PARALLELIZING THE SIMULATION OF SHIPBOARD POWER SYSTEMS

Technical Report

Submitted to:
The Office of Naval Research

Contract Number: N0014-08-1-0080

Submitted by:
Fabian Uriarte (UT), Robert Hebner (UT), Michael Mazzola (MSU) Greg Henley (MSU), Tomasz Haupt (MSU), Angela Card (MSU), Sherif Abdelwahed (MSU), Jian Shi (MSU), Mohammed Alattar (MSU)

June 2014

ELECTRIC SHIP RESEARCH AND DEVELOPMENT CONSORTIUM

ESRDC

MISSION STATEMENT

The Electric Ship Research and Development Consortium brings together in a single entity the combined programs and resources of leading electric power research institutions to advance near- to mid-term electric ship concepts. The consortium is supported through a grant from the United States Office of Naval Research.

Florida State University

Massachusetts Institute of Technology

Mississippi State University

Naval Postgraduate School

Purdue University

University of South Carolina

United States Naval Academy

The University of Texas at Austin

2000 Levy Avenue, Suite 140 | Tallahassee, FL 32310 | www.esrdc.com

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 EXECUTIVE SUMMARY

As a result of this research the Navy has a simulation approach for ship power systems that is computationally effective enough to permit efficient simulation. In addition to simulating the basic power system, significant progress has been made in the simulation of the control system.

This research is necessary because the technology leading to effective simulation has slowed its rate of advance significantly over the past decade. The ESRDC was not alone in recognizing this situation. Within the government broadly, this is an issue being addressed by the Office of Science and Technology Policy, who has staff focused on finding a good solution for the U.S, government. In Defense, DARPA has staff members that recognize the need to help maintain military superiority while transitioning from an environment driven by Moore's Law. Outside of DOD, this is one of the top technical challenges being addressed by the IEEE, the leading global technical organization in the field of electro-technology.

The ESRDC was among the leaders in recognizing the issue because the development of future ships that are efficient, effective, and employ emerging technology requires exhaustive simulation before and after their construction. Today, however, it is not possible to conduct the required simulations of large shipboard models due to long-running solution times obtained when using commercial software on desktop computers.

The ESRDC solution has been developed over time to match the needs of the Navy and the shipbuilding community while also being sensitive to the relevant commercial development. Discussion with shipyard leaders led to a strong endorsement to have a solution as close to *MATLAB*/*Simulink* as possible. This is an understandable constraint as most engineers today graduate being competent in said program. A different approach would increase training costs and thus overall costs. The ESRDC approach meets this need by using *Simulink* as the user interface.

To make the capability available to as wide an audience as possible, the initial development is available to all users inside ESRDC. To move the capability from the ESRDC to the Navy, the ESRDC will make the software available on the web to students at the Naval Postgraduate School to help support their research. This is a research environment involving naval officers and the system is helping them improve the quality of their education during their limited time at NPS. The arrangement is that the program is provided through a private link between NPS and the University of Texas at Austin. The developer in Austin can monitor performance and quickly help students resolve any problems. Discussions are underway to open the link to research projects at the U.S. Naval Academy after the system is sufficiently robust to be used by less experienced users.

The next step would be to transfer the capability to NAVSEA and to the shipyards. Those implementations must be even more robust, however, as much of the information they process is classified. That is, in moving from research to production, the penalty for failure is higher.

Significant progress has been made in four areas:

- Accelerating the simulation of shipboard power systems:
  Accelerations near 80x [1] have been measured in circuits with relevant levels of complexity.
- Assessing confidence in the accuracy of the accelerated simulations:
  The comparisons showed that simulations get faster but with no major reduction in accuracy when compared to commercial systems. [2]
- Providing benchmarks for industrial development
  Collaboration [3] with commercial suppliers of software has guided this work and provided benchmarks for the developers of both commercial and open source software.
- Simulation of dc systems
  Previous work focused on ac systems. The work herein was on a dc system, and it brought together for the first time a combined power grid and control system simulation. In addition, it showed the approach was robust with respect to the simulation of solid-state switching, a key requirement for dc system simulations.

2 INTRODUCTION

An important contributor to U.S. military superiority is the continued superiority in computing and communications. For the last few decades, military superiority in this area has rested in a large part on Moore's Law, which is a description of the fact that investment in appropriate semiconductor technology led to better performance, which led to new products that organizations and individuals would buy, which led to further investment in the technology. The Department of Defense has learned to exploit this rapid change in technology even though it does not fit well with its budget or procurement cycles. This ability to manage technological change has helped achieve superior capability.

But Moore's Law growth has ended. In a purely technological level, we can still double the density of the components on a processor chip, but it does not lead to sufficient system improvement to warrant the investment. So, the Department of Defense, as well as companies needing a competitive advantage, must find other solutions.

The ESRDC was not alone in recognizing this situation. Within the government broadly, this is an issue being addressed by the Office of Science and Technology Policy, who has staff focused on this finding a good solution for the U.S, government. In Defense, DARPA has staff members that recognize the need to help maintain military superiority while transitioning from an environment driven by Moore's Law. Outside of DOD, this is one of the top technical challenges being addressed by the IEEE, the leading global technical organization in the field of electrotechnology.

The ESRDC was early in recognizing the issue because the development of future ships that are efficient, effective, and employ emerging technology requires exhaustive simulation before and after their construction. Today, however, it is not possible to conduct the required simulations of large shipboard models due to the length of time required to complete the solution when using commercial software and desktop computers.

This led to the ESRDC exploring three options:

- Use of special purpose computers:
  The ESRDC does have access to special purpose computer systems to address appropriate near term problems. And this has been successful. The need for hardware procurement and training costs coupled with the historical concerns over the longevity of special purpose computing have suggested this will be a valuable research tool, but it will not be widely adopted within the Navy and the shipbuilding industry
- Use of Field-Programmable Gate Arrays (FPGA's):
  FPGA's can be considered quasi-special-purpose computers that provide excellent computational speed by limiting functionality. They have found use, for example, in control systems. The ESRDC explored with ONR the possibility of exploiting this technology for ship simulation. The challenge, however, was that ONR was already exploring this technology in general, but the anticipated progress was expected to be too slow to support ship design. Furthermore, some of the issues cited for special purpose computers would likely prevail with the FPGA alternative.
- Use of technologies that are in the mainstream of computer evolution:
  The computer industry has been evolving to more parallel systems to compensate for lack of speed on a given processor. Multicore systems provide promise for continued improvement and are commercially available for decreasing cost. The primary challenge is that legacy software typically must be rewritten to operate with best efficiency on such systems. For adaption to ship power system design, the major challenge was automated model partitioning and parallelization into subsystems of less computational burden to facilitate the use of legacy systems. The ESRDC has made a significant contribution to solving this problem.

The ESRDC solution has been developed over time to match the needs of the Navy and the shipbuilding community while also being sensitive to the relevant commercial development. Discussion with ship yard leaders led to a strong endorsement to have a solution as close to *Simulink* [4-6] as possible. This is an understandable constraint as most engineers today graduate being competent in such program. A different approach would increase training costs and thus overall costs. The ESRDC approach meets this need by *Simulink* as the user interface.

To make the capability available to as wide an audience as possible, the initial development will be made available to all users under Navy-sponsored programs. To move the capability from the ESRDC to the Navy, the ESRDC will make the software available on the web to students at the Naval Postgraduate School to help support their research. This is a research environment involving naval officers and the system is helping them improve the quality of their education during their limited time at NPS. The arrangement is that the program is provided on the web through a private link between NPS and the University of Texas at Austin. The developer in Austin can monitor performance and quickly help students resolve any problems. Discussions are underway to next open the link to research projects at the U.S. Naval Academy after the system is sufficiently robust to be used by less experienced users.

The next step would be to transfer the capability to NAVSEA and to the shipyards. Those implementations must be even more robust, however, as much of the information they process is classified. That is, moving from research to production, the penalty for failure is higher.

A widely accessible accelerated simulation approach provides key Navy users with early capability to use emerging software approaches to modeling. The open structure permits the focus to be on a structure that works for the application. In addition, collaboration with software vendors permits the information developed in this project to help inform the commercial development.

3 RESEARCH PROGRAM STRUCTURE

The Center for Electromechanics (CEM) of The University of Texas at Austin (UT) is developing a parallel solver for the Office of Naval Research and Electric Ship Research and Development Consortium (ESRDC). The aim is to accelerate the simulation of large ship power systems models created in *Simulink* and the *SimPowerSystems* blockset.[1] Among its salient features, *CEMSolver* is designed for use on everyday multicore desktop (i.e., *Windows*-based) computers to circumvent the acquisition of specialized hardware.

Although *CEMSolver* accelerates the simulation of electrical network models, shipboards include controls as well. To address this aspect, Mississippi State University, in close lock-step collaboration with UT, is developing a complimentary solver named *MSUSolver*. *MSUSolver* compliments *CEMSolver* by solving the controls portion of a model while *CEMSolver* solves the electrical portion of a model. The development of these two solvers, and testing their communication, speedup, and accuracy are the outcomes from this research project (Appendix A).

4 ADVANCES

Significant progress has been made in four areas:

- Accelerating the simulation of shipboard power systems
- Assessing the confidence in the accuracy of the accelerated simulations
- Providing benchmarks for industrial development
- Simulation of dc systems

Each of these is a critical step toward the final application.

4.1 Accelerating Simulation

Previous research [3],[7-10] in parallelizing and accelerating desktop simulation focused on ac power systems as they are the most common in both the Navy and in worldwide applications. Said application is expected to have the greatest impact, but early in the research, it was decided to exclude control systems and focus the attention to the computational burden of the problem: electromagnetic simulation of large-scale electrical networks.

This work has advanced the research by focusing on a scaled-down version of the dc shipboard model created by Florida State University. The key technical challenge was to continue automatic partitioning of the electrical system, but this time including the solution of the related

[1] *MATLAB*/*Simulink* including the *SimPowerSystems* blockset is termed *Simulink* hereinafter

control network. Solving the control network required the development of a new solver called *MSUSolver*, which in addition to providing accurate results, had to communicate correctly, promptly, and be synchronized with *CEMSolver*.

It is well known that to use multiple processors, a problem must be partitioned so that the processors can work efficiently in parallel. The breakthrough of *CEMSolver* was to use graph-partitioning software. This mathematical process can be performed by the computer without requiring the operator to have knowledge of graph theory. Following partitioning, a model must be reformulated by using appropriate, unknown current sources and sinks to make each partition perform electrically as it would in the unpartitioned system. Again, this is all done in software with no intervention by the operator.

It was anticipated and demonstrated that the degree of acceleration would depend on the nature of the model simulated. The components for each of the cases simulated are summarized in Table 4 (page 20 in Appendix). The observed accelerations will show that using *Simulink* on computer 2 took ~21 minutes, but only ~6 minutes using *CEMsolver* (a 3.2x speedup)—and in all cases, the operator developed only one model in *Simulink*. The computer used by *CEMSolver* was the *same* computer as used by *Simulink* to do the simulation. Therefore, while the complexity is transparent to the user, the benefit is very apparent.

4.2 Confidence in the Results

Assuring the accuracy of a simulation is a challenging task. In this case, it was determined acceptable to assess relative accuracy against *Simulink* because it is a commercial product that is widely used and generally produces trusted results. Nevertheless, there are issues with the commercial software as it leaves choosing the integration algorithm to the user, which is expected to be familiar with the problem being solved. Each integration algorithm is meant to address a different type of problem. *Simulink* allows specifying Tustin or Backward Euler integration.

Comparing the results on the same circuit (or model) between *Simulink* and *CEMSolver* suggests that the enhanced speed of *CEMSolver* does not degrade the accuracy. Referring to the measurement locations on the one-line diagram of Fig. 1, the accuracy comparisons appear in the Appendix as Fig. 5 through Fig. 11.

It should be recognized that the agreement of results between the slower commercial approach and the faster ESRDC approach was achieved by overcoming important differences. These included:

- Commercial development of *Simulink* vs. academic development of *CEMSolver*
- State-space formulation in *Simulink* vs. nodal one in *CEMSolver*
- Unpartitioned as opposed to parallelized solutions
- Different integration methods
- Different switch models

4.3 Benchmarking Commercial Development

The ESRDC is not developing its acceleration capability in a vacuum. Rather it is collaborating with industry to develop long-term sustainable solutions that are useful for the Navy. In a published comparison [3], the ESRDC approach achieved a 30-times acceleration of the simulation compared to *MATLAB/Simulink* while the commercial product achieved about a 7x acceleration. The comparison with the ESRDC solution provided the company with information needed to improve their approach. To determine accuracy, point-by-point comparisons were made of simulated waveforms. All three approaches produced nearly identical results, showing the strength of each.

4.4 Simulation of DC Systems

The outcome of this work was demonstrated by accelerating a *scaled-down* version of the notional MVDC model built by ESRDC researchers at FSU. This work is different from previous research in that it also simulates the behavior of the control system through a software package named the *MSUSolver*. These two emerging solvers are being designed to import existing *Simulink* models and automatically parallelize their simulation.

In the past, *CEMSolver* (alone) solved electrical networks for AC shipboard models. In this work, *CEMSolver* was tested on a DC shipboard model and —in addition—was restructured to cooperate with *MSUSolver* toward a comprehensive electrical-and-control solution. While the solution of DC systems did not present a technical challenge to *CEMSolver*, restructuring of *CEMSolver* to cooperate with *MSUSolver* introduced new wait states in *CEMSolver* that reduced its performance. Despite the performance reduction, acceleration was still possible (but to a lesser degree).

The metrics of success for DC shipboard simulation (electrical and controls) were defined as solver communication, speed, and accuracy. The solvers communicated successfully by 1) exchanging valid data throughout the simulation, 2) running to completion without abnormal terminations, and 3) by producing results consistent with those observed in *Simulink*. The details of the DC shipboard simulation are given in the Appendix.

5 CONCLUSIONS

The change in computing necessitated by the end of improvements driven by Moore's law has stimulated creative solutions to achieving the speed needed to support the simulation of ship power systems. This research program has succeeded in achieving significant acceleration of the simulation time by automated partitioning of the problem.

This approach has the advantage of using conventional inputs so that engineers do not need to be retrained to use the system. In addition, the solution is being offered on the web to students at the Naval Postgraduate School to help keep them current with the newest technology.

The approach is fast enough to be a significant improvement. In addition, the speed is gained with no loss of accuracy. The development also serves as a benchmark to industry to guide their proprietary improvements. Moreover, it is not an attempt to develop a new software program as that is the role of industry. Rather it is designed to seamlessly merge with selected commercial software so it can be widely useful.

In addition to a circuit solver, the research has developed a system to integrate the control system into the circuit simulation. While the addition has naturally added overhead, the acceleration is still significant. Further research can optimize the combined performance.

This research is developing the tools needed by the Navy today. But it is also setting the path for the commercial, hardware-tuned, and widely-accessible software of tomorrow.

6 REFERENCES

[1] F. M. Uriarte, R. E. Hebner, and A. L. Gattozzi, "Accelerating the simulation of shipboard power systems," in *Proc. 2011 Grand Challenges in Modeling & Simulation*, June 27 - 30, 2011.

[2] F. M. Uriarte and R. E. Hebner, "Assessing Confidence in Parallel Simulation Results," in *Proc. 2013 Electric Ship Technologies Symposium*, Apr. 22-23, 2013.

[3] F. M. Uriarte and C. Dufour, "Multicore Methods to Accelerate Ship Power System Simulations," in *Proc. 2013 Electric Ship Technologies Symposium*, April 22-23, 2013.

[4] The MathWorks, Inc. (2010). *Simulink 7 User's Guide.* [Online]. Available: http://www.mathworks.com/help/toolbox/simulink/

[5] The MathWorks, Inc. (2010). *SimPowerSystems 5 User's Guide.* [Online]. Available: http://www.mathworks.com/help/toolbox/physmod/powersys/

[6] L.-A. Dessaint, K. A.-Haddad, H. Le-Huy, G. Sybille, *et al.*, "A Power System Simulation Tool Based on Simulink," *IEEE Trans. Industrial Electronics*, vol. 46, 1999.

[7] F. M. Uriarte. (2013). *Multicore Simulation of Power System Transients.* (1st ed.) London: IET.

[8] F. M. Uriarte and R. Hebner, "Development of a multicore power system simulator for ship systems," in *Proc. 2011 Electric Ship Technologies Symposium*, pp. 106-110, April 10-13, 2011.

[9] F. M. Uriarte, "Multicore simulation of an ungrounded power system," *IET Electrical Systems in Transportation*, vol. 1, pp. 31-40, Mar. 2011.

[10] F. M. Uriarte and K. L. Butler-Purry, "Multicore simulation of an AC-radial shipboard power system," in *Proc. 2010 Power Engineering Society General Meeting*, pp. 1-8, July 25-29, 2010.

[11] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: applications in VLSI domain," in *Proc. 1997 Design and Automation Conference*, pp. 526-529, 1997.

[12] G. Karypis and V. Kumar. (1998). *hMETIS: A Hypergraph Partitioning Package Version 1.5.3.* Minneapolis: Dept. Computer Science & Engineering, University of Minnesota. [Online]. Available: http://glaros.dtc.umn.edu/gkhome/metis/hmetis/download

[13] F. M. Uriarte, "On Kron's diakoptics," *Electr. Power Syst. Res.*, vol. 88, pp. 146-150, Jul. 2012.

[14] C. Dufour, J. Mahseredjian, J. Bélanger, and J. L. Naredo, "An advanced real-time electro-magnetic simulator for power systems with a simultaneous state-space nodal solver," in *Proc. 2010 IEEE/PES Transmission and Distribution Conference and Exposition: Latin America*, 2010.

[15] J. Schutt-Aine, "Latency Insertion Method (LIM) for the Fast Transient Simulation of Large Networks," *IEEE Trans. Circuits and Systems I: Fundamental Theory and Applications*, vol. 48, pp. 81-89, 2001.

[16] S. Esmaeili and S. M. Kouhsari, "A Distributed Simulation Based Approach for Detailed and Decentralized Power System Transient Stability Analysis," *Electric Power Systems Research*, vol. 77, pp. 673-684, 2007.

[17] J. A. Hollman and J. R. Marti, "Real time network simulation with PC-cluster," *IEEE Trans. Power Systems*, vol. 18, pp. 563-569, 2003.

[18] S. Jiwu, X. Wei, and Z. Weimin, "A Parallel Transient Stability Simulation for Power Systems," *IEEE Trans. Power Systems*, vol. 20, p. 1709, 2005.

[19] J. R. Marti, L. Linares, J. A. Hollman, and F. A. Moreira, "OVNI: Integrated Software/Hardware Solution for Real-Time Simulation of Large Power Systems," in *Proc. 2002 Power Systems Computation Conference (PSCC'02)*, pp. 1-7, 2002.

[20] J. R. Marti and L. R. Linares, "Real-Time EMTP-based Transients Simulation," *IEEE Trans. Power Systems*, vol. PWRS-9, pp. 1309-1317, 1993.

[21] T. Noda and S. Sasaki, "Algorithms for Distributed Computation of Electromagnetic Transients toward PC Cluster Based Real-Time Simulations," in *Proc. 2003 International Conference on Power System Transients*, 2003.

[22] Y. Xie, G. Seenumani, J. Sun, Y. Liu, *et al.*, "A PC-cluster based real-time simulator for all-electric ship integrated power systems analysis and optimization," in *Proc. 2007 Electric Ship Technologies Symposium (ESTS)*, May 22-23, 2007.

[23] K. Strunz and E. Carlson, "Nested Fast and Simultaneous Solution for Time-Domain Simulation of Integrative Power-Electric and Electronic Systems," *IEEE Trans. Power Delivery*, vol. 22, p. 277, 2007.

[24] M. Armstrong, J. R. Marti, L. R. Linares, and P. Kundur, "Multilevel MATE for efficient simultaneous solution of control systems and nonlinearities in the OVNI simulator," *IEEE Trans. Power Systems*, vol. 21, pp. 1250-1259, Aug. 2006.

[25] P. T. Norton, P. Deverill, P. Casson, M. Wood, *et al.*, "The reduction of simulation software execution time for models of integrated electric propulsion systems through partitioning and distribution," in *Proc. 2007 Electric Ship Technologies Symposium (ESTS)*, pp. 53-59, May 22-23, 2007.

[26] Y. Gong, L. Chen, Y. Chen, and Y. Xu, "A parallel based real-time electromagnetic transient simulator for IPS," in *Proc. 2011 Electric Ship Technologies Symposium*, pp. 96-101, April 10-13, 2011.

[27] J. Langston, S. Suryanarayanan, M. Steurer, M. Andrus, *et al.*, "Experiences with the simulation of a notional all-electric ship integrated power system on a large-scale high-speed electromagnetic transients simulator," in *Proc. 2006 Power Engineering Society General Meeting*, June 18-22, 2006.

[28] Y. Zhang, R. Dougal, B. Langland, J. Shi, *et al.*, "Method for partitioning large system models when using latency insertion method to speed network solution," in *Proc. 2009 Grand Challenges in Modeling and Simulation*, 2009.

[29] Q. Huang, J. Wu, J. L. Bastos, and N. N. Schulz, "Distributed Simulation Applied to Shipboard Power Systems," in *Proc. 2007 Electric Ship Technologies Symposium, 2007. ESTS '07. IEEE*, pp. 498-503, 2007.

[30] A. Benigni, P. Bientinesi, and A. Monti, "Benchmarking different direct solution methods for large power system simulation," in *Proc. 2010 Grand Challenges in Modeling and Simulation*, 2010.

[31] F. M. Uriarte, "A partitioning approach for parallel simulation of AC-radial shipboard power systems," PhD dissertation, Dept. Electrical and Computer Eng., Texas A&M Univ., College Station, 2010.

[32] F. M. Uriarte and K. L. Butler-Purry, "A Partitioning Approach for the Parallel Simulation of Ungrounded Shipboard Power Systems using Kron's Diakoptics and Loop Analysis," in *Proc. 2007 Summer Computer Simulation Conference 2007 (SCSC'07)*, Jul. 16-17, 2007.

[33] IEEE Power Systems Engineering Committee, "Parallel Processing in Power Systems Computation," *IEEE Trans. Power Systems*, vol. 7, pp. 629-38, 1992.

[34] K. K. C. Yu and N. R. Watson, "A Comparison of Transient Simulation with EMTDC and State Space Diakoptical Segregation Methodology," in *Proc. June 19-23, 2005 the International Conference on Power System Transients*, June 19-23, 2005.

[35] A. Kalantari and S. M. Kouhsari, "An Exact Piecewise Method for Fault Studies in Interconnected Networks," *International Journal of Electrical Power & Energy Systems*, vol. 30, pp. 216-225, 2008.

[36] T. Watanabe, Y. Tanji, H. Kubota, and H. Asai, "Fast Transient Simulation of Power Distribution Networks Containing Dispersion Based on Parallel-Distributed Leapfrog Algorithm," *IEICE Trans. Fundamentals*, vol. E90, pp. 388-397, 2007.

[37] Z. Quming, S. Kai, K. Mohanram, and D. C. Sorensen, "Large Power Grid Analysis Using Domain Decomposition," in *Proc. 2006 Design, Automation and Test in Europe, 2006. DATE '06. Proceedings*, pp. 1-6, 2006.

[38] P. Zhang, J.R.Marti, and H. W. Dommel, "Network Partitioning for Real-Time Power System Simulation," in *Proc. 2005 International Conference on Power System Transients*, pp. 1-6, 2005.

[39] C. Yue, X. Zhou, and R. Li, "Node-splitting approach used for network partition and parallel processing in electromagnetic transient smiulation," in *Proc. 2004 International Conference on Power System Technology*, 2004.

[40] K. W. Chan, R. C. Dai, and C. H. Cheung, "A Coarse Grain Parallel Solution Method for Solving Large Sets of Power System Network Equations," in *Proc. 2002 International Conference on Power System Technology (PowerCon '02)*, pp. 2640-2644, 2002.

[41] A. Y. Zomaya. (1996). *Parallel and Distributed Computing Handbook.* New York: McGraw-Hill.

[42] J. Langston, M. Steurer, J. Crider, S. Sudhoff, *et al.*, "Waveform-Level Time-Domain Simulation Comparison Study of Three Shipboard Power System Architectures," in *Proc. 2012 Grand Challenges in Modeling and Simulation*, 2012.

[43] M. Steurer, S. Woodruff, R. Wen, H. Li, *et al.*, "Accuracy and Speed of Time Domain Network Solvers for Power Systems Electronics Applications," in *Proc. 2003 10th European Conference on Power Electronics and Applications (EPE2003)*, Sept 2-4, 2003.

[44] A. Yamane and J. Bélanger. (2010). *Comparison between ARTEMIS 5th order Integration method used with the eMEGAsim Simulation Platform and the classical TUSTIN 2nd order method used in PSCAD and SimPowerSystems software.* [Online]. Available: http://www.opal-rt.com/technical_documents

[45] C. Dufour and J. Belanger, "Discrete Time Compensation of Switching Events for Accurate Real-Time Simulation of Power Systems," in *Proc. 2001 The 27th Annual Conference of the IEEE Industrial Electronics Society*, pp. 1533-1538, 2001.

[46] A. M. Gole, "Electromagnetic transient simulation of power electronic equipment in power systems: challenges and solutions," in *Proc. 2006 Power Engineering Society General Meeting*, pp. 1301-1306, Oct., 2006.

[47] M. O. Faruque, V. Dinavahi, and W. Xu, "Algorithms for the accounting of multiple switching events in digital simulation of power-electronic systems," *IEEE Trans. Power Delivery*, vol. 20, pp. 1157-1167, 2005.

7 APPENDIX: DC SIMULATION

The objective of this work is to continue the development of *CEMSolver*, debut *MSUSolver*, and to test the communication, speedup, and accuracy of the two solvers. *CEMSolver* and *MSUSolver* were tested by cooperatively solving a scaled-down version of the MVDC model built by FSU. This approach was motivated by a "walk before you run" approach. It is important to determine the performance characteristics of the new system prior to trying interestingly complex problems.

The remainder of this appendix is organized as follows. Section 7.1 gives an overview of the MVDC model used in this work. Section 7.2 gives a brief description of how each solver works. Section 7.3 presents the results of the work's objectives. Section 7.4 closes the appendix with observations and conclusions.

Readers are apprised that the effort described in the remaining of this appendix revolves around the ability of *CEMSolver* and *MSUSolver* to cooperatively accelerate a scaled-down MVDC model built in *Simulink* and does not delve into the model details, power system analysis, nor software design.

7.1 Power System Model

The objectives of this work are demonstrated by parallelizing the simulation of a scaled-down version of FSU's MVDC power system model[2]. The scaled-down model includes several characteristics of the full-scale model, but it does not fully represent said system. The first part of this section presents a one-line diagram of the model and describes its salient attributes. The second part compares the sizes of the scaled-down model against the full-size model.

7.1.1 Model Description

This scaled-down version of the MVDC model was built by UT and MSU in *Simulink* version 2012b. The scaled-down model shares important attributes of its larger counterpart such as voltage and power levels, distribution architecture, number of generators, generation control, number of power converters, and number of loads.

The three major differences between the full-size and scaled-down models are the non-dependence on external files, how the loads are modeled, and the controls complexity. The scaled-down mode does not require external scripts (.m files) to initialize or run the model. The scaled-down model exists as a stand-alone file (.slx file) that be run directly. The loads were simplified to static loads behind their respective power converters to reduce the dependence on controls. The number, function, and complexity of the other controls throughout the model were

[2]This report focuses on details of the solution approach, not on details of the specific models. For readers interested in details of this model, they can be found at Y. Lee, E. Zivi, J. Langston, M. Steurer, J. Crider, S. D. Sudhoff, R. A. Dougal, Y. Zhang, R. Hebner, and A. Ouroua, "Waveform-Level Time-Domain Simulation Comparison Study of Three Shipboard Power System Architectures," in Proc. 2012 Conference on Grand Challenges in Modeling and Simulation Conference, Genoa Italy, July 8-11, 2012

also limited in count and complexity. These three differences did not alter significantly the expected behavior of the scaled-down model in comparison to its larger counter-part.

A one-line diagram of the scaled-down MVDC model is shown in Fig. 1, which is presented as a high-level view to favor readability. The cables around the main dc bus were modeled as series resistive-inductive (RL) sections. (All parameters were taken from the full-size MVDC model.) The generators on the schematic represent two 47 MVA main turbine generators (MTG) and two auxiliary 5 MVA turbine generators (ATG). Each generator produces power at 4.16 kV, 240 Hz and includes a control model for its corresponding prime mover (gas turbine), governor, voltage regulator, and exciter. Downstream of each generator are three-phase circuit breakers programmed to close when the simulation time reaches $t = 0.25$ s. Behind each three-phase breaker are six-pulse (uncontrolled) rectifiers (ac-dc converters) that power the main dc ring bus at 4,160 x 1.35 = 5.62 kV dc. Between each rectifier and the main dc bus are disconnect switches. These disconnect switches were included in the model for completeness but they remained in their closed positions. (The same is true for all other switches around the ring bus.)

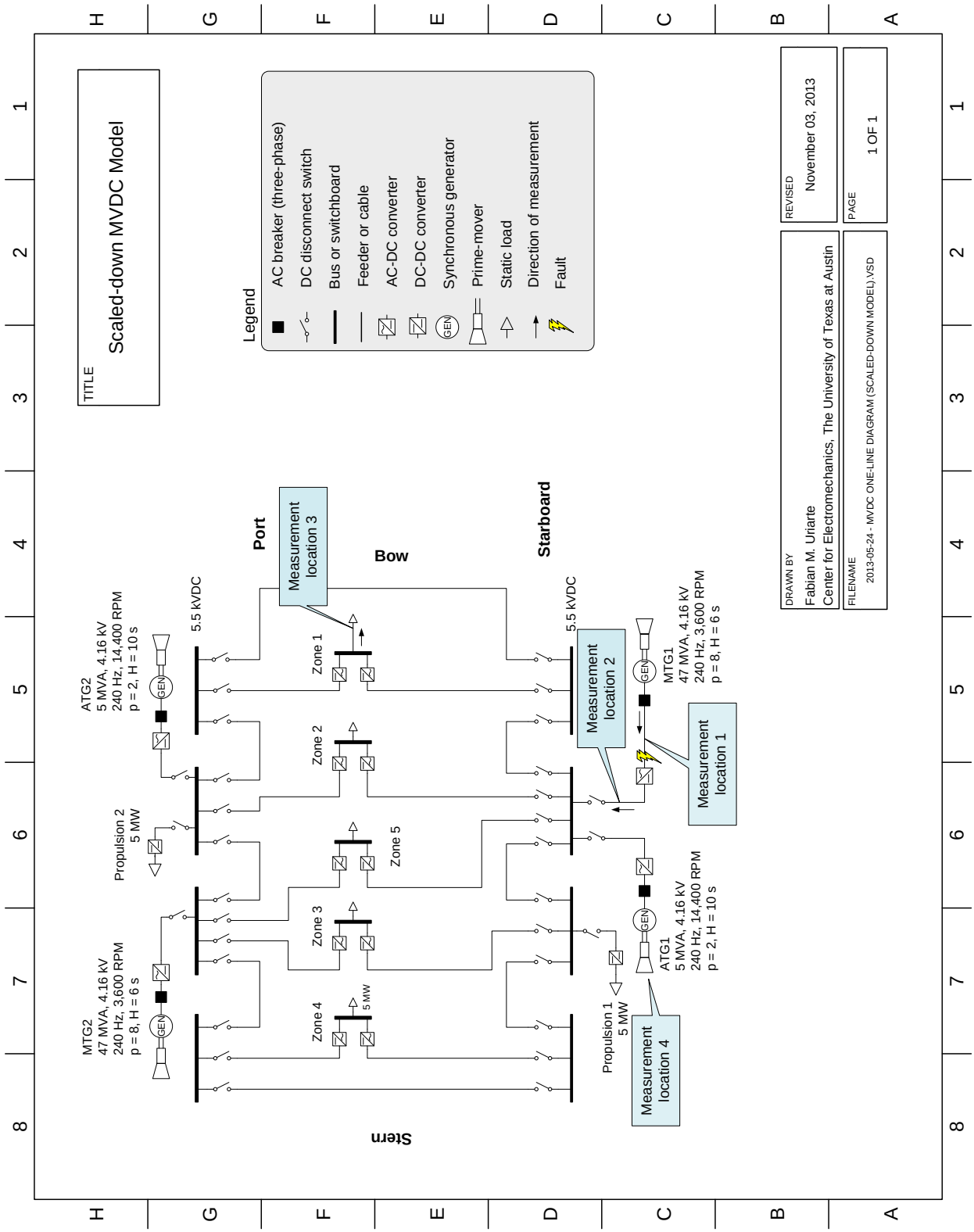


Fig. 1: One-line diagram of scaled-down MVDC model

The main dc bus serves two propulsion loads and five zones. A screenshot of propulsion load 1 is shown in Fig. 2. Each of the two propulsion loads consists of a three-phase inverter controlled by an open-loop pulse-width modulation (PWM) controller. The inverters were modeled with three arms, six ideal switches, and six snubbers. The load served by each inverter represents a

propulsion load, which was modeled as a three-phase static load served at 3.7 kV ac (5 MVA, 60 Hz, 0.85 power factor).

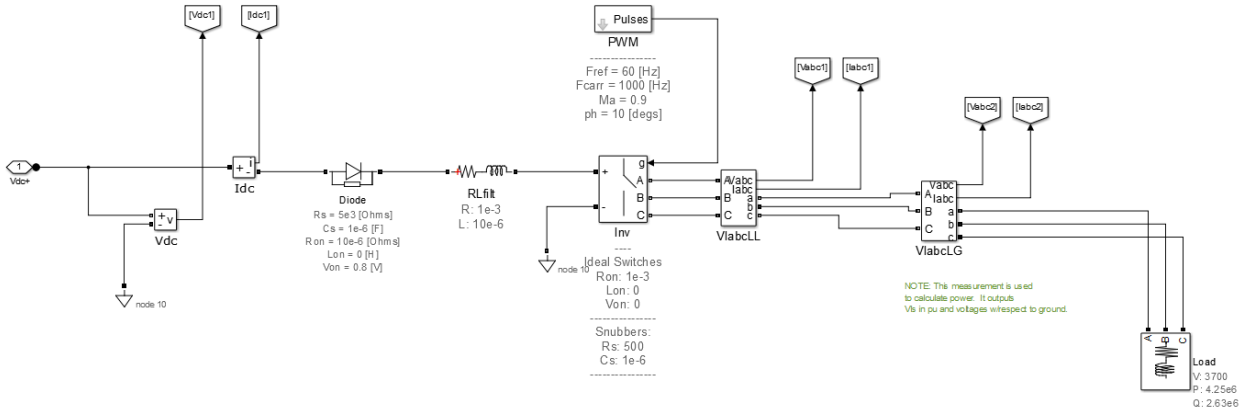


Fig. 2: Screenshot of propulsion load 1

The five zones were each modeled as two parallel dc/dc converters serving a common resistive load. A high-level diagram of zone 1 is shown in Fig. 3. From left to right, each dc/dc converter was modeled as having a front-end two-arm inverter controlled in open loop by its own PWM controller. The output of each inverter was a bi-polar square wave applied to single-phase isolation transformers. The isolation transformers (two per zone) stepped down the voltage from ~5 kV to 1 kV. The output stage of each dc/dc converter connected the transformer's secondary side to a two-arm uncontrolled diode rectifier. Each of the paralleled rectifiers served the same (common) zonal load modeled as a static resistance.

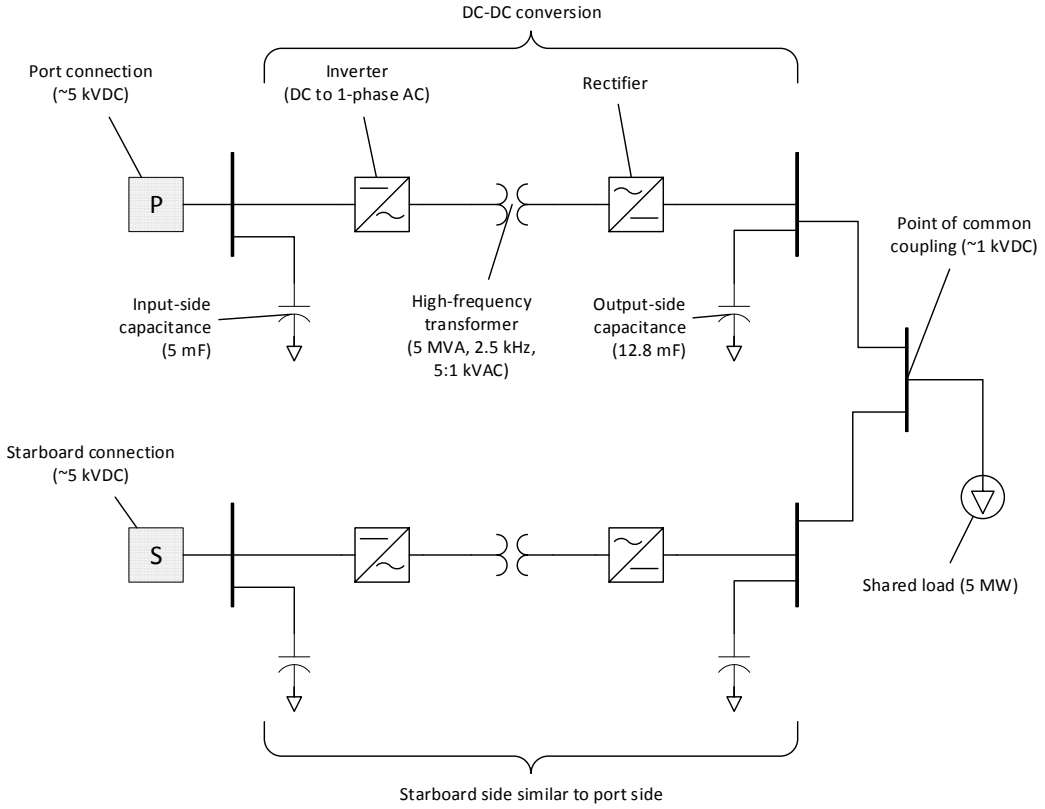


Fig. 3: Diagram of zone 1

7.1.2 Model Size

The scaled-down model used in this work is intended to be smaller than the full-scale model. Terms such as *smaller* or *larger*, however, are often subject to different interpretations. These terms may refer to block count, file size, run time, system capacity, spatial dimensions, equipment count, bus count, the number of power electronic valves, state-variables, nodes, meshes, branches, or number of non-zeros in the matrix factors (to name a few).

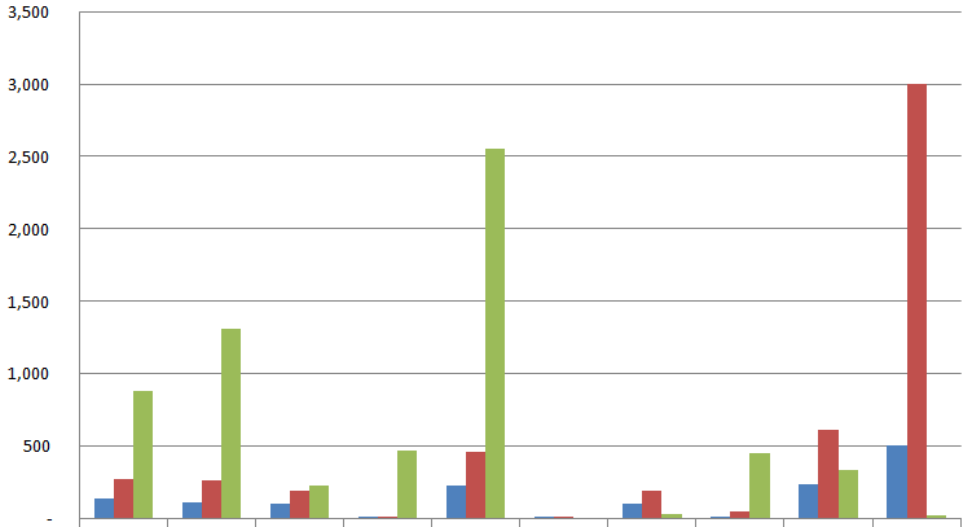
This section provides information on the size of the scaled-down model in relation the full scale model by using different metrics to assess model size. Although not part of this work, for completeness, a third model used by UT in the past to benchmark *CEMSolver* is included for reference. This reference model is deemed large, and it represents a legacy electromechanical 450 VAC, 5 MW AC-radial systems and is described in [1-3],[8]. The third model is referred to because it has been used in the past to demonstrate that *CEMSolver* is capable of speedups above 50x on select models [1-3],[10].

Fig. 4 shows several metrics to contrast the scaled-down model, the full-scale model, and the reference AC-radial model. The meanings of each category are defined below:

- *Nodes*: the number of circuit nodes or junctions where two or more electrical branches interconnect
- *State-variables*: the number of independent state-variables as reported by *Simulink*. These variables are related to the linear portion of the model and include inductor currents, capacitor voltages, and transfer block states.
- *Power-electronic switches*: the total number of switches found in all power electronic converter blocks together. The frequent commutation of these switches is known to degrade simulation performance, and it often serves as a metric to gauge model complexity.
- *Breaker switches*: the total number of switches found in all three-phase breakers and dc disconnect switches together. These switches do not toggle often and neither their electrical presence nor operation affects simulation performance as do power electronic switches. However, breakers typically include relays that require calculating throughput (e.g., three-phase voltage and current rms measurements) and can degrade simulation performance.
- *RLC branches*: the number of resistor, inductor, or capacitor branches. Combinations are considered a single branch.
- *Machines*: the number of generator or motors included in the model. These blocks are considered burdensome due to their time-varying characteristics and are also known to be a source of numerical instability.
- *Measurements*: the number of voltage and current measurements included in the model. These can be single- or three-phase voltage and current measurements, and do not include the computation of power nor harmonics. Calculating instantaneous voltage and current is not computationally expensive, but storing their data at a timestep Δt = 10 µs and stop time t_{stop} = 10 s can exert noticeable memory pressure. Memory pressure degrades simulation performance and can block computer (e.g., keyboard and mouse) responsiveness.

- *Run Time*: the amount of time spent by a solver or program during the time loop portion of a simulation. The run time ratio between *Simulink* and *CEMSolver* is the speedup. The run times are typically approximate and vary by computer used and by the state of other background processes.
- *Electrical Blocks*: the number of electrical blocks on the schematic that were extracted from the *SimPowerSystems* blockset (a sub-library in *Simulink*).
- *Control Blocks*: the number of blocks extracted from the *Simulink* native library or from the *SimPowerSystems* "Extras" blocks.

In Fig. 4, when comparing the scaled-down model (blue columns) against the full-scale model (red columns), a salient difference is in the number of control blocks (508 vs. 2,997). The difference in control complexity (i.e., block count, logic, and function) is one of the three aspects mentioned earlier where the models differ most. The full-scale model is also larger in control complexity than the AC-radial system. The AC-radial system, however, is the largest in most other counts including power electronic switches and state-variables, which are noticeably responsible for the larger *Simulink* run time of 450 minutes (for t_{stop} = 10 s, Δt = 10 µs). One desirable feature included in the scaled-down model was over 100 power electronic switches. The need to accurately model such switches is critical in dc systems.



	Nodes	State Variables	Power Elec. Swit.	Breaker Switches	RLC Branches	Machines	Measure ments	Run Time (mins)	Electrical Blocks	Control Blocks
MVDC (scaled-down)	138	111	104	12	224	4	100	10.8	239	508
MVDC (FSU)	274	260	187	12	456	6	190	45	610	2997
AC-Radial	876	1310	228	471	2557	0	30	450	330	19

Fig. 4: Comparison of model metrics to assess model sizes

Although several metrics were presented to assess model size, the authors do not claim which model is larger or smaller as size is a subjective perception. Nevertheless, readers can assess model complexity by contrasting their *Simulink* run times in Fig. 4.

7.2 Solvers

CEMSolver is (strictly) an electrical network solver developed by UT, and it is designed to solve electrical network problems. *MSUSolver* is a control network solver developed by MSU, and it

was developed to compliment *CEMSolver* by addressing the controls portion of models. The combined use of these two solvers is sought to accelerate the simulation of the scaled-down MVDC model.

A depiction of how the solvers cooperate and accepted the reduced MVDC model is shown in Fig. 9. The top part of Fig. 9 depicts the full-scale MVDC model developed by FSU in *Simulink*. The full-scale his MVDC model is highly detailed and available to the ESRDC, but due to its runtime, the full-scale model was not considered practical to reach the outcomes of this work. Below the schematic in Fig. 9 are depictions of the interactions between *CEMSolver* and *MSUSolver*.

At each time step of the simulation, *CEMSolver* sends the electrical network solution to *MSUSolver* as instantaneous voltages and currents. *MSUSolver* uses the electrical network solution as input to produce a solution for the control network. When the control network is solved, *MSUSolver* passes back to *CEMSolver* the control network solution as commanding actions (e.g., trip signals, mechanical power set point, etc.). The interaction between *CEMSolver* and *MSUSolver* is carried out throughout a simulation to produce a parallel simulation that is intended to be faster than simulations produced with *Simulink*.

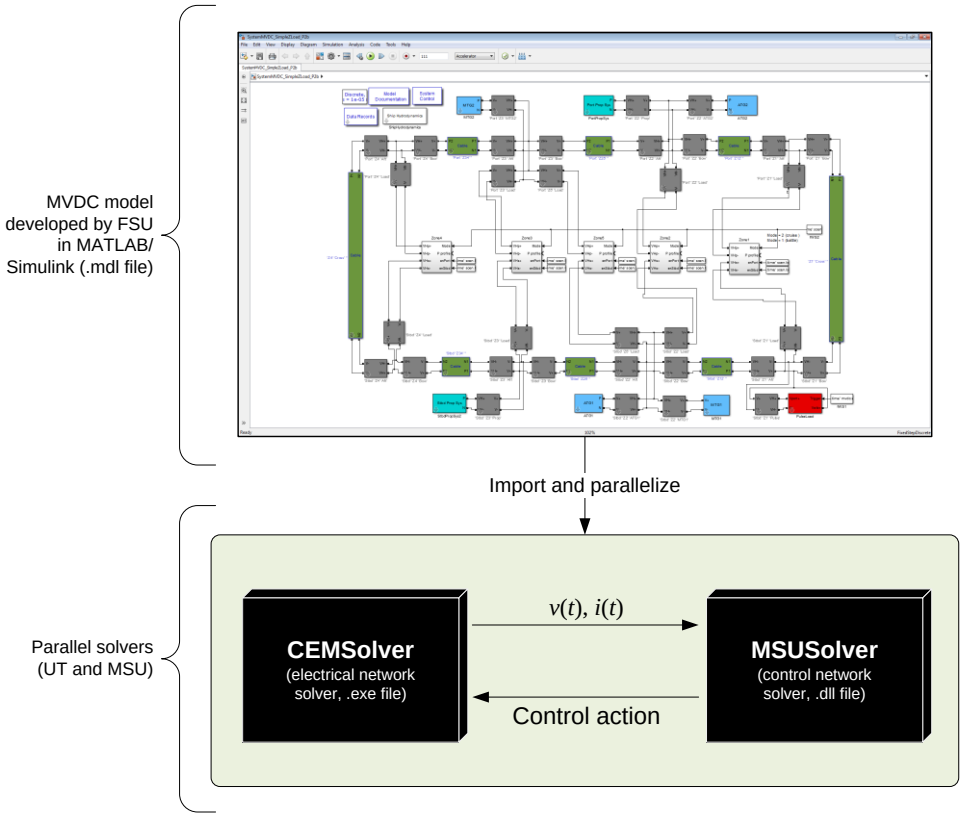


Fig. 9: Depiction of cooperative solution between CEMSolver and MSUSolver.

The remainder of this section presents a brief description of each solver.

7.2.1 CEMSolver

CEMSolver is an *electrical network* solver that imports, partitions, parallelizes, and accelerates the simulation of models built in *Simulink*. Although *CEMSolver* cannot import *any Simulink* model, among the models it has successfully imported is the scaled-down MVDC model used in this work (Fig. 1).

CEMSolver imports *Simulink* models by accepting a single model-file as input and identifying all electrical blocks and their interconnections. After identifying the electrical blocks, *CEMSolver* creates a hyper-graph that represents the power system model, and then partitions it by invoking *hMetis* [11],[12] (a free hyper-graph-partitioning tool developed by the University of Minnesota). Invoking *hMetis* by passing-in the hyper-graph of the power system at hand returns *p* different vertex sets, where *p* represents the number of partitions specified by the user, and each vertex set represents the electrical blocks (i.e., power apparatus) belonging to the same partition.

After partitioning the representative graph, *CEMSolver* constructs electrical network matrices for each partition. The parallel solution of these partitioned matrices on multicore processors results in speedups that typically vary by model size, number of partitions, and hardware used. For example, small models (e.g., <100 states), do not exhibit high speedups. For such cases, users are encouraged to continue solving their models in *Simulink*. For larger models (e.g., >1,000 states), however, speedups can result in 50x. (Exact speedup is not currently predictable by the authors.)

Technical details on how *CEMSolver* partitions power system models are available in [1-3],[8],[13]. The partitioning strategy used by *CEMSolver*, however, is one of many possible ways to partition models. There are other partitioning methods available in the literature [14-24] as well—but those used *specifically* on shipboards can be found in [1],[8],[9],[22],[25-32]. The references above suggest that the field of power system partitioning is not new [33]; however, it has progressed significantly over the last two decades through its many contributors [16],[18],[21],[23],[24],[28],[34-40] and it is an opportune time to apply it to desktop multicore computers.

7.2.2 MSUSolver

MSUSolver is a control network solver that focuses on solving control blocks (the non-electrical blocks) in *Simulink* models. At each time-step, it accepts input from *CEMSolver* as an array of signal values generated by the electrical blocks and transforms it into another array of responses to be assimilated by *CEMSolver*. The output array of *MSUSolver* controls the behavior of *CEMSolver* during run time.

MSUSolver accepts the same *Simulink* file (.slx file) as input. From this input, it identifies the list of signals to be exchanged between the solvers, as well as the control blocks together with their connectivity and dependencies. The list of signals defines the input and output arrays. The network of the control blocks (represented as a set of hyper-graphs) defines the algorithm for the array transformation. Because many individual *Simulink* control blocks represent a very simple functionality (e.g., a sum or a product), the blocks are combined into groups. For more complex components (such as a time integrator) *MATLAB* code is developed to mimic the functionality of

the corresponding *Simulink* component. The resulting algorithm is then optimized. In the final step, the optimized algorithms are translated into C# code, compiled, and packaged as a *Windows* Dynamic Load Library (DLL) to be invoked by *CEMSolver*. At present, most of this is done manually, with automation of this process planned for the future.

The validation of *MSUSolver* was performed in two steps. First the optimized algorithm, implemented as a set of *MATLAB* functions, is compared against the original *Simulink* implementation, and then the C# code is compared against the optimized algorithm. The first step verifies that the physical phenomena are captured correctly; while the other step verifies the correctness of our procedures for generation of the equivalent C# code.

7.3 Results

This section presents the results of using *CEMSolver* and *MSUSolver* to import and cooperatively accelerate the simulation of the scaled-down MVDC model. Since solver communication was designed, tested, and verified over the course of the work, the results next focus on speedup and accuracy rather than on data exchange and communication performance.

Before presenting the results, the simulation events are described. The events provide readers with information on what events were included in the simulation, and at what times they were triggered. Following, a description of the hardware shows the equipment used to run the simulation. The speedup results will compare simulation performance on two machines (a desktop and a laptop). Accuracy will be presented as "visual" and will show how the results produced by *CEMSolver* and *MSUSolver* relate to those produced with *Simulink*.

7.3.1 Simulation Events

The simulation events are listed in Table 1. After starting the simulation at t =0 s, the first event was the simultaneous closing of all (four) generator breakers at t = 0.25 s. The closing time allowed sufficient time for all four generators to produce rated voltage at rated speed. (All machines started from a full-speed initial condition.). The next event at t = 0.5 s was a single-phase fault (phase *a* to ground) at the input terminals of MTG1's rectifier (noted by fault symbol in Fig. 1). The fault was a temporal fault triggered at t = 0.5 s that cleared itself at t = 0.6 s. The fault impedance was set arbitrarily to 1 mΩ. To limit control complexity, it was assumed there was no breaker operation to clear the fault.

The simulation stop time was varied as t_{stop} = 1 s, t_{stop} = 5 s, and t_{stop} = 10 s. Each simulation stop time imposed different memory storage requirements to contain the simulation data sets. It will be shown later how memory pressure exerted by large data set sizes affected simulation performance on each computer.

Table 1: Simulation Events

Time (s)	Event
0	Simulation starts
0.25	Three-phase ac breakers of all four generator close
0.5	Fault applied at the input terms of MTG1's three-phase rectifier (phase *a* to ground)
0.6	Fault clears itself
1, 5, 10	Simulation stops time (three different runs)

7.3.2 Hardware

Two computers were used to benchmark the run times of *Simulink* against *CEMSolver*. The basic specifications of these computers are listed in Table 2 and Table 3. These computers are commercial, every-day off-the-shelf multicore computers running *Windows* 7 with four cores each. The differences between these two computers are the processor type and the amount of memory (RAM) to contain the data sets. Both computers ran the same versions of *Simulink* (2012b) and *CEMSolver*.

Table 2: Computer 1 (Laptop)

Brand & Model	Acer TravelMate 8481T-6873
Memory (RAM)	8 GB
Operating System	*Windows* 7 (64-bit) with Service Pack 1
Processor	Intel Core™ i7-2637M
Speed	1.7 GHz with Turbo Boost up to 2.8 GHz
Cores	4

Table 3: Computer 2 (Desktop)

Brand & Model	Dell Precision T7500
Memory (RAM)	12 GB
Operating System	*Windows* 7 (64-bit) with Service Pack 1
Processor	Intel Xeon™ E5630
Speed	2.53 GHz
Cores	4

Since *CEMSolver* was only tested on quad-core computers, *CEMSolver* partitioned the scaled-down MVDC model into four partitions (in all runs). Partitioning models into fewer partitions than cores has shown not to be optimal for select models in the past. However, in other cases, partitioning models into a different number of partitions than cores has increased performance. Further testing is needed on machines with more- and less-than four cores.

7.3.3 Speedup

Equation (1) defines speedup [41] s the ratio of unpartitioned (t_{unp}) and partitioned (t_{part}) simulation run times, where runtime is considered the *solution* time of each program (excluding including initialization time). Speedup was computed using (1). For parallel simulations to be considered effective, speedup should be much larger than one.

$$Speedup = \frac{t_{unp}}{t_{part}} \tag{1}$$

The numerator in (1) is the time it took *Simulink* to complete a simulation run using its fixed-step discrete solver, the backward Euler integration method, a timestep of Δt = 10 µs, and running in normal simulation mode. It should be noticed that the results presented were averaged over several runs, and that they are specific to the machines listed in Table 2 and Table 3. The machines that conducted the simulations are ordinary *Windows* desktop machines, which imply that they simultaneously ran other programs in the background during the simulations. The execution of other programs produced variance in the simulation run times, which was mitigated by averaging several run times.

Table 4 shows the results of benchmarking *Simulink* vs. *CEMSolver* using the two computers described in Table 2 and Table 3. Each computer executed several simulations using three different stop times (t_{stop} = 1, 5, and 10 s) to solve the reduced model. The data bars in Table 4 are (courtesy) visual cues to show how values change across columns (i.e., for each t_{stop} setting). The blue data bars refer to run time. The green data bars refer to speed up. The data bar width (or size) shows how the numerical number it represents compares to the largest value in the same row.

Table 4: Timing Results

Computer 1 (laptop, 8 GB, 4 cores)						
	1-sec run		5-sec run		10-sec run	
	Simulink	*CEMSolver*	*Simulink*	*CEMSolver*	*Simulink*	*CEMSolver*
Initialization time (mm:ss)	00:09	00:04	00:12	00:08	00:07.6	0:00:49
Run time (mm:ss)	02:54	00:47	14:01	04:19	0:30:36	0:19:46
Frame time (µs)	1,740	470	1,680	518	1,830	1,190
Frame rate (fps)	575	2,128	595	1,931	546	840
Speedup	3.7		3.2		1.5	

Computer 2 (desktop, 12 GB, 4 cores)						
	1-sec run		5-sec run		10-sec run	
	Simulink	*CEMSolver*	*Simulink*	*CEMSolver*	*Simulink*	*CEMSolver*
Initialization time (mm:ss)	00:10	00:03	00:08	00:10	00:07.6	0:00:18
Run time (mm:ss)	02:08	00:34	10:45	03:04	0:21:36	0:06:44
Frame time (µs)	1,280	340	1,290	368	1,294	404
Frame rate (fps)	781	2,941	775	2,717	773	2,475
Speedup	3.8		3.5		3.2	

The timings quantified in Table 4 are defined as follows:

- *Initialization time*: the time required to initialize each solver. In *Simulink*, this is the time measured from the moment users click "Play" to right before the beginning of the simulation. In *CEMSolver*, this time is measured from the program's launch to immediately *before* starting the simulation. This time normally includes, but is not limited to, validating user input (e.g., parameters), allocating memory to store simulation data, creating network matrices, and executing all routines required to prepare a solver for simulation. This initialization time was *not* used to measure speedup.

- *Run time*: the time spent solving the power system model over the intended time span. This run time is normally lengthy for large models, and represents the time spent on the *time loop* (the crux of the simulation). Run time was used to calculate speedup.

- *Frame time*: the average time spent on each simulation time step. In real time simulators, this number equals the simulation timestep size (Δt). In offline programs such as *Simulink* and *CEMSolver*, this number is less than Δt for small models, but typically much larger than Δt for large models as the one used for this work.

- *Frame rate*: the average rate (speed in frames per seconds, fps) at which the solver advances the simulation. The frame rate is the inverse of the frame time.

- *Speedup*: the ratio of *Simulink*'s run time and *CEMSolver*'s run time as defined by equation (1).

7.3.4 Performance on Computer 1

Referring to the initialization times of the three runs on computer 1 in Table 4, it is seen that *Simulink*'s initialization times are consistent. *CEMSolver*'s initialization times increased noticeably for the 10-sec run. The increased initialization time was caused by *CEMSolver*'s pre-allocation of memory for data storage, which exerted considerable pressure on the laptop's memory capacity (6 out of 8 GB were utilized). The initialization time of *CEMSolver* for the 10-sec run degraded considerably as noted by the increase from 4 s to 49 s.

The run times for computer 1 are highlighted with blue data bars. It is shown that *CEMSolver* ran faster than *Simulink* on all counts. In the best case, the simulation reduced from nearly 3 minutes to less than 1 minute. In the slowest case, the simulation reduced from half-an-hour to less than 20 minutes (a 10 minute reduction).

The frame times on computer 1 for *Simulink* were all in the millisecond range. *CEMSolver* showed microsecond (μs) performance levels for all but the 10-sec run. It is interesting to note that *CEMSolver* reached microsecond performance, as this is indicative that it may be possible to approach real time simulation speeds in the future.

The highest speedup for computer 1 was 3.7x; however, speedup decayed for each run of increasing stop time. The decay is because, at present, *CEMSolver* requires improvement to manage large data sets. Currently, *CEMSolver* pre-allocates memory for the data it produces, which exerts considerable memory pressure commensurate to the simulation stop time. The pressure can be alleviated, for example, by persisting data to the hard disk during run time.

7.3.5 Performance on Computer 2

Computer 2 is considered a workstation of slightly higher performance than typical desktop computers. However, this computer is available commercially and inexpensively. Comparing the initialization times for the three runs on computer 2, *Simulink* remained consistent for the three stop times. *CEMSolver*'s initialization was faster than for computer 1, but it also increased with stop time (as expected). For the 10-run case, 6 GB of data storage in memory only constituted 50 % of the computer 2's memory capacity (vs. 75 % of the capacity of computer 1).

The best run time reduction was on computer 2 for the 1-sec run. The simulation time reduced from approximately two minutes to nearly 30 seconds. This is an important result as it permits running more case studies per day when using *CEMSolver*—even for a small model such as the one used in this work. Similarly, the 5-sec run time was reduced from approximately 10 minutes to 3 minutes. These run time reductions constitute an important result toward the acceleration of FSU's larger shipboard model, which will be the subject of continued work.

On computer 2, *CEMSolver* performed in the microsecond frame-time range for all runs while *Simulink* performed in the millisecond range. It is desirable to perform in microsecond ranges as it suggests potential for real-time simulation (albeit far away).

The best speedup was 3.8x on computer 2 for the 1-sec run. Although higher speedups have been obtained with *CEMSolver* with the AC-radial model, they were obtained for a model that is

deemed larger than the model used for this work. It has been reported before that *CEMSolver* may not be usable for small models. The scaled-down model used in this work can be considered as a "lower-limit" to define what a small model means as its speedups exceeded 3x and are considered acceptable by the authors.

The following observations are made on performance. In its current version, *MSUSolver* is not a parallel solver. *MSUSolver* is invoked sequentially by *CEMSolver* at the end of each time step, which puts *CEMSolver* in a "wait state." Although the computational work exercised by *MSUSolver* is not significant due to the limited controls considered, transitioning *CEMSolver* in and out of the wait state was found to affect simulation performance significantly.

Waiting for data arrival in parallel (multithreaded) programs is expensive due to operating-system kernel-mode thread-blocking transitions. The blocking transitions took longer than it did for *MSUSolver* to solve the control network at each simulation timestep. That is, theoretically, *CEMSolver* would run faster if it did not have to block and wait for *MSUSolver*; however, without *MSUSolver*, *CEMSolver* cannot produce complete results.

7.3.6 Accuracy

CEMSolver produces the same results whether *CEMSolver* runs a partitioned simulation or an un-partitioned one. However, *CEMSolver* does *not* return the same results as *Simulink* (not even when *CEMSolver* runs in its unpartitioned mode). Because solvers developed by different entities are expected to show differences [42],[43] , the results produced by *CEMSolver* were not expected to agree entirely with *Simulink*—moreover, this expectation deferred a thorough investigation of the differences at this stage of the work.

The simulation results shown in this section were taken, arbitrary, from measurement locations 1, 2, 3, and 4 in Fig. 1. A thorough contrast of the results produced by *CEMSolver* and *Simulink* would require orchestrating ubiquitous measurements throughout the model, for all phases, and for each time step of the simulation. Such analysis can be important, but it was not addressed in this work. Instead, a visual comparison (side by side) of the measurements taken at the four locations is presented next.

7.3.7 Measurements at Location 1

Fig. 5 shows the three-phase voltage and current waveforms collected from location 1. The left column shows the results produced by *Simulink* while the column on the right shows the results produced by *CEMSolver*. Comparing the voltage envelopes (top row) of both programs, the results appear to be consistent. The voltages are zero behind the three-phase breaker until the four generator breakers close simultaneously at $t = 0.25$ s. Upon closing the breakers, full voltage appears at the input terminals of each generator rectifier as expected.

The bottom row compares the current output of both programs. At $t = 0.25$ s, a current transient is observed due to the charging currents for the generator rectifiers. The peak of these charging currents appears consistent in both programs. Although FSU's MVDC model includes charging circuits to mitigate these large charging currents, at present the control logic for the charging circuits was not modeled in the scaled-down version of the MVDC model. From Fig. 5, the charging currents reached per-unit (p.u.) levels of approximately +/- 2 p.u. on a current base of

I_{base}= 6,500 A. Although the charging currents of both programs appear similar, their differences were not investigated.

From the bottom row of Fig. 5, which shows currents, it is noticed that the fault at t = 0.5 s caused only phase *a*'s current (blue waveform) to increase as expected. Comparing the results of both programs, the *envelope* of the faulted current appears consistent in both programs.

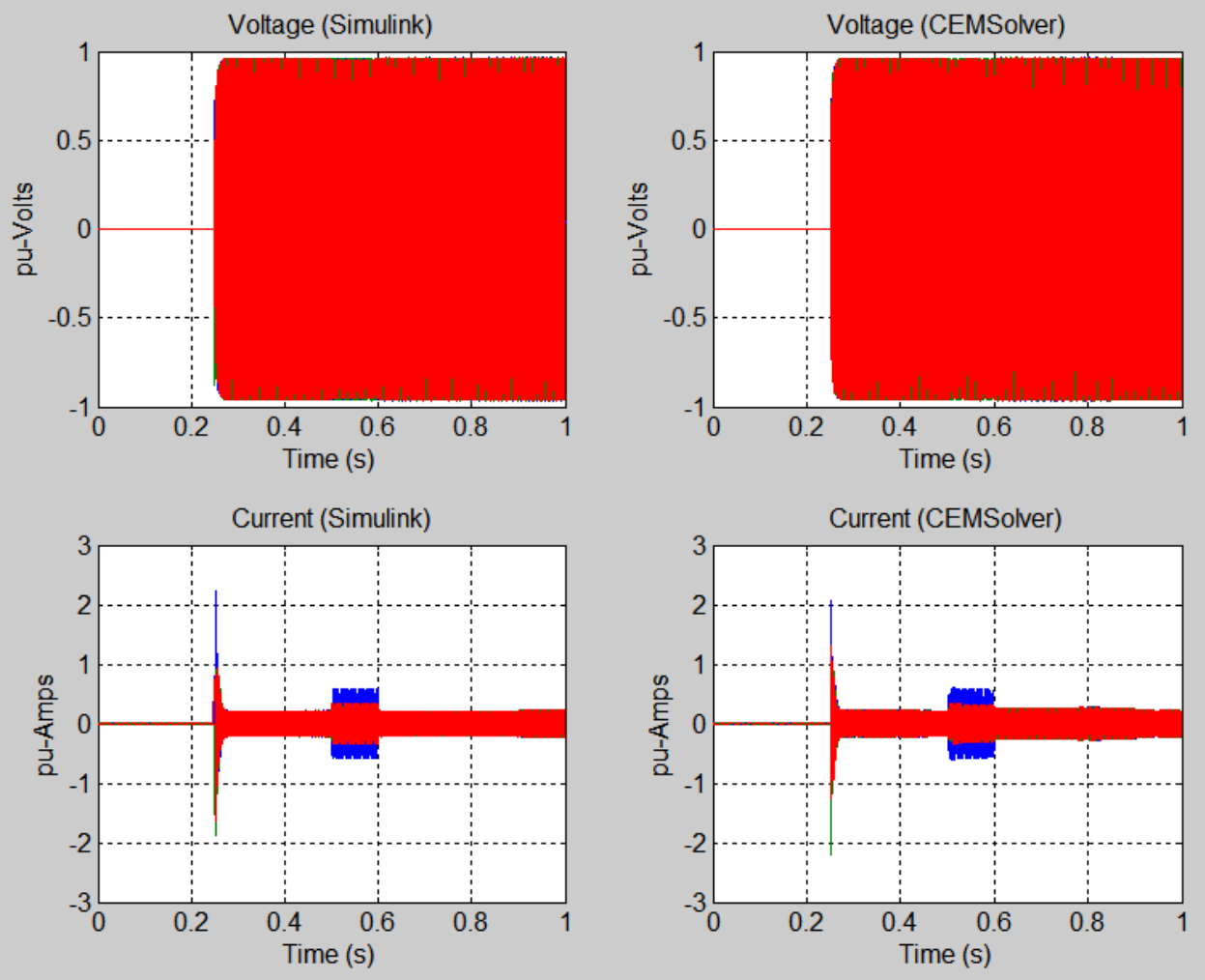


Fig. 5: Measurements at location 1 (three-phase AC waveforms, where colors green, blue represent phases *a*, *b*, and *c*, respectively)

To examine the results more closely, Fig. 6 shows a close-up of Fig. 5 between t = 0.49 s and t = 0.51 s. The top row shows that the three-phase line voltages are similar in both magnitude and frequency, but not in phase. The phase difference is noticed by contrasting the colors of the top two voltage traces. Colors red, green, and blue represent phases *ab*, *bc*, and *ca*, respectively. Visual inspection of the voltage trace colors suggests the results of *CEMSolver* lead the results produced by *Simulink*. Similarly, the frequency appears to be the same, but the number of peaks in each voltage trace is slightly different. This suggests that the generator frequencies are slightly different in both programs as will also be shown for measurement location 4.

The bottom row in Fig. 6 elucidates the fault current in phase *a* (blue trace) and seems consistent in magnitude in both programs. However, it is noted that the breaker closing's incident angle is

different because of the aforementioned displacement (phase) differences in the waveforms. This caused the faulted phase's current waveform to appear 180^{o} out of phase.

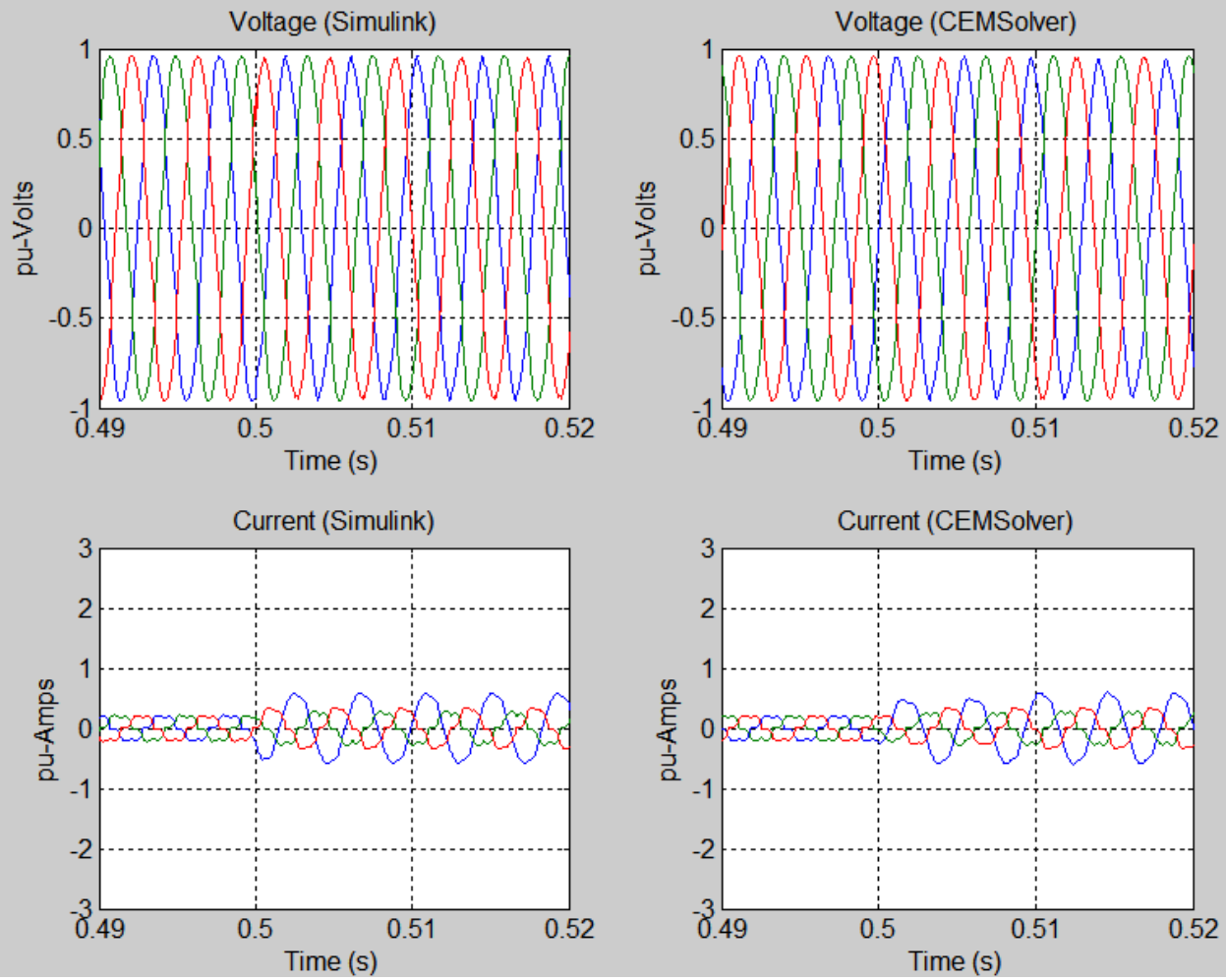


Fig. 6: Measurements at location 1 (three-phase ac waveforms in-front of fault).

7.3.8 Measurements at Location 2

Fig. 7 shows the dc voltage and current measurements collected at location 2. The left column shows the results produced by *Simulink* while the column on the right shows the results produced by *CEMSolver*. Comparing the dc voltage envelopes (top row), the results of both programs appear consistent in rise time and lower voltage value. (The dc voltages are also zero until the generator breakers closed at t = 0.25 s.)

The bottom row in Fig. 7 compares the dc current flowing out of the rectifier terminals (measured before the dc capacitor). The peak charging current returned by *Simulink* is slightly higher than the one produced by *CEMSolver*. Differences noted in RLC branches are due to the use of different integration algorithm's [2]. *Simulink* uses either backward Euler or the trapezoidal rule. *CEMSolver* uses root matching. (5th order methods are also in use [44].) These three integration algorithms normally agree during steady-state, but not during transient regimes. In both programs, however, the dc charging currents seem to have reached similar values at the same time. From the voltage and current charts, the envelopes throughout the simulation appear

consistent—even during the fault. However, the voltage and current envelopes suggest a mismatch in the ripple.

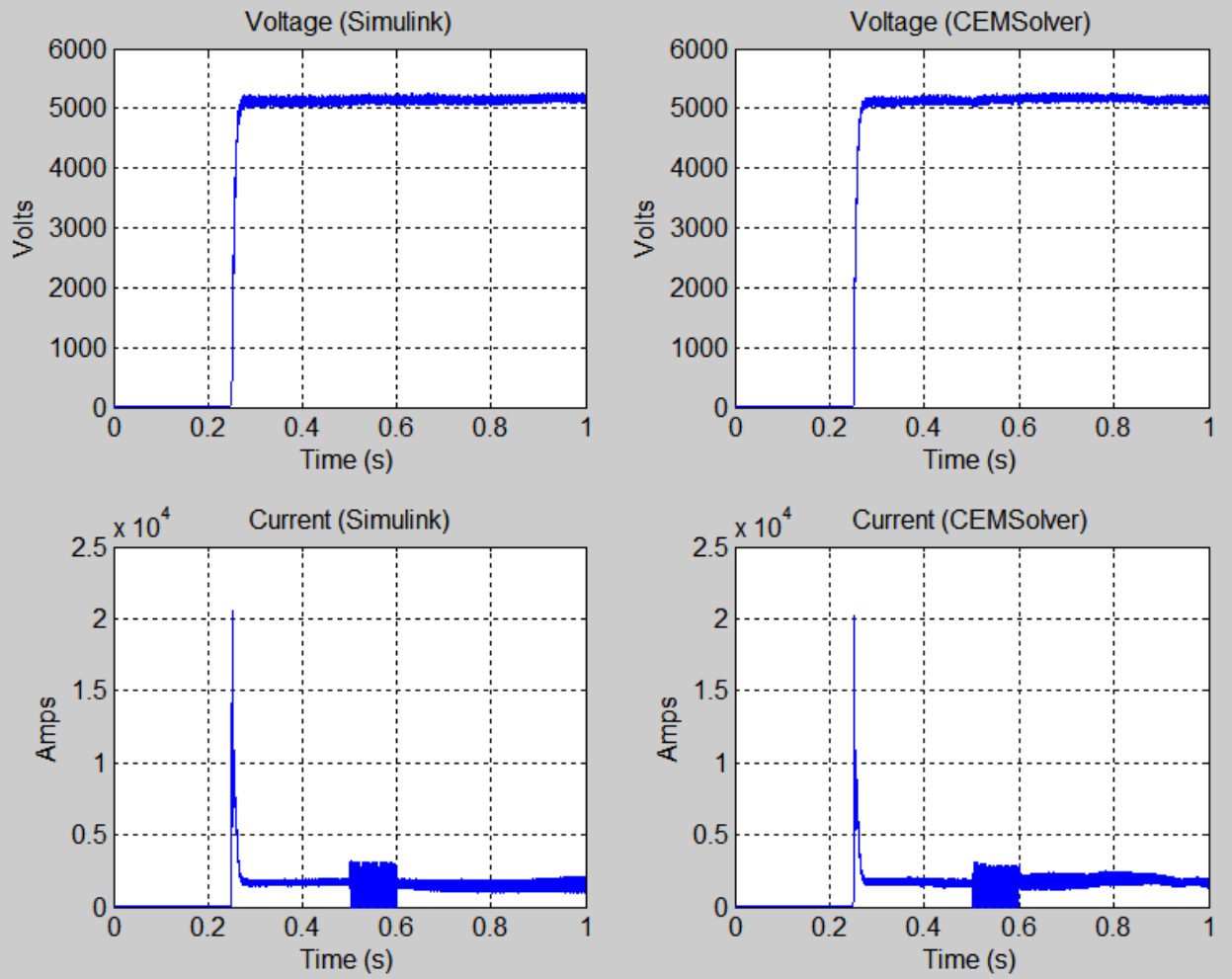


Fig. 7: Measurements at location 2 (DC waveforms at the output terminals of MTG1's rectifier).

Fig. 8 shows a close-up of Fig. 7 to highlight initiation of the single-phase fault. During the fault, the voltages produced by *CEMSolver* seem similar to those returned by *Simulink*'s except for the dip observed when the fault initiated. The voltage produced by *CEMSolver* undergoes a small dip that could be caused by larger branch impedances upstream of measurement location 2. In both cases, however, the upper and lower voltage limits appear similar.

The bottom row in Fig. 8 shows a close-up of the rectifier output current during the fault. Both programs seem to have produced similar currents leading up to the fault at $t = 0.5$ s. At the time the fault was applied, the current in *Simulink* shot upwards while the current in *CEMSolver* shot downwards. The different in current direction may be related to the displacement differences noted earlier. The source of displacement difference was not investigated, but it is also (likely) responsible for switching-instant differences in power converters (although it was not investigated in depth.) Similarly, it is noticed that while the current peaks seem similar, *CEMSolver* shows a decay in fault current while *Simulink* maintains a quasi-steady level. The reasons for these differences were not investigated.

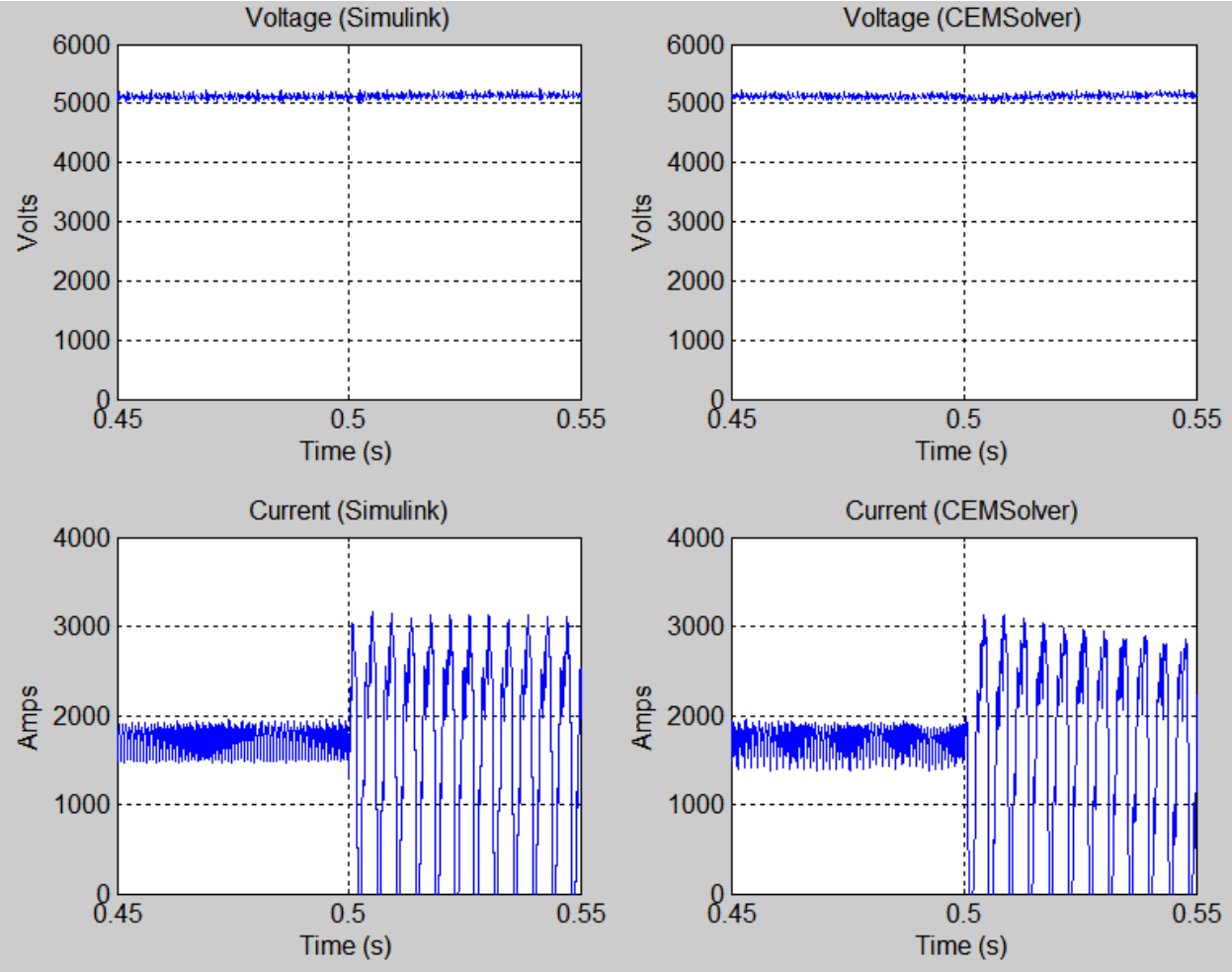


Fig. 8: Measurements at location 2 (close-up of Fig. 7).

7.3.9 Measurements at Location 3

Fig. 9 shows the dc voltage and current measurements at location 3 (load bus in zone 1). The left column shows the results produced by *Simulink* while the column on the right shows the results produced by *CEMSolver*. The voltage levels in both programs show the result of the dc/dc converters stepping-*down* the main dc voltage from ~5 kVdc to ~800 Vdc. The conversion took place through the parallel converters in each of the five zones as shown in Fig. 3. In each zone, each dc/dc converter consisted of a two-arm, PWM-controlled inverter, a single-phase high-frequency isolation transformer, and two-arm rectification.

The voltage and current output of both programs is nearly consistent, which suggests that the converters were modeled consistently in both programs. The load was modeled as a static load for simplicity, but subsequent versions of the solvers will support variable loads as found in FSU's larger MVDC model. Similar to the voltage, the load's current envelope was consistent in both programs.

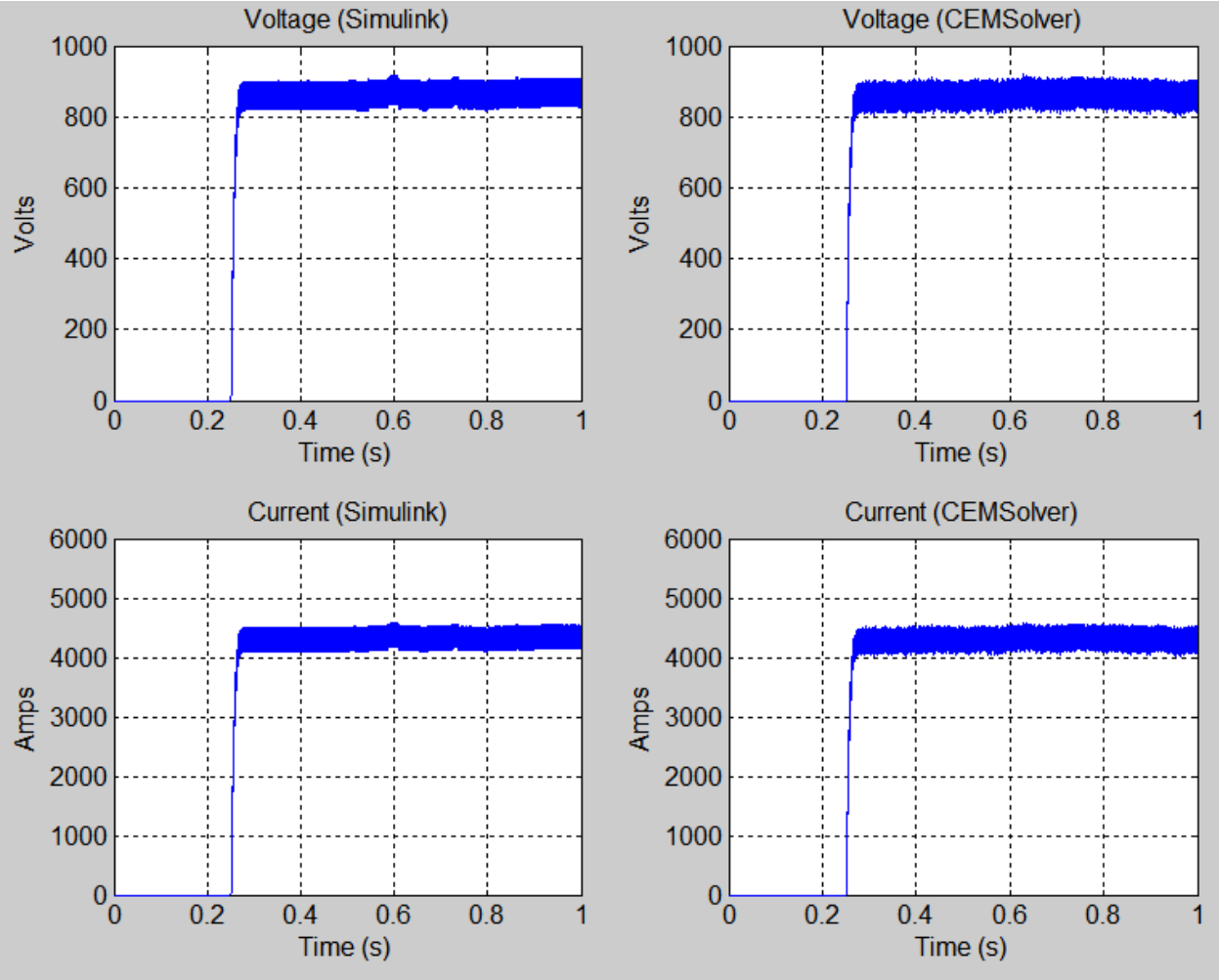


Fig. 9: Measurements at location 3 (DC waveforms at the load of zone 1).

Fig. 10 shows a close-up of Fig. 9. Although the dc envelopes in Fig. 9 appeared consistent earlier, a close-up of the results shows jitter [45] in *CEMSolver*'s voltage and current waveforms. The cause of the jitter was not investigated, but it was likely caused by late (i.e., switched at the next timestep rather *between* timesteps) switching transitions in the converters [44],[46]. The voltage and current results in all other zones (2 through 5) are similar to what is shown in Fig. 9 for zone 1.

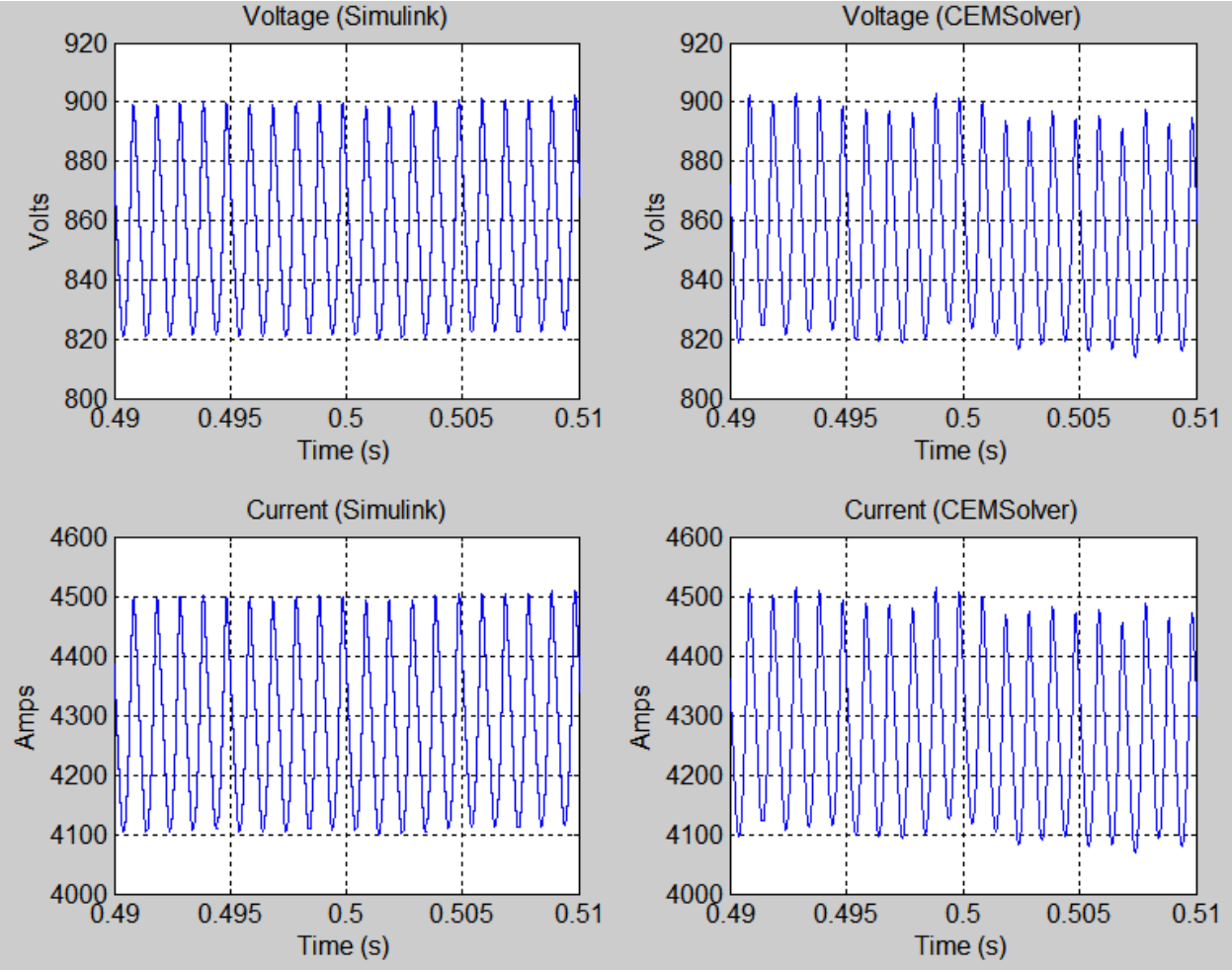


Fig. 10: Measurements at location 3 (close-up of Fig. 9).

7.3.10 Measurements at Location 4

Fig. 11 shows ATG1's stator induced voltage (EMF), power relations, and mechanical speed (measurement location 4 in Fig. 1). The left column shows the results produced by *Simulink* while the column on the right shows the results produced by *CEMSolver*. The impressed electromotive force (EMF) on the machine stators in both programs are consistent, which suggests the line voltages for ATG1 computed by *CEMSolver* are correct, and that the excitation control solved with *MSUSolver* returned acceptable values.

The second row in Fig. 11 shows the power relations in ATG1. The red waveform shows the mechanical input power to the machine (prime-mover power computed by *MSUSolver*). The green waveform shows the electrical power demand seen by ATG1 (computed by *CEMSolver*). The red waveform shows the accelerating power, which is the difference between mechanical and electrical power. Comparing the results of both programs, the electrical power in *CEMSolver* shows spurious spikes that did not appear in *Simulink*. The instantaneous power spikes apparently are caused by voltage or current spikes, which may have been produced by late switching transitions in a converter. (The issue of preventing negative currents in semi-conductors remains a challenging simulation topic [47]—even for *CEMSolver*.) Despite the spurious transients reported by *CEMSolver*, the power traces have consistent envelopes.

The lower chart row shows slight differences in the speed profile for ATG1. It appears that the rotor speed decayed more rapidly in *CEMSolver* than it did in *Simulink*. The reason for the larger decay was not investigated, but it may be related to how each program solves for the rotor speed. *Simulink* uses a torque-based (linear) variant of Newton's second law for rotational motion to compute speed, while *CEMSolver* uses a power-based (non-linear) version of the same equation to solve for speed. Although the two equations should give the same result, how *Simulink* calculates torque or how *CEMSolver* calculates power may produce implementation differences that reflect on the speed. A comparison of these two rotor-speed calculation methods was not investigated. In addition to the differences in speeds, there is a slight overshoot in the speed reported by *CEMSolver*. Although the speed (and hence ac frequency) under- and over-shoots are small, they may be responsible for the phase and frequency deviations noted earlier.

Fig. 11 demonstrates the interaction between *CEMSolver* and *MSUSolver*. After solving the electrical network, *CEMSolver* sends the electrical power output of each generator (P_{elec} trace) to *MSUSolver*. *MSUSolver* takes the electrical power signal from *CEMSolver* and solves the equations corresponding to the same machine's prime mover and governor controllers. When the net mechanical power out of the prime mover is calculated, *MSUSolver* passes this mechanical power signal (P_{mech} trace) back to *CEMSolver*. When *CEMSolver* receives the mechanical power input signal, it integrates the accelerating power ($P_{accel} = P_{mech} - P_{elec}$) to calculate rotor speed (lower magenta trace) using the power-based version of Newton's second law for rotational motion.

Overall, the data exchanged between *CEMSolver* and *MSUSolver* was successful; however, the data exchange is also sensitive. Errors in data exchanged (or corrupted data) can accumulate over the duration of the simulation and lead to long-term divergence. Although divergence effects were not investigated, the simulation runs through $t_{stop} = 10$s appeared to have exchanged good data—that is, significant divergence was not noticed in the present state of the two solvers.

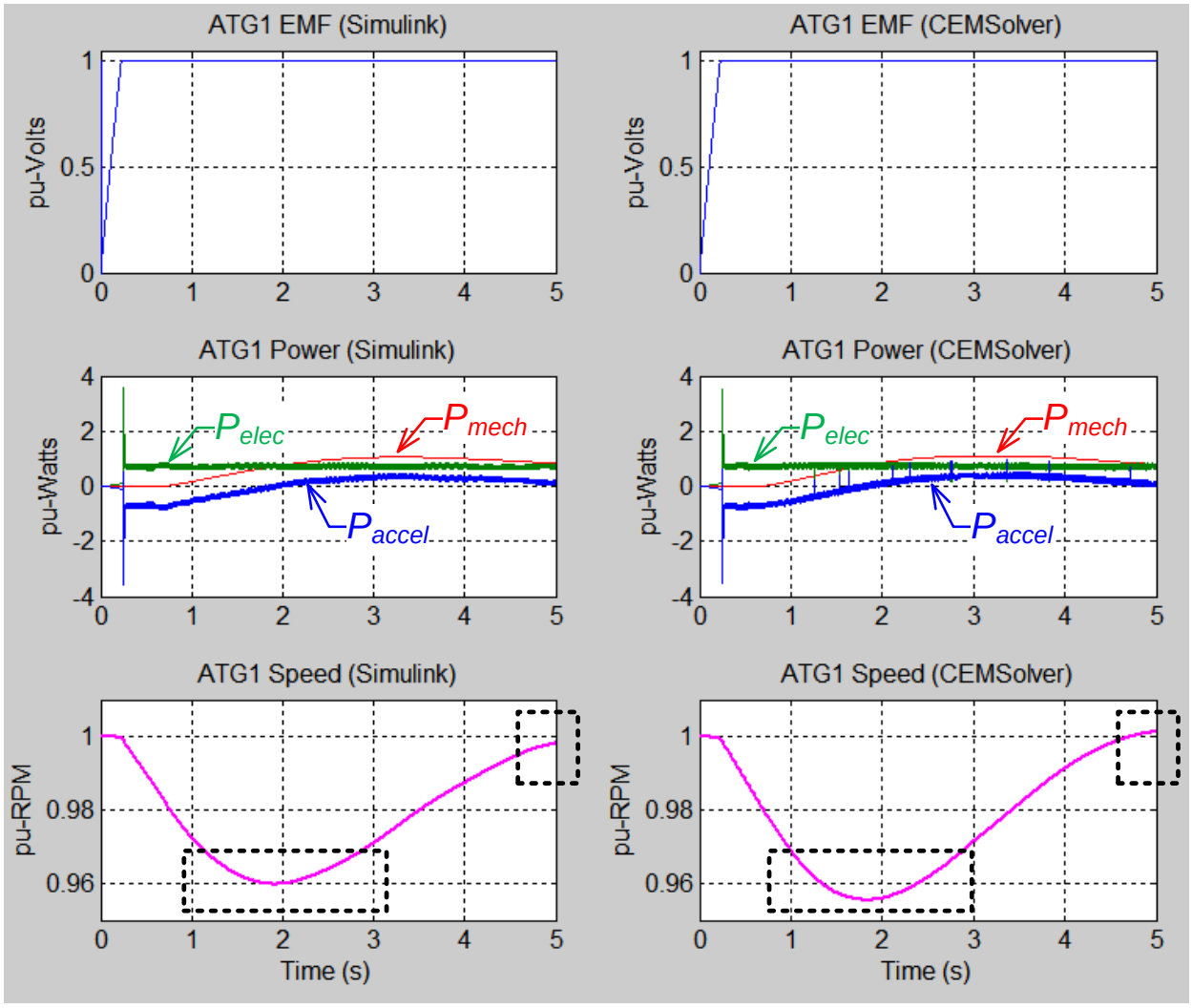


Fig. 11: Measurements at location 4 (EMF, power, and speed profile for ATG1).

7.4 Appendix Summary and Conclusions

This work tested the ability of two ESRDC solvers to accelerate a scaled-down version of the MVDC model built by FSU. The first solver (*CEMSolver*) is a parallel solver that accelerates the simulation of electrical networks by partitioning (and parallelizing) power system models into smaller subsystems of less computational burden (developed by UT). The second solver is a sequential solver that solves control networks (developed by MSU).

The metrics of success were defined as solver communication, speed, and accuracy. The solvers communicated successfully by 1) exchanging valid data throughout the simulation, 2) running to completion without abnormal terminations, and 3) by producing results consistent with *Simulink*. The data exchanged was valid and bounded—that is, neither incongruous values such as "NaN" (not a number) nor infinite values were produced by either solver. Although the solvers communicated valid values, occasional spurious spikes and minor divergences were noted. These data anomalies, however, did not prevent the results from being in general agreement with the results produced by *Simulink*. Additional work, however, is required to find and eliminate the numerical imperfections.

Speed was measured by comparing the run time of *Simulink* vs. *CEMSolver* over simulations of different length (i.e., different stop times) using two different quad-core computers (a laptop and a desktop). The speedup varied with the chosen simulation stop time due to the memory pressure exerted by the data produced. Large data sets were generated and stored by both programs, but *Simulink* appeared to be more effective in managing large data sets and alleviating memory pressure. With a time step of Δt=10 μs, the amount of data stored by *CEMSolver* was large, but it could have been persisted to disk to avoid incurring wait penalties.

It was also noted that speedup varied by computer used and simulation stop time. Computer 1 (with less memory) had added memory pressure compared to computer 2, which caused *CEMSolver* not to perform as well as it did on computer 2. However, in simulations of short duration (smaller stop time), *CEMSolver* performed well on both machines.

The speedups varied between 1.5 and 3.7x, which is small compared to results obtained previously with *CEMSolver* alone (without *MSUSolver*) on a large shipboard model. This reduction in performance was not due to the work done by *MSUSolver*. Instead, it was partly due to the small model size being solved (*CEMSolver* is not efficient at accelerating small models) and a newly-introduced "wait state" in which *CEMSolver* blocks all of its threads until *MSUSolver* solves the control network. The time spent by *MSUSolver* doing work was relatively short and did not contribute to a low acceleration. In this particular simulation scenario, solving the control network was not as large a factor in determining the total system solution time as that of the physical problem handled by *CEMSolver*. Nevertheless, real-world power problems normally cannot be solved without both electrical (*CEMSolver*) and control (*MSUSolver*) solutions.

It is considered that the speedup obtained in this work is small, but it is commensurate to the model size used and to the number of independent cores available to solve it (four). It has been reported before [1],[9] that *CEMSolver*'s speedups follow model size rather than core count—and in this work, the scaled-down MVDC model used was small. Additional work is required to improve how *CEMSolver* handles large data sets and how it should handle (or avoid) unnecessary wait states.

The reasons for low acceleration also include a "flat-fee" overhead imposed by the *Windows* operating system when running parallel programs on shared-memory multicore processors. Details of this overhead is outside the scope of this work, but the reasons include a high amount of kernel-mode thread context switching and thread cross-core migrations. *CEMSolver* does not control how many threads are used or what cores the threads are executed on— *Windows* does. A *Windows*-based thread-pool-managed program is advantageous to the programmer as it results in less code and thread management, but the resulting overhead cannot be controlled by the programmer. Additional contributors to simulation overhead (also outside the scope of this work) are the corrective actions taken by *CEMSolver* to prevent negative semi-conductor currents. These actions ensure the simulation results are valid when simulating networks containing power electronic devices.

It should is also be pointed out that neither *Simulink* nor *CEMSolver* were the only programs running during the simulations. Background programs also received processor time (time slices) by the operating system, and it is not clear whether the same background programs ran during the execution of *Simulink* and *CEMSolver*. Running the simulation several times was an attempt to mitigate inconsistencies in the state of background programs.

Accuracy was an assumed byproduct of speedup, but it was seen that it is not the case. Confidence in accuracy is important when developing parallel solvers and it is of equal (or more) importance than speed. Although accuracy studies were not part of this work, side-by-side results showed that the results produced by *CEMSolver* were consistent at the envelope level. Several differences in the results were noted as well—especially when zooming-in on the waveforms. The differences in results stem from a variety of reasons including: difference in integration algorithms, how negative semiconductor currents are prevented, the way power apparatus and switches are modeled, the one-step delay between *CEMSolver* and *MSUSolver*, the numerical conditioning of the network matrices, and perhaps internal code-optimizations that sacrifice accuracy for speed. The presence of internal code optimizations in *Simulink* is unknown to the authors, but some exist in *CEMSolver*.

Overall, the goal between FSU, MSU, and UT was accomplished. FSU provided a well-characterized reference model of a notional MVDC system. This model was used by UT and MSU to build a scaled-down version (also in *Simulink*) to test the two solvers. The combined effort of the solvers developed by UT and MSU successfully imported the *Simulink* model, partitioned it, communicated between them, accelerated the simulation, and returned results in general consistency with those produce by *Simulink.*