Pacific Northwest
NATIONAL LABORATORY

PNNL-29301

Power Grid Computational Challenges and Metrics for Hardware Accelerator Evaluation

October 2019

Shrirang Abhyankar
Slaven Peles
Draguna Vrabie

U.S. DEPARTMENT OF ENERGY

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

DISCLAIMER

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from
the Office of Scientific and Technical
Information,
P.O. Box 62, Oak Ridge, TN 37831-0062
www.osti.gov
ph: (865) 576-8401
fox: (865) 576-5728
email: reports@osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312
ph: (800) 553-NTIS (6847)
or (703) 605-6000
email: info@ntis.gov
Online ordering: http://www.ntis.gov

PNNL-29301

Power Grid Computational Challenges and Metrics for Hardware Accelerator Evaluation

October 2019

Shrirang Abhyankar
Slaven Peles
Draguna Vrabie

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

Power Grid Computational Challenges and Metrics for Hardware Accelerator Evaluation

Shrirang Abhyankar, Slaven Peles, Draguna Vrabie

1. Objective

The objective of this document is to support the DMC initiative's research on hardware and software development. Our focus is on the power grid domain that has a rich diversity of numerical and computationally challenging applications. Through this document, we highlight two core building blocks, power flow and transient stability equations, for power grid applications. Improving computational robustness and acceleration of these core blocks will tremendously benefit all power grid applications. We also highlight the most computationally dominant numerical kernels that are being used in power flow and transient stability. The objective of this document is three-folds:

- Provide a description of the application workflows and associated hardware and software needs.
- Identify the fundamental computing challenges that are a recurring theme in many of the power grid applications.
- Provide a baseline compute performance on the CPUs to motivate the DMC research on accelerator hardware and software technologies.

2. Power grid structure, analysis needs, and current state-of-art in computing

2.1 Structure

Before we delve on the power grid structure, a few notes, to compare and contextualize the current scientific computing applications, are necessary. Most of the problems studied in large-scale scientific computing stem from applications that use partial differential equations (for e.g. heat equation or advection-diffusion). These partial differential equations (PDEs) are discretized over grids that have regular structure/topology (equal distance grid points) and are typically have 1, 2, or 3 dimensions. Such problems also have homogeneity in the equations (i.e., the same equation (or set of equations) and same type of variable set for each grid point). This greatly eases the data access and computational patterns since identical equations need to be evaluated. A second class of problems in scientific computing are the unstructured grid problems that are again described by partial differential equations. The computations are done on "cells" (a collection of grid points that have irregular patterns, such as a hexagon, triangle, or other patterns) instead of the grid points. Such problems also do (mostly) have homogeneity in the problem equations and variables.

Power grids fall in the category of "network" problems. Such networks (topological collection of nodes that have irregular connectivity) are typically associated with infrastructure networks (power grid, communication, water networks, gas networks, and others), social networks, biological networks etc. In scientific computing parlance, problems emanating from such network are loosely termed as discrete network problems. Unlike structured or unstructured grid problems, where the topology is decided to suit the problem needs, network problem use the irregular topology of the network that is a physical layout of the network itself. Moreover, power grids have an irregular topology that follow a power law distribution [Atkins2009]. Each node can have one or more incident edges. The power grid equations are typically described by purely algebraic equations or differential algebraic equations. The equipment/devices at the nodes and edges dictate the type and the number of equations at each power grid node. For instance, a node with wind generators will have different type and number of equations than a node that has industrial motors.

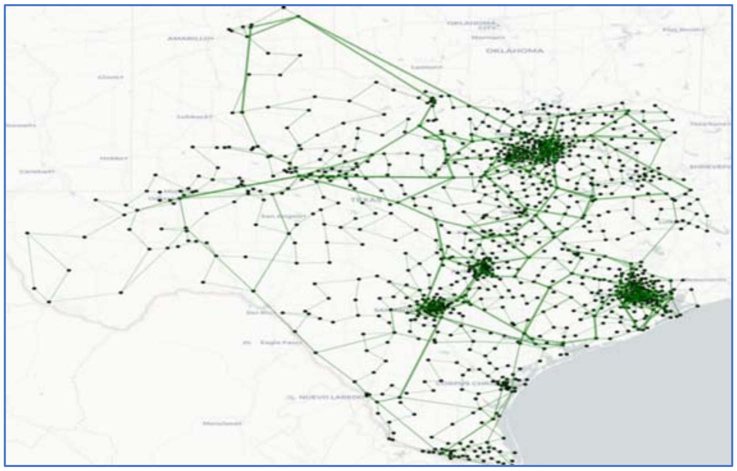


Fig. 1: Topology of the synthetic Texas 2000-bus network from TAMU Electrical Grid Test Repository. Dense connectivity at urban centers (Houston, Dallas, Austin, San Antonio) can be seen, while there is sparse connectivity in urban areas.

For analysis of these grids, the most important element is the graph Laplacian of the network. The Laplacian of the above networks, shown in Fig. 3 and Fig. 4, exhibits this interweaving between sparsity and denseness in the matrix blocks, following the topology of the grid that has densely connected urban areas and sparsely connected rural regions.

Fig. 2: Topology of the synthetic Western Interconnect from TAMU Electrical Grid Test Repository shows a different connectivity pattern with urban centers and the suburbs having a denser connection.

As described above, power grids fall in the category of networks with unstructured topology. This is a typical feature for most infrastructure networks such as transportation, gas networks, water networks, communications, and infrastructure. Such networks are described by ‘nodes’ and ‘edges’ with the topology of the network dictating the connectivity and thereby the network strength. Figure 1 and 2 show the topology of power grids for the Texas and Western Interconnect bulk power system (above 115KV). The Texas network has 2000 nodes, called “buses” in power grid parlance, and 3206 lines, while the Western Interconnect has 12706 lines. These power grids are synthetic networks created for the ARPA-E GridData competition by the Texas A&M university [Birchfield2017].

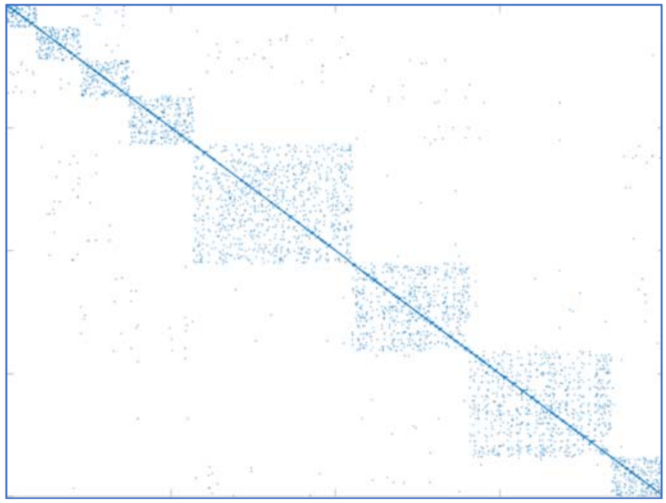


Fig. 3: Laplacian of the Texas 2000-bus synthetic system. It can be seen that the Laplacian reflects the physical topology through clustered blocks at the diagonal (representing the urban centers) and sparse connectivity between them.

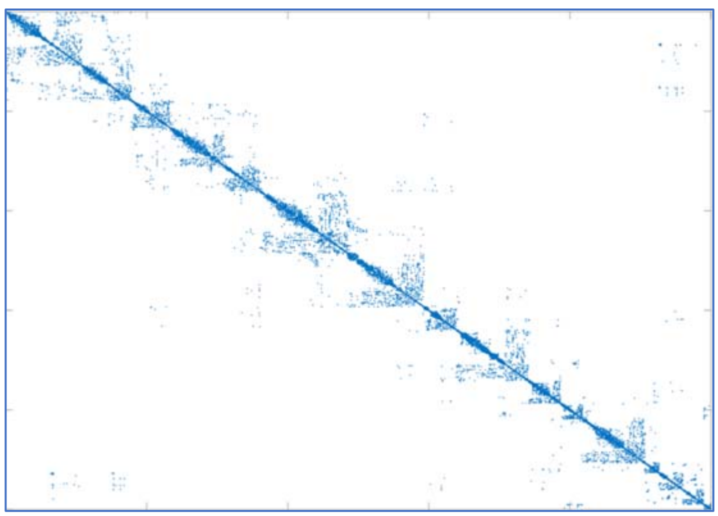


Fig. 4: Laplacian of the Western Interconnect synthetic system that has a similar structure as the It can be seen that the Laplacian reflects the physical topology through clustered blocks at the diagonal (representing the urban centers) and sparse connectivity between them. The size of the Western Interconnect (10000 nodes) is bigger than the Texas Interconnection (2000 nodes) and as such the sparsity pattern of the Laplacian is different.

2.2 Power grid applications

Power grid analysis needs span a multitude of timescales, as shown in Fig. 5, ranging from microseconds timescales for studying the effects of lightning propagation on transmission lines to years or decades for assessing the long-term planning needs such as building new transmission lines or generation facilities. This rich diversity in time-scales have led to a multitude of applications that have been developed by the power industry. These applications have modeling assumptions that help segregate the analysis based on different time-scales. Roughly, the applications can be divided into three types:

- **Electromagnetic transients**: These applications range from the microsecond (or lower) to millisecond timeframe. They have detailed modeling of the network which is akin to circuit simulation. They have detailed modeling of the network with power electronics switches. The numerical methods used for such studies are typically piecewise nonlinear differential equations with switching (hybrid nature of the problem). Due to the detailed modeling involved and the time-scales, such applications are restricted to small networks only (in the order of tens of nodes).
- **Electromechanical transients**: Applications to study electromechanical transients range in the time-frame of milliseconds to minutes. With certain modeling assumptions, they greatly reduce the size and the complexity of the equations to be solved. The numerical methods used for solving such problems are discrete-nonlinear-differential algebraic (hybrid). Such applications are used to assess the impact of disturbances on the grid to the stability of the system. With the modeling assumptions, analysis of large networks (in the order of thousands of nodes) is possible, but it is time-consuming as the system size increases.
- **Steady-state:** Steady-state studies are used for finding optimal operating point for the grid or within context of quasi-steady-state transient simulations that span the time-frame of minutes to years. They make further modeling assumptions to reduce the system to algebraic equations that model the power flow in the network. There are various applications that span the steady-state regime with their own numerical solution needs. The most common examples are (a) power flow – nonlinear algebraic problem, (b) optimal power flow or economic dispatch – nonlinear or linear constrained optimization problem, (c) unit commitment – linear mixed integer programming.

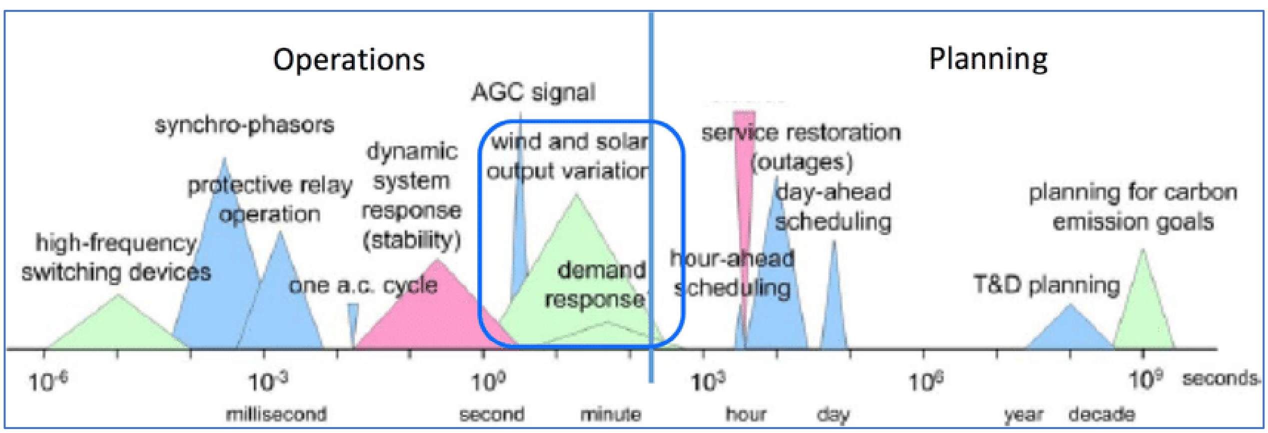


Figure 5: Power grid operations and planning time-scales. Source: California Energy Commission

2.3 Computing in Power Grid

Most of power grid applications run on Windows platforms with customized solvers and modeling. Many of these applications were developed before the computing revolution in the early 2000s and as such are legacy-based. A drawback of the legacy code is that it was developed for single-core or single-threaded applications. With the advent of multi-core machines, there have been efforts to make use of the shared-memory parallelism, but this has been more of an “inserted” effort rather than starting from ground-up. There are vendors like Opal-RT [Opal-RT], RTDS [RTDS], and Typhoon HIL [TyphoonHIL] that use a Linux platform, but their tools are restricted to specialized applications such as those for studying electromagnetic transients.

2.3.1 Parallel Computations

Research on parallel computation in the power grid has been done since the early 80s. [Alvardo1979] proposed the first parallel algorithm for transient stability analysis. They used a parallel-in-time approach for partitioning the problem into separate time chunks. Subsequent efforts in this area were pursued by Ilic and Crow who prototyped methods for parallel transient simulations, sensitivity analysis, and computational load balancing [Ilic1987]. Unfortunately, this work coincided with dramatic increase in CPU clock speed and the need for parallel computing in power systems diminished.

Today, the power industry does use parallel computation for embarrassingly parallel applications such as contingency analysis. Many of the utilities and ISOs have smallish clusters to run several power flow calculations simultaneously.

Another parallelization strategy that is gaining traction in power systems community as of recently is co-simulations (also known as operator splitting) [HELICS, PCKrause2009]. Parts of the system are co-simulated independently and data between them is exchanged at each solver iteration. Since this approach considers co-simulated parts as black-box simulations, power systems practitioners find it very effective given diversity and lack of standardization among grid simulation tools. The downside of this approach is lack of ability to control (or even estimate) numerical error, as well as lack of ability to balance computational load. Therefore, this approach does not scale accuracy- or performance-wise.

Several efforts for using distributed memory computations have also been carried out, but these have been mainly limited to the research community. There have been promising results presented that sped up power grid applications. Electronic circuit simulator Xyce [XyceManual], developed at Sandia National Laboratory, has power systems component model library, which enables users to compose Netlists for power system models. Xyce supports distributed and shared memory parallelism and has been used in circuit simulations with $\sim 10^7$ unknowns. However, for large scale simulations that require distributed memory parallelism, convergence is not guaranteed for general problems because of ill-conditioned nature of grid models. GridPACK [GridPACK2013] is a software development kit for high-performance computing applications targeting transmission systems. GridPACK also supports shared and distributed memory parallelism. RTDS is using causalization approach to partition grid models and deploy computations on distributed memory architecture. This approach assumes power flow from bulk power systems to distribution grids (no backflow) and uses time delays to break coupling between different parts of grid. Such approach is less and less viable with higher penetration of renewables and distributed energy resources. DMNetwork [DMNetwork] is a framework developed in the PETSc library for supporting rapid development of unstructured problems typically found in "network" applications, such as power grids, gas pipelines, communications, water networks, and others. DMNetwork provides toplology management and data migration capabilities on distributed memory architectures. At this time, none of these tools support computations on accelerator devices.

3. Power grid applications performance testing

In this section, we take a deeper dive on two power grid applications with specific details on application work flows, numerical details, and baseline performance metrics. These two power grid applications are the fundamental building blocks of many of the power grid applications. As such, any performance improvement for them would greatly improve the performance of a large variety of power grid applications.

3.1 Hardware, software, and test networks

The performance of these applications was done on a 2.2 GHz Intel icore7 Mac OSX with 16 GB DDR4 RAM. This machine has one node with 6 physical cores and 12 logical cores (hyperthreading on). Each physical core has a 256 KB L2 cache and all cores share 9216 KB L3 cache.
The application code is written in C and uses the numerical library PETSc [Balay2016]. It is compiled with Clang compiler with optimization. (-03). PETSc is an open-source numerical library that provides serial and parallel linear, nonlinear, and time-stepping solvers. The tests were run on the synthetic Texas and the Western Interconnect (WI) networks available via the TAMU electrical grid repository. The details of these networks are given below in Table I.

3.2. AC Power Flow

Power flow is one of the most fundamental application in power grid. The equations used in power flow are used in a variety of applications. A subset of these applications is shown in Fig. 6

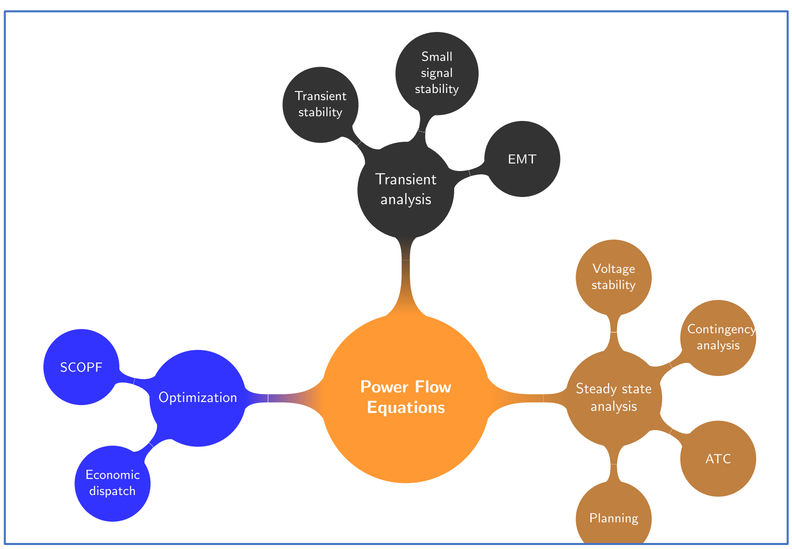


Fig. 6: Power flow equations form the basis of a large variety of power grid problems. These equations, based on steady-state AC Kirchoff's current and voltage laws, represent the power flow balance at the network nodes, i.e., the summation of power flowing in and out at each node is zero.

3.2.1 Equations

For a n-bus power system, the AC power flow equations are described by the real and reactive power balance equations at each bus i . Following Kirchoff's current law (KCL) and Kirchoff's voltage laws (KVL), the power balance equations state that for each bus i, the summation of the power injected in the network P_i^{inj} and the power flowing in the network (over k lines) should be zero.

$$P_i^{inj} - \sum_{\substack{k=1 \\ a_{ij}=1}}^{n} V_i V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) = \Delta \mathrm{P}_i = 0$$

$$Q_i^{inj} - \sum_{\substack{k=1 \\ a_{ij}=1}}^{n} V_i V_j (G_{ij} \cos\theta_{ij} + B_{ij} \sin\theta_{ij}) = \Delta \mathrm{Q}_i = 0$$

3.2.2 Application Workflow

AC power flow problem solves balanced system of nonlinear algebraic equations. These problems are typically solved using a variant of a Newton method, which obtains solution for the nonlinear problem through a sequence of linear problem solutions as shown in Figure 7.

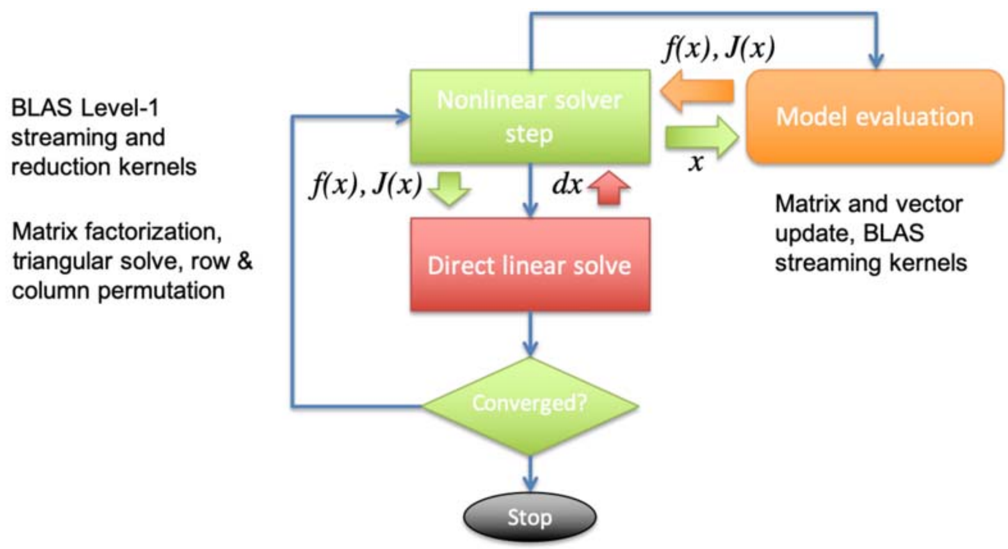


Figure 7: AC power flow computation workflow. Block arrows represent data movement and line arrows represent control signals. Nonlinear solver requests residual evaluation, Jacobian evaluation, or both from the model.

The Newton solver itself performs only BLAS Level-1 operations and schedules runs of the linear solver. The Newton solver computes error norms to check if the convergence criterion is met. If not, the solver requests updated values for the system residual vector and Jacobian matrix from the grid model, passes those to the linear solver and calls linear solver run for the problem

$$J\,dx = -f.$$

Here J is the Jacobian matrix, f is the residual vector for the AC optimal power flow problem, and dx is the solution of the linear system. The Newton solver then updates the nonlinear solution $x \leftarrow x + dx$ and passes it to the power system model. More advanced Newton solvers perform several streaming BLAS Level-1 operations to improve the solution update based on solver iteration history and/or error norm values. This sequence is repeated until convergence criteria are met or maximum number of allowed iterations is exceeded. In this workflow, most of the operations are done during the linear solve, and therefore most of the computational cost comes from there. Jacobian matrix evaluation is typically the second most expensive part of the workflow.

Power grid problems typically lead to ill-conditioned linear problems, so *direct* linear solvers are tool of choice in this area. Effective preconditioners for iterative linear solvers in this area are still mainly in the research domain. However, sparse direct linear solvers cannot be fully parallelized, so due to Amdahl's law, the linear solver becomes the computational bottleneck in a parallel computation environment.

3.2.3 Numerical Details

Direct sparse linear solve is the most expensive operation within AC power flow computation. State-of-the-art solvers execute these four sequential steps:

1. Ordering
2. Symbolic factorization
3. Numeric factorization
4. Triangular solve

The time complexity of the numeric factorization dominates the linear solve and hence this step carries most of the computational cost for all nontrivial AC power flow problems. The complexity slightly varies from implementation to implementation. Current state of the art is $\sim O(NNZ \times \log N)$, where *N* is the size of the problem and *NNZ* is the number of non-zeros in the sparse matrix [Gilbert2004].

Since the termination criterion for nonlinear solver is expressed in terms of a residual norm, the accuracy of the iterative solution depends only on model residual and not on model Jacobian. Inaccuracies in Jacobian may affect the convergence rate of the Newton method, but not the solution accuracy (to within prescribed tolerance). Because of that, it is often advantageous to use *inexact* Newton method where Jacobian matrix and its factorization are reused over several solver iterations. In that case, triangular solve (step 4) is applied on a stale Jacobian factorization. This approach may take more nonlinear solver iterations to get to the solution, but the overall computational cost is lower as there are fewer Jacobian matrix evaluations and matrix factorizations. State-of-the-art nonlinear solvers employ sophisticated control algorithms to reduce the overall computational cost by judiciously deciding when to recompute and factorize Jacobian [Chai1991].

3.2.4 Performance Evaluation

The performance of the power flow program was evaluated on the Texas and Western Interconnect power grid networks. The size of the network is given in Table I.

Network	**Buses (nodes)**	**Lines (edges)**	**# of variables**
Texas	2000	3206	4000
WI	10000	12706	20000

Table I: Test power grid networks used for performance evaluation of applications. The topology of these networks are shown in Figs. 1 and 2, respectively

We evaluated the performance of four main application kernels: (a) function (residual) evaluation, (b) Jacobian evaluation, (c) time-spent only in the nonlinear solver operations, and (d) time-spent in the linear solver operations. The performance of these application kernels for the two test networks is shown in Figs 8 and 10, respectively. Table III gives information on the total time spent in the application and the number of calls for each of the kernels. As seen, the linear solver kernel took the largest time, so we also evaluated the performance of the different linear solver kernels, as shown in Figs. 9 and 11, respectively. From the linear solver performance evaluation, it is seen that the numerical factorization routine is the most dominant linear solver kernel.

Texas:

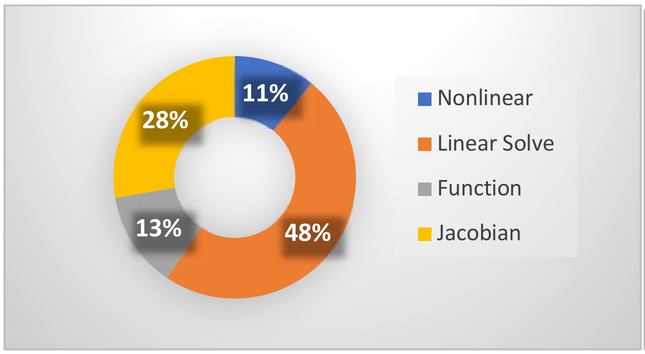


Fig. 8: Execution time distribution of application kernels for the Teas network. Linear solver is the most dominant kernel taking 49% of total execution time.

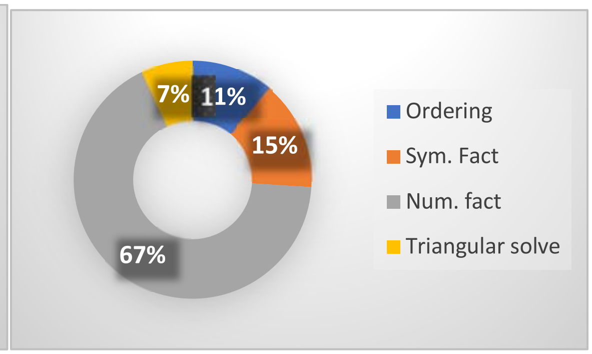


Fig. 9: Execution time distribution of linear solver kernels for the Teas network. Numerical factorization is the most dominant kernel taking 67% of total linear solver execution time.

Western Interconnect:

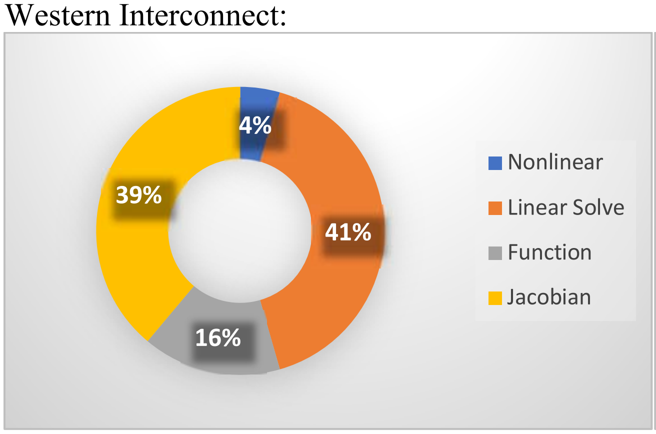


Fig. 10: Execution time distribution of application kernels for the WI network. Linear solver is the most dominant kernel taking 41% of total execution time.

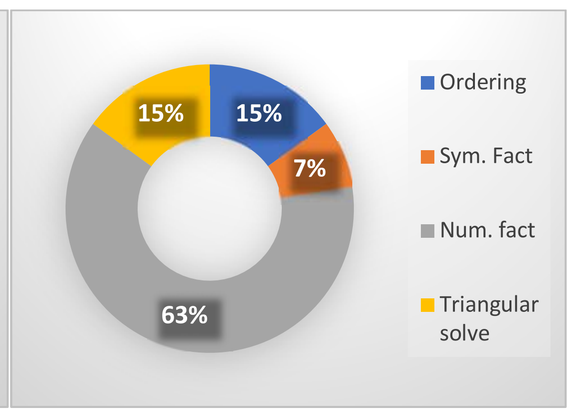


Fig. 11: Execution time distribution of linear solver kernels for the WI network. Numerical factorization is the most dominant kernel taking 63% of total linear solver execution time

Table III: Total solution time and number of calls for application kernels

Network	Total Solution time (sec)	Function evaluation # of calls	Jacobian evaluation # of calls	Nonlinear Solver % time # of calls	Linear Solver # of calls
Texas	0.022	5	4	1	4

WI	0.0855	6	5	1	5

The linear solver used above is a direct solver based on LU factorization with an approximate minimum degree ordering scheme from SuiteSparse [SuiteSparse2015] package's AMD library. From our experiments with different linear solver packages and ordering schemes, this combination of the linear solver package and ordering was found to be the best. Table IV and V show a comparison of linear solver performances using different linear solver packages and ordering schemes for the Texas and WI networks, respectively. All these linear solver packages and ordering schemes are available through the PETSc library (PETSc has interfaces with external linear solver and ordering packages such as SuiteSparse, MUMPS [MUMPS2001], etc.)

Linear Solver	**Ordering**	**Linear solver time (sec)**	**Ordering time (%)**	**Symbolic factorization time (%)**	**Numerical factorization (%)**	**Triangular solve (%)**
PETSc	**AMD**	**0.0083**	**14**	**14**	**65**	**7**
PETSc	QMD	0.0175	53	13	27	7
PETSc	ND	0.043	4	9	85	2
KLU	AMD	0.0176	13	12	69	6
UMFPACK	AMD	0.035	4	15	77	4
MUMPS	METIS	0.102	5	16	61	19

Table IV: Performance of Linear solver with different ordering techniques and linear solvers for the Texas network. UMFPACK, KLU, and AMD packages are available via the SuiteSparse package. MUMPS is a linear solver package that uses multi-frontal method for solving linear system. QMD (Quotient minimum degree) and ND (nested dissection) orderings are available through the PETSc library

Linear Solver	**Ordering**	**Linear solver time (sec)**	**Ordering time (%)**	**Symbolic factorization time (%)**	**Numerical factorization (%)**	**Triangular solve (%)**
PETSc	AMD	0.034	10	20	60	10
PETSc	QMD	0.15	82	4	13	1
PETSc	ND	0.91	1	3	95	1
KLU	AMD	0.064	13	12	69	6
UMFPACK	AMD	0.18	3	5	87	5
MUMPS	METIS	0.49	2	16	64	18

Table V: Performance of Linear solver with different ordering techniques and linear solvers for the WI network. UMFPACK, KLU, and AMD packages are available via the SuiteSparse package. MUMPS is a linear solver package that uses multi-frontal method for solving linear system. QMD (Quotient minimum degree) and ND (nested dissection) orderings are available through the PETSc library

3.2. Transient Stability

Power systems undergo disturbances of various types such as balanced and unbalanced short circuits, equipment outage of generators, transmission lines and other equipment, breaker tripping, and others [Sauer1998]. Maintaining stability of the power system on incidence of such disturbances is of prime importance to system planning engineers for ensuring a secure and uninterrupted supply to the consumers. Power system stability is the ability of an electric power system, for a given initial operating condition, to regain a state of operating equilibrium after being subjected to a physical disturbance, with most system variables bounded so that practically the entire system remains intact [Kundur2004]. Transient stability analysis studies the ability of the power system to keep its rotating machines in synchronism, when subjected to a disturbance (perturbation), during and after the perturbation. A common disturbance simulation involves applying a fault at a given bus at some pre-specified time and removing the fault by opening a circuit element at another pre-specified time. This type of disturbance scenario can help determine the critical clearing time of the circuit breakers and thereby protect the electrical machines from going out of synchronism.

3.2.1 Equations

In transient stability analysis, the electrical power system is expressed as a set of discrete nonlinear differential algebraic equations (DAEs) as given below.

$$dx/dt = f(x, y)$$
$$0 = g(x, y)$$
$$p(x^-) \leq p(x) \leq p(x^+)$$

Here, the differential equations model dynamics of the rotating machines (e.g., generators and motors) and the algebraic equations represent the transmission system and quasi-static loads. The inequalities in the last equation represent the discontinuities on the differential variables x, due to the presence of nonlinearities such as saturation, over and under excitation limits, valve limiters, and others. Thus, the transient stability problem can be classified as a discontinuous differential-algebraic model (DDAE) or hybrid model.

The solution of the dynamic model given in (1) needs the following:

- A numerical integration scheme to convert the differential equations in algebraic form
- A nonlinear solution scheme to solve the resultant nonlinear algebraic equations
- A linear solver to solve the update step at each iteration of the nonlinear solution

3.2.2 Application workflow

There are two approaches to solve (1) - alternating explicit and simultaneous implicit [1]. In the alternating approach, the differential variables are first updated, $x^{n+1} \leftarrow H(x^n)$, where H is an explicit integration scheme. Once the dynamic variables are updated, the algebraic variables y are solved for the $n+1$ time-instant by solving the algebraic equations $g(y^{n+1}) = 0$. A typical choice of the explicit integration scheme is a second-order Runge-Kutta method or Trapezoidal method. The advantage of an alternating explicit approach is the ease of implementation due to separation of the differential and algebraic parts. However, they may need to use shorter time-steps to ensure stability of the numerical integration scheme. Commercial transient stability tools, for e.g., PSSE [PSSE], PowerWorld [PowerWorld], use an alternating explicit approach for the simulation of power system transient stability equations.

In the simultaneous implicit approach, the differential equations are first converted to algebraic form using an implicit numerical integration formula and the two-sets of nonlinear equations solved together using Newton's method. A simultaneous implicit approach requires the solution of a bigger nonlinear algebraic system given in (2), thus more time-consuming than the alternating explicit approach. But it has better stability properties even for larger time-steps. EUROSTAG

[Sanchez] and Powertech's TSAT[PowerTech] are commercial softwares that employ implicit integration schemes. The figure below shows the application workflow for transient stability analysis with an implicit integration scheme.

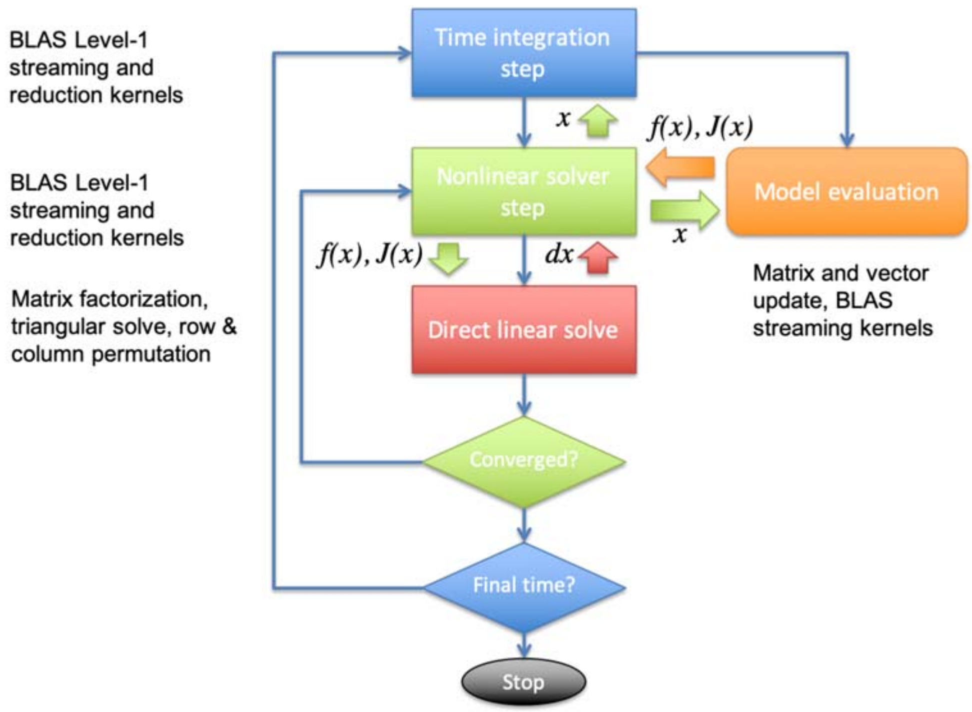


Figure 12: Transient stability computation workflow. Block arrows represent data movement and line arrows represent control signals.

3.2.3 Performance Evaluation

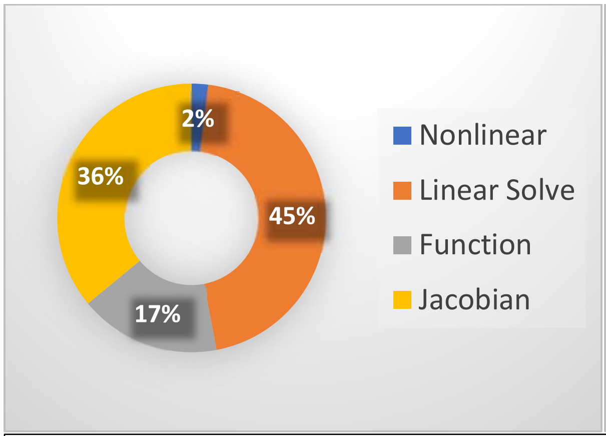
The evaluation of the transient stability computational performance was done on the Texas and WI networks for a 10-second simulation with a time-step of 0.01 second. As seen from Figs. 13 and 15, half of the application time is for the function and Jacobian evaluation, while the other half is on the linear solver. This is a similar computational pattern as found for the power flow application. Moreover, the numerical factorization, similar to power flow application, is the most dominant linear solver kernel as shown in Figs. 14 and 16.

Network	**Buses (nodes)**	**Lines (edges)**	**# of variables**
Texas	2000	3206	7456
WI	10000	12706	35496

Table VI: Test power grid networks used for performance evaluation of applications. The topology of these networks are shown in Figs. 1 and 2, respectively

Texas network

Network	Solution time (sec)	Time-steps	Function evaluation # of calls	Jacobian evaluation # of calls	Nonlinear Solver # of calls	Linear Solver # of calls
Texas	13.24	1000	3998	1998	1003	2004



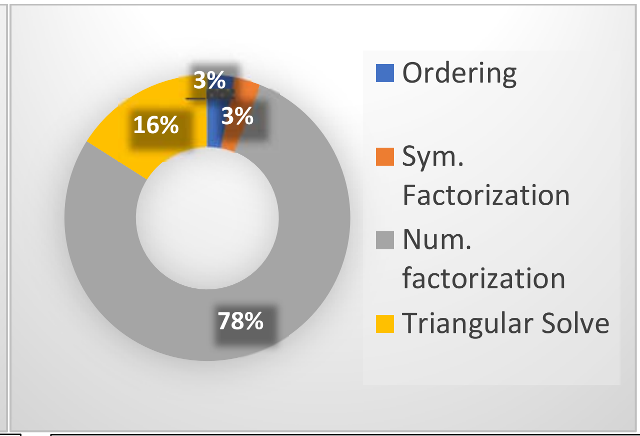
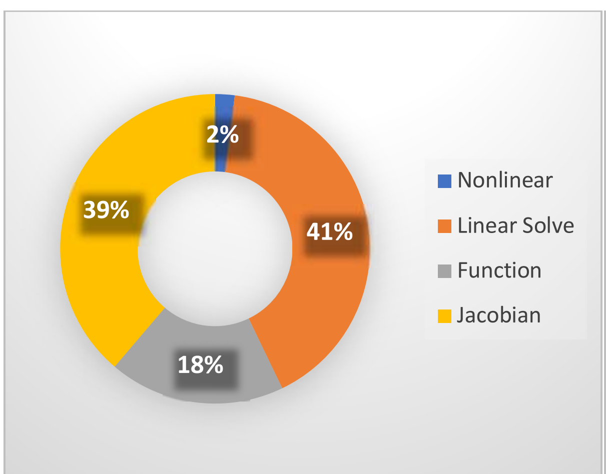


Fig. 13: Execution time distribution of application kernels for the Texas network. Linear solver is the most dominant kernel taking 45% of total execution time.

Fig. 14: Execution time distribution of linear solver kernels for the Texas network. Numerical factorization is the most dominant kernel taking 75% of total linear solver execution time

Western Interconnect

Network	Solution time (sec)	Time-steps	Function evaluation # of calls	Jacobian evaluation # of calls	Nonlinear Solver # of calls	Linear Solver # of calls
WI	56.13	1000	4073	2073	1003	2080



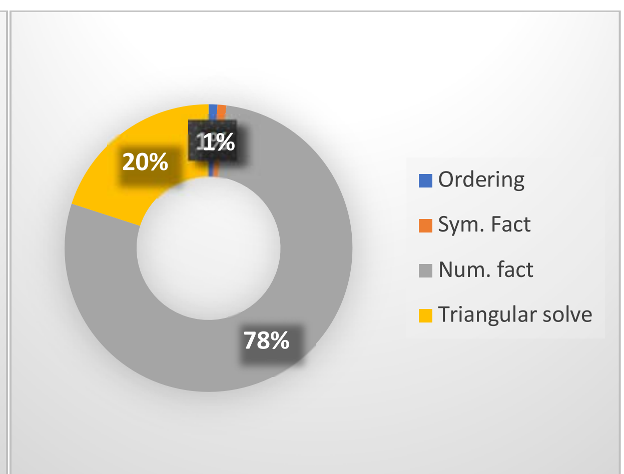


Fig. 15: Execution time distribution of application kernels for the WI network. Linear solver is the most dominant kernel taking 41% of total execution time.

Fig. 16: Execution time distribution of linear solver kernels for the WI network. Numerical factorization is the most dominant kernel taking 78% of total linear solver execution time

4. Summary and Recommendations

In this document, an in-depth evaluation of two of the most common power grid applications – power flow and transient stability, has been presented. The computational runs were conducted on large synthetic networks that have the modeling complexity typical of large utility and interconnect networks. The most dominant application and computational kernels were identified. Based on the profiling of different simulations, the following recommendations/observations for DMC initiative researchers are recommended. We note that while the performance evaluation was done only for power grid network problems, the recommendations, in general, are applicable to a broad class of application domains.

Recommendation 1: Accelerate function/Jacobian evaluation on accelerators

Two of the most dominant power grid application kernels identified by this work were (a) function/residual evaluation and (b) Jacobian evaluation. These two application kernels contributed more than 50% of the total application time combined. For accelerating the application, these two application kernels are the prime candidates. However, power grid problems are unstructured (irregular topology) and heterogenous (different number of variables per node) leading to different computational patterns and data access. As such, careful consideration and thought must be given to port these two kernels on accelerators that have SIMD architecture.

Recommendation 2: Accelerate linear system solution on accelerators

The biggest gain for power grid applications would be acceleration of linear solver kernel. In our simulations, linear solver kernel took almost half of the total application solution time. The linear solver kernel consists of four operations that have irregular and sequential computational pattern. These are: (a) matrix reordering – to reduce the fill-ins in the factored matrix, (b) symbolic factorization – get locations and allocate memory for the L (lower triangular) and U (upper triangular) factored matrices, (c) numerical factorization – compute the values in L and U matrices, and (d) triangular solve – compute the solution via forward and backward solves with the L and U matrices. In our simulations, the numerical factorization contributed to almost 70% of the total linear solver kernel time.

Recommendation 3: Develop efficient computational patterns and data structures for irregular/heterogenous network for SIMD architectures

One of the differentiating factors for power grid networks is their unstructured topology and heterogeneity which lead to logic branching, jumping, and irregular data access. For example, a certain node (bus) may be connected, via a branch, to only one node, while another one may be connected to ten other nodes. In addition, a node may or may not have a generator or a load or have multiple of them with status ON or OFF. These are some of the irregular and logic-based computational patterns that may not be amenable to SIMD architectures for accelerators.

Recommendation 4: Need software stack/methods for fast access and computations of sparse matrix data structures

Power grid applications are extremely sparse (typically < 1% sparsity) and as such matrices are typically stored in sparse compressed data structures, such as sparse compressed row. Software stacks on accelerators need to support such data structures or provide alternate data structures that are efficient on accelerators.

Recommendation 5: Developing profiling tools for evaluation and benchmarking

All the profiling done in this work was possible because of the excellent profiling tools available via the PETSc library. The development of profiling tools/codes is central to performance evaluation and benchmarking. As such, profiling tools on accelerators for different application and numerical kernels are important for development and benchmarking.

References

[Abhyankar2011] S. Abhyankar, B. Smith, H. Zhang, and A. Flueck. Using PETSc to develop scalable applications for next-generation power grid. In Proceedings of the 1st International Workshop on High Performance Computing, Networking and Analytics for the Power Grid. ACM, 2011.

[Abhyankar2013] S. Abhyankar, E. Constantinescu, and B. Smith. "Evaluation of overlapping restricted additive Schwarz preconditioning for parallel solution of very large power flow problems." 3rd International Workshop on High Performance Computing, Networking, and Analytics for the Power Grid, November 2013.

[Abhyankar2014] S. Abhyankar, Q. Cui, and A. Flueck. "Fast power flow analysis using a hybrid current-power balance formulation in rectangular coordinates." Proceedings of the IEEE Transmission and Distribution Conference and Exposition, April 2014.

[Alvarado1979] Alvarado, F. L., "Parallel solution of transient problems by trapezoidal integration", IEEE Transactions on Power Apparatus and Systems, PAS-98 (May/June 1979): 1080-1090.

[Atkins2009] K. Atkins, J. Chen,. V. S. Kumar, and A. Marathe, "The structural of electrical networks: a graph theory-based analysis", Int. Journal on Critical Infrastructures, vol. 5, no. 3, 2009.

[Balay2016] S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. Gropp, D. Karpeyev, D. Kaushik, M. Knepley, L. Curfman McInnes, R. Mills, T. Munson, K. Rupp, B. Smith, S. Zampini, H. Zhang, and H. Zhang. 2019. "PETSc Users Manual. Technical Report ANL-95/11 - Revision 3.12". Argonne National Laboratory. https://www.mcs.anl.gov/petsc/petsc-current/docs/manual.pdf

[Birchfield2017] A. B. Birchfield; T. Xu; K. M. Gegner; K. S. Shetye; T. J. Overbye, "Grid Structural Characteristics as Validation Criteria for Synthetic Networks," in *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3258-3265, July 2017.

[Chai1991] Chai., J. S., N. Zhu, A. Bose, and D.J. Tylavsky. "Parallel Newton type methods for power system stability analysis using local and shared memory multiprocessors", IEEE Transactions on Power Systems, 6.4 (November 1991): 9-15.

[Davis2010] T. A. Davis and E Natarajan. Algorithm 907: Klu, a direct sparse solver for circuit simulation problems. ACM Transactions on Mathematical Software, 37:3426–3446, 2010.

[DMNetwork] S. Abhyankar, B. Getnet, D. Maldonado, L. McInnes, B. Smith, and H. Zhang, "PETSc DMNetwork: A Scalable Network PDE-Based Multiphysics Simulator", To appear in ACM Transactions on Mathematical Software (TOMS), 2019.

[Gilbert2004] J. Gilbert, "Sparse Matrix Algorithms", lecture notes CS 200H, UCSB, 2004.

[GridPACKManual] GridPACK User Manual, available at https://www.gridpack.org/wiki/images/8/86/GridPack_User_Manual_3.3.pdf, 2013

[HELICS] B. Palmintier, D. Krishnamurthy, P. Top, S. Smith, J. Daily, and J. Fuller, "Design of the HELICS high-performance transmission-distribution-communication-market co-simulation framework", Proceedings of the 2017 workshop on modeling and simulation of cyber-physical energy systems (MSCPES), 2017. DOI: 10.1109/MSCPES.2017.8064542

[Ilic1987] Ilic., M., M. L. Crow, and M. A. Pai., "Transient stability simulation by waveform relaxation methods", IEEE Transactions on Power Systems, 2.4 (November 1987): 943-952.

[Kundur2004] P. Kundur, J. Paserba, V. Ajjarapu, G. Andersson, A. Bose, C. Canizares, N. Hatziargyriou, D. Hill, A. Stankovic, C. Taylor, T. V. Cutsem, and V. Vittal, "Definition and classification of power system stability ieee/cigre joint task force on stability terms and definitions," IEEE Transactions on Power Systems, vol. 19, no. 3, pp. 1387–1401, Aug 2004

[MUMPS2001] P. R. Amestoy, I. S. Duff, J-Y. L'Excellent, and J. Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. SIAM Journal on Matrix Analysis and Applications, 23(1):15–41, 2001.

[Opal-RT] OPAL-RT Technologies, https://www.opal-rt.com/

[PCKrause2009] Wasynczuk O, Lucas CE, Walters EA, Jatskevich JV, inventors; PC Krause, Assoc Inc, assignee. "Distributed simulation". United States patent US 7,490,029. 2009 Feb 10.

[RTDS] RTDS Technologies, https://www.rtds.com/

[Falcao1996] Falcao., D., "High performance computing in power system applications", Invited Lecture at the 2 International Meeting on Vector and Parallel Processing, Porto, Portugal, (1996).

[Grijvala2011] Grijalva S., "Research needs in multi-dimensional, multi-scale modeling and algorithms for next generation electricity grids". DOE conference on computing challenges for the next-generation power grid (2011)

[Smith2010] Smith, B., and H. Zhang, "Sparse Triangular Solves for ILU Revisited: Data Layout Crucial to Better Performance", International Journal of High Performance Computing Applications, 2010. DOI 10.1177/10 94342010384857.

[Eto2011] Joseph H. Eto and R. J. Thomas. "Computational needs for the next generation electric grid." Technical report, U.S. Department of Energy, 2011. http://energy.gov/sites/prod/files/FINAL CompNeeds Proceedings2011.pdf.

[PowerWorld] PowerWorld Users Guide, PowerWorld Corporation.

[PowerTech] Powertech, "Power TSAT Web page," http://www.powertechlabs. com/areas-of-focus/software-technologies-/dsatools-software/ transient-security-assessment-tool/, 2017. Available: http://www.powertechlabs.com/areas-of-focus/software-technologies-/ dsatools-software/transient-security-assessment-tool/

[PSSE] PSSE 34.1 - Program Operational Manual, SIEMENS.

[Sanchez1995] J. J. Sanchez-Gasca, R. D'Aquila, W. W. Price, and J. J. Paserba, "Variable time-step, implicit integration for extended-term power system dynamic simulation," in IEEE Conference Proceedings of Power Industry Computer Application Conference. IEEE, 1995.

[Sauer1998] P. Sauer and M. A. Pai. Power System Dynamics and Stability. Prentice Hall Inc., 1998.

[SuiteSparse2015] T. Davis. SuiteSparse Web page. http://faculty.cse.tamu.edu/davis/suitesparse.html, 2015.

[TyphoonHIL] Typhoon HIL, webpage access https://www.typhoon-hil.com/

[XyceManual] E. R. Keiter, T. V. Russo, R. L. Schiek, H. K. Thornquist, T. Mei, J. C. Verley, P. E. Sholander, K. V. Aadithya, "Xyce™ Parallel Electronic Simulator Users Guide, version 6.11", May 2019, report available at https://xyce.sandia.gov/downloads/_assets/documents/Users_Guide.pdf

Pacific Northwest
National Laboratory

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354
1-888-375-PNNL (7665)

www.pnnl.gov