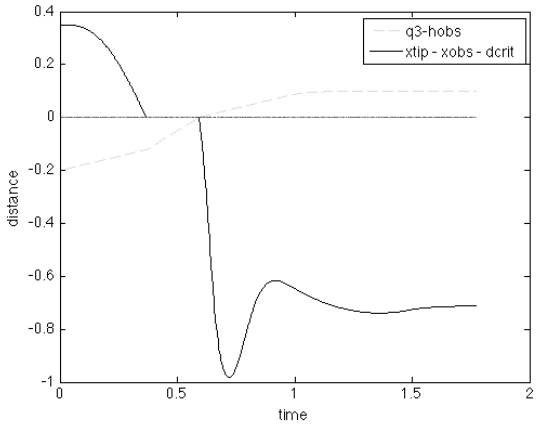


Figure 6: Distance between the joint vector and the obstacle in Simulink.

The reason for this deviance is that after the second event occurs the values of the velocities and the armature current are set to zero in the MATLAB model. In Simulink this is not necessary, therefore the velocity is bigger and the tooltip has to spin around to reach the target point - marked as a red dot in Figure 7.

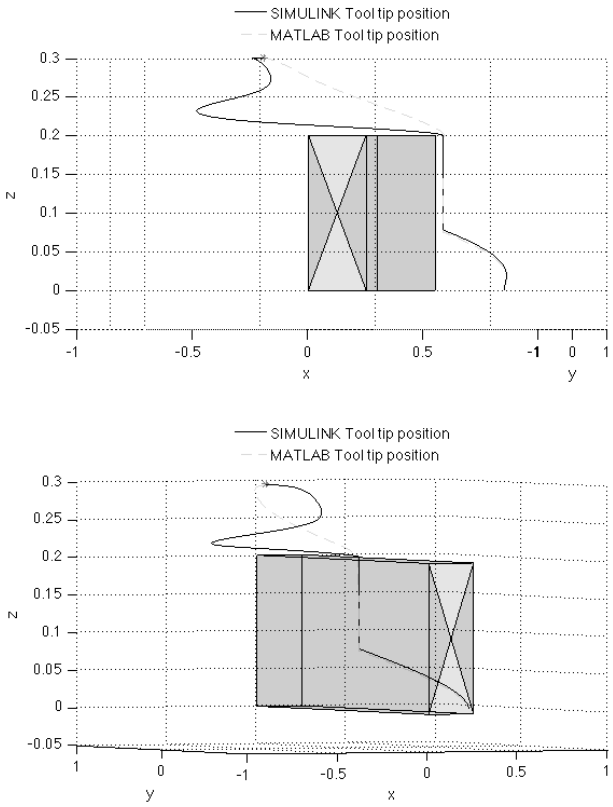


Figure 7: Tooltip position simulated in MATLAB and Simulink.

Furthermore, this also means that the simulation time of Simulink, which the tooltip needs to reach the target point, is much longer than of MATLAB.

Model	Simulation time
MATLAB	1.4555
Simulink	1.7736

Table 3: Simulation time of MATLAB and Simulink.

In Simulink it is possible to reduce the deceleration-factor to $dec = 0.005$, but the implementation in MATLAB cannot cope with this situation, because the slowdown of the movement in the *xy*-plane is too little. So the tooltip would not have enough time to exceed the height of the obstacle, which would mean that the tooltip would crash into the obstacle. So Simulink is much steadier in this situation than MATLAB.

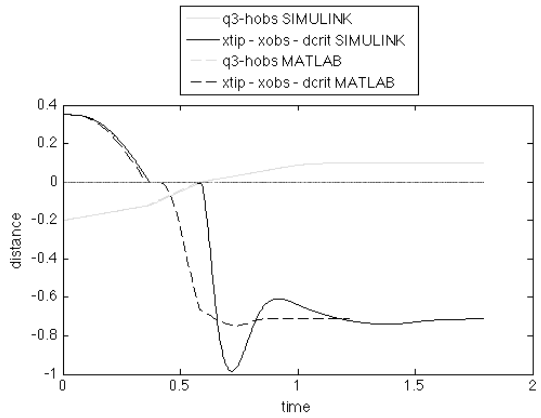


Figure 8: Distance between the joint vector and the obstacle in Simulink and MATLAB with $dec = 0.005$.

4 Discussion

As already mentioned two different kinds of comparisons have been done. On the one hand the explicit with the implicit implementation in MATLAB only were compared. Here the runtimes of both implementations were observed and resulting from this the explicit model was faster than the implicit one as already shown in Section 1.1.

Therefore, the further comparison of MATLAB and Simulink only refers to the explicit description of the given system. One big difference of those two kinds of implementation is that MATLAB cannot solve a second order differential equation system and therefore it had to be transferred into a first order differential equation system. But Simulink can handle such second order differential equation systems as given by connecting two 'Integrator'-blocks in series.

For the simple point-to-point motion there is no big difference in using either the MATLAB model or the Simulink model, both tooltips follow the same movements and the simulation times are also equal. The only difference is that the real runtime is a lot faster for the Simulink simulation than for the MATLAB one.

The last task which was implemented is the obstacle avoidance. The results of the two different implementations show that not only the simulation times and runtimes differ but also the movement of the tooltip. As the MATLAB model is quicker in solving this task it seems to be obvious to prefer the MATLAB implementation for the obstacle avoidance task. But one should not disregard that the Simulink model is steadier in respect of changing the factor dec for slowing down, especially in terms of increasing this factor and there is no need of changing the initial conditions in order to use the point-to-point model after crossing the height of the obstacle in the collision avoidance model.

Model sources

For this C11 Benchmark approach, all MATLAB model m-files, all Simulink model mdl-files, all parametrization m-files, and a short file description can be downloaded (zip format) by EUROSIM societies' members from SNE website, or are availably from the corresponding author.

References

[1] Ecker H, Comparison 11: SCARA Robot. EUROSIM-*Simulation News Europe*. 1998; 22: 30-32

[2] mathworks: Products and Services [Internet]. The MathWorks, Inc.: c1994-2014 [cited 2014 Nov 13]. Available from: http://uk.mathworks.com/products/index.html?s_tid=gn_loc_drop

[3] Hairer E, Wanner G. *Solving Ordinary Differential Equations II*. Germany: Springer; 2002. 491p.

Discussion of two Case Studies on DAEs using Different Approaches for Regularisation

Carina Pöll[1*], Irene Hafner[2], Bernhard Heinzl[3]

[1] Institute of Analysis and Scientific Computing, Vienna University of Technology, Wiedner Hauptstraße 8-10, 1040 Vienna, Austria; *carina.poell@tuwien.ac.at*

[2] dwh GmbH, Simulation Services, Neustiftgasse 57-59, Vienna, Austria

[3] Institute of Computer Aided Automation, Vienna University of Technology, Treitlstraße 3, 1040 Vienna, Austria

Simulation Notes Europe SNE 24(3-4), 2014, 179 - 184
DOI: 10.11128/sne.24.tn.10267
Received: July 15, 2014; Revised September 14, 2014;
Accepted: October 20, 2014;

Abstract. The object-oriented model description of physical or mechanical systems leads to differential-algebraic equations. In general the numerical solution of such equation systems is very complex, numerically extensive or may even be impossible. Therefore it is important to find methods for solving given equation system, this leads to the so-called index reduction and regularization methods. This paper gives a short overview of common methods of index reduction. Additionally a classification of these different approaches is made. Afterwards each approach is presented in detail and the advantages and disadvantages of the different methods are discussed. In order to compare the different index reduction methods, the methods described above are demonstrated by various examples. For the comparability of the different methods the obtained numerical solutions and the deviation from the constraint equations are displayed graphically. Therefore the distinct approaches can be compared with regard to their numerical solutions. The two examples are mechanical systems with differential index three. The equations of motion of a pendulum on a circular path in Cartesian coordinates and the motion of the double pendulum in Cartesian coordinates, which shows a chaotic behaviour, are used as case studies.

Introduction

An object-oriented acausal model description for physical or mechanical systems, such as Modelica or MATLAB/Simscape, leads to differential-algebraic equations with non-trivial differential index.

The numerical solution of these equations with methods for ordinary differential equations is generally very complex and therefore numerically extensive or may even be impossible. This problem leads to the so-called index reduction or regularisation methods. These methods transform the given differential-algebraic equation into a differential-algebraic equation with lower differential index or into an ordinary differential equation. Due to the large differences (structure, properties, etc.) of differential-algebraic equations, in the literature there can be found a number of different approaches and methods for the reduction of the differential index and the regularisation. Some of these approaches will be compared in this paper and evaluated by means of case studies. These approaches can be split into three topics, see [1]:

- index reduction using differentiation
- stabilization by projections
- methods based on local state space transformations

Each of these topics is discussed in detail in Section 2.

1 Basic Definitions

In this section some basic definitions, which are used in the following, are presented. A differential-algebraic equation (DAE), see [2], is given by an implicit equation,

$$F(t, x, \dot{x}) = 0, \tag{1}$$

with $F: I \times D_x \times D_{\dot{x}} \to \mathbb{R}^n$ $(n \in \mathbb{N})$, where I is a real interval, $\dot{x}$ denotes the derivative of x with respect to t and D_x, $D_{\dot{x}}$ are open subsets of $\mathbb{R}^n$. A differential-algebraic equation consists of differential as well as algebraic variables and equations.

The algebraic equations of the given differential-algebraic equation have the form

$$g(x) = 0 \tag{2}$$

where g is a function $g: \mathbb{R}^n \to \mathbb{R}^k$ and $k < n$, and are called constraints or constraint equations.

Equation (1) has differential index $m \in \mathbb{N}$, if m is the minimal number of derivatives such that from the system

$$F(t, x, \dot{x}) = 0, \frac{dF(t,x,\dot{x})}{dt} = 0, \dots, \frac{d^m F(t,x,\dot{x})}{dt^m} = 0 \tag{3}$$

an ordinary differential equation system can be extracted via algebraic manipulations, see [1]. After these algebraic calculations the given system can be transformed into an ordinary differential equation $\dot{x} = \varphi(t, x)$ with $\varphi: I \times D_x \to \mathbb{R}^n$. In the following the differential index is also called only index.

2 Regularisation Methods

In the following six regularization approaches are discussed, see [1].

2.1 Differentiation and substitution of the constraint

A first idea for the reduction of the index is to differentiate the constraint equations and substitute the constraint equations by its derivatives. This procedure is repeated until the differential-algebraic equation has differential index 1. For example, let a DAE with differential index 3 be given, the constraint equations are substituted by the second derivative with respect to the time, i.e. the new constraint equation is

$$\ddot{g}(x) = 0. \tag{4}$$

The resulting system has index 1. A problem of this method is that due to the derivation there is a loss of information. Therefore the necessary initial values for the integration are unknown. This fact causes a numerical "drift-off", i.e. the numerical solution departs from the solution manifold.

2.2 Baumgarte-Method

Another approach using differentiation is the Baumgarte-Method, see [3]. This method substitutes the constraint equation (2) by a linear combination of $g, \dot{g}$ and $\ddot{g}$, i.e. instead of (2) the equation

$$\ddot{g} + 2\alpha\dot{g} + \beta^2 g = 0 \tag{5}$$

is used. This approach is motivated by the special form of the new constraint equation.

The shape of equation (5), i.e. the appearance of g and $\dot{g}$, ensures that there is no loss of information like in the approach explained above.

The parameters α and β in equation (5) have to be chosen such that the ordinary differential equation (5) is asymptotically stable. Therefore the zeros of the characteristic polynomial

$$\lambda^2 + 2\alpha\lambda + \beta^2 \tag{6}$$

of the ordinary differential equation have to be computed, which results in

$$\lambda_{1,2} = -\alpha \pm \sqrt{\alpha^2 - \beta^2}. \tag{7}$$

Therefore follows $\alpha > 0$. A problem of this approach is the choice of the two parameters.

2.3 Pantelides Algorithm

The procedure of the Pantelides-Algorithm has a fixed routine for every constraint equation. This algorithm is given by the following steps, see [4].

- The constraint equation has to be differentiated.
- The differentiated constraint has to be added to the system of equations. If there is an algebraic variable in the constraint equation, then the derivative of this variable becomes a so-called dummy derivative, for example the derivative of the algebraic variable y is written as dy, which is called dummy derivative.
- An integrator which has a connection to the constraint equation and the derivative of the constraint respectively is eliminated, i.e. for example $\dot{x}$ is eliminated and instead of $\dot{x}$ a new variable called dx is used.
- By differentiation of the constraint it can occur that a new variable is generated, i.e. for example through differentiation y, which is an algebraic variable, becomes dy and there is an equation where y can be computed in the system (otherwise the constraint equation would not be a constraint equation).
- Therefore the equation of which y can be computed also has to be differentiated.
- The procedure of the last two points has to be repeated until no new variables are created.

A problem of this method is that during the procedure of the algorithm a lot of variables and equations may be created. Therefore the system of the resulting equations is getting large and can be unclear. Compared with the other methods which are discussed in this paper, this is the only approach where it is not necessary to compute the differential index.

2.4 Orthogonal Projection method

General assumptions:

- A DAE with differential index $k > 1$ is given.
- The algebraic variables can be expressed by the $(k-1)^{th}$ derivative of the constraint equation with respect to t.

The solution manifold M is given by the constraint equation and the first till the $(\mathrm{k}-2)^{\mathrm{th}}$ derivative of the constraint equation with respect to t, i.e.

$$M = \{x \in \mathbb{R}^n : g(x) = 0, \frac{d^i g}{dt^i} = 0,\ i \in I\}, \quad (8)$$

where $I = \{1, \dots, k-2\}$.

The idea of the orthogonal projection method is to project orthogonally onto the solution manifold M, if the numerical solution does not fulfil the constraints.

For one step the procedure of this method is given as follows, see [6]:

- $\hat{y}_{n+1} = \Phi(y_n)$, where Φ is a numerical integrator.
- y_{n+1} is the orthogonal projection of $\hat{y}_{n+1}$ onto the solution manifold M.

In Figure 1 one step of this approach is shown.

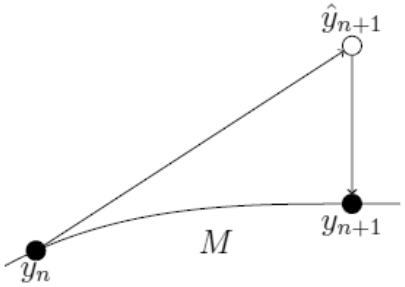


Figure 1: Schematic illustration of the orthogonal projection method.

The problem of this method is to find the orthogonal projection. Another fact is, that the numeric integration has to be stopped after each step for checking whether the numerical solution fulfils the constraint equations and if it is not fulfilled, the orthogonal projection has to be applied.

2.5 Symmetric projection method

The idea of the symmetric projection method is to perturb y_n so that it is not on the solution manifold M and then apply a symmetric one-step-method. The distance of the new value (computed with the symmetric one-step-method) to the manifold corresponds to the absolute value of the perturbation.

For the general assumptions see section 2.4. The solution manifold is given by equation (8).

A one-step-method Φ_h (with step size h) is symmetric if $\Phi_h = {\Phi_{-h}}^{-1}$.

For one step the procedure of this method is: (see [7])

- $\hat{y}_n = y_n + \frac{d\tilde{g}}{dy}\mu$ where $\tilde{g}(y_n) = 0$.
- $\hat{y}_{n+1} = \Phi_h(\hat{y}_n)$, where Φ_h is a symmetric one-step-method.
- $y_{n+1} = \hat{y}_{n+1} + \frac{d\tilde{g}}{dy}\mu$ where $\tilde{g}(y_{n+1}) = 0$.

It is important that in the first and in the third step the same μ is used. Therefore the calculations described above have to be implemented with an iteration. The function $\tilde{g}$ consists of the constraint equations and of the derivatives of the constraint equations, which are used for describing the solution manifold.

In Figure 2 a schematic presentation of one step of this approach is shown.

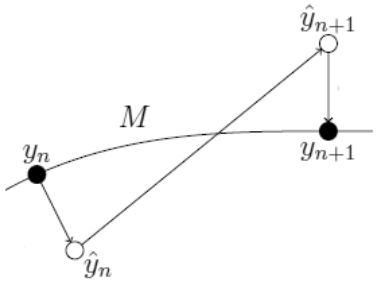


Figure 2. Schematic illustration of the symmetric projection method.

The disadvantage of this method is that the calculation has to be iterative and therefore the solution time is getting bigger in contrast to the methods where an ode-solver of MATLAB is used.

2.6 Methods based on local state space transformation

For the general assumptions see Section 2.4. The solution manifold is also given by (8).

The general idea of this approach is instead of solving the system on the whole state space only to solve it on a manifold, which is realized with an appropriate local coordinate transformation ψ.

For one step the procedure of this method is, see [6]:

- z_n is calculated with $\psi(z_n) = y_n$.
- $z_{n+1} = \Phi(z_n)$, where Φ is a numerical integrator.
- $\psi(z_{n+1}) = y_{n+1}$.

The local coordinatisation can be changed in every step. If there is a global coordinatisation, this method leads to a simple equation system. The difficulty of this approach is to find a suitable local or global coordinatisation.

3 Case Study 'Pendulum'

The first example is the motion of a pendulum in Cartesian coordinates, which is shown in Figure 3.

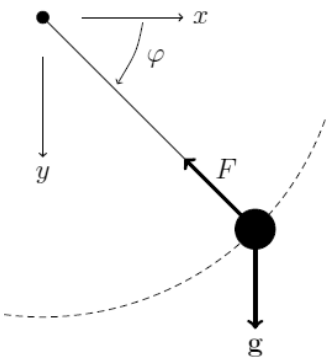


Figure 3: Schematic illustration of the motion of a pendulum in Cartesian coordinates.

The equations of motion of the pendulum are given by the following equations, see [5],

$$\dot{x} = v_x \quad (9)$$
$$\dot{y} = v_y \quad (10)$$
$$\dot{v}_x = -Fx \quad (11)$$
$$\dot{v}_y = g - Fy \quad (12)$$
$$x^2 + y^2 = 1, \quad (13)$$

where g is the gravitational acceleration and F is the force. Equation (13) is the constraint equation. The DAE (9)-(13) has differential index three, which can be seen from the third derivative with respect to t of equ. (13).

All simulations are realized with MATLAB R2012b. The first approach leads to the numerical 'drift-off'. Therefore this approach is not suitable for the simulation of the given DAE. This can be seen in Figure 4, where the result of the simulation with the ode-solver `ode15s` is shown.

The orthogonal projection method uses an explicit *Euler* method for the integration. The numerical solution of this method is not correct due to the increasing speed, which is shown in Figure 5.

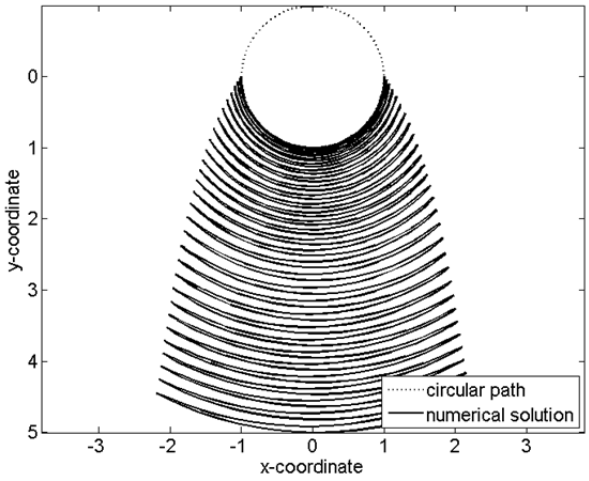


Figure 4: Result of the simulation using differentiation and substitution of the constraint.

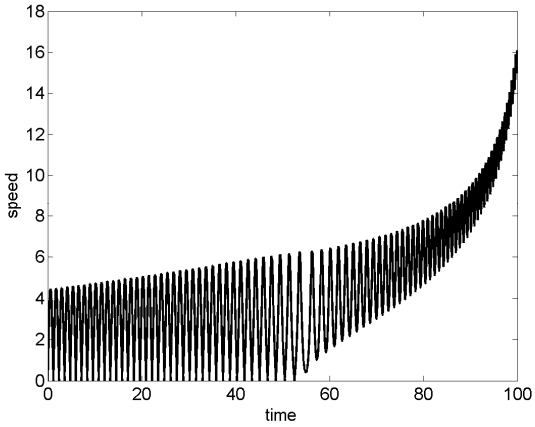


Figure 5: Speed of the numerical solution obtained with the orthogonal projection method.

The Baumgarte-Method leads to a system of ordinary differential equations, which is solved with the ode-solver `ode45`.

The Pantelides-Algorithm leads to four equation systems because of equation (13), which are used for four different regions of the unit circle. For solving these four equation systems the ode-solver `ode15i` is used.

The symmetric projection method is solved iteratively, where the trapezoidal rule, which is a symmetric one-step-method, is used. In contrast to the orthogonal projection method, this method does not show increasing speed.

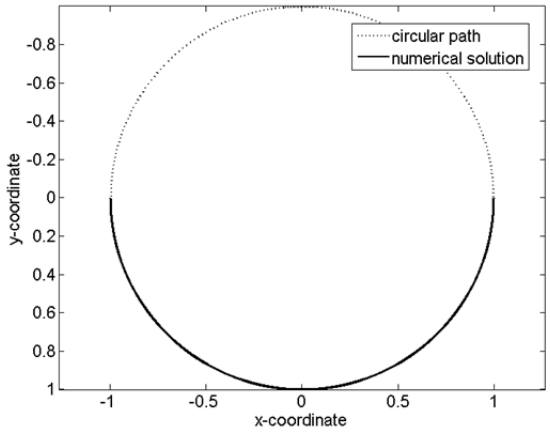


Figure 6: Result of the phase space trajectory.

The state space transformation is solved using polar coordinates for the coordinatisation. This coordinatisation is global, which is a great advantage of this method. This method leads to a two-dimensional system of ordinary differential equations, which is solved with `ode45`.

The numerical solutions obtained with the Baumgarte-Method, Pantelides-Algorithm, symmetric projection method and the state space transformation result in quite similar results with respect to the phase space trajectory. Therefore only one of these solutions is shown graphically, see Figure 6. While the phase space trajectory looks similar, the deviations of the numerical solutions obtained with the different methods show big differences.

The state space transformation has the smallest deviation from the numerical solution to the circular path and is easy to implement. Therefore this method would be the recommended approach for the given DAE.

4 Case Study 'Double Pendulum'

The second case study treats a double pendulum. The equations of motion of the pendulum are given by the following equations,

$$\dot{x}_1 = v_{x_1} \quad (14)$$

$$\dot{y}_1 = v_{y_1} \quad (15)$$

$$\dot{x}_2 = v_{x_2} \quad (16)$$

$$\dot{y}_2 = v_{y_2} \quad (17)$$

$$\dot{v}_{x_1} = -F_1 x_1 - F_2(x_1 - x_2) \quad (18)$$

$$\dot{v}_{y_1} = g - F_1 y_1 - F_2(y_1 - y_2) \quad (19)$$

$$\dot{v}_{x_2} = -F_2(x_2 - x_1) \quad (20)$$

$$\dot{v}_{y_1} = g - F_2(y_2 - y_1) \quad (21)$$

$$x_1{}^2 + y_1{}^2 = 1 \quad (22)$$

$$(x_1 - x_2)^2 + (y_1 - y_2)^2 = 1 \quad (23)$$

Where g is the gravitational acceleration and F_1 and F_2 are forces. Equations (22) and (23) are the constraint equations. The DAE (14)-(23) has differential index three, which can be seen from the third derivative with respect to t of equations (22) and (23). Like before all simulations are realized with MATLAB R2012b.

The approach using differentiation and substitution of the constraint is, like for the first case study, not suitable for the numerical simulation of the given DAE.

The orthogonal projection method has unbounded speed, which leads to 'wrong' positions on the solution manifold. Therefore this method cannot lead to reasonable results.

The Baumgarte-Method leads to very different results for different values of the parameters. In Figure 7 one result with the Baumgarte-Method is shown, where the red line is the numerical solution of the second pendulum until 10 seconds and from 10 till 100 seconds the numerical solution of this pendulum is shown in black.

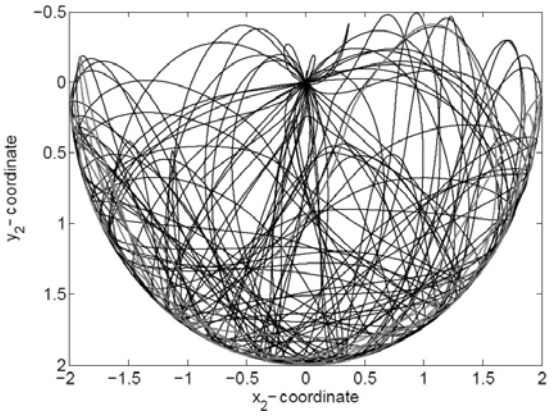


Figure 7: Result of the pendulum (x_2, y_2) with the Baumgarte-Method with $\alpha = 100$ and $\beta = 1000$.

The simulation of the Pantelides-Algorithm leads to a problem. The ode-solver has problems to solve the equations, which leads to too small step sizes. Therefore the ode-solver `ode15i` has to be stopped and restarted with new values. Furthermore there are sixteen different equation systems which are a little complex to implement.

The symmetric projection method again has a long simulation time because of the iterative calculation, but with this approach the speed is bounded in contrast to the orthogonal projection method.

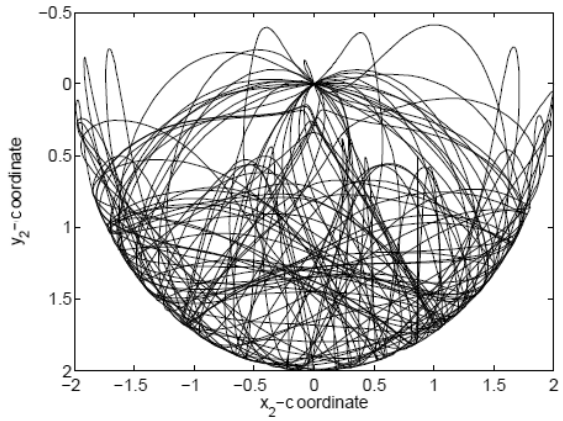


Figure 8: Result of the pendulum (x_2, y_2) with the state space transformation.

Using the state space transformation the given DAE can be transformed into a system of four ordinary differential equations with the use of a coordinatisation using cosine and sine. Simulating this approach using ode45 or `ode23t` for example leads to different results, which can be explained by the chaotic behaviour of the double pendulum, see [8]. In Figure 8 the result for the second pendulum with the state space transformation is shown. Comparing Figure 7 and Figure 8 it is obvious that the numerical solution using Baumgarte-Method and state space transformation is not equal. Still one can observe that until 10 seconds the results of most of the approaches are similar, but from 10 till 100 seconds the numerical solutions show big differences.

In general it is a fact that for this case study one cannot say whether a simulation result is the "right" or the "best" because there exists no analytical solution and the double pendulum shows a chaotic behaviour.

5 Conclusion and Outlook

After analysing the presented methods with the two examples some facts can be observed.

The method using differentiation and substitution of the constraint equations and the orthogonal projection method do not lead to suitable results. Using the first method leads to the numerical 'drift-off'. The second method does not show the numerical 'drift-off', which means that the numerical solution stays on the solution manifold, but due to the increasing speed the positions are not correct.

- The Baumgarte-Method results in small deviations to the constraint equations for a suitable choice for the two parameters.
- The implementation of the Pantelides-Algorithm is a little complex because of the many equations and unknowns and for the second case study the ode-solver had to be stopped an restarted because of too small step sizes.
- The symmetric projection method has a long simulation time because there is iteration, but the results stay close to the solution manifold.
- The last method is the state space transformation, which can be done global for the two used case studies. This is a very important fact, because therefore the resulting equations where the most simple.

For further research it would be interesting to study DAEs with a differential index unequal to three and analyse the use of the methods for such DAEs. Furthermore an improvement with respect to the implementation for the Pantelides-Algorithm would be to write an own solver.For the simulation of the orthogonal projection method it would be interesting to use another numerical integration method for investigating whether the speed is increasing too.

Another further study would be to test the state space transformation for DAEs where no global transformation can be used.

References

[1] Hairer E, Wanner G. *Solving Ordinary Differential Equations II.* Germany: Springer; 2002. 491 p.

[2] Kunkel P, Mehrmann V. *Differential-Algebraic Equations: Analysis and Numerical Solution.* Finland: European Mathematical Society; 2006. 385 p.

[3] Eich E, Hanke M. Regularization Methods for Constrained Mechanical Multibody Systems. *ZAMM - Journal of Applied Mathematics and Mechanics.* 1995; 75(10): 761-773. doi: 10.1002/zamm.19950751013

[4] Cellier FE. *The Structural Singularity Removal Algorithm by Pantelides*. http://www:inf:ethz:ch/personal/fcellier/Lect/MMPS/Ppt/mmps_6_engl:ppt, date: 10.02.2014.

[5] Cellier FE, Kofman E. *Continuous System Simulation.* USA: Springer; 2006. 658 p.

[6] Hairer E. Geometric Integration of Ordinary Differential Equations on Manifolds. *BIT Numerical Mathematics.* 2001; 41(5): 996-1007. doi: 10.1023/A:1021989212020

[7] Hairer E. Symmetric Projection Methods for Ordinary Differential Equations on Manifolds. *BIT Numerical Mathematics*. 2000; 40(4): 726-734. doi: 10.1023/A:1022344502818

[8] Stolze J. *Einführung in die nichtlineare Dynamik*. http://t1:physik:tu-dortmund:de/stolze/teaching/TNF/tnf_chaos:pdf, date: 24.06.2014.

Implementation of Quantized State Systems in MATLAB/Simulink

Patrick Grabher[1], Matthias Rößler[2*], Bernhard Heinzl[3]

[1]Inst. of Analysis and Scientific Computing, Vienna University of Technology, Wiedner Haupstraße 8-10, 1040 Vienna, Austria; * *matthias.roessler@tuwien.ac.at*

[2]dwh Simulation Services, Neustiftg. 57-59, A-1070 Wien, Austria

[3]Inst. of Computer Aided Automation, Vienna University of Technology, Treitlstraße 3, 1040 Vienna, Austria.

Simulation Notes Europe SNE 24(3-4), 2014, 185 - 190
DOI: 10.11128/sne.24.tn.10269
Received: June 12, 2014; Revised September 25, 2014;
Accepted: October 25, 2014;

Abstract. Common practice in the simulation of continuous systems is to discretize the time in order to obtain a numerical solution. The Quantized State System (QSS) approach makes it possible that the discretisation is applied to the state variables, instead of the time range. In other words continuous systems can be simulated event-based with the QSS method. It also effects a new orientation and leads among other things to very efficient state event detection. Ernesto Kofman and Sergio Junco presented the QSS method in the DEVS formalism [1]. This paper describes how the implementation of the QSS method in Simulink/SimEvents works and which restrictions still exist.

1 Introduction

The content of this chapter is taken from [1]. Considering a model which is described by the equation

$$\dot{\boldsymbol{x}}(t) = f\big(\boldsymbol{x}(t), \boldsymbol{u}(t)\big), \tag{1}$$

the associated QSS system is

$$\dot{\boldsymbol{x}}(t) = f\big(\boldsymbol{q}(t), \boldsymbol{u}(t)\big). \tag{2}$$

The variable $\boldsymbol{q}(t)$ is the quantized state of $\boldsymbol{x}(t)$ and will be obtained by a quantized hysteretic function and is defined by:

Let $Q = \{Q_0, Q_1, Q_2, \dots, Q_r\}$ be a set of real numbers and $Q_{i-1} \le Q_i$ for $1 \le i \le r$. Let $x \in \Omega$ be a continues trajectory, where $x: \mathbb{R} \to \mathbb{R}$. Let $b: \Omega \to \Omega$ be a mapping and $q = b(x)$ the trajectory which satisfies

$$q(t) = \begin{cases} Q_m, & if\ t = t_0 \\ Q_{i+1}, if\ x(t) = Q_{i+1} \wedge \mathrm{q}(\mathrm{t}^-) = \mathrm{Q_i} \wedge \mathrm{i} < r \\ Q_{i-1}, if\ x(t) = Q_i - \varepsilon \wedge \mathrm{q}(\mathrm{t}^-) = \mathrm{Q_i} \wedge \mathrm{i} > 0 \\ q(t^-), & else \end{cases} \tag{3}$$

and

$$m = \begin{cases} 0, & if\ x(t_0) < Q_0 \\ r, & if\ x(t_0) \ge Q_r \\ j, & if\ Q_j \le x(t_0) < Q_{j+1} \end{cases}, \tag{4}$$

then b is the *Quantized Function with Hysteresis.* ε is called hysteresis width und describes a delay for some events. Generally you use a uniform quantum ΔQ like in Figure 1.

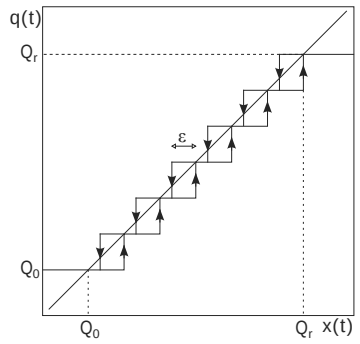


Figure 1: Quantized Function with Hysteresis.

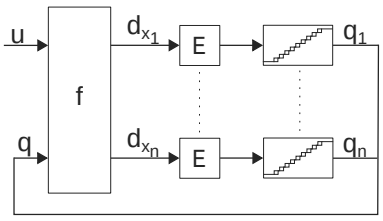


Figure 2: Block diagram of a Quantized State System, which consists of the static function f and the quantized intergrator with hysteresis.

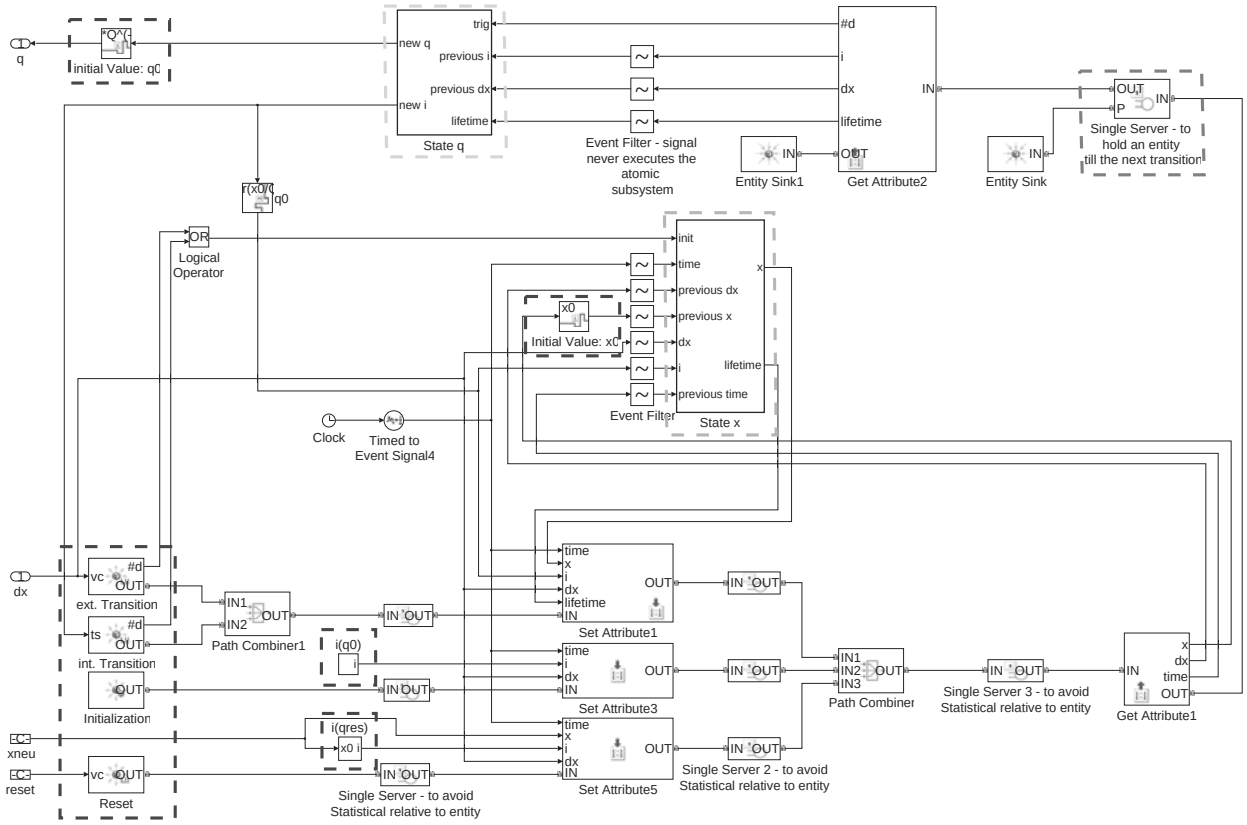


Figure 3: Implementation of the Quantized Integrator of QSS method in Simulink.

Figure 2 shows the block diagram of a QSS. Since it is a first order method, the signals $\boldsymbol{q}$ and $\boldsymbol{d_x}$ are always piecewise constant for every static function f (note that f is time-invariant). So it is sufficient to implement only the quantized integrator to obtain a legitimate QSS system.

1 Implementation of the QSS Method in Simulink

A large part of the implementation was built with elements from the SimEvents library. The general functionality of this integrator is: An event, like a change in the state derivative or in the quantized state, generates an entity. This new entity replaces the last entity. In this moment it describes the new state and specifies when the quantized state has to change.

1.1 Events and Entity Generators

Considering a quantized integrator, computations have to be performed only in moments of events. In the QSS method there are two types of events. The first type is the input event. It describes any kind of change in the state derivative. The other type is the output event, it is a change of the quantized state. In figure 3 one can see on the left side the quantized state q and the state derivative d_x as inputs for the QSS integrator.

Generally each event in the QSS system generates an entity in at least one integrator. Additional entities are generated in the beginning of the simulation or if the integrator is reset. The single purpose of these kinds of entities is to initialize the system with the according initial values, so the lifetime of these entities are zero seconds. For each case there exists an entity generator (see the bottom left corner in figure 3). In some models an input event occurs simultaneously with an output event. For this case the event priority in the entity generator for the input events is set to 1000 in contrast to 5, so that input events are replaced. It should be mentioned, that it does not matter which event is preferred because both entities get the same attributes.

1.2 Attributes and computations

The generated entities also get some attributes to describe the exact state of the integrator. Let S_n be an entity (with sequencing entity S_{n+1}). Its related attributes *time of arising* t_n, *exact state* x_n, *index* $i_n \in \mathbb{Z}$ of the quantized state $q_n = i_n \cdot \Delta Q$, *state derivative* $d_{x,n}$ and the *lifetime* σ_n are computed and stored in the moment of the creation of S_n. The initial values x_0, i_0 and q_0 (see the blue outlined blocks in Figure 3) are given by

$$\begin{aligned} x_0 &= x(0), \\ i_0 &= \left\lfloor \frac{x(0)}{\Delta Q} \right\rfloor, \\ q_0 &= i_0 \cdot \Delta Q. \end{aligned} \tag{5}$$

The single purpose of the usage i_n, which is an Int32 data type, is that simple rounding errors can be avoided in q_n. The state x_n (see the green outlined block in Figure 3) is obtained by

$$x_n = \begin{cases} x_0, \ if\ n = 0 \\ x_{n-1} + d_{x,n-1} \cdot (t_n - t_{n-1}), \ else \end{cases}. \tag{6}$$

How long this integrator holds his state q_n depends on σ_n, which is given by

$$\sigma_n = \begin{cases} 0, \ if\ n = 0 \\ \frac{(q_{n-1}+\Delta Q)-x_n}{d_{x,n}}, \ if\ n > 0 \wedge d_{x,n} > 0 \\ \frac{x_n-(q_{n-1}-\varepsilon)}{|d_{x,n}|}, \ if\ n > 0 \wedge d_{x,n} < 0 \\ \infty, \ if\ n > 0 \wedge d_{x,n} = 0 \end{cases}. \tag{7}$$

After this attributes were assigned to S_n, the entity reaches the entity (single) server (see the red outlined block in Figure 3). It stays for σ_n, if meanwhile no other entity replaces it.

If S_n reaches σ_n in this entity server and if $\sigma_n \neq 0$ (i.e no initialization) it generates an output event and the quantized state changes its values by

$$q_{n+1} = \begin{cases} q_n + \Delta Q, \ if\ d_{x,n} > 0 \\ q_n - \Delta Q, \ if\ d_{x,n} < 0 \end{cases}. \tag{8}$$

In the other cases q_{n+1} stays unchanged with $q_{n+1} = q_n$ (see the yellow outlined block in Figure 3).

1.3 Interaction between event and time-based blocks

The compatibility between different blocks in Simulink is an important issue. In principle, the integrator is built with event-based blocks, but for the attributes q_n and x_n computations are necessary. Therefore in Simulink exist the possibilities to use the *Attribute Function* (it is an embedded MATLAB-function), the *Gateway-blocks* (does not work for every signal and produces sometimes algebraic loops) or time-based math operations, which need to be in an *Atomic Subsystem.*

In the end the third possibility is the best in view of reliability and efficiency at the moment. Figure 3 also shows, that all atomic subsystems are executed by changes of only one signal, on that account all other input signals use an event filter.

1.4 Comments

In Figure 3 you can see some additional entity servers, that haven't been described in the previous chapter. They do not cause any delay for the entities. These blocks ensure that each entity receives its corresponding attributes. If they would not exist, the entities could receive attributes, which belong to the last entity.

The integrator allows some types of signals and changes to be reset. You can find these properties in the entity generator which is responsible for the reset (see Figure 3 in the bottom left corner).

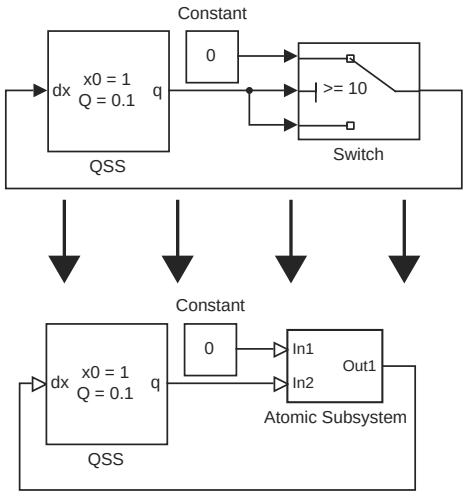


Figure 4: Incompatible time-based blocks have to be placed inside an Atomic Subsystem.

If you create a model sometimes time-based blocks are not compatible with the integrator, because they don't work with the event-based signal type which is coming from the integrator. In this case you have to put the blocks which provoke an error into an atomic subsystem (like Figure 4).

2 Case Studies and Results

In this chapter the properties of the QSS method will be presented. First it should be mentioned, that the solution of QSS method usually doesn't converge in the equilibrium point and so the state is finally oscillating. Also the behaviour with stiff equations is different. We know from time discrete systems that their solution diverges from the analytical solution if they are not stable.

This could not happen with the QSS method, because the error is always bounded with the quantum, although it is fully explicit. But how does the rate of stiff systems affect the solution of the QSS method? The answer is, the more stiff the equation is, the faster is the oscillation of the solution.

2.1 Simple equation

Let following initial value problem be given:

$$\begin{aligned} \dot{x}(t) &= -100 \cdot (x(t) - 0.1), \\ x(0) &= 1\,. \end{aligned} \tag{9}$$

Already this example shows that it is sufficient to take a small step size h for an explicit solver like the Forward Euler (FE) method. Figure 5 shows what happens, if h is chosen too large with $h = 0.025$, namely the solution of the FE diverges from the analytic solution.

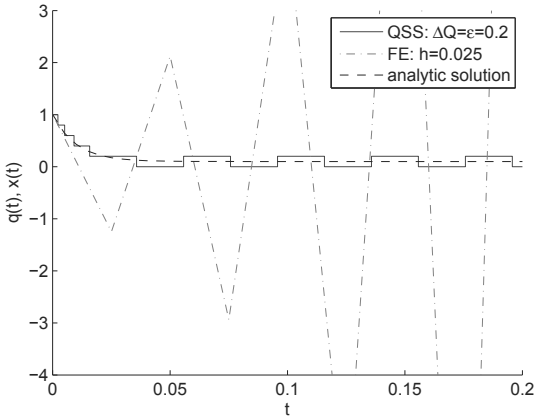


Figure 5: Numerical solutions from the different types of explicit solvers.

This kind of problem does not appear with the QSS method. Usually the quantized state is not able to attain the equilibrium point, so the solution oscillates around it, but it is always bounded with the quantum ΔQ. Figure 5 shows this behavior of the QSS method with $\Delta Q = 0.2$.

2.2 Stiff equation

Following equation represents a stiff problem:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t), \\ \dot{x}_2(t) &= -1000 \cdot (x_1(t) + x_2(t)) + 9500, \\ x_1(0) &= 0, \qquad x_2(0) = 10. \end{aligned} \tag{10}$$

To solve this system a quantum $\Delta Q = 1$ is chosen. Figure 6 shows the behaviour of the QSS method, which is overwhelmed with this situation. The second state variable has 2500 changes in this simulation.

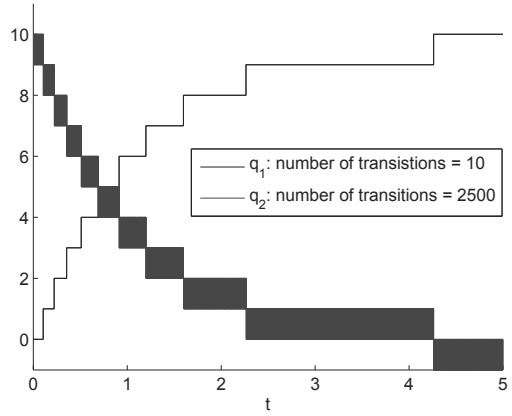


Figure 6: Stiff equations effect a fast oscillation in the state.

The conclusion is that the QSS method is always stable, but requires potentially many computations (see IQSS [2], if you are interested to avoid this problem).

Let us now consider other properties of the QSS method. The event-based system has another two advantages over common time discrete solvers. Firstly, events can be directly detected by the QSS system, so especially for models with a large number of events this method is more efficient than time discrete solvers. The second point is the asynchronous procedure, which is not possible for time discrete solvers. This could be very interesting in large models. The following examples show these capabilities.

2.3 H-Bridge

This model comes from drive engineering and is used to control a DC motor with frequently toggled (with a PWM signal) voltage supply and shows the advantages caused by the asynchronous procedure of the QSS method. Basically this model [3] is described by an electrical part

$$u_S = u_{EMF} + u_R + u_L, \tag{11}$$

and a mechanical part

$$I \cdot \ddot{\omega} = M - k \cdot \dot{\omega}. \tag{12}$$

In the first equation u_S describes the DC *voltage supply*, which is controlled by a PWMsignal, u_{EMF} the *voltage of the electromotive force*, u_R the *voltage at the resistance* and u_L the *voltage of the inductance*. In the mechanical part (12) I is the *moment of inertia* of the rotor, M the *motor torque*, k the *motor viscous friction constant* and ω stands for the *angle* ($\dot{\omega}$ *angular velocity*, $\ddot{\omega}$ *angular acceleration*).

A PWM signal with a frequency of 1kHz controls the H-Bridge for the acceleration and braking procedure. The great advantage of the QSS is that the most computations, which depends on the PWMsignal, must be made in the electronic equation (11) only, whereas the mechanical part (12) depends on a change in the electric current i (input event) or $\dot{\omega}$ (output event). In contrast a time discrete solver, which works synchronously, has to update the whole system if the PWMsignal produces an event.

Figure 7 shows a simulation, where the motor is accelerated the first 8 seconds, then it runs free for 4 seconds (no PWM signal) and at last it brakes.

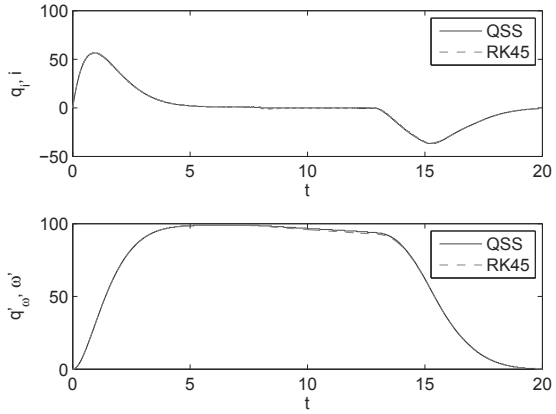


Figure 7: Simulation results of the DC motor.

For the quantum ΔQ the QSS method has 394 output events at the first integrator ($\frac{d}{dx} i \to i$) and 397 at the second ($\ddot{\omega} \to \dot{\omega}$), i.e. that the second one has overall only 791 events (output and input events). In contrast the RK45 uses 29082 grid points for this simulation. Although the motor was running free in the interval (8,12) (i.e. no PWM and $i = 0$ A) the RK45 solver has unnecessary many computations, caused by missing the event $i = 0$ A, see Figure 8. The QSS method does not have this problem and manages the running free operation perfectly with only 7 events in the interval (8,12).

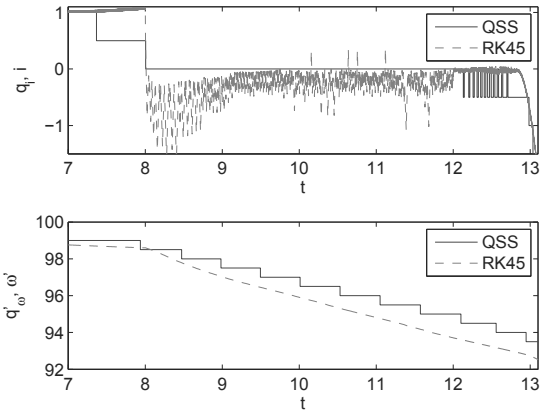


Figure 8: Simulation results of the DC motor in the interval (7,13.1).

2.4 Bouncing ball

Let us now consider a model of a bouncing ball on the floor [4]. The attention here should be pointed at the events when the ball touches the ground and especially what happens if the ball's *vertical speed* $v(t)$ goes to 0 m/s. The fall condition is described by

$$\begin{aligned} \dot{v}(t) &= -g, \\ \dot{x}(t) &= v(t). \end{aligned} \tag{13}$$

In (13) g describes the *earth gravity constant*. If the height $x(t) = 0$ the ball touches the ground and v has to be reset with $v(t) = k \cdot v_{old}$, with the *coefficient of restitution* $k \epsilon (-1,0)$. Normally (in time discrete systems) also the height $x(t)$ will be reset in the moment of an event, which won't be necessary here.

Figure 9 shows three numerical solutions. One can see that is important which quantum is chosen.

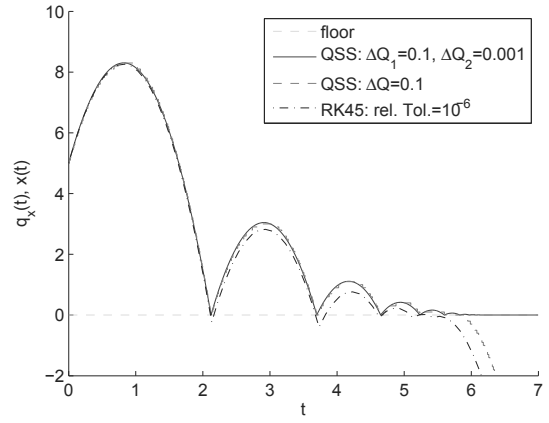


Figure 9: Simulation results of the Bouncing Ball.

Although the QSS method recognizes every event, some parameters effect that the ball does not come up from the floor at a definite time so that it has only the fall condition from this moment. Figure 10 shows the moment of the QSS ($\Delta Q = 0.1$), where the ball does not come up from the floor any more.

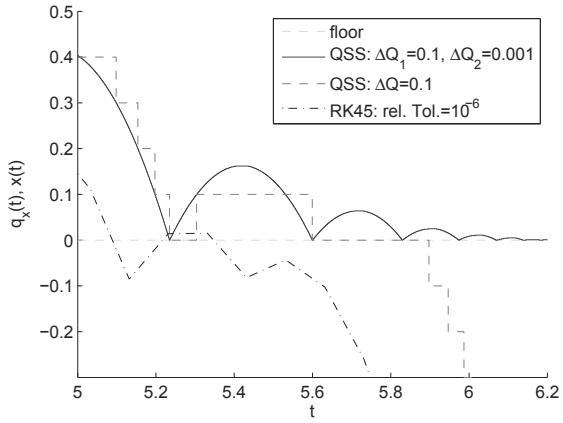


Figure 10: Simulation results of the Bouncing Ball in the interval (5,6.2).

Nevertheless with the QSS method it is also possible to reach the rolling condition. Figure 11 shows that with well chosen parameters ($\Delta Q_1 = 0.1, \Delta Q_2 = 0.001$) a periodic oscillation on the floor can be obtained.

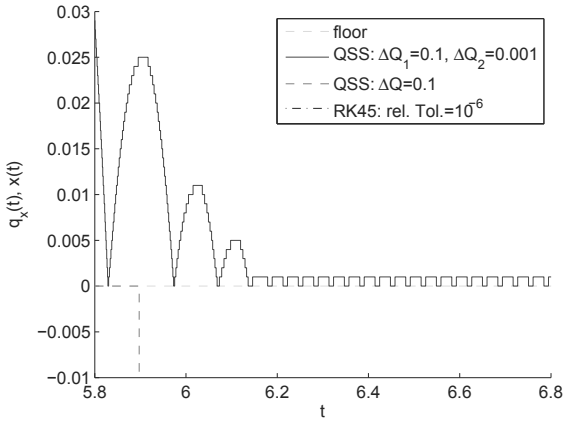


Figure 11: Simulation results of the Bouncing Ball in the interval (5.8,6.8).

Such a rolling condition, which is computed with the QSS method, is inconceivable for time discrete solvers unless they obtain help from additional approaches, like zero crossing detection or a reset of the height $x(t)$ in the moment of every event to handle this *Zeno Phenomenon*.

References

[1] Kofman E, Junco S. Quantized-State Systems: A DEVs Approach for Continuous System Simulation. *Transactions of the Society for Computer Simulation International - Recent advances in DEVS Methodology--part I*. 2001; 18(3): 123-132. ISSN: 0740-6797.

[2] Cellier FE, Kofman E. *Continuous System Simulation*. New York: Springer; 2006. 643 p. ISBN: 978-0-387-26102-7.

[3] *Control Tutorials for MATLAB and Simulink*. University of Michigan. Bill Messner, Dawn Tilbury. cited 2014 Sep 26. Available from: http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SimulinkModeling

[4] Heinzl B, Rößler M, Körner A, Zauner G, Ecker H, Breitenecker F. BCP - A Benchmark for Teaching Structural Dynamical Systems. In Breitenecker F, Troch I, editors. Mathematical Modelling. *MATHMOD 2012 - 7th Vienna Conference on Mathematical Modelling;* 2012 Feb; Vienna University of Technology. Zürich: International Federation of Automatic Control. 896-901. ISBN: 978-3-902823-23-6.

SNE Simulation News

EUROSIM Data and Quick Info

EUROSIM 2016

9th EUROSIM Congress on Modelling and Simulation

City of Oulu, Finland, September 16-20, 2016

www.eurosim.info

Contents

Simulation Notes Europe SNE is the official membership journal of EUROSIM and distributed / available to members of the EUROSIM Societies as part of the membership benefits. **SNE** is published in a printed version (Print ISSN 2305-9974) and in an online version (Online ISSN 2306-0271). With **Online SNE** the publisher **ARGESIM** follows the **Open Access** strategy for basic **SNE** contributions. Since 2012 **Online SNE** contributions are identified by DOI 10.11128/sne.xx.nnnnn. for better web availability and indexing.

Print SNE, high-resolution **Online SNE**, and additional **SNE** contributions are available via membership in a **EUROSIM** society.

This *EUROSIM Data & Quick Info* compiles data from EUROSIM societies and groups: addresses, weblinks, officers of societies with function and email, to be published regularly in **SNE** issues.

If you have any information, announcement, etc. you want to see published, please contact a member of the editorial board in your country or the editorial office. For scientific publications, please contact the EiC.

EUROSIM
Federation of European Simulation Societies

General Information. **EUROSIM**, the Federation of European Simulation Societies, was set up in 1989. The purpose of **EUROSIM** is to provide a European forum for simulation societies and groups to promote advancement of modelling and simulation in industry, research, and development. → *www.eurosim.info*

Member Societies. **EUROSIM** members may be national simulation societies and regional or international societies and groups dealing with modelling and simulation. At present **EUROSIM** has fourteen *Full Members* and three *Observer Members*:

ASIM	Arbeitsgemeinschaft Simulation *Austria, Germany, Switzerland*
CEA-SMSG	Spanish Modelling and Simulation Group *Spain*
CROSSIM	Croatian Society for Simulation Modeling *Croatia*
CSSS	Czech and Slovak Simulation Society *Czech Republic, Slovak Republic*
DBSS	Dutch Benelux Simulation Society *Belgium, Netherlands*
FRANCOSIM	Société Francophone de Simulation *Belgium, France*
HSS	Hungarian Simulation Society *Hungary*
ISCS	Italian Society for Computer Simulation *Italy*
LIOPHANT	LIOPHANT Simulation Club *Italy & International, Observer Member*
LSS	Latvian Simulation Society *Latvia*
PSCS	Polish Society for Computer Simulation *Poland*
SIMS	Simulation Society of Scandinavia *Denmark, Finland, Norway, Sweden*
SLOSIM	Slovenian Simulation Society *Slovenia*
UKSIM	United Kingdom Simulation Society *UK, Ireland*
KA-SIM	Romanian Society for Modelling and Simulation, *Romania, Observer Member*
ROMSIM	Romanian Society for Modelling and Simulation, *Romania, Observer Member*
RNSS	Russian National Simulation Society *Russian Federation, Observer Member*

EUROSIM Board / Officers. EUROSIM is governed by a board consisting of one representative of each member society, president and past president, and representatives for SNE Simulation notes Europe. The President is nominated by the society organising the next **EUROSIM** Congress. Secretary and Treasurer are elected out of members of the Board.

President	Esko Juuso (SIMS) *esko.juuso@oulu.fi*
Past President	Khalid Al.Begain (UKSIM) *kbegain@glam.ac.uk*
Secretary	Borut Zupančič (SLOSIM) *borut.zupancic@fe.uni-lj.si*
Treasurer	Felix Breitenecker (ASIM) *felix.breitenecker@tuwien.ac.at*
SNE Repres.	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*

SNE – Simulation Notes Europe. **SNE** is a scientific journal with reviewed contributions as well as a membership newsletter for **EUROSIM** with information from the societies in the *News Section*. **EUROSIM** societies are offered to distribute to their members the journal **SNE** as official membership journal. **SNE** Publishers are **EUROSIM**, ARGESIM and ASIM.

Editor-in-chief	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*

→ *www.sne-journal.org,*

✉ *office@sne-journal.org*

EUROSIM Congress. **EUROSIM** is running the triennial conference series **EUROSIM** Congress. The congress is organised by one of the **EUROSIM** societies.

EUROSIM 2016 will be organised by SIMS in Oulu, Finland, September 16-20, 2016.

Chairs / Team EUROSIM 2016

Esko Juuso EUROSIM President, *esko.juuso@oulu.fi*
Erik Dahlquist SIMS President, *erik.dahlquist@mdh.se*
Kauko Leiviskä EUROSIM 2016 Chair,
kauko.leiviska@oulu.fi

→ *www.eurosim.info*

✉ *office@automaatioseura.fi*

EUROSIM Member Societies

ASIM
German Simulation Society
Arbeitsgemeinschaft Simulation

ASIM (Arbeitsgemeinschaft Simulation) is the association for simulation in the German speaking area, servicing mainly Germany, Switzerland and Austria. ASIM was founded in 1981 and has now about 700 individual members, and 30 institutional or industrial members.

→ *www.asim-gi.org* with members' area

info@asim-gi.org, admin@asim-gi.org

✉ ASIM – Inst. f. Analysis and Scientific Computing
Vienna University of Technology
Wiedner Hauptstraße 8-10, 1040 Vienna, Austria

ASIM Officers	
President	Felix Breitenecker *felix.breitenecker@tuwien.ac.at*
Vice presidents	Sigrid Wenzel, *s.wenzel@uni-kassel.de*
	T. Pawletta, *pawel@mb.hs-wismar.de*
Secretary	Ch. Deatcu, *christina.deatcu@hs-wismar.de*
Treasurer	Anna Mathe, *anna.mathe@tuwien.ac.at*
Membership Affairs	S. Wenzel, *s.wenzel@uni-kassel.de*
	W. Maurer, *werner.maurer@zhwin.ch*
	Ch. Deatcu, *christina.deatcu@hs-wismar.de*
	F. Breitenecker, *felix.breitenecker@tuwien.ac.at*
Universities / Research Inst.	S. Wenzel, *s.wenzel@uni-kassel.de*
	W. Wiechert, *W.Wiechert@fz-juelich.de*
	J. Haase, *Joachim.Haase@eas.iis.fraunhofer.de*
	Katharina Nöh, *k.noeh@fz-juelich.de*
Industry	S. Wenzel, *s.wenzel@uni-kassel.de*
	K. Panreck, *Klaus.Panreck@hella.com*
Conferences	Klaus Panreck *Klaus.Panreck@hella.com*
	J. Wittmann, *wittmann@htw-berlin.de*
Publications	Th. Pawletta, *pawel@mb.hs-wismar.de*
	Christina Deatcu, *christina.deatcu@hs-wismar.de*
	F. Breitenecker, *felix.breitenecker@tuwien.ac.at*
Repr. EUROSIM	F. Breitenecker, *felix.breitenecker@tuwien.ac.at*
	N. Popper, *niki.popper@drahtwarenhandlung.at*
Education / Teaching	A. Körner, *andreas.koerner@tuwien.ac.at*
	N. Popper, *niki.popper@drahtwarenhandlung.at*
	Katharina Nöh, *k.noeh@fz-juelich.de*
International Affairs	A. Körner, *andreas.koerner@tuwien.ac.at*
	O. Rose, *Oliver.Rose@tu-dresden.de*
Editorial Board SNE	T. Pawletta, *pawel@mb.hs-wismar.de*
	Ch. Deatcu, *christina.deatcu@hs-wismar.de*
Web EUROSIM	Anna Mathe, *anna.mathe@tuwien.ac.at*

Last data update December 2013

ASIM Working Committee. **ASIM**, part of GI - Gesellschaft für Informatik, is organised in Working Committees, dealing with applications and comprehensive subjects in modelling and simulation:

ASIM Working Committee	
GMMS	Methods in Modelling and Simulation Th. Pawletta, *pawel@mb.hs-wismar.de*
SUG	Simulation in Environmental Systems Wittmann, *wittmann@informatik.uni-hamburg.de*
STS	Simulation of Technical Systems H.T.Mammen, *Heinz-Theo.Mammen@hella.com*
SPL	Simulation in Production and Logistics Sigrid Wenzel, *s.wenzel@uni-kassel.de*
EDU	Simulation in Education/Education in Simulation N. Popper, *niki.popper@dwh.at* A. Körner, *andreas.koerner@tuwien.ac.at*
	Working Groups for Simulation in Business Administration, in Traffic Systems, for Standardisation, for Validation, etc.

CEA-SMSG – Spanish Modelling and Simulation Group

CEA is the Spanish Society on Automation and Control In order to improve the efficiency and to deep into the different fields of automation, the association is divided into thematic groups, one of them is named 'Modelling and Simulation', constituting the group.

→ *www.cea-ifac.es/wwwgrupos/simulacion*

→ *simulacion@cea-ifac.es*

✉ CEA-SMSG / María Jesús de la Fuente,
System Engineering and AutomaticControl department,
University of Valladolid,
Real de Burgos s/n., 47011 Valladolid, SPAIN

CAE - SMSG Officers	
President	M. À. Piera Eroles, *MiquelAngel.Piera@uab.es*
Vice president	Emilio Jiminez, *emilio.jiminez@unirioja.es*
Repr. EUROSIM	Emilio Jiminez, *emilio.jiminez@unirioja.es*
Edit. Board SNE	Emilio Jiminez, *emilio.jiminez@unirioja.es*
Web EUROSIM	Mercedes Peres, *mercedes.perez@unirioja.es*

Last data update December2013

CROSSIM – Croatian Society for Simulation Modelling

CROSSIM-*Croatian Society for Simulation Modelling* was founded in 1992 as a non-profit society with the goal to promote knowledge and use of simulation methods and techniques and development of education. CROSSIM is a full member of **EUROSIM** since 1997.

→ *www.eurosim.info*

vdusak@foi.hr

CROSSIM / Vesna Dušak
Faculty of Organization and
Informatics Varaždin, University of Zagreb
Pavlinska 2, HR-42000 Varaždin, Croatia

CROSSIM Officers	
President	Vesna Dušak, *vdusak@foi.hr*
Vice president	Jadranka Božikov, *jbozikov@snz.hr*
Secretary	Vesna Bosilj-Vukšić, *vbosilj@efzg.hr*
Executive board members	Vlatko Čerić, *vceric@efzg.hr*
	Tarzan Legović, *legovic@irb.hr*
Repr. EUROSIM	Jadranka Božikov, *jbozikov@snz.hr*
Edit. Board SNE	Vesna Dušak, *vdusak@foi.hr*
Web EUROSIM	Jadranka Bozikov, *jbozikov@snz.hr*

Last data update December2012

CSSS – Czech and Slovak Simulation Society

CSSS -The *Czech and Slovak Simulation Society* has about 150 members working in Czech and Slovak national scientific and technical societies (*Czech Society for Applied Cybernetics and Informatics*, *Slovak Society for Applied Cybernetics and Informatics*). The main objectives of the society are: development of education and training in the field of modelling and simulation, organising professional workshops and conferences, disseminating information about modelling and simulation activities in Europe. Since 1992, CSSS is full member of **EUROSIM**.

→ *www.fit.vutbr.cz/CSSS*

snorek@fel.cvut.cz

CSSS / Miroslav Šnorek, CTU Prague
FEE, Dept. Computer Science and Engineering,
Karlovo nam. 13, 121 35 Praha 2, Czech Republic

CSSS Officers	
President	Miroslav Šnorek, *snorek@fel.cvut.cz*
Vice president	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Treasurer	Evžen Kindler, *ekindler@centrum.cz*
Scientific Secr.	A. Kavička, *Antonin.Kavicka@upce.cz*
Repr. EUROSIM	Miroslav Šnorek, *snorek@fel.cvut.cz*
Deputy	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Edit. Board SNE	Mikuláš Alexík, *alexik@frtk.fri.utc.sk*
Web EUROSIM	Petr Peringer, peringer@fit.vutbr.cz

Last data update December2012

DBSS – Dutch Benelux Simulation Society

The Dutch Benelux Simulation Society (DBSS) was founded in July 1986 in order to create an organisation of simulation professionals within the Dutch language area. DBSS has actively promoted creation of similar organisations in other language areas. DBSS is a member of **EUROSIM** and works in close cooperation with its members and with affiliated societies.

→ www.eurosim.info

a.w.heemink@its.tudelft.nl

DBSS / A. W. Heemink
Delft University of Technology, ITS - twi,
Mekelweg 4, 2628 CD Delft, The Netherlands

DBSS Officers	
President	A. Heemink, *a.w.heemink@its.tudelft.nl*
Vice president	W. Smit, *smitnet@wxs.nl*
Treasurer	W. Smit, *smitnet@wxs.nl*
Secretary	W. Smit, *smitnet@wxs.nl*
Repr. EUROSIM	A. Heemink, *a.w.heemink@its.tudelft.nl*
Deputy	W. Smit, *smitnet@wxs.nl*
Edit. Board SNE	A. Heemink, *a.w.heemink@its.tudelft.nl*

Last data update April 2006

FRANCOSIM – Société Francophone de Simulation

FRANCOSIM was founded in 1991 and aims to the promotion of simulation and research, in industry and academic fields. Francosim operates two poles.

- Pole Modelling and simulation of discrete event systems. Pole Contact: *Henri Pierreval, pierreva@imfa.fr*
- Pole Modelling and simulation of continuous systems. Pole Contact: *Yskandar Hamam*, *y.hamam@esiee.fr*

→ *www.eurosim.info*

y.hamam@esiee.fr

FRANCOSIM / Yskandar Hamam
Groupe ESIEE, Cité Descartes,
BP 99, 2 Bd. Blaise Pascal,
93162 Noisy le Grand CEDEX, France

FRANCOSIM Officers	
President	Karim Djouani, *djouani@u-pec.fr*
Treasurer	François Rocaries, *f.rocaries@esiee.fr*
Repr. EUROSIM	Karim Djouani, *djouani@u-pec.fr*
Edit. Board SNE	Karim Djouani, *djouani@u-pec.fr*

Last data update December2012

HSS – Hungarian Simulation Society

The Hungarian Member Society of EUROSIM was established in 1981 as an association promoting the exchange of information within the community of people involved in research, development, application and education of simulation in Hungary and also contributing to the enhancement of exchanging information between the Hungarian simulation community and the simulation communities abroad. HSS deals with the organization of lectures, exhibitions, demonstrations, and conferences.

→ *www.eurosim.info*

javor@eik.bme.hu

HSS / András Jávor,
Budapest Univ. of Technology and Economics,
Sztoczek u. 4, 1111 Budapest, Hungary

HSS Officers	
President	András Jávor, *javor@eik.bme.hu*
Vice president	Gábor Szűcs, *szucs@itm.bme.hu*
Secretary	Ágnes Vigh, *vigh@itm.bme.hu*
Repr. EUROSIM	András Jávor, *javor@eik.bme.hu*
Deputy	Gábor Szűcs, *szucs@itm.bme.hu*
Edit. Board SNE	András Jávor, *javor@eik.bme.hu*
Web EUROSIM	Gábor Szűcs, *szucs@itm.bme.hu*

Last data update March 2008

ISCS – Italian Society for Computer Simulation

The Italian Society for Computer Simulation (ISCS) is a scientific non-profit association of members from industry, university, education and several public and research institutions with common interest in all fields of computer simulation.

→ *www.eurosim.info*

Mario.savastano@uniina.at

ISCS / Mario Savastano,
c/o CNR - IRSIP,
Via Claudio 21, 80125 Napoli, Italy

ISCS Officers	
President	M. Savastano, *mario.savastano@unina.it*
Vice president	F. Maceri, *Franco.Maceri@uniroma2.it*
Repr. EUROSIM	F. Maceri, *Franco.Maceri@uniroma2.it*
Secretary	Paola Provenzano, *paola.provenzano@uniroma2.it*
Edit. Board SNE	M. Savastano, *mario.savastano@unina.it*

Last data update December2010

LIOPHANT Simulation

Liophant Simulation is a non-profit association born in order to be a trait-d'union among simulation developers and users; Liophant is devoted to promote and diffuse the simulation techniques and methodologies; the Association promotes exchange of students, sabbatical years, organization of International Conferences, organization of courses and stages in companies to apply the simulation to real problems.

→ *www.liophant.org*

info@liophant.org

LIOPHANT Simulation, c/o Agostino G. Bruzzone,
DIME, University of Genoa, Polo Savonese,
via Molinero 1, 17100 Savona (SV), Italy

LIOPHANT Officers	
President	A.G. Bruzzone, *agostino@itim.unige.it*
Director	E. Bocca, *enrico.bocca@liophant.org*
Secretary	A. Devoti, *devoti.a@iveco.com*
Treasurer	Marina Masseimassei@itim.unige.it
Repr. EUROSIM	A.G. Bruzzone, *agostino@itim.unige.it*
Deputy	F. Longo, *f.longo@unical.it*
Edit. Board SNE	F. Longo, *f.longo@unical.it*
Web EUROSIM	F. Longo, *f.longo@unical.it*

Last data update December2013

LSS – Latvian Simulation Society

The Latvian Simulation Society (LSS) has been founded in 1990 as the first professional simulation organisation in the field of Modelling and simulation in the post-Soviet area. Its members represent the main simulation centres in Latvia, including both academic and industrial sectors.

→ *briedis.itl.rtu.lv/imb/*

merkur@itl.rtu.lv

LSS / Yuri Merkuryev, Dept. of Modelling and Simulation Riga Technical University Kalku street 1, Riga, LV-1658, LATVIA

LSS Officers	
President	Yuri Merkuryev, *merkur@itl.rtu.lv*
Secretary	Artis Teilans, *Artis.Teilans@exigenservices.com*
Repr. EUROSIM	Yuri Merkuryev, *merkur@itl.rtu.lv*
Deputy	Artis Teilans, *Artis.Teilans@exigenservices.com*
Edit. Board SNE	Yuri Merkuryev, *merkur@itl.rtu.lv*
Web EUROSIM	Oksana Sosho, *oksana@itl.rtu.lv*

Last data update December2013

PSCS – Polish Society for Computer Simulation

PSCS was founded in 1993 in Warsaw. PSCS is a scientific, non-profit association of members from universities, research institutes and industry in Poland with common interests in variety of methods of computer simulations and its applications. At present PSCS counts 257 members.

→ *www.ptsk.man.bialystok.pl*

leon@ibib.waw.pl

PSCS / Leon Bobrowski, c/o IBIB PAN, ul. Trojdena 4 (p.416), 02-109 Warszawa, Poland

PSCS Officers	
President	Leon Bobrowski, *leon@ibib.waw.pl*
Vice president	Tadeusz Nowicki, *Tadeusz.Nowicki@wat.edu.pl*
Treasurer	Z. Sosnowski, *zenon@ii.pb.bialystok.pl*
Secretary	Zdzislaw Galkowski, *Zdzislaw.Galkowski@simr.pw.edu.pl*
Repr. EUROSIM	Leon Bobrowski, *leon@ibib.waw.pl*
Deputy	Tadeusz Nowicki, *tadeusz.nowicki@wat.edu.pl*
Edit. Board SNE	Zenon Sosnowski, z.sosnowski@pb.ed.pl
Web EUROSIM	Magdalena Topczewska *m.topczewska@pb.edu.pl*

Last data update December2013

SIMS – Scandinavian Simulation Society

SIMS is the *Scandinavian Simulation Society* with members from the four Nordic countries Denmark, Finland, Norway and Sweden. The SIMS history goes back to 1959. SIMS practical matters are taken care of by the SIMS board consisting of two representatives from each Nordic country (Iceland one board member).

SIMS Structure. SIMS is organised as federation of regional societies. There are FinSim (Finnish Simulation Forum), DKSIM (Dansk Simuleringsforening) and NFA (Norsk Forening for Automatisering).

→ *www.scansims.org*

esko.juuso@oulu.fi

SIMS / Esko Juuso, Department of Process and Environmental Engineering, 90014 Univ.Oulu, Finland

SIMS Officers	
President	Esko Juuso, *esko.juuso@oulu.fi*
Vice president	Erik Dahlquist, *erik.dahlquist@mdh.se*
Treasurer	Vadim Engelson, *vadim.engelson@mathcore.com*
Repr. EUROSIM	Esko Juuso, *esko.juuso@oulu.fi*
Edit. Board SNE	Esko Juuso, *esko.juuso@oulu.fi*
Web EUROSIM	Vadim Engelson, *vadim.engelson@mathcore.com*

Last data update December2013

SLOSIM – Slovenian Society for Simulation and Modelling

SLOSIM - Slovenian Society for Simulation and Modelling was established in 1994 and became the full member of **EUROSIM** in 1996. Currently it has 69 members from both slovenian universities, institutes, and industry. It promotes modelling and simulation approaches to problem solving in industrial as well as in academic environments by establishing communication and cooperation among corresponding teams.

→ www.slosim.si

slosim@fe.uni-lj.si

SLOSIM / Rihard Karba, Faculty of Electrical Engineering, University of Ljubljana, Tržaška 25, 1000 Ljubljana, Slovenia

SLOSIM Officers	
President	Vito Logar, *vito.logar@fe.uni-lj.si*
Vice president	Božidar Šarler, *bozidar.sarler@ung.si*
Secretary	Aleš Belič, *ales.belic@sandoz.com*
Treasurer	Milan Simčič, *milan.simcic@fe.uni-lj.si*
Repr. EUROSIM	B. Zupančič, *borut.zupancic@fe.uni-lj.si*
Deputy	Vito Logar, *vito.logar@fe.uni-lj.si*
Edit. Board SNE	Rihard Karba, *rihard.karba@fe.uni-lj.si*
Web EUROSIM	Vito Logar, *vito.logar@fe.uni-lj.si*

Last data update December2013

UKSIM - United Kingdom Simulation Society

UKSIM has more than 100 members throughout the UK from universities and industry. It is active in all areas of simulation and it holds a biennial conference as well as regular meetings and workshops.

→ *www.uksim.org.uk*

david.al-dabass@ntu.ac.uk

✉ UKSIM / Prof. David Al-Dabass
Computing & Informatics,
Nottingham Trent University
Clifton lane, Nottingham, NG11 8NS
United Kingdom

UKSIM Officers	
President	David Al-Dabass, *david.al-dabass@ntu.ac.uk*
Vice president	A. Orsoni, *A.Orsoni@kingston.ac.uk*
Secretary	Richard Cant, *richard.cant@ntu.ac.uk*
Treasurer	A. Orsoni, *A.Orsoni@kingston.ac.uk*
Membership chair	K. Al-Begain, *kbegain@glam.ac.uk*
Univ. liaison chair	R. Cheng, *rchc@maths.soton.ac.uk*
Repr. EUROSIM	Richard Zobel, *r.zobel@ntlworld.com*
Deputy	K. Al-Begain, *kbegain@glam.ac.uk*
Edit. Board SNE	Richard Zobel, *r.zobel@ntlworld.com*

Last data update December2013

EUROSIM OBSERVER MEMBERS

KA-SIM Kosovo Simulation Society

Kosova Association for Modeling and Simulation (KA – SIM, founded in 2009), is part of Kosova Association of Control, Automation and Systems Engineering (KA – CASE). KA – CASE was registered in 2006 as non Profit Organization and since 2009 is National Member of IFAC – International Federation of Automatic Control. KA-SIM joined EUROSIM as Observer Member in 2011.

KA-SIM has about 50 members, and is organizing the international conference series International Conference in Business, Technology and Innovation, in November, in Durrhes, Albania, an IFAC Simulation worksh ops in Pristina.

→ *www.ubt-uni.net/ka-case*

ehajrizi@ubt-uni.net

✉ MOD&SIM KA-CASE
Att. Dr. Edmond Hajrizi
Univ. for Business and Technology (UBT)
Lagjja Kalabria p.n., 10000 Prishtina, Kosovo

KA-SIM Officers	
President	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Vice president	Muzafer Shala, *info@ka-sim.com*
Secretary	Lulzim Beqiri, *info@ka-sim.com*
Treasurer	Selman Berisha, *info@ka-sim.com*
Repr. EUROSIM	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Deputy	Muzafer Shala, *info@ka-sim.com*
Edit. Board SNE	Edmond Hajrizi, *ehajrizi@ubt-uni.net*
Web EUROSIM	Betim Gashi, *info@ka-sim.com*

Last data update December2013

ROMSIM – Romanian Modelling and Simulation Society

ROMSIM has been founded in 1990 as a non-profit society, devoted to theoretical and applied aspects of modelling and simulation of systems. ROMSIM currently has about 100 members from Romania and Moldavia.

→ *www.ici.ro/romsim/*

sflorin@ici.ro

✉ ROMSIM / Florin Stanciulescu,
National Institute for Research in Informatics, Averescu Av. 8 – 10, 71316 Bucharest, Romania

ROMSIM Officers	
President	Florin Stanciulescu, *sflorin@ici.ro*
Vice president	Florin Hartescu, *flory@ici.ro* Marius Radulescu, *mradulescu@ici.ro*
Repr. EUROSIM	Florin Stanciulescu, *sflorin@ici.ro*
Deputy	Marius Radulescu, *mradulescu@ici.ro*
Edit. Board SNE	Florin Stanciulescu, *sflorin@ici.ro*
Web EUROSIM	Zoe Radulescu, *radulescu@ici.ro*

Last data update December2012

RNSS – Russian Simulation Society

NSS - The Russian National Simulation Society (Национальное Общество Имитационного Моделирования – НОИМ) was officially registered in Russian Federation on February 11, 2011. In February 2012 NSS has been accepted as an observer member of **EUROSIM**.

→ *www.simulation.su*

yusupov@iias.spb.su

RNSS / R. M. Yusupov,
St. Petersburg Institute of Informatics and Automation RAS, 199178, St. Petersburg, 14th lin. V.O, 39

RNSS Officers	
President	R. M. Yusupov, *yusupov@iias.spb.su*
Chair Man. Board	A. Plotnikov, *plotnikov@sstc.spb.ru*
Secretary	M. Dolmatov, *dolmatov@simulation.su*
Repr. EUROSIM	R. M. Yusupov, *yusupov@iias.spb.su*
Deputy	B. Sokolov, *sokol@iias.spb.su*
Edit. Board SNE	Y. Senichenkov, *sneyb@dcn.infos.ru*

Last data update February 2012

SNE – Simulation Notes Europe

Simulation Notes Europe publishes peer revie wed *Technical Notes*, *Short Notes* and *Overview Notes* on developments and trends in modelling and simulation in various areas and in application and theory. Furthermore **SNE** documents the **ARGESIM Benchmarks** on *Modelling Approaches and Simulation Implementations* with publication of definitions, solutions and discussions (*Benchmark Notes*). Special *Educational Notes* present the use of modelling and simulation in and for education and for e-learning.

SNE is the official membership journal of **EUROSIM**, the Fede ration of Europe an Si mulation Societies. A News Section in **SNE** provides inf ormation f or **EUROSIM** Simulation Societies and Sim ulation Groups. In 2013, SNE introduced an extended subm ission strategy i) individual s ubmissions of scientific papers, and ii) submissions of selected c ontributions from conferences of EUROSIM societies for post-conference publication (suggested by conference organizer and authors) – both with peer review.

SNE is published in a printed version (Print ISSN 2305-9974) a nd in a n online ve rsion (Online ISSN 2306-0271). W ith **Online SNE** the publisher **ARGESIM** follows the **Open Access** strategy, allowing download of published contributions for free. Since 2012 **Online SNE** contributions are identified by an DOI (Digital Object Identifier) assigned to the publisher **ARGESIM** (DOI prefix 10 .11128). **Print SNE**, high -resolution **Online SNE**, source codes of the *Benchmarks* and other additional sources are available for subscription via membership in a **EUROSIM** society.

Authors Information. Authors are i nvited to subm it contributions which have n ot been p ublished a nd ha ve not being considere d for pu blication else where t o the SNE Editorial Office. SNE di stinguishes different types of contributions (*Notes*):

- *Overview Note* – State-of-the-Art report in a specific area, up to 14 pages, only upon invitation
- *Technical Note* – scientific publication on specific topic in modelling and simulation, 6 – 8 (10) pages
- *Education Note* – modelling and simulation in / for education and e-learning; max. 6 pages
- *Short Note* – recent development on specific topic, max. 4 pages
- *Software Note* – specific implementation with scientific analysis, max 4 pages
- *Benchmark Note* – Solution to an ARGESIM Benchmark;basic solution 2 pages, extended and commented solution 4 pages, comparative solutions on invitation

Interested authors may find further information at **SNE's** website → **www.sne-journal.org** (layout templates for *Notes*, requirements for benchmark solutions, etc.).

SNE Editorial Office /ARGESIM

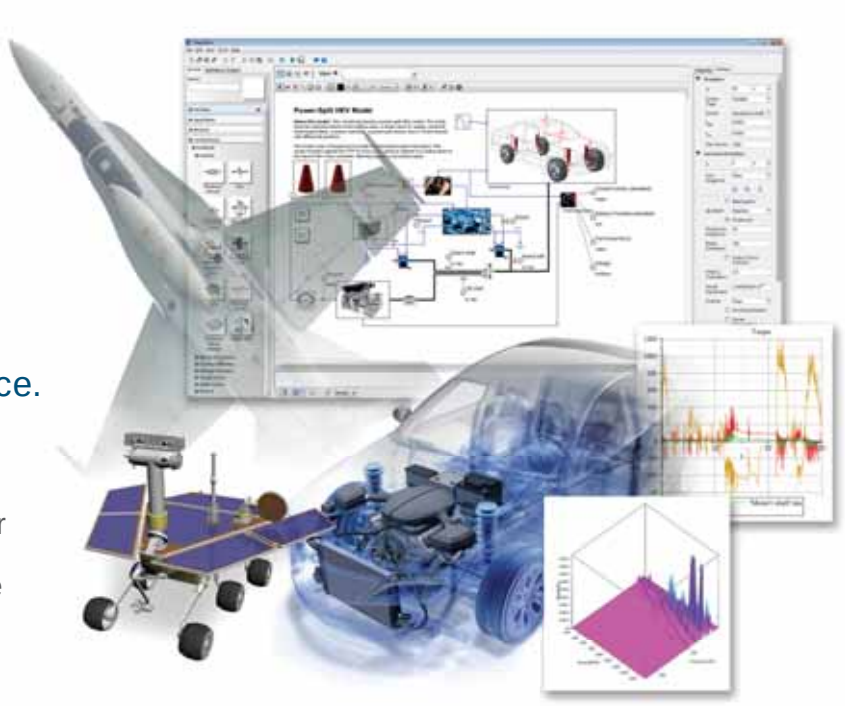
→ *www.sne-journal.org, www.eurosim.info*

office@sne-journal.org (info, news)

eic@sne-journal.org Felix Breitenecker (publications)

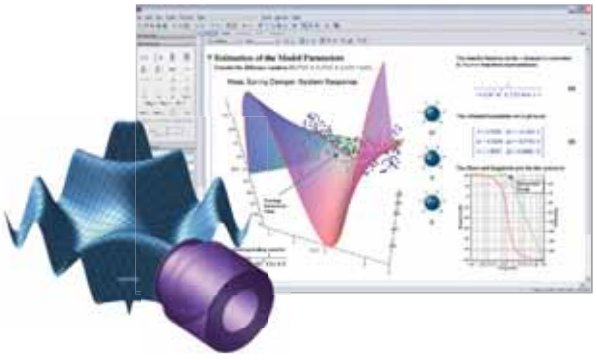
A modern approach to modeling and simulation

MapleSim™



With MapleSim, educators have an industry-proven tool to help bridge the gap between theory and practice.

- MapleSim illustrates concepts, and helps students learn the connection between theory and physical behavior
- A wide variety of models are available to help get started right away



Maple™

MapleSim is built on Maple, which combines the world's most powerful mathematical computation engine with an intuitive, "clickable" user interface.

To learn more about how you can reinforce engineering concepts using a combination of theory, simulation, and hardware, view this webinar.

www.maplesoft.com/SNEWebinar

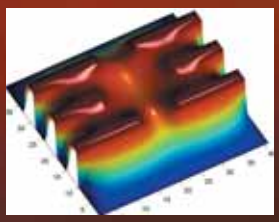
Contact us: +49 (0)241/980919-30

Maplesoft
Mathematics • Modeling • Simulation
A Cybernet Group Company

www.maplesoft.com | germany@maplesoft.com

Parlez-vous MATLAB?

Über eine Million Menschen weltweit sprechen MATLAB. Ingenieure und Wissenschaftler in allen Bereichen – von der Luft- und Raumfahrt über die Halbleiterindustrie bis zur Biotechnologie, Finanzdienstleistungen und Geo- und Meereswissenschaften – nutzen MATLAB, um ihre Ideen auszudrücken. Sprechen Sie MATLAB?



Modellierung eines elektrischen Potentials in einem Quantum Dot.

Dieses Beispiel finden Sie unter: www.mathworks.de/ltc

Image: Kim Young-Sang, Jung HeeJun, Quantum Device Lab, Hanyang Univ.

MATLAB®

The language of technical computing