COMPLEX VECTOR LOAD FLOW

G. L. VIVIANI

Lamar University
P.O. Box 10029 LUS
Beaumont, Texas 77710

Abstract – A complex vector approach to the load flow problem with two important and unique contributions is presented. Traditionally, the intermediate calculations for the Newton-Raphson solution of the load flow problem require operations on elements of matrices (as opposed to operating on vectors themselves). In this approach all intermediate calculations are performed by conventional vector operations plus one other operation. Therefore, only a specific set of operations requires optimization to make the entire algorithm more efficient. This is particularly useful for array processor applications and dedicated VLSI circuitry. In addition, this approach allows for easy inclusion of tap changing transformers and phase shifters into the analysis. A more concise and understandable load flow problem statement is presented, which ultimately results in dramatic improvements in computation efficiency, making microcomputers able to perform load flow calculations faster than minicomputers.

1. Introduction

There is a voluminous amount of research regarding the power system load flow problem. This document will not review this body of material, but instead it will improve upon existing approaches. The interested reader may study any of the following to become familiar with the load flow algorithm [1,2,3]. Additionally, there have been some "improvements" to the basic algorithm such as the decoupled load flow, [4,5]. In this paper a complex formulation is proposed which follows directly from a desire to produce a formulation that is both analytic and vector "oriented". Another paper, [6], proposed a similar complex formulation of the load flow problem. Still, this previous work relied on operations with the individual elements of matrices. Hence there was no benefit in utilizing a complex formulation as opposed to a more traditional approach, with the possible exception of some slight computational efficiencies.

The primary motivation for complex vector load flow (CVLF) is increased throughput. This is the focus of a recent paper, [7], which indicates that two orders of magnitude reduction in computational time is possible by application of VLSI technology. The CVLF algorithm is formulated to require only a specific set of routine operations. Hence, it is conceivable to develop hardware devices which perform these operations (see next section) very efficiently. The resulting increased throughput should be even better than two orders of magnitude, since in [7], only the VLSI implementation of the solution of linear equations was considered. Since the operations required by the approach of this paper are "standard," hardware to support these calculations already exists; it also satisfies the computational needs of other technical specialties in engineering and science.

2. Operations

The CVLF requires the following conventional vector mathematical operations:

1. matrix-matrix multiplication: $C^{n \times k} \quad C^{n \times m} \times C^{m \times k}$

2. matrix addition(subtraction): $C^{n \times k} \quad C^{n \times k} \pm C^{n \times k}$

3. complex conjugation $(\ )^*$

4. complex solution of simultaneous equations.

In addition, another complex matrix-vector operation is required. The symbol $(\ )^D$ denotes using the vector inside the parenthesis to form a square diagonal matrix. A mapping from complex n space to complex n x n space is implied.

3. Problem Formulation

Following [6], it is desirable to determine a solution to the complex vector system of complex vector arguments,

$F(V,V^*) = 0$
where
$F \in C^{2n}$; $V \in C^n$; n = number of nodes.

Here, V denotes complex nodal voltages and F denotes complex mismatch power, a nonlinear function of V. A Newton-Raphson solution approach suggests the following equation, for all nodes except the slack node:

$$\begin{bmatrix} J^k \end{bmatrix} \begin{bmatrix} V^k \\ V^{*k} \end{bmatrix} = \begin{bmatrix} F^k \end{bmatrix}$$

and

$$V^{k+1} = V^k + \delta V^k \, .$$

For convenience, the V vectors are partitioned so that the first element is the slack node, followed by all PV (voltage controlled) nodes, followed by all PQ (load) nodes, $V^t = (V_{slack}, V_{pv}, V_{pq})$. The J matrix is termed the Jacobian matrix in power system literature. For this formulation, unlike previous formulations, the elements of the J matrix are conveniently stated in matrix notation and conventional vector mathematics. The vector F is divided into four subvectors associated with the various nodes in the system. These have also been partitioned accordingly. F is written as:

$$F = (f_1, f_2, f_3, f_4)^t \ .$$

Defining P_{sk} as the specified power for a generator node k, S_k as the nodal complex power injected into node k, S_{sk} as the specified complex power at a load node, and V_s as the specified voltage at a generator node, the following definitions for the subelements of the vector F are given:

$f_{1k} \equiv P_{sk} - 0.5(S_k + S_k^*)$	for generator node k
$f_{2k} \equiv S_{sk} - S_k$	for load node k
$f_{3k} \equiv V_{sk}^2 - V_k V_k^*$	for generator node k
$f_{4k} \equiv S_{sk}^* - S_k^*$	for load node k.

Because of this formulation, it is possible to state elements of the J matrix, in matrix notation, as:

$$\frac{\partial P}{\partial V} = \frac{1}{2}\left((I^*)^D + (V)^D Y \right)$$

$$\frac{\partial P}{\partial V^*} = \frac{\partial P}{\partial V}^*$$

$$\frac{\partial S^*}{\partial V} = (V^*)^D Y$$

$$\frac{\partial S^*}{\partial V^*} = (I)^D$$

$$\frac{\partial V^2}{\partial V^*} = (V)^D$$

$$\frac{\partial V^2}{\partial V} = \left(\frac{\partial V^2}{\partial V^*}\right)^*$$

$$\frac{\partial S}{\partial V} = \left(\frac{\partial S^*}{\partial V^*}\right)^*$$

$$\frac{\partial S}{\partial V^*} = \left(\frac{\partial S^*}{\partial V}\right)^*$$

where

$$S = V^D I^*$$

$$I = (Y\ V)$$

$$V^2 = V^D V^*$$

$$Y \equiv \text{nodal admittance matrix}$$

All the elements of J are found as submatrices (in block form) of the above matrices, which are elements of $C^{n \times n}$. Further, many of the above matrices result from a complex conjugation operation which should require very little CPU time, regardless of the particular implementation. Hence, formation of the Jacobian matrix is considerably more efficient than conventional approaches.

A reduced set of equations is determined as follows:

$$[J_o] [\delta V] = [F_o]$$

with

$$[J_o] = [J_1 - J_2 J_4^{-1} J_3]$$

$$[F_o] = [F_1 - J_2 J_4^{-1} F_2].$$

For this reduced set, J and F are conveniently partitioned as,

$$J = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \qquad J_1, J_2, J_3, J_4 \in C^{(n-1) \times (n-1)}$$

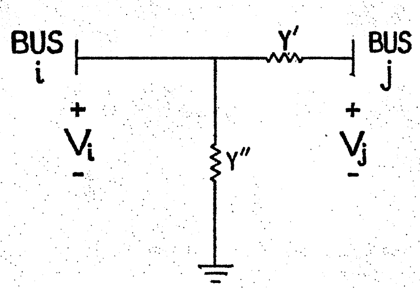
and

$$F = (F_1 , F_2)^t = ((f_1,f_2),(f_3,f_4))^t.$$

Solution of this system of equations proceeds as in conventional Newton-Raphson solutions to the load flow problem. Reference [6] sites some improvements in computational efficiencies in addition to a method for even faster convergence, [8], as compared to a conventional Newton-Raphson load flow. By expressing the formulation in vector format, dedicated hardware can further improve throughput. At the same time, the software development necessary to implement an algorithm is more efficient and understandable.

4. Related Results

A complex vector formulation of the load flow problem also suggests a more elegant and efficient method for treating transformers and phase shifters. It is not difficult to show that an equivalent model for a branch from node i to node j, including an ideal transformer and ideal phase shifter, is given as (see [9], or [12]):



(Y' and Y'' are embedded in the calculations which follow.)

This suggests an equivalent representation for the nodal admittance matrix as:

$$Y_{nodal} = B'^{*t}\, Y_{prim}\, [\alpha]\, B' + Y_{shunt}.$$

(note $*t \equiv$ hermitian operation)

Here, B' is the complex branch incidence matrix, including the effect of phase shifters in a branch. B' is formed as follows:

$$B' = \begin{cases} +1\angle\frac{\theta}{2} & \text{if starting node and phase shift reference} \\ -1\angle\frac{\theta}{2} & \text{if ending node and not phase shift reference} \\ 0 & \text{otherwise .} \end{cases}$$

($B' \in C^{m \times n}$;if no phase shifter present, $\theta = 0$; m = number of branches)

$[\alpha]$ = diagonal matrix ($C^{m \times m}$) of tap ratios

Y_{prim} = diagonal matrix ($C^{m \times m}$) of branch admittances

The angle, θ, is the total amount of phase shift present in the line (branch). The diagonal matrix Y_{shunt} represents shunt admittance between each node and the reference (ground) node. Assuming that untapped sides of transformers always coincide with phase shift reference nodes, then the elements of the diagonal matrix Y_{shunt} are determined as follows:

$$Y^1_{shunt\ j} = \begin{cases} \sum_k \alpha_k y_k \ Q_{1k} & \text{if node } j \text{ is untapped side of branch } k \\ \sum_k \alpha_k y_k \ Q_{2k} & \text{if node } j \text{ is tapped side of branch } k \end{cases}$$

where

$Q_{1k} = (\alpha_k - 1)$

$Q_{2k} = (\alpha_k^{-1} - 1)$

y_k = branch admittance for branch k

α_k = turns ratio of transformer k.

Other shunt admittances may also exist. For convenience, these will be lumped together in a diagonal matrix called Y_{other}. Hence,

$$Y_{shunt} = Y^1_{shunt} + Y_{other}.$$

Note: It may be possible to specify Y_{shunt} in vector format if a distinction between tapped and untapped nodes is maintained. Such a formulation would be useful for optimal power flow studies. Y_{shunt} is never zero if transformers are present.

5. Discussion

The operations of section 2 are readily implemented with most commercially available array processors. Hence, the CVLF software consists of two major components. The first is the portion of code for setting up required matrices and the second is the number crunching part. The information presented in this paper is sufficient to devise a scheme for efficient setup of appropriate matrices. Improved number crunching, however, is the real benefit of the CVLF. In this portion of the code, all operations are described in section 2. No other significant operations are necessary.

For routine iterations of the CVLF algorithm, only the specified operations of section 3 are required for updating the complex Jacobian matrix and solving for the revised voltages. Assuming hardware to perform these operations is considerably faster than conventional approaches, there is no need to further approximate the Jacobian matrix as is done in the Stott decoupled approach. In [11], Tinney suggests the future demise of sparsity programming approaches to load flow solutions with the development of vector based computational devices. He also indicates the need for suitable vector approaches to load flow solutions which ultimately would eliminate the dependence on more conventional approaches and take advantage of new technologies. The CVLF algorithm is intended to ride the crest of future technological achievements in scientific and engineering numerical analysis.

6. Example

The algorithm described has been programmed on a microcomputer. All aspects of the problem formulation have been tested. Within the speed and memory limitations of the computer, the algorithm performs very well. For all cases tested the algorithm requires exactly the same number of iterations to converge to a solution as the conventional Newton-Raphson approach for a specified mismatch tolerance and initial voltage profile. In [6] the non-vector formulation of the problem was tested for larger systems, verifying the numerical stability of the approach. It is assumed that with the availability of appropriate hardware, tremendous reductions in computational time would be possible as described in [7]. A sample application is shown in Appendix A.

To get a feel for the sorts of speed improvements possible with a vectorized approach to load flow in conjunction with a commercially available array processor (AP), several more sample systems were tested. The conclusions are best summarized by the following table.

TIME per ITERATION

CASE:	IEEE 14 NODE	IEEE 30 NODE	IEEE 118 NODE
with AP	< 1 sec	3 sec	79 sec
without AP	9	76 sec	4900 sec

TABLE 1

As indicated, dramatic speed improvements are possible by application of an array processor. The array processor utilized in the above calculations is not as fast as what is presently available. Since, other existing array processors are 15 times faster than the one employed, another 15 times decrease in time per iteration is readily realized, for a small additional expense.

7. Conclusions

The CVLF algorithm provides an efficient formulation of the load flow problem, suitable for use with modern computational advances such as array processors and/or VLSI circuitry. The increased throughput makes this algorithm very attractive for real time control applications, such as those encountered in dynamics and stability studies of power systems. The proposed formulation may lead to useful efficiencies for optimal power flow studies.

8. Acknowledgement

The author would like to thank IBM and SKY Computers for their support of this research. Indirect support of this research was sponsored by Gulf States Utilities.

9. References

[1] Stott, B., "Review of load-flow calculation methods," Proc. IEEE., vol 62, 1974, pp. 916-929.

[2] Tinney, W. F., Hart, C. E., "Power Flow Solution by Newton's Method," IEEE Transactions on PAS, vol PAS-86, 1967, pp.1449-1456.

[3] Bandler, J. W., El-Kady, M. A., "A New Method for Computerized Solution of Power Flow Equations," IEEE Transactions on PAS. vol PAS-101, 1982, pp. 1-10.

[4] Stott, B., "Decoupled Newton Load Flow," IEEE Transactions on PAS, vol PAS-91, 1972, pp. 1955-1959.

[5] Stott, B., Alsac, O., "Fast Decoupled Load Flow," IEEE Transactions on PAS, vol PAS-93, 1974, pp. 859-869.

[6] Gomez, F., Semlyen, A., "The Newton-Raphson Load Flow with Complex Variables," IEEE PES Summer Meeting, Conference Paper C 72 483-6, 1972.

[7] Shanblatt, M. A., Leung, Y., "The Promise of VLSI for Load Flow Computation Enhancement," to appear IEEE Transactions on PAS, 84 WM 095-6, 1984.

[8] Sasson, A. M., Trevino, C., Aboytes, F., "Improved Newton's Load Flow Through a Minimization Technique," IEEE Transactions on PAS, 71 TP 18-PWR, 1971, pp. 1974 - 1981.

[9] Pai, M., Computer Techniques in Power System Analysis, Tata McGraw-Hill, New Delhi, 1980.

[10] Stagg, G., El-Abiad, A., Computer Methods in Power System Analysis, McGraw-Hill, 1968, New York.

[11] Erisman, Neves, Dwarakanath (editors), "Electric Power Problems: The Mathematical Challenge," SIAM, Philadelphia, 1980, pp. 17 and 520 - 523.

[12] Britton, J., "Improved Load Flow Performance Through a More General Equation Form," IEEE Transactions on PAS, vol. PAS-90, 1971, pp. 109-115.

Appendix A

A simple example for a small system is presented to verify and illustrate the CVLF algorithm. For clarity, important matrices are specified in full. Complex matrices are presented in two parts, real then imaginary, in rectangular coordinates. The sample system is shown in Figure A1. All quantities are specified in per unit.

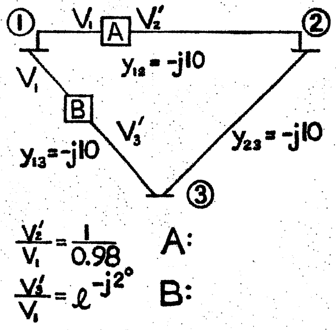


Figure A1

For this system, the nodal admittance matrix Y_{bus} is determined as:

Y_{bus} =

```
       0                0                -0.348995
       0                0                 0
       0.348995         0                 0

      -20.4123          10.2041           9.99391
       10.2041         -20                10
        9.99391         10               -20
```

B' =

```
      -1                1                0
      -0.999848         0                0.999848
       0                1               -1

       0                0                0
       0.0174525        0                0.0174525
       0                0                0
```

Y_{prim} [α] =

```
       0                0                0
       0                0                0
       0                0                0

      -10.2041          0                0
       0               -10               0
       0                0               -10
```

Y_{shunt} =

```
        0              0              0
        0              0              0
        0              0              0

        -0.208247      0              0
        0              0.208247       0
        0              0              0
```

The specified complex powers and voltage magnitudes are given by, F^t,

F^t =

```
        0.9     -0.2    1.1025 -0.2

        0       -0.012 0       0.012
```

Setting the slack voltage to 1.05 pu, a solution to the load flow problem in three iterations, with the CVLF algorithm, is determined as,

V^t =

```
        1.05    1.04931         1.04868

        0       0.038902        -0.00880855
```

I^t =

```
        -0.666667      0.849905       -0.190606

        -0.242202      0.214933       0.0130436
```

S^t =

```
        -0.7           0.9            -0.2

        0.254313       -0.193157      -0.12
```

Manuscript received in final form, April 8, 1985
Request reprints from G. L. Viviani