DELFT UNIVERSITY OF TECHNOLOGY

REPORT 12-08

TOWARDS FASTER SOLUTION OF LARGE POWER FLOW PROBLEMS

R. IDEMA, G. PAPAEFTHYMIOU, D.J.P. LAHAYE, C. VUIK AND
L. VAN DER SLUIS

ISSN 1389-6520

Reports of the Department of Applied Mathematical Analysis

Delft 2012

Towards Faster Solution of Large Power Flow Problems

Reijer Idema, *Member, IEEE*, Georgios Papaefthymiou, *Member, IEEE*,
Domenico Lahaye, Cornelis Vuik, and Lou van der Sluis, *Senior Member, IEEE*

***Abstract*—Current and future developments in the power system industry demand fast power flow solvers for larger power flow problems. The established methods are no longer viable for such problems, as they are not scalable in the problem size.**

In this paper, the use of Newton-Krylov power flow methods is proposed, and a multitude of preconditioning techniques for such methods are discussed and compared. It is shown that incomplete factorizations can perform very well as preconditioner, resulting in a solver that scales in the problem size. It is further shown that using a preconditioned inner-outer Krylov method has no significant advantage over applying the preconditioner directly to the outer iterations. Finally, Algebraic Multigrid is demonstrated as a preconditioner for Newton-Krylov power flow and argued to be the method of choice in some scenarios.

***Index Terms*—power flow analysis, Newton-Krylov methods, preconditioning, incomplete factorizations, flexible inner-outer Krylov methods, Algebraic Multigrid.**

I. Introduction

In recent years the power systems industry is experiencing a radical change, driven by the imperative to shift to a more competitive and less carbon intensive energy system. As the penetration of variable renewables and distributed energy sources increases, and power markets get more integrated, existing infrastructures are expected to evolve in two major directions [1]:

1) *Supergrids*: much longer and higher rated transmission lines are needed to transport renewable energy from distant areas, and to enable the coupling of power markets. This increased interconnection dictates the integrated management of power systems of continental scale.
2) *Smartgrids*: ICT technologies and local energy storage will allow the integration of intelligence in the demand, and enable large scale demand response actions in the system. Distribution networks will be transformed into active network clusters (smartgrids), consisting of loads and local generation and storage, which will assume a significant role in the management of the power system.

In the light of this system transformation, new computational algorithms are needed that allow the simulation of continental wide systems in short time, for operational purposes. The integrated operation of transmission and distribution systems, spanning vast geographic areas—as dictated by the above-mentioned developments—translates into the need for analysis and simulation of very large networks.

Typically, operational security assessment involves off-line contingency analysis [2], resulting in a large number of power flow simulations for slightly modified network configurations. In the new competitive environment, system security assessment has to be performed as close as possible to real time, with sufficient speed to either trigger an automatic control action, or to allow time for the operator to react [3]. Further, the incorporation of variable renewable generation creates uncertainty in the expected infeeds, and thus in the conditions for the chosen network configurations. To include this uncertainty, Monte-Carlo techniques can be employed, which consist of the sampling of stochastic infeeds and the simulation of a large number of system states [4]. For all these tasks the main computational burden lies in the repetitive simulation of slightly modified versions of a power flow problem.

For the typical size of networks analyzed in control rooms today, classic power flow solvers offer good performance. However, these solvers are not so efficient when the problem size is increasing, and they become extremely slow for very large networks. An approach in dealing with the computational burden of operational tasks, is to distribute computations among multiple servers [5].

Taking into account the size of future networks, new solvers are needed that are scalable in the problem size. In this paper we propose the use of Newton-Krylov power flow methods, and analyze a multitude of preconditioning techniques to optimize performance. The good results of incomplete LU factorizations [6] are explained, and extended with incomplete Cholesky factorizations. Further, inner-outer Krylov methods are investigated, with proper attention to the accuracy of the inner solves. And finally, Algebraic Multigrid is introduced as a preconditioner for Newton-Krylov power flow methods.

The presented methods perform much better than classic methods for large network sizes, and are better suited for operational tasks as they allow more information to be reused when solving similar problems [6]. Algebraic Multigrid is also well-suited for a parallel computing environment.

The paper is structured as follows. Section II introduces the setting of the power flow problem. Then, Section III discusses inexact Newton methods, and Section IV presents the theory of Krylov subspace methods and preconditioning. Section V discusses the preconditioning of the power flow problem, followed by the presentation and analysis of the results of numerical experiments in Section VI. Finally, the conclusions are presented in Section VII.

R. Idema, D.J.P. Lahaye, and C. Vuik are with the Delft Institute of Applied Mathematics, Delft University of Technology, The Netherlands.

G. Papaefthymiou is with the Power Systems and Markets Department, Ecofys Germany GmbH and with the Electrical Power Systems Department, Delft University of Technology, The Netherlands.

L. van der Sluis are with the Electrical Power Systems Department, Delft University of Technology, The Netherlands.

II. The Power Flow Problem

The power flow equations, are equations that relate the power to the voltage in each bus in the power system. Let $|V_i|$ be the voltage magnitude, δ_i the voltage angle, P_i the active power, Q_i the reactive power, and $Y = G + jB$ the admittance matrix. Further, define $\delta_{ij} = \delta_i - \delta_j$. The power flow equations in bus i can then be written as

$$\sum_{k=1}^{N} |V_i|\,|V_k|\,(G_{ik}\cos\delta_{ik} + B_{ik}\sin\delta_{ik}) = P_i, \quad (1)$$

$$\sum_{k=1}^{N} |V_i|\,|V_k|\,(G_{ik}\sin\delta_{ik} - B_{ik}\cos\delta_{ik}) = Q_i, \quad (2)$$

Combining the power flow equations (1), (2) in all buses, yields a nonlinear system of equations

$$\boldsymbol{F}(\boldsymbol{x}) = \boldsymbol{0}, \quad (3)$$

where $\boldsymbol{F}$ is known as the power mismatch function.

Given the supply and demand in the power system, the power flow problem (3) can be solved to reveal the steady-state voltages in the power system. For more information on power systems and power flow, see for example [7].

Traditionally, the power flow problem is solved using the Newton-Raphson method with a direct solver [8], [9], or using the fast decoupled load flow (FDLF) method [10]–[12]. In [6] we showed that the LU factorization—which is used by both these traditional methods—is not viable for very large power flow problems. As an alternative, we proposed the use of Newton-Krylov methods: inexact Newton methods that incorporate Krylov methods to solve the linear problems.

III. Inexact Newton Methods

The Newton-Raphson method, for the solution of nonlinear systems of equations, is an iterative method that updates the iterate $\boldsymbol{x}_i$ in each iteration by adding a Newton step. The Newton step $\boldsymbol{s}_i$ is calculated by solving the linearized system in the current iterate, i.e.,

$$J(\boldsymbol{x}_i)\,\boldsymbol{s}_i = -\boldsymbol{F}(\boldsymbol{x}_i), \quad (4)$$

where J is the Jacobian matrix of the power mismatch $\boldsymbol{F}$.

Inexact Newton methods use the same principle, except that the linear system (4) is not solved to full accuracy. Instead, a solution is calculated that satisfies

$$\|J(\boldsymbol{x}_i)\,\boldsymbol{s}_i + \boldsymbol{F}(\boldsymbol{x}_i)\| \leq \eta_i \|\boldsymbol{F}_i\|. \quad (5)$$

The values $\eta_i \in (0, 1)$ are called the forcing terms.

It has been proven that—if the forcing terms are chosen correctly—inexact Newton methods exhibit the same quadratic convergence as the Newton-Raphson method [13]. Too large forcing terms lead to slower convergence, whereas choosing the forcing terms too small leads to oversolving. Especially in early iterations, the forcing terms can be chosen quite large without compromising convergence. In the numerical experiments presented in this paper, the forcing terms are chosen using the method by Eisenstat and Walker [14].

It is important to note that the choice of linear solver does not fundamentally influence the convergence of the inexact Newton method. It is the accuracy up to which the Jacobian system is solved that mainly determines the convergence, and thus the robustness of the method. Provided that the linear solver is robust, an inexact Newton method with correctly chosen forcing terms is as robust as the Newton-Raphson method with a direct solver.

Further note that the exact Newton step is generally not the best of all the steps that satisfy equation (5). An inexact solution of the Jacobian system may lead to a slightly worse iterate than the exact solution; however, it may also very well lead to a better iterate. If the convergence of two Newton methods differs a lot for the same nonlinear problem, then either one of the methods got lucky with the iterates, or one of the methods is using forcing terms that are too large.

IV. Krylov Methods and Preconditioning

Krylov subspace methods are iterative linear solvers that generate iterates within Krylov subspaces based on the linear system of equations [15]. For a linear system $A\boldsymbol{x} = \boldsymbol{b}$ with given initial iterate $\boldsymbol{x}_0$, the initial residual is $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, and the Krylov subspace of dimension j is defined as

$$\mathcal{K}_j(A, \boldsymbol{r}_0) = \text{span}\left\{\boldsymbol{r}_0, A\boldsymbol{r}_0, \ldots, A^{j-1}\boldsymbol{r}_0, \right\}. \quad (6)$$

A Krylov method produces iterates such that

$$\boldsymbol{x}_j \in \boldsymbol{x}_0 + \mathcal{K}_j(A, \boldsymbol{r}_0). \quad (7)$$

Krylov methods that calculate the best iterate within the Krylov subspace—in the sense that the residual $\boldsymbol{r}_j = \boldsymbol{b} - A\boldsymbol{x}_j$ is minimized—are referred to as minimal residual methods. Another desirable property for Krylov methods is that of short recurrences. An algorithm is said to have short recurrences, if in each iteration only data of a fixed low number of previous iterations is needed. It has been proven that Krylov methods can not have both the minimal residual property and short recurrences [16], [17]. Bi-CGSTAB [18], [19] and IDR(s) [20] are examples of methods that have short recurrences, but not the minimal residual property. GMRES [21] is a minimal residual method, but the amount of data and work grows with every iteration. It is possible to restart GMRES after a certain amount of iterations to reset the amount of data and work, but then the minimal residual property is lost.

Preconditioning is a technique that changes the Krylov subspace, and thus the iterates produced by a Krylov method. Good preconditioning is essential for the performance of Krylov methods [15]. In the numerical experiments presented in this paper, we use right preconditioning. This means that instead of solving the original linear system $A\boldsymbol{x} = \boldsymbol{b}$, the preconditioned system

$$AP^{-1}\boldsymbol{y} = \boldsymbol{b} \quad (8)$$

is solved, after which the solution to the original system is calculated by solving $P\boldsymbol{x} = \boldsymbol{y}$. The advantage of right preconditioning is that the residual of the preconditioned system is the same as that of the original system, which is not the case for left or split preconditioning.

The closer the preconditioner matrix P resembles the coefficient matrix A, the faster Krylov methods can be expected to converge. However, a linear system of the form $P\boldsymbol{u} = \boldsymbol{v}$ has to be solved in every iteration, and one more such system

at the end to obtain the solution of the original problem. Thus it is imperative that such systems can be solved with relatively little computational effort. Note that if $P = A$, then the preconditioned Krylov system is simply solved by $\boldsymbol{y} = \boldsymbol{b}$, but the step to get the original solution from the preconditioned one is exactly equal to solving the original problem.

Krylov methods usually expect the preconditioner P to be the same in each linear iteration. However, so-called flexible Krylov methods allow the preconditioner to vary. Examples of such methods are GMRESR [22] and FGMRES [23].

V. Preconditioning the Power Flow Problem

In each Newton iteration, a preconditioner P_i is needed for the Jacobian system (4). This gives the linear system

$$J_i P_i^{-1} \boldsymbol{z}_i = -\boldsymbol{F}_i, \tag{9}$$

from which the Newton step $\boldsymbol{s}_i$ is calculated by solving

$$P_i \boldsymbol{s}_i = \boldsymbol{z}_i. \tag{10}$$

In this paper we investigate LU and Cholesky factorized matrices as preconditioner, preconditioned Krylov methods as preconditioner (also known as inner-outer Krylov methods), and Algebraic Multigrid as preconditioner. Newton-Krylov power flow with factorized preconditioners was previously explored in [6], [24]–[26]. In [27] GMRES as preconditioner for Newton-Krylov power flow was investigated.

All treated methods of preconditioning are based on one of three matrices: the coefficient matrix J_i, the initial Jacobian J_0, or the matrix Φ^*, a special symmetric positive definite M-matrix derived from the fast decoupled load flow method.

The FDLF matrix Φ^* is constructed as follows. Shunts are removed from the power system model, transformer ratios are set to 1, and the phase shift of phase-shifting transformers are set to 0. For this modified model the fast decoupled load flow matrices B' and B'' are calculated, according to the BX scheme. Then

$$\Phi^* = \begin{bmatrix} B' & 0 \\ 0 & B'' \end{bmatrix}. \tag{11}$$

In the absence of negative reactances, the result is a symmetric positive definite M-matrix (see also [28]).

The special structure of the matrix Φ^* allows the use of a Cholesky factorization, the Conjugate Gradient (CG) [29] method, and Algebraic Multigrid. If the power system model contains negative reactances, some extra adaptations may be needed to use these methods. These methods can not be used with the J_i and J_0 base matrices.

Factorizations of matrices similar to Φ^* were already shown to be good preconditioners in [6], [24], [30]. Tests showed that preconditioning with Φ^* was not noticeably worse than with the unmodified version that was used in [6].

A. Factorizations

Preconditioners P in the form of a triangular factorization—like the LU factorization or Cholesky factorization—are popular because systems $P\boldsymbol{u} = \boldsymbol{v}$ can be solved with just a forward and backward substitution, which is very fast.

The LU decomposition is a factorization $P = LU$, where the matrix L is lower triangular and U is upper triangular. Such a factorization exists for every invertible matrix P, provided that row permutations are allowed. The Cholesky factorization is a decomposition $P = CC^T$, where C is a lower triangular matrix. Only symmetric positive definite matrices allow a Cholesky factorization. The Cholesky factorization is more memory efficient, as only a single factor needs to be stored.

For large matrices, calculating the factorization is computationally very expensive. Also, for sparse matrices the factors generally contain many more nonzero entries than the original matrix. This not only increases memory usage, but also the computational cost of the forward and backward substitution operations. Smart reordering of the rows and columns of the matrix can significantly reduce the fill-in.

Incomplete factorizations [31], [32] are factorizations that merely approximate the original matrix. The aim is to reduce computational time needed to calculate the factors, and reduce the fill-in, while retaining a good approximation. When used as a preconditioner for a Krylov method, an incomplete factorization generally leads to slower convergence compared to the full factorization. However, for large problems the extra iterations of the linear solver are generally much cheaper than the extra computational cost of a full factorization.

An incomplete LU (ILU) factorization of a matrix Q is a product $P = LU$ that approximates Q. Similarly, an incomplete Cholesky (ICC) factorization $P = CC^T$ also approximates Q. ILU(k) and ICC(k) factorizations use the number of levels k to determine the approximation quality; higher k gives a better approximation, but takes longer to calculate and also leads to more fill-in.

In [6] we showed the following:

- LU factorizations (and thus also direct solvers) are not viable for large power flow problems, but ILU(k) factorizations scale very well in the problem size.
- The Approximate Minimum Degree (AMD) [33] reordering should be used for all factorizations. It reduces the fill-in for both complete and incomplete factorizations, and improves the quality of incomplete factorizations.
- A single factorization of a well-chosen preconditioner matrix should be used throughout all Newton iterations.

Therefore, in this paper we consider ILU(k) factorizations of J_0, and ICC(k) factorizations of Φ^*, with AMD reordering, as preconditioners. Complete LU factorizations, also with AMD reordering, are only used as a reference.

Note that both the calculation of a factorization, and the forward and backward substitution operations, are inherently sequential. A block diagonal approximation of the matrix can be used to parallelize factorizations, at the cost of some of the quality of the preconditioner.

B. Krylov Methods as Preconditioner

The application of any number of iterations of a Krylov method can be written as a linear operation, and can thus be used as a preconditioner. The iterations of the method used to solve the Jacobian system are called the outer iterations, while the iterations of the method that is used as preconditioner are

called the inner iterations. Note that it is usually desirable to use preconditioning on the inner Krylov method also.

Most Krylov methods are nonstationary, meaning that the linear operation that results from a fixed number of iterations is generally not the same for all right-hand side vectors. When using a nonstationary iterative method as preconditioner, the outer Krylov method needs to be flexible, like FGMRES.

This type of preconditioning is often used when a high quality factorization is unavailable, for example if it is too costly too calculate, if parallelization does not allow it, or if the matrix is only available implicitly as an operation on a vector and not explicitly as a given matrix.

In general it does not make sense to only do a single inner iteration, or to solve the inner problem to such high accuracy that the outer method converges in a single iteration. As long as the accuracy of the inner solve is somewhere well between these extremes, the overall speed of the outer solve is usually not very sensitive to the precise inner accuracy.

Special care should be taken if the inner iterative solver operates on a different coefficient matrix than the outer Krylov method, e.g., if the Jacobian system is solved using FGMRES preconditioned with CG on the Φ^* matrix. This causes a similar situation to that of Newton-Krylov methods, where a full accuracy linear solve leads to oversolving. There is only a certain amount of convergence that can be achieved in each outer iteration, when the preconditioner is based on the Φ^* matrix. Solving the inner problem up to an accuracy higher than that, is a waste of computational effort.

In this paper we consider GMRES on J_i and CG on Φ^* as preconditioners, with FGMRES to solve the Jacobian systems. The GMRES preconditioner is in turn preconditioned with ILU(k) factorizations of J_0, and the CG preconditioner is preconditioned with ICC(k) factorizations of Φ^*. The results are compared with using incomplete factorizations as preconditioner on the outer iterations directly.

C. Algebraic Multigrid

Multigrid methods [34] are iterative methods that originate from the field of solving discretized differential equations. Multigrid methods are optimal in the sense that the convergence is independent of the number of grid points. The basic idea is to combine cheap methods on grids with varying sizes into an update for the iterate.

Basic iterative methods, like Jacobi (diagonal scaling) and Gauss-Seidel (forward substitution using the lower triangular part of the coefficient matrix), generally smooth the error very quickly, without necessarily making the error much smaller. Thus, high frequency errors disappear, but low frequency errors remain. On a coarser grid, these low frequency errors show up as high frequency errors, and can be smoothed again cheaply using a basic iterative method. When the grid is coarse enough, the linear system on that grid can be solved efficiently with any method, usually a direct solver.

More formally, let $A_h \boldsymbol{x}_h = \boldsymbol{b}_h$ be a fine grid discretization, and $A_H \boldsymbol{x}_H = \boldsymbol{b}_H$ a coarse grid discretization of the same problem. First, the current iterate $\boldsymbol{x}_h^j$ is smoothed using a pre-smoother S_h:

$$\bar{\boldsymbol{x}}_h^j = \boldsymbol{x}_h^j + S_h^{-1}\left(\boldsymbol{b} - A_h \boldsymbol{x}_h^j\right) \tag{12}$$

Then, using a restriction operator I_h^H, the residual is brought to the coarse grid

$$r_H^j = I_h^H\left(\boldsymbol{b} - A_h \bar{\boldsymbol{x}}_h^j\right), \tag{13}$$

where it is used to solve the defect equation

$$A_H \boldsymbol{e}_H^j = r_H^j. \tag{14}$$

Next, the coarse grid error $\boldsymbol{e}_H^j$ is brought to the fine grid using an interpolation operator I_h^H, and the smoothed iterate $\bar{\boldsymbol{x}}_h^j$ is updated by

$$\bar{\bar{\boldsymbol{x}}}_h^j = \bar{\boldsymbol{x}}_h^j + I_H^h \boldsymbol{e}_H^j \tag{15}$$

Finally, a post-smoother S_h is used to smooth any high frequency errors that may have been introduced by the interpolated coarse grid error:

$$\boldsymbol{x}_h^{j+1} = \bar{\bar{\boldsymbol{x}}}_h^j + S_h^{-1}\left(\boldsymbol{b} - A_h \bar{\bar{\boldsymbol{x}}}_h^j\right) \tag{16}$$

Note that the smoothers should be stationary iterative schemes like Jacobi or Gauss-Seidel, but that the pre-smoother and post-smoother do not necessarily have to be the same.

The above process describes a single cycle of a two grid method. When more grids are used, there are several methods of traversing through the finer and coarser grids. The simplest method is to smooth and restrict all the way down to the coarsest grid, and then interpolate and smooth all the way back up to the finest. This method is referred to as a V-cycle. Provided that smoothers and a coarse grid solver are used that allow effective parallelization, multigrid cycles are very well-suited for parallel computing.

Multigrid can be used as an iterative linear solver, but also as a preconditioner. If a stationary solver is used on the coarsest grid, then multigrid is a stationary solver itself. Therefore, if a fixed number of cycles is used as preconditioner, there is no need to use a flexible Krylov solver.

In Geometric Multigrid methods, the grids and the corresponding restriction and interpolation operators I_h^H and I_H^h are constructed based on the geometry of the problem. For structured grids such operators are readily available, but for unstructured grids the construction may be very challenging.

In Algebraic Multigrid (AMG) methods, the construction of the grids and restriction and interpolation operators is automated, based on the properties of the coefficient matrix. The classical Ruge-Stüben approach to AMG needs a symmetric positive definite M-matrix as coefficient matrix. However, modern implementations of this approach often show some leniency regarding this requirement.

The power flow problem is not a discretized differential equation, but has a similar structure. It is not immediately clear how to construct restriction and interpolation operators based on the geometry of the problem, thus Algebraic Multigrid is a logical choice. The Jacobian matrices are generally far from symmetric positive definite M-matrices, so AMG can not be used directly as a solver for the Jacobian systems. Instead,

we solve these systems using GMRES, preconditioned with a fixed number of AMG cycles on the modified FDLF matrix Φ^*. Note that the tolerance of modern solvers, regarding the structure of the matrix, may be useful to deal with power system models that contain some negative reactances.

VI. Numerical Experiments

In this section, numerical experiments with the discussed preconditioning techniques are treated. These experiments are presented in subsections ordered and named identical to Section V.

The test cases used are based on the UCTE winter 2008 study model [1]. The model has been copied and interconnected to create larger test cases. Table I shows the number of buses and branches in the test problems, as well as the number of nonzeros in the Jacobian matrix nnz(J). The naming convention used is uctewXXX, where XXX is the number of times the model is copied and interconnected.

TABLE I
Power flow test problems

	buses	branches	nnz(J)
uctew001	4.25k	7.19k	62.7k
uctew002	8.51k	14.4k	125k
uctew004	17.0k	28.8k	251k
uctew008	34.0k	57.6k	502k
uctew016	68.0k	115k	1.00M
uctew032	136k	231k	2.01M
uctew064	272k	462k	4.02M
uctew128	544k	924k	8.05M
uctew256	1.09M	1.85M	16.1M

The power flow solver is implemented in C++ using PETSc (Portable, Extensible Toolkit for Scientific Computation) [35]. All experiments were performed on a single core of a machine with Intel Core i5 3.33GHz CPU and 4Gb memory, running a Slackware 13 64-bit Linux distribution. The problems were solved from a flat start, up to an accuracy of 10^{-6} p.u.

A. Factorizations

In this section, experiments are presented with ILU(k) factorizations of J_0 and ICC(k) factorizations of Φ^* as preconditioner, as discussed in Section V-A. For the factorization levels k, the numbers 4, 8, and 12 are demonstrated. Lower levels led to significantly slower solution times, due to the reduced speed of convergence of the linear solver. Higher levels led to more expensive factorizations, and more fill-in, without improving convergence significantly.

Bi-CGSTAB is used when preconditioning with factorizations with 4 levels. With these preconditioners, a significant amount of linear iterations (30+) is needed in some Newton steps. The short recurrences property of Bi-CGSTAB makes it outperform GMRES for these cases. For factorizations with 8 and 12 levels, less iterations are needed per Newton step, and GMRES outperforms Bi-CGSTAB.

[1]UCTE is a former association of transmission system operators in Europe. As of July 2009, the European Network of Transmission System Operators for Electricity (ENTSO-E), a newly formed association of 42 TSOs from 34 countries in Europe, has taken over all operational tasks of the existing European TSO associations, including UCTE. See http://www.entsoe.eu/

Fig. 1 and 2 show the solution time in seconds, when using ILU(k) factorizations of J_0 and ICC(k) factorizations of Φ^*, respectively. In both figures the results are compared with Newton power flow with a direct linear solver.

All of the presented incomplete factorizations are scalable in the problem size. The factorizations with 12 levels give the best results, with those with 8 levels right behind. The experiments clearly illustrate that a direct solver is not viable for very large problems.

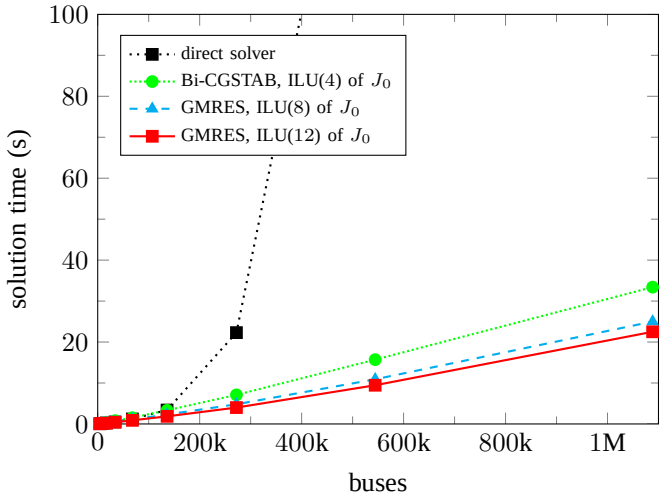

Fig. 1. Comparison of Newton-Krylov power flow preconditioned with ILU(k) factorizations of J_0, and Newton power flow with a direct solver.

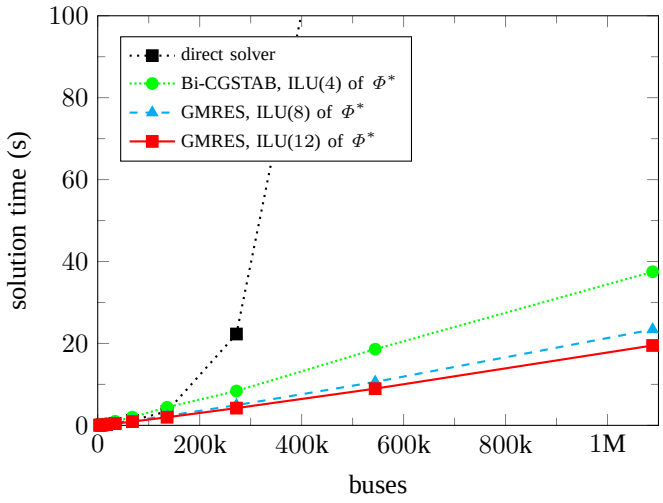

Fig. 2. Comparison of Newton-Krylov power flow preconditioned with ICC(k) factorizations of Φ^*, and Newton power flow with a direct solver.

Table II shows a breakdown of the computation times for the largest test case. The reported times are for the calculation of factorizations (PCSetUp), the forward and backward substitution operations (PCApply), the total time spent on linear solves (KSPSolve), and the total time to solve the problem.

The total time is made up for the better part of linear solves. The remaining time is mostly spent on the calculation of the power mismatch function and Jacobian matrix. The linear

solves are made up from factorizations, forward and backward substitution, and other operations of the GMRES algorithm. Note that direct linear solves only consist of a factorization and a forward and backward substitution.

The results show that J_0 leads to a slightly better preconditioner, in the sense that less overall GMRES iterations are needed to solve the problem. On the other hand, the factorization of Φ^* is faster, and 68 applications are still faster than 58 applications of the factorization of J_0.

Overall, the ICC(12) factorization of Φ^* leads to a slightly faster solution of the uctew256 problem. However, this is mostly due to converging in 6 Newton iterations, where the ILU(12) factorization of J_0 leads to 8 Newton iterations. This can be assumed to more be a matter of some luck, than a fundamental property of Newton-Krylov power flow.

TABLE II
COMPUTATION TIMES FOR THE UCTEW256 TEST CASE

	direct		ILU(12) of J_0		ICC(12) of Φ^*	
	count	time	count	time	count	time
PCSetUp	8	2359	1	5.84	1	3.07
PCApply	8	2	58	5.59	68	4.81
KSPSolve	8	2361	8	16.3	6	14.3
Total		2367		22.5		19.5

B. Krylov Methods as Preconditioner

In this section, experiments with a preconditioned Krylov method as preconditioner—as discussed in Section V-A—are presented. To support this type of preconditioning, FGMRES is used as outer Krylov method. GMRES on J_i and CG on Φ^* are both tested as inner Krylov methods. As preconditioner for the inner iterations, incomplete factorizations with 4 levels are used. With higher level factorizations, convergence is too fast to have both inner and outer iterations perform a meaningful amount of iterations. Lower level factorizations were also tested, and yielded similar results.

When using GMRES on J_i as preconditioner, the Jacobian system can be solved in one outer iteration by solving to high accuracy in the inner iterations. However, since the desired accuracies for the outer iterations, i.e., the forcing terms, are generally between 10^{-1} and 10^{-6} it makes no sense to solve the inner iterations beyond an accuracy of 0.1. The method proved insensitive to the inner iteration accuracy between 0.5 and 0.1, as this ensures that a meaningful amount of inner iterations was executed without ever oversolving the accuracy desired in the outer iterations. The results presented in this section are for an inner tolerance of 0.3

When using CG on Φ^* as preconditioner, the convergence of one outer iteration can never be better than when applying an LU factorization of Φ^* as preconditioner directly. Solving the inner iterations beyond that convergence factor would lead to oversolving. In our experiments this factor was found to be around 0.6, and the best results were attained using this very value as tolerance for the inner iterations.

Fig. 3 shows the solution times for these two techniques, as well as the solution times when applying the used incomplete factorizations as preconditioner for the outer iterations directly. For these test cases, preconditioned Krylov methods as preconditioner do not give significantly better results than applying the incomplete factorization as preconditioner directly.

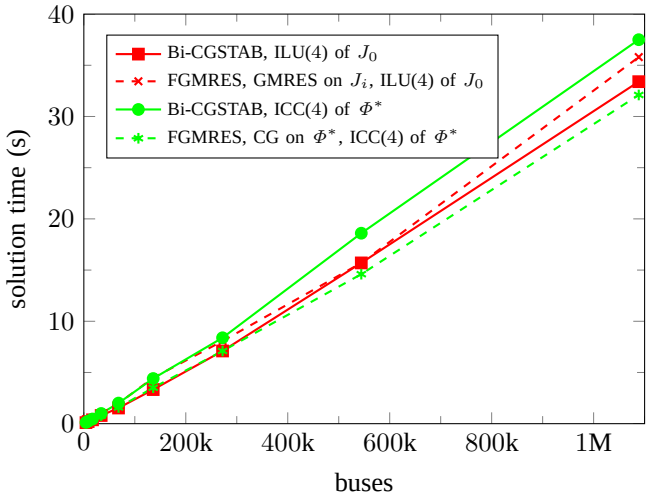

Fig. 3. Comparison of Newton-Krylov power flow with 4-level incomplete factorizations as preconditioner, and with Krylov methods as preconditioner that are preconditioned with the same incomplete factorizations.

C. Algebraic Multigrid

This section reports on experiments with Algebraic Multigrid on Φ^* as preconditioner for Newton-Krylov power flow, as discussed in Section V-C.

Similar to when using preconditioned CG on Φ^* as preconditioner (Section VI-B), setting up the AMG preconditioner to be too good only leads to oversolving. In our experiments, the best results were attained using a single V-cycle with a full Gauss-Seidel sweep as both pre-smoother and post-smoother. On the coarsest grid a direct solver was used, so that the resulting AMG method is stationary. The coarse grid solution is only a minor part of computational time of each V-cycle.

Fig. 4 compares AMG with the ICC(12) factorization of Φ^* as preconditioner. The AMG preconditioner scales very well in the problem size. This is to be expected, because the defining operations of a V-cycle scale linearly in the number of nonzeros in the coefficient matrix, (which is approximately linear in the problem size), and multigrid convergence is independent of the problem size. However, preconditioning with the ICC(12) factorization was still significantly faster than using the AMG preconditioner. Both methods needed about the same amount of linear iterations to converge, but—provided that the fill-in is low—forward and backward substitution operations are much faster than an AMG V-cycle. AMG cycles are easier to parallelize than a factorization, though, and may therefore be preferred in parallel computing environments, including GPU computing.

Multigrid solvers are known to be the best available method for some problems. For example, for Poisson equations discretized on a structured grid. The reason that AMG preconditioning here is slower than preconditioning with an incomplete factorization, is likely due to the structure of the network.

If a power system network consists of many smaller clusters of buses, that may be tightly connected within the cluster but only have a few branches between clusters, then the Jacobian matrix can be reordered to a near block diagonal structure.

Such a structure is very beneficial for factorizations, as it leads to little fill-in. Thus for power systems networks of this type, incomplete factorizations are expected to perform very well as preconditioner.

If, on the other hand, the entire network is tightly connected, then factorizations generally lead to a lot of fill-in, which gets worse the larger the problem becomes. AMG does not share this issue, and can be expected to perform much better for such networks, especially for very large problems.

Our test cases are based on a model of the European grid. Since different countries are generally only connected by very few branches, the structure of our test cases is of the type that favors factorizations. For more tightly connected power systems, AMG is expected to outperform factorization methods for large power flow problems.

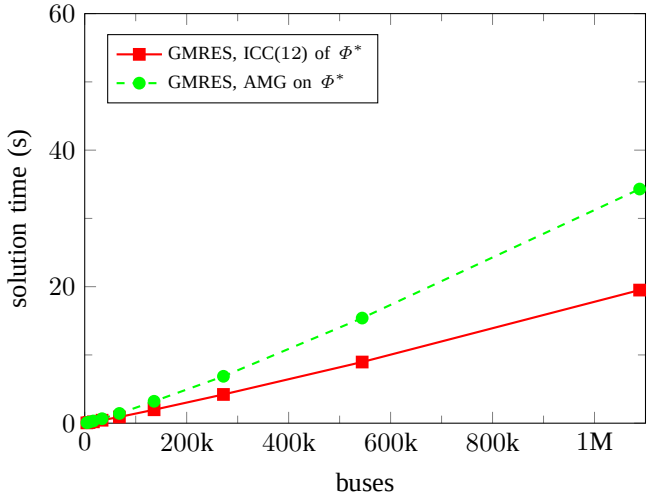

Fig. 4. Comparison of Newton-Krylov power flow preconditioned with the ICC(12) factorization of Φ^* and with AMG on Φ^*.

VII. Conclusions

In this paper, preconditioning techniques for Newton-Krylov power flow solvers have been investigated. Preconditioning based on factorizations, preconditioned Krylov methods, and Algebraic Multigrid were tested and compared.

For the available set of test problems, the best results were attained when using incomplete LU (ILU) factorizations of the initial Jacobian J_0, or incomplete Cholesky (ICC) factorizations of the modified FDLF matrix Φ^*, with 8–12 levels. Using an inner Krylov method—preconditioned with an incomplete factorization—as preconditioner for the outer Krylov iterations, did not provide a fundamental improvement over applying that incomplete factorization as preconditioner to the outer iterations directly.

Algebraic Multigrid on the modified FDLF matrix Φ^* performed very well as preconditioner, but was slower than the best performing incomplete factorizations. It was argued that the used test cases favor factorizations because they consist of a number of loosely connected subnetworks. For more densely connected networks factorizations may suffer from much higher fill-in, and AMG is expected to perform better. Algebraic Multigrid is, further, much better suited for parallel computing than factorizations.

From the results of our research on the solution of large power flow problems, the following recommendations can be made. In a sequential computing environment, use a Newton-Krylov method, preconditioned with incomplete factorizations as detailed in this paper. The fill-in ratio of the factorization should be kept track of, and if it grows too large Algebraic Multigrid can be used as an alternative. In a parallel computing environment, we recommend using Algebraic Multigrid on the modified FDLF matrix Φ^* as preconditioner.

Acknowledgment

The authors would like to thank Robert van Amerongen for his many contributions, providing hands-on experience, and invaluable insight on the subject of power flow analysis. Thanks are also extended to Barry Smith for his help with the PETSc package, and UCTE/ENTSO-E for providing the UCTE study model. Further thanks are due to the JRC Institute for Energy and Transport for partial funding of the research.

References

[1] A. Gomez-Exposito, A. Abur, A. de la Villa Jaen, and C. Gomez-Quiles, "A multilevel state estimation paradigm for smart grids," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 952 – 976, June 2011.

[2] J. McCalley, S. Asgarpoor, L. Bertling, R. Billinton, H. Chao, J. Chen, J. Endrenyi, R. Fletcher, A. Ford, C. Grigg, G. Hamoud, D. Logan, A. P. Meliopoulos, M. Ni, N. Rau, L. Salvaderi, M. Schilling, Y. Schlumberger, A. Schneider, and C. Singh, "Probabilistic security assessment for power system operations," in *IEEE PES General Meeting, Denver, USA*, July 2004.

[3] D. Bakken, A. Bose, K. M. Chandy, P. P. Khargonekar, A. Kuh, S. Low, A. von Meier, K. Poolla, P. P. Varaiya, and F. Wu, "Grip grids with intelligent periphery: Control architectures for grid2050," in *Proceedings IEEE SmartGridComm Conference*, Brussels, Belgium, October 2011.

[4] P. Pinson, H. Madsen, H. A. Nielsen, G. Papaefthymiou, and B. Klöckl, "From probabilistic forecasts to statistical scenarios of short-term wind power production," *Wind Energy*, vol. 12, no. 1, pp. 51–62, 2009. [Online]. Available: http://dx.doi.org/10.1002/we.284

[5] K. Morison, L. Wang, and P. Kundur, "Power system security assessment," *IEEE power & energy magazine*, vol. 2, no. 5, pp. 30 – 39, September - October 2004.

[6] R. Idema, D. J. P. Lahaye, C. Vuik, and L. van der Sluis, "Scalable Newton-Krylov solver for very large power flow problems," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 390–396, February 2012.

[7] P. Schavemaker and L. van der Sluis, *Electrical Power System Essentials*. Chichester: John Wiley & Sons, 2008.

[8] W. F. Tinney and C. E. Hart, "Power flow solution by Newton's method," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-86, no. 11, pp. 1449–1449, 1967.

[9] W. F. Tinney and J. W. Walker, "Direct solutions of sparse network equations by optimally ordered triangular factorization," *Proceedings of the IEEE*, vol. 55, no. 11, pp. 1801–1809, 1967.

[10] B. Stott and O. Alsac, "Fast decoupled load flow," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-93, no. 3, pp. 859–869, 1974.

[11] R. A. M. van Amerongen, "A general-purpose version of the fast decoupled loadflow," *IEEE Transactions on Power Systems*, vol. 4, no. 2, pp. 760–770, 1989.

[12] A. J. Monticelli, A. Garcia, and O. R. Saavedra, "Fast decoupled load flow: Hypothesis, derivations, and testing," *IEEE Transactions on Power Systems*, vol. 5, no. 4, pp. 1425–1431, 1990.

[13] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, "Inexact Newton methods," *SIAM J. Numer. Anal.*, vol. 19, no. 2, pp. 400–408, 1982.

[14] S. C. Eisenstat and H. F. Walker, "Choosing the forcing terms in an inexact Newton method," *SIAM J. Sci. Comput.*, vol. 17, no. 1, pp. 16–32, 1996.

[15] Y. Saad, *Iterative methods for sparse linear systems*, 2nd ed. SIAM, 2000.

[16] V. V. Voevodin, "The problem of non-self-adjoint generalization of the conjugate gradient method is closed," *U.S.S.R. Comput. Math. and Math. Phys.*, vol. 22, pp. 143–144, 1983.

[17] V. Faber and T. Manteuffel, "Necessary and sufficient conditions for the existence of a conjugate gradient method," *SIAM J. Numer. Anal.*, vol. 21, pp. 352–362, 1984.

[18] H. A. van der Vorst, "Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for solution of nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 13, pp. 631–644, 1992.

[19] G. L. G. Sleijpen, H. A. van der Vorst, and D. R. Fokkema, "BiCGstab(ℓ) and other hybrid Bi-CG methods," *Numerical Algorithms*, vol. 7, pp. 75–109, 1994.

[20] P. Sonneveld and M. B. van Gijzen, "IDR(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations," *SIAM J. Sci. Comput.*, vol. 31, no. 2, pp. 1035–1062, 2008.

[21] Y. Saad and M. H. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 7, pp. 856–869, 1986.

[22] H. A. van der Vorst and C. Vuik, "GMRESR: a family of nested GMRES methods," *Num. Lin. Alg. Appl.*, vol. 1, pp. 369–386, 1994.

[23] Y. Saad, "A flexible inner-outer preconditioned GMRES algorithm," *SIAM J. Sci. Comput.*, vol. 14, no. 2, pp. 461–469, March 1993.

[24] A. J. Flueck and H. D. Chiang, "Solving the nonlinear power flow equations with an inexact Newton method using GMRES," *IEEE Transactions on Power Systems*, vol. 13, no. 2, pp. 267–273, 1998.

[25] F. de León and A. Semlyen, "Iterative solvers in the Newton power flow problem: preconditioners, inexact solutions and partial Jacobian updates," *IEE Proc. Gener. Transm. Distrib*, vol. 149, no. 4, pp. 479–484, 2002.

[26] D. Chaniotis and M. A. Pai, "A new preconditioning technique for the GMRES algorithm in power flow and $P-V$ curve calculations," *Electrical Power and Energy Systems*, vol. 25, pp. 239–245, 2003.

[27] Y.-S. Zhang and H.-D. Chiang, "Fast Newton-FGMRES solver for large-scale power flow study," *IEEE Transactions on Power Systems*, vol. 25, no. 2, pp. 769–776, May 2010.

[28] H. Dag and F. L. Alvarado, "Toward improved uses of the conjugate gradient method for power system applications," *IEEE Transactions on Power Systems*, vol. 12, no. 3, pp. 1306–1314, August 1997.

[29] M. R. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *J. Res. Nat. Bur. Stand.*, vol. 49, no. 6, pp. 409–436, December 1952.

[30] R. Idema, D. J. P. Lahaye, C. Vuik, and L. van der Sluis, "Fast Newton load flow," in *Transmission and Distribution Conference and Exposition, 2010 IEEE PES*, April 2010, pp. 1–7.

[31] J. A. Meijerink and H. A. van der Vorst, "An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix," *Mathematics of Computation*, vol. 31, no. 137, pp. 148–162, January 1977.

[32] ——, "Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems," *Journal of Computational Physics*, vol. 44, no. 1, pp. 134–155, 1981.

[33] P. R. Amestoy, T. A. Davis, and I. S. Duff, "An approximate minimum degree ordering algorithm," *SIAM J. Matrix Anal. Appl.*, vol. 17, no. 4, pp. 886–905, October 1996.

[34] U. Trottenberg, C. W. Oosterlee, and A. Schüller, *Multigrid*. Academic Press, 2001.

[35] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang, "PETSc users manual," Argonne National Laboratory, Tech. Rep. ANL-95/11 - Revision 3.1, 2010, http://www.mcs.anl.gov/petsc/.

Reijer Idema obtained his MSc in Applied Mathematics (Computational Science and Engineering specialization) at the Delft University of Technology in 2007. Currently he is a PhD student at the Numerical Analysis research group of the Delft Institute of Applied Mathematics, Delft University of Technology.

Georgios Papaefthymiou received the Dipl. Eng. degree in electrical and computer engineering from the University of Patras, Patras, Greece, in 1999, and the PhD degree from Delft University of Technology, Delft, The Netherlands, in 2007. He currently works as a consultant with the Power Systems and Markets Group, Ecofys, and as Research Associate with the Power Systems Group, of Delft University of Technology. His current research interests include modeling of uncertainty in power systems and the planning and operation of power systems with high penetration of renewables and energy storage, wind power forecasting, and market participation issues.

Domenico Lahaye obtained his MSc in Applied Mathematics at the Free University of Brussels in 1994, his post-graduate degree in Mathematics for the Industry at the Eindhoven University of Technology in 1996, and his PhD in Computer Science at the Catholic University of Leuven in 2001. After having held positions at the Center for Advanced Studies, Research and Development in Sardinia, and at the National Research Center for Mathematics and Computer Science in the Netherlands, he joined the Numerical Analysis research group of the Delft Institute of Applied Mathematics, Delft University of Technology, as assistant professor in 2007.

Kees Vuik obtained his MSc in Applied Mathematics at the Delft University of Technology in 1982. After a short stay at the Philips Research Laboratories, he obtained his PhD in Mathematics at Utrecht University in 1988. Thereafter he became employed at the Delft University of Technology, where he holds the position of full professor of the Numerical Analysis research group. In 2007 he additionally became director of the Delft Center of Computational Science and Engineering.

Lou van der Sluis obtained his MSc in Electrical Engineering at the Delft University of Technology in 1974. He joined the KEMA High Power Laboratory in 1977. In 1990 he became a part-time professor at the Delft University of Technology, and since 1992 he is employed as full professor of the Power Systems Department of the Delft University of Technology.