SAND2017-5973C

Algorithmic Aspects of a Commercial-Grade Distribution System Load Flow Engine

Francis Therrien[1], Marc Belletête[1], Jean-Sébastien Lacroix[1], and Matthew J. Reno[2]

[1] CYME International T&D, St-Bruno, QC, J3V 3P8, Canada

[2] Sandia National Laboratories, Albuquerque, NM, 87185, USA

Abstract — **The increasing penetration of PV requires more and more load flow calculations to assess the corresponding impact on planning and operation of distribution systems, e.g., during high-resolution quasi-static time-series calculations. There is therefore a strong incentive to optimize the speed of distribution load flow engines as much as possible without compromising on solution accuracy. This paper details the structure of a commercial-grade topology-independent Newton-Raphson distribution load flow engine, the reasoning behind some of its design choices, and recent improvements made to three rarely discussed aspects: initialization, calculation and discretization of tap changer positions, and solution of linear equations. Computer studies demonstrate the reduction in computational time on various distribution systems, allowing for faster interconnection studies with high PV penetration.**

I. Introduction

The large amount of distributed energy resources (DERs) and especially PV now found at the low- and medium-voltage levels require rethinking the planning and operation of distribution grids. Distribution system analysis tools are becoming more important than ever, as they can assess the various impacts of DERs on operating constraints, protection schemes, losses, and so on [1]. While it is now recognized that precise electromagnetic transient (EMT) studies are sometimes needed at the distribution level, load flow calculations remain ubiquitous, whether in "standalone" mode or within quasi-static time-series (QSTS) studies [2]. Improving the speed of load flow engines will therefore facilitate the execution of more comprehensive studies with PV, such as year-long high-resolution QSTS studies [2].

Traditional distribution system load flow engines often use the ladder iterative method (often known as backward-forward sweep), which is tailored for radial and passive systems. While the ladder iterative method can be adapted to handle meshes [3] and voltage-controlled nodes [4], its efficiency decreases quickly as the number of loops and DERs in voltage control mode increases. In fact, ladder iterative methods will rarely converge in urban meshed networks .

Modern commercial-grade load flow engines often use matrix-based methods. Some of these engines are based on a full Newton-Raphson approach [5]–[8], while others use a fixed point scheme [8]–[10]. Full Newton-Raphson approaches can naturally handle continuous controls by integrating them in the constraint vector, and typically converge in a few iterations; however, they require matrix re-factorizations, which are typically considered as computational bottlenecks. Fixed point schemes keep the same matrix factors, and therefore are considered faster; however, they may have more difficulty converging. The use of a smaller external sensitivity matrix can improve convergence in fixed point techniques [10], but its calculation and solution may require non-negligible computational effort when the number of control devices is large.

The CYME power engineering software has two load flow engines for unbalanced distribution systems: Voltage Drop Unbalanced (VDU, based on the ladder iterative approach), and the more recent Newton-Raphson Unbalanced (NRU, based on a full Newton-Raphson scheme). This paper first gives an overview of NRU, which is built on the work set forth in [8]. We discuss in detail three aspects of NRU that are often overlooked in the literature but that can have a significant impact on computational speed: the initialization method, the calculation and discretization of load tap changer (LTC) positions, and the solution of the resulting system of linear equations. Specifically, we present the advantages and disadvantages of initializing NRU with a fixed point (FP) or flat start (FS) approach, and briefly describe how to generate the FS profile considering NRU's modified nodal formulation. The benefits of integrating the control variables within NRU's set of equations are also discussed. Two LTC position normalization approaches are then presented. Finally, we challenge the often-held belief that constant-matrix engines are necessarily faster than full Newton-Raphson ones unless the former requires many more iterations. A brief discussion on two linear solvers, PARDISO [11] and KLU [12], is also included. Computer studies on various distribution systems are then presented to support the claims and hypotheses made throughout this paper.

II. Load Flow Formulation and Algorithmic Aspects

A. Fundamentals

Using the principle of modified nodal analysis (MNA, sometimes referred to as modified *augmented* nodal analysis [8]), the main load flow constraints can be described by

$$\mathbf{i}_n = \mathbf{f}_n(\mathbf{v}_n, \mathbf{i}_d, \mathbf{x}_c) \tag{1}$$

$$\mathbf{v}_d = \mathbf{f}_d(\mathbf{v}_n, \mathbf{i}_d, \mathbf{x}_c) \quad (2)$$

$$\mathbf{y}_c = \mathbf{f}_c(\mathbf{v}_n, \mathbf{i}_d, \mathbf{x}_c) \quad (3)$$

where $\mathbf{f}_n$, $\mathbf{f}_d$, and $\mathbf{f}_c$ are – potentially nonlinear – functions enforcing Kirchhoff's current law at every node, Kirchhoff's voltage law for ideal components (voltage sources, switches, and branch-dependent devices such as transformers and regulators), and additional network/component constraints, respectively; $\mathbf{v}_n$, $\mathbf{i}_d$, and $\mathbf{x}_c$ are state vectors comprising the nodal voltages, the currents of the ideal components, and the additional state variables (e.g., the equivalent admittance of DERs), respectively; and $\mathbf{i}_n$, $\mathbf{v}_d$, and $\mathbf{y}_c$ are nodal injection currents, voltages of ideal voltage sources, and the desired values of the additional network constraints. Complex values are represented using rectangular components. A multiphase framework is considered to model unbalanced conditions and topologies typical of North American distribution systems.

Since some of the constraints are nonlinear, (1)–(3) must be solved iteratively, herein using a full Newton-Raphson approach. Writing the Taylor series of (1)–(3) and truncating it therefore yields the main NRU equation

$$\begin{bmatrix} \mathbf{Y}_n^{''k} & \mathbf{D}_c^k & \mathbf{C}_{c1}^k \\ \mathbf{D}_r^k & \mathbf{S}_d^k & \mathbf{C}_{c2}^k \\ \mathbf{C}_{l1}^k & \mathbf{C}_{l2}^k & \mathbf{C}_d^k \end{bmatrix} \begin{bmatrix} \Delta\mathbf{v}_n^k \\ \Delta\mathbf{i}_d^k \\ \Delta\mathbf{x}_c^k \end{bmatrix} = \begin{bmatrix} \mathbf{i}_n^k \\ \mathbf{v}_d^k \\ \mathbf{y}_c^k \end{bmatrix} - \begin{bmatrix} \mathbf{f}_n^k \\ \mathbf{f}_d^k \\ \mathbf{f}_c^k \end{bmatrix} \quad (4)$$

or, written in compact form,

$$\mathbf{J}^k \Delta\mathbf{x}^k = \mathbf{b}^k - \mathbf{f}^k . \quad (5)$$

In the above, the superscript k indicates the iteration count; $\mathbf{Y}_n^{''k}$, $\mathbf{D}_c^k$, $\mathbf{D}_r^k$, $\mathbf{S}_d^k$, $\mathbf{C}_{c1}^k$, $\mathbf{C}_{c2}^k$, $\mathbf{C}_{l1}^k$, $\mathbf{C}_{l2}^k$, and $\mathbf{C}_d^k$ are submatrices of the system Jacobian $\mathbf{J}^k$; and the Δ operator indicates the difference between a given vector at the present and past iterations (e.g., $\Delta\mathbf{v}_n^k = \mathbf{v}_n^k - \mathbf{v}_n^{k-1}$). Additional information regarding the formulation of the Jacobian matrix and the right-hand side vectors $\mathbf{b}^k$ and $\mathbf{f}^k$ can be found in [8], [13]. In particular, $\mathbf{Y}_n^{''k}$ is a modified version of the standard admittance matrix, which accounts for state-variable-dependent equivalent current injections (see Section II-B).

Following the Newton-Raphson approach, (4) or (5) is solved iteratively until the largest normalized voltage mismatch has fallen below a user-specified tolerance, while ensuring that all dependent and independent variables are within their physical boundaries (e.g., minimum and maximum tap positions, reactive power capabilities, …).

B. Current Injections vs. Additional Network Constraints

Due to the flexibility of MNA, loads can be included either as equivalent current injections in $\mathbf{f}_n^k$ [5], or through additional network constraints in $\mathbf{f}_c^k$ [7], [8]. On one hand, the former approach does not increase the system dimension, but its partial derivatives added in $\mathbf{Y}_n^{''k}$ are fairly involved when voltage-sensitive load models are used [5]. On the other hand, the latter approach increases the system size (the load currents are added to $\mathbf{x}_c^k$), but only a few straightforward elements must be added to $\mathbf{C}_{l1}^k$ and $\mathbf{C}_d^k$ (e.g., 8 for a single-phase wye-grounded load) [7], [8]. Moreover, $\mathbf{C}_{c1}^k$ acts as an adjacency matrix for loads, and therefore its only non-zero values will be 1s and potentially –1s.

Tests on various networks have shown that no approach is unequivocally faster, owing in part to the high efficiency of modern linear solvers. However, it was noted that the equivalent current injection approach converged more robustly, likely due to its more linear formulation. The robustness of the equivalent current injection approach was further improved by representing loads using a fixed admittance in $\mathbf{Y}_n^{''k}$ and a small current in $\mathbf{i}_n^k$ compensating for the difference between the desired power and the power consumed by the fixed admittance. For a constant impedance load, the load equation becomes fully linear as the current injection is always equal to 0.

Other types of power injections or network constraints, such as DERs in voltage control mode, cannot be formulated linearly or quasi-linearly. In this case, their control equations are added to $\mathbf{f}_c^k$ along with the corresponding states in $\mathbf{x}_c^k$ [7], [8].

C. Initialization

The Newton-Raphson approach is not self-starting: its initial state vector $\mathbf{x}^0$ must be defined to solve (4) or (5). The FS approach remains the initialization method of choice in traditional transmission-level load flow engines, as the state vector is only comprised of nodal voltages in polar coordinates, and the phase shift of transformers is typically ignored. Initialization is more complicated in NRU, as its formulation includes state variables that are not nodal voltages (see (4)), and transformer phase shifts are considered. Moreover, the selected transformer modeling approach [14] involves internal nodes, whose equivalent FS voltage is not always trivial to define.

Due to the above reasons, NRU was originally initialized using a single-iteration FP solution [7], [8]. Therein, all power injection devices are represented using equivalent admittances. The initial reactive power must be guessed for generators in voltage control mode, which can be problematic in some transmission systems. All load tap changers (LTCs, see Section II-D) are fixed. Vector $\mathbf{f}_c^0$ is filled with 0s, $\mathbf{C}_{c1}^0$ and $\mathbf{C}_{c2}^0$ are empty matrices, and $\mathbf{C}_d^0$ is an identity matrix. This creates a linear equivalent to (4), allowing the computation of an initial guess of the state vector. Moreover, if the linearization is accurate, $\mathbf{x}^0$ will be closer to the final solution than a FS, possibly reducing the number of iterations. To increase speed, the FP matrix is padded with "hard" 0s wherever the Jacobian $\mathbf{J}^k$ might have non-zeros. Consequently, $\mathbf{J}^k$ can be built upon the structurally identical FP matrix, thereby significantly reducing CPU time since matrices are stored in a compressed sparse format [15].

The main caveat of this approach is that it requires two successive symbolic factorizations (for the FP solution and the first Newton-Raphson iteration), which are costly operations. More detail on this subject is given in Section II-E. To reduce the number of symbolic factorizations and hopefully the CPU time, an augmented FS approach is proposed. It makes use of a network-model iterator, which computes the complex nominal voltage at each node. For transformers, the internal nodal voltages are initialized based on the corresponding external nodal voltages and winding configuration. Vector $\mathbf{i}_d^0$ can be set to 0, since the corresponding equations are almost always linear; whereas the values of $\mathbf{x}_c^0$ are set using the desired power injection and the corresponding FS voltage.

D. LTC Discretization

There are two main methods to calculate LTC positions in load flows: between iterations (externally, sometimes referred to as the error-feedback approach), or by adding tap positions as state variables (internally) [16]. The former approach usually fits well into constant-matrix solutions [17], [18], but requires the external calculation of a sensitivity factor (empirically or through a more sound analysis [10]). Hunting between different LTCs may also occur [16]. In the latter approach, the LTC voltage set point is added to the constraint vector (e.g., in $\mathbf{f}_c^k$ in NRU). When using a Newton-Raphson approach, the solution usually converges to the desired set point in a few iterations. The main challenge is that the LTC positions are treated as continuous variables, whereas only discrete positions are physically feasible.

Since NRU does not have a constant matrix, and for the sake of simplicity, LTC positions are included in $\mathbf{x}_c^k$ along with the corresponding voltage set-point constraint in $\mathbf{f}_c^k$. Another benefit of this approach is that it meshes naturally with CYME's "infinite taps" mode, where LTC positions are considered as continuous variables. This mode allows planners to overcome the well-known problem of the multiplicity of load flow solutions due to LTC bandwidths, which cannot guarantee a worst-case solution. By using the lower bandwidth value as the voltage set point along with continuous LTC positions, planners can obtain under-voltages that are lower or equal to the worst physically possible under-voltage, and thus plan their network safely and reliably. This approach can be easily tweaked to study over-voltages arising due to high PV penetration and reverse power flow.

Since many distribution engineers require feasible (discrete) LTC positions, a normalization scheme is also needed. A "step discretization" approach was first developed in NRU. Based on the fact that downstream voltage regulators/transformers have little impact on the upstream voltage, after initial convergence, LTCs were discretized one by one following a downstream path. The main disadvantages of this approach are twofold. First, it is numerically costly for networks with many LTCs, since two extra iterations are usually needed per LTC. Second, for meshed networks, the discretization order is not always obvious. To overcome these problems, this paper presents a "joint discretization" approach. After the first convergence, all LTCs are normalized at once. Due to the concurrent discretization, the controlled voltage of some LTCs may end up outside of the specified bandwidth. A detection-correction step has been devised. It checks if all constraints are respected. When it is not the case, the corresponding LTC positions are changed by ±1. This is repeated until all constraints are respected. A break condition is added if hunting is detected, but this is unlikely on real customer networks. Finally, the transformer LTC position (primary or secondary) and the regulator operating mode (e.g., bi-directional, co-generation [19], …) must be considered to choose whether the LTC positions must be incremented or decremented to move towards the bandwidth.

E. Solution of Linear Systems

Roughly speaking, efficient numerical solution of (4) or (5) using sparse methods includes permuting and pre-ordering the unsymmetric indefinite Jacobian matrix to reduce fill-ins (symbolic factorization), calculating the L and U factors (numerical factorization), and solving using backward-forward substitution. When considering matrix-based load flows, it has become almost axiomatic to say that the solution of the linear equations is the "main computational bottleneck [11]". Consequently, since factorization is costlier than backward-forward substitution, it is pretty much accepted that methods with constant matrices (therefore requiring only one factorization) will be more efficient than load flows with changing matrices, unless they require significantly more iterations. For instance, in the 1970s, the Fast Decoupled formulation was shown to be around 5 times faster per iteration than the full Newton-Raphson formulation [16].

However, in view of today's highly efficient linear solvers such as KLU [12] and PARDISO [11], and based on our practical experience, we make the two following claims: 1) in complex commercial-grade packages such as CYME, the percentage of time of a full Newton-Raphson load flow spent on factorization is relatively small (this will be quantified in Section III-D); and 2) numerical factorization is computationally cheap. Since the same symbolic factorization can be reused from iteration to iteration when the matrix non-zero pattern remains constant, based on the 2nd claim, a full Newton-Raphson approach can nowadays be competitive with constant-matrix methods, and sometimes even faster if it requires fewer iterations. This is especially true when care is taken to avoid symbolic factorizations as much as possible, for instance by discretizing all taps jointly as explained in Section II-D. In that approach, the corrective steps do not require additional symbolic factorizations since only the numerical values change.

PARDISO [11], which is included in Intel MKL, was initially used as NRU's solver. It is a general-purpose package that calls different solvers depending on the matrix properties (e.g., real or complex numbers, symmetric or unsymmetric, positive (semi-)definite or indefinite, …). This is an interesting feature for a wide-ranging distribution system analysis program such as CYME, as the matrices generated by different modules often have different properties. KLU [12] was recently tested as it is designed for circuit matrices, which, according to [12], are extremely sparse. This is especially true of the Jacobian of radial systems, but less so for urban meshed networks. The performance of PARDISO and KLU for different test systems are presented in Section III-D.

III. Case Studies

The numerical performance of NRU for each pair of proposed initialization methods, discretization techniques, and solvers are tested on 6 different networks [20]–[21]. The salient features of these test systems are presented in Table I. In particular, the IEEE 342 and Util1 test systems are urban meshed networks, the latter being extremely large for distribution systems. In addition to having several inline voltage regulators, the Util2 and Util3 test systems also have high PV penetration. The number of nodes reported in Table I differ from those of the official IEEE test feeders; this is due to a different (and inconsequential) way of counting nodes. Moreover, to provide additional insight, the detailed CPU timings for one of these systems is provided, and a comparison between the numerical efficiency of NRU and VDU serves as a closing statement.

All tests are executed on a special version of CYME 7.2 used in the course of the Department of Energy-sponsored “Rapid QSTS Project”. This version runs on a dedicated benchmark machine (3.4 GHz i7-2600 CPU with 8 GB of RAM). A tolerance of 0.01% on the voltage mismatch is used as the stopping criterion.

A. Initialization

NRU’s numerical efficiency using the FP and FS initialization methods is presented in Table II for the subject test systems (all LTCs are operating in “infinite taps” mode and PARDISO is used). The CPU time is presented in normalized form with respect to the FP approach. For two networks, an additional iteration is required with the FS approach; whereas for the other four, the iteration count is identical with both initialization procedures. This is explained by the fact that the FP approach typically yields a state vector that is closer to the final solution. This can be observed in Fig. 1, where the voltage mismatch of four of the test systems is plotted as a function of the iteration. Fig. 1 also demonstrates that both approaches converge almost quadratically. Only the IEEE 342 system converges slower near the end, due to its ungrounded nature and the resulting ill-conditioned matrix. Whether an additional iteration is required or not, the solution time remains smaller with the FS method, ranging from 83% to 92.7% of the CPU time of the FP approach.

Table I
Main properties of the test systems.

Network	Nb. Nodes	Nb. Loops	Nb. LTCs
CKT5 [9]	3003	0	0
IEEE 342 [20]	390	71	0
IEEE 8500 [21]	4875	0	4
Util1	28425	3180	14
Util2	2370	1	10
Util3	4066	0	14

Table II
Number of iterations and normalized CPU time for a load flow solution using fixed point and flat start initialization.

	Fixed Point Init.		**Flat Start Init.**	
Network	**Nb. Iter.**	**CPU Time**	**Nb. Iter.**	**CPU Time**
CKT5	2	100%	3	92.7%
IEEE 342	2	100%	2	83.0%
IEEE 8500	3	100%	4	90.6%
Util1	5	100%	5	90.6%
Util2	3	100%	3	86.8%
Util3	3	100%	3	88.9%

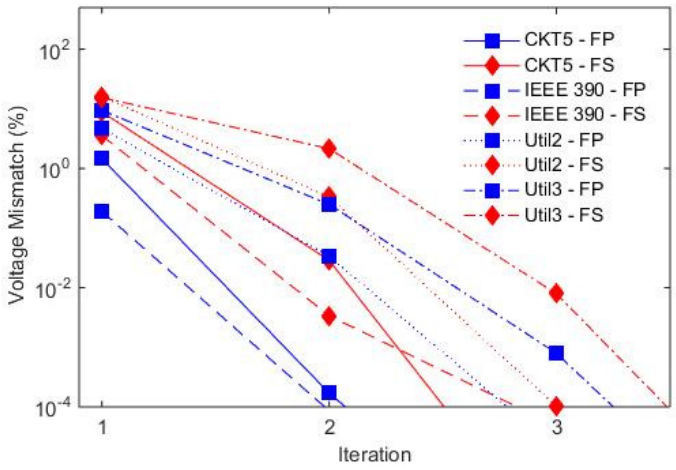

Fig. 1. Convergence pattern of four of the test systems using the fixed point (FP – blue square markers) and flat start (FS – red diamonds markers) initialization approaches.

To verify the generalization of these results, the NRU load flow is executed on all the distribution systems of an entire region of a large utility (657 feeders for a total of 73 independent networks). The majority of these networks are fed by Yg-Delta transformers with a grounding transformer on the secondary side, and the peak load scenario is considered. The combination of these two conditions often causes problems to distribution load flow engines (in fact, 2 of the networks do not converge after 200 iterations when using CYME’s robust VDU engine with a relaxed tolerance of 0.1%). With NRU, all networks converge independently of the initialization approach. The FS approach requires an extra iteration for 50 networks; whereas the same number of iterations is needed for the other 23.

TABLE III
NUMBER OF ITERATIONS AND NORMALIZED CPU TIME FOR A LOAD FLOW SOLUTION USING STEP AND JOINT LTC POSITION DISCRETIZATION.

	Step Discret.		**Joint Discret.**	
Network	**Nb. Iter.**	**CPU Time**	**Nb. Iter.**	**CPU Time**
IEEE 8500	12	100%	9	87.2%
Util1	28	100%	7	31.5%
Util2	23	100%	5	43.2%
Util3	30	100%	7	42.4%

TABLE IV
NUMBER OF ITERATIONS AND NORMALIZED CPU TIME FOR A POWER FLOW SOLUTION USING THE PARDISO AND KLU SOLVERS.

	PARDISO		**KLU**	
Network	**Nb. Iter.**	**CPU Time**	**Nb. Iter.**	**CPU Time**
CKT5	3	100%	3	89.2%
IEEE 342	2	100%	2	92.8%
IEEE 8500	9	100%	9	74.2%
Util1	7	100%	7	225.4%
Util2	5	100%	5	75.8%
Util3	7	100%	7	73.4%

B. Discretization

The number of iterations and normalized CPU time for the test systems using the step and joint LTC discretization approaches are summarized in Table III (the FS method is used along with PARDISO). The results for CKT5 and IEEE 342 are not presented as they do not have LTCs. As expected, the iteration count and CPU times are reduced with the joint discretization scheme; this reduction is drastic for networks with several LTCs (e.g., 7 iterations instead of 28 for Util1, requiring only 31.5% of the original CPU time). As each discretization usually results in 2 (sometimes 3) additional iterations, it can be seen by comparing Tables II and III that the first joint discretization sometimes resulted in controlled voltages outside of the specified bandwidth, and that corrective tap changes were needed to find an adequate solution.

C. Solvers

The numerical performance of NRU using the PARDISO and KLU linear solvers are presented in Table IV for the 6 test systems (the FS and joint discretization techniques are used). As it is optimized for extremely sparse circuit matrices, KLU outperforms PARDISO on 5 of the 6 systems; however, for the very large urban meshed network, PARDISO is more than twofold faster than KLU. It is therefore suggested to use KLU for radial and weakly meshed systems, and PARDISO for highly meshed networks. A criterion based on the loop-to-node ratio can be added to the code to automatically select the preferred linear solver for any given system.

TABLE V
DETAILED CPU TIMINGS OF AN NRU LOAD FLOW SOLUTION.

Total Load Flow	58.61 ms
Create Segment	19.90 ms
Data Exchange (CYME → NRU)	1.57 ms
Initialization (build Jacobian)	14.79 ms
Iteration #1	13.87 ms
Update Jacobian	0.494 ms
Create Vector ($\mathbf{b}^k - \mathbf{f}^k$)	1.162 ms
Symbolic Factorization	9.081 ms
Numerical Factorization	2.879 ms
Solve	0.164 ms
Convergence Check	0.080 ms
Iteration #2	4.93 ms
Update Jacobian	0.527 ms
Create Vector ($\mathbf{b}^k - \mathbf{f}^k$)	1.182 ms
Numerical Factorization	2.990 ms
Solve	0.169 ms
Convergence Check	0.056 ms
Get Results (NRU → CYME)	2.821 ms

D. Detailed Timings

To better assess the numerical performance of NRU, several timers were added to the major code sections. The resulting CPU timings for the IEEE 342 test system with PARDISO are presented in Table V. The "Create Segment" part of the code consists of reading and treating the network file: it accounts for around a third of the CPU time. Creation of the matrix is also non-negligible, as it requires approximately 1/4th of the total CPU time, which is almost the same as for symbolic and numerical factorizations combined. This contradicts the well-known idea that matrix factorization is the main computational bottleneck in load flow calculations. Numerical factorization is also about 3 times faster than symbolic factorization. Due to this, the 2nd iteration accounts for less than 10% of the CPU time. While these exact ratios depend on the test system and the solver, they clearly demonstrate that a full Newton-Raphson approach is not much slower than constant-matrix solutions.

E. Comparison Between NRU and VDU

Finally, the CPU times necessary to solve the load flows of the six test systems with CYME's NRU and VDU engines are presented in Table VI. NRU is initialized using FS, the LTCs are discretized jointly, and either KLU or PARDISO is used, based on the criterion explained in Section III–C. It is observed that VDU does not converge on 3 of the 6 systems. The IEEE 342 and Util1 test systems diverge due to their highly meshed structure; whereas hunting between the 14 LTCs occurs when trying to solve the Util3 network. It is emphasized that Table VI is not meant as the definitive comparison between matrix-based and ladder iterative methods, but simply as a comparison between two optimized

TABLE VI
NUMBER OF ITERATIONS AND NORMALIZED CPU TIME USING THE NRU AND VDU LOAD FLOW SOLVERS (THE DASH (–) INDICATES DIVERGENCE).

	NRU		**VDU**	
Network	**Nb. Iter.**	**CPU Time**	**Nb. Iter.**	**CPU Time**
CKT5	3	0.174 s	4	0.142 s
IEEE 342	3	0.122 s	–	–
IEEE 8500	9	0.531 s	19	0.733 s
Util1	5	5.400 s	–	–
Util2	5	0.211 s	13	0.205 s
Util3	8	0.298 s	–	–

commercial-grade distribution load flow solvers. As previously demonstrated in [6] in different environments, the cost per iteration is smaller for VDU than NRU; whereas the former typically requires fewer iterations. In particular, NRU converges in much fewer iterations than VDU on highly loaded systems with several controls, such as the IEEE 8500 system. Overall, the computational cost of NRU is absolutely competitive with VDU; in fact, NRU is even faster than VDU on some systems.

IV. CONCLUSION

This paper presents the general structure of a commercial-grade load flow engine suitable for distribution systems of all topologies, called Newton-Raphson Unbalanced (NRU). Three often overlooked aspects of distribution load flow calculations are discussed in detail: initialization, calculation and discretization of tap changer positions, and solution of the resulting linear equations. Computer studies on multiple distribution systems support the design choices made regarding the three aforementioned aspects. They also demonstrate that the versatile NRU engine is very competitive in terms of CPU time with a topology-limited ladder iterative engine. As a consequence of its high numerical efficiency, NRU is well suited to meet the increasing demand of distribution system load flow solutions arising from higher levels of PV penetration.

REFERENCES

[1] R. F. Arritt and R. C. Dugan, “Distribution system analysis and the future smart grid,” *IEEE Trans. Ind. Appl.*, vol. 47, no. 6, pp. 2343–2350, Nov./Dec. 2011.

[2] M. J Reno, J. Deboever, and B. Mather, “Motivation and requirements for quasi-static time series (QSTS) for distribution system analysis,” to appear in *Proc. IEEE Power Energy Soc. General Meeting*, 2017.

[3] D. Shirmohammadi, H. W. Hong, A. Semlyen, and G. X. Luo, “A compensation-based power flow method for weakly meshed distribution and transmission networks,” *IEEE Trans. Power Syst.*, vol. 3, no. 2, pp. 753-762, May 1988.

[4] C. S. Cheng and D. Shirmohammadi, “A three-phase power flow method for real-time distribution system analysis,” *IEEE Trans. Power Syst.*, vol. 10, no. 2, pp. 671-679, May 1995.

[5] P. A. N. Garcia, “Three-phase power flow calculations using the current injection method,” *IEEE Trans. Power Syst.*, vol. 15, no. 2, pp. 508-514, May 2000.

[6] L. R. de Araujo *et al.*, "Comparisons between the three-phase current injection method and the forward/backward sweep method," *Elect. Power Energy Syst.*, vol. 32, pp. 825-833, 2010.

[7] W. Xu, J. R. Martí, and H. W. Dommel, “A multiphase harmonic load flow solution technique,” *IEEE Trans. Power Syst.*, vol. 6, no. 1, pp. 174-182, Feb. 1991.

[8] I. Kocar *et al.*, “Multiphase load-flow solution for large-scale distribution systems using MANA,” *IEEE Trans. Power Del.*, vol. 29, no. 2, pp. 908-915, Apr. 2014.

[9] R. C. Dugan, T. E. McDermott, “An open source platform for collaborating on smart grid research,” in *Proc. IEEE Power Energy Soc. General Meeting*, 2011.

[10] I. Dzafic, R. A. Jabr, E. Halilovic, and B. C. Pal, “A sensitivity approach to model local voltage controllers in distribution networks,” *IEEE Trans. Power Syst.*, vol. 29, no. 3, pp. 1419-1428, May 2014.

[11] O. Schenk and K. Gärtner, “Solving unsymmetric sparse systems of linear equations with PARDISO,” *Future Generation Comp. Syst..*, vol. 20, no. 3, pp. 475-487, 2004.

[12] T. A. Davis and E. Palamadai Natarajan, “Algorithm 907: KLU, a direct sparse solver for circuit simulation problems,” *ACM Trans. Math. Softw*, vol. 37, no. 3, Sep. 2010.

[13] I. Kocar, J.-S. Lacroix, and F. Therrien, “General and simplified computation of fault flow and contribution of distributed sources in unbalanced distribution networks,” in *Proc. IEEE Power Energy Soc. General Meeting*, 2012.

[14] I. Kocar and J.-S. Lacroix, “Implementation of a modified augmented nodal analysis based transformer model into the backward forward sweep solver,” *IEEE Trans. Power Syst.*, vol. 27, no. 2, pp. 663-670, May 2012.

[15] T. A. Davis, *Direct Methods for Sparse Linear Systems*, Philadelphia, PA: SIAM, 2006.

[16] B. Stott, “Review of load-flow calculations,” *Proc. IEEE*, vol. 62, no. 7, pp. 916-929, Jul. 1974.

[17] B. Stott and O. Alsaç, “Fast decoupled load flow,” *IEEE Trans. Power App. Syst.*, vol. PAS-93, no. 3, pp. 859-869, 1974.

[18] S.-K. Chang and V. Brandwajn, “Adjusted solutions in fact decoupled load flow,” IEEE Trans. Power Syst., vol. 3, no. 2, pp. 726-733, May 1988.

[19] Cooper Power Systems, *S225-10-10: Voltage Regulators*, Oct. 2001.

[20] K. Schneider, P. Phanivong, and J.-S. Lacroix, “IEEE 342-node low voltage networked test system,” in *Proc. IEEE Power Energy Soc. General Meeting,* 2014.

[21] R. F. Arritt and R. C. Dugan, “The IEEE 8500-node test feeder,” in *Proc. IEEE Power Energy Soc. T&D*, 2010.

This research was supported in part by the DOE SunShot Initiative, under agreement 30691. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA0003525.