An efficient load flow solution method for radial distribution network is presented in this paper. The directed graph of a radial network shows power drawn and path of power flow from the reference node to the leaf end. A simple technique is used to sort input data and based on network graphical information power flow equations are formulated in matrix form to satisfy the need of distribution automation. In the algorithm input data are arranged to obtain sorted power injections, losses, branch currents and complex voltages using the novel equations. The ordered form of collection of input data and formulation of load flow information matrix are found to be excellent convergent over conventional methods. The methodology proposed here has been successfully demonstrated on an IEEE 15-Node network. The test results obtained are validated through MATLAB Ver. 7.0. The proposed method presented in the paper is found to be flexible and efficient.

**Keywords:** Distribution network, node load, branch load, sorted information matrix.

1. **INTRODUCTION**

It has been realized that Modern Distribution System (MDS) requires precise formulation and efficient algorithm to resolve power flow solution in a complex radial distribution network. Power flow study is the backbone of power system analysis and design. It is necessary for planning, operation, economic scheduling and exchange of power between utilities. In addition, power flow analysis is required for many other analyses such as transient stability and contingency studies as in [1]. Some of the basic power flow algorithms, which are already developed and applied, include methods of Newton Raphson and Gauss Seidal. Some of the prominent features of the electric distribution systems are: radial or weakly meshed network with wide range of R/X; multiphase, unbalanced, grounded or ungrounded operation; dispersed generator; unbalanced distributed loads; extremely large number of network branches/ nodes. The method developed as in [2] for transmission network, expresses that NR method of power flow algorithm for solving high R/X ratio of distributed network was not successful, as it diverged for several network studied. The decoupling assumptions necessary for simplifications used in the standard fast-decoupled NR method as documented in [3] are often not valid for distribution system. Implementation of fast decoupled power flow for unbalanced radial distribution systems as in [4] requires special process of ordering for the input data format as mentioned in [5].

The ladder network theory as in [6] considers dependency of the load demands on voltage changes, which is much similar to the forward-backward substitution method as reported in [7], which consider current as variable to obtain power flow solution. Likewise, many other algorithms for radial distribution network have been developed as in [8-9]. A series of interconnected ladder network based method as in [10] found less efficient to assess, modify, update, delete in sorted form due to the complexity in the numbering scheme. The network topology based algorithm as in [11] using current as variable has problems for...
large radial networks, like developing the two constant matrices by writing special program. It also takes more computation time to converge the solution, when compared with the backward-forward technique and ladder network theory methods as expressed in [12] for the given data as in [13]. In proposed method, it is identified that sorted information of constant power is the independent of assumed voltage. Because of the losses in the network are about 8% of the system demands and hence the initial error anticipation will be small. These points are found to be useful to obtain fast and efficient solution technique.

Tellegen’s theorem as in [14] states that the algebraic sum of complex powers meeting at a node is zero. Using TT, the backward sweep of load power and losses are performed from downstream to upstream of a network. The sorted information matrix developed during the backward sweep is helpful to compute injected branch powers. The current drawn by each load is calculated using ratio of conjugate power to voltage. Finally, directly apply KVL in the forward sweep to obtain power flow solution. The validation of algorithm performance is carried out by writing programs in MATLAB specification as in [15] for the networks data as reported in [16].

1. SOLUTION METHODOLOGY

Numbering and arrangement of input data

The node oriented numbering for a typical radial distribution network is depicted in the figure 1 having ‘n’ nodes and b (= n-1) branches. The nodes in the network are numbered level by level from left to right side of the network. Likewise, proceeds till the end of network.

In case branch, which lies between top kth node and bottom (k+1)th node of radial network, it is suggested that the branch number is the downstream node number itself. The same suggestion is applicable to branch, wherein power flow from upstream to downstream of network.

For the network as shown in Fig. 1, tabulate the input information for both line data and node data in such way that the receiving end node must be in an ascending order as
### TABLE I. Tabulation of input data

<table>
<thead>
<tr>
<th>Sending Node</th>
<th>Receiving Node</th>
<th>Line Impedance</th>
<th>Nodal Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>(Z_2)</td>
<td>(S_2)</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>(Z_3)</td>
<td>(S_3)</td>
</tr>
<tr>
<td>3</td>
<td>(i)</td>
<td>(Z_i)</td>
<td>(S_i)</td>
</tr>
<tr>
<td>(r)</td>
<td>(i+1)</td>
<td>(Z_{i+1})</td>
<td>(S_{i+1})</td>
</tr>
<tr>
<td>(k)</td>
<td>(i+2)</td>
<td>(Z_{i+2})</td>
<td>(S_{i+2})</td>
</tr>
<tr>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
</tr>
<tr>
<td>(j)</td>
<td>(n)</td>
<td>(Z_n)</td>
<td>(S_n)</td>
</tr>
</tbody>
</table>

Here \(i, i+1, i+2, \ldots, n\) are numbers assigned to receiving end nodes. The given impedance data \([ZD]\) and node data \([SD]\) dimensions are read as \((n-1) \times 1\) from Table I.

\[
[ZD] = \begin{bmatrix}
Z_2 \\
Z_3 \\
Z_i \\
Z_{i+1} \\
Z_{i+2} \\
\vdots \\
Z_n
\end{bmatrix}
\]

(1)

\[
[SD] = \begin{bmatrix}
S_2 \\
S_3 \\
S_i \\
S_{i+1} \\
S_{i+2} \\
\vdots \\
S_n
\end{bmatrix}
\]

(2)

**Formation of node-load to branch-load [NLBL] information matrix**

The paths \(P_i\) are marked for IEEE 15-Node network data [16] as shown in Fig.2. These paths are always traveling along the branches from the reference node to the node at which the voltage is to be determined.
Fig. 2 Directed graph represents branch, impedance and node information of network.

Consider the 2\textsuperscript{nd} path traveling along branch path impedance $Z_{12}$ in between node 1 and node 2 i.e line section 1-2. Similarly 3\textsuperscript{rd} path travels along the line section 1-2 -2-3 having two branches $Z_{12} - Z_{23}$ and three nodes 1-2-3. Tabulate path information as shown in Table II. In the same way path information are identified from $P_1$ to $P_n$ till the end of the network.

Table II. Information about (Pi), Number of nodes and branches in path of Fig. 2

<table>
<thead>
<tr>
<th>Path $P_i$ to know voltage at the end</th>
<th>Nodal path information and (Total No. of Nodes)</th>
<th>Branch-Path information ( Total No. of branches)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_2$</td>
<td>1,2 (2)</td>
<td>$Z_{12}$ (1)</td>
</tr>
<tr>
<td>$P_3$</td>
<td>1,2,3 (3)</td>
<td>$Z_{12}, Z_{23}$ (2)</td>
</tr>
<tr>
<td>$P_4$</td>
<td>1,2,3,4 (4)</td>
<td>$Z_{12}, Z_{23}, Z_{34}$ (3)</td>
</tr>
<tr>
<td>$P_5$</td>
<td>1,2,3,4,5 (5)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{45}$ (4)</td>
</tr>
<tr>
<td>$P_6$</td>
<td>1,2,3,4,5,6 (6)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{45}, Z_{56}$ (5)</td>
</tr>
<tr>
<td>$P_7$</td>
<td>1,2,3,4,5,6,7 (7)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{45}, Z_{56}, Z_{67}$ (6)</td>
</tr>
<tr>
<td>$P_8$</td>
<td>1,2,3,4,5,6,7,8 (8)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{45}, Z_{56}, Z_{67}, Z_{78}$ (7)</td>
</tr>
<tr>
<td>$P_9$</td>
<td>1,2,3,4,5,6,7,8,9 (9)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{45}, Z_{56}, Z_{67}, Z_{78}, Z_{89}$ (8)</td>
</tr>
<tr>
<td>$P_{10}$</td>
<td>1,2,3,4,10 (5)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{410}$ (4)</td>
</tr>
<tr>
<td>$P_{11}$</td>
<td>1,2,3,4,10,11 (6)</td>
<td>$Z_{12}, Z_{23}, Z_{34}, Z_{410}, Z_{1011}$ (5)</td>
</tr>
<tr>
<td>$P_{12}$</td>
<td>1,2,3,12 (4)</td>
<td>$Z_{12}, Z_{23}, Z_{1212}$ (3)</td>
</tr>
<tr>
<td>$P_{13}$</td>
<td>1,2,3,12,13 (5)</td>
<td>$Z_{12}, Z_{23}, Z_{1212}, Z_{1213}$ (4)</td>
</tr>
<tr>
<td>$P_{14}$</td>
<td>1,2,3,12,13,14 (6)</td>
<td>$Z_{12}, Z_{23}, Z_{1212}, Z_{1213}, Z_{1314}$ (5)</td>
</tr>
<tr>
<td>$P_{15}$</td>
<td>1,2,3,12,13,14,15 (7)</td>
<td>$Z_{12}, Z_{23}, Z_{1212}, Z_{1213}, Z_{1314}, Z_{1415}$ (6)</td>
</tr>
</tbody>
</table>
Procedure to build [NLBL] information matrix
Create [NLBL] having null matrix of order $n \times n$. Obtain the first path branch impedance i.e $Z_{12}$ from the information Table II and mention +1 in the first column of [NLBL]. Similarly, as per branch information available for each path place number of +1’s in [NLBL] as information along path. The number of 1’s in the column of [NLBL], gives the total number of branches or number of load drawn at nodes in that path sequence. The necessary detail information is tabulated in column 3 and 4 of Table II and remaining positions are filled with zeros. As an example [NLBL] for IEEE 15-Node network data [16] can be written as:

\[
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

(3)
The suggested programming steps to build [NLBL] are as follows:

a) Create [NLBL] having null matrix of order $n \times n$.
b) Consider path $P_i$, if $Z_{i,i+1}$ is available in that path, then place +1 in that column position.
c) For the range of node number, $i=2,3,\ldots,n$ select 1 as incremental value.
d) Set +1 in the diagonal position of [NLBL]$_{n \times n}$
e) In [NLBL] receiving node column position $1,\ldots,(i+1)^{th}$, $(i+2)^{th}$,...$n$ are filled with 1’s as per the nodal connectivity information available along the path. Then simply add $i^{th}$ node column number information position to $(i+1)^{th}$ node column position. In the same way add column wise for the subsequent positions.
f) The dimension of the [NLBL] is reduced to $(n-1) \times (n-1)$ by removing the first row and first column of matrix, which is suitable to compute $(n-1)$ unknown nodal voltages.

2.3 Calculation of Power Injections $[S_i]$ at nodes
Fig.4 shows power injection in the upstream power as summation of loads and losses in the downstream
Fig. 4. Representation of load at nodes and losses in the branches.

\[ [S_i] = \sum_{k=\text{downstream}}^{n-\text{upstream}} \text{Loads} + \sum_{k=\text{downstream}}^{n-\text{upstream}} \text{Losses} \quad \text{branches} \]

\[ [S_i] = \sum_{k=\text{downstream}}^{n-\text{upstream}} [S_d(k)] + \sum_{k=\text{downstream}}^{n-\text{upstream}} [S_i(k)] \quad \text{nodes} \]

The first term (loads) of Equation (5) is independent of assumed voltage, whereas second term (losses) depends on square of absolute value of voltage. It is noted that the losses are about 8% of the system demand and therefore the initial error anticipation will be small.

a) To obtain the summation of power injections \([S]\) at each node excluding losses in each branch multiply information matrix\([NLBL]\) and sorted load data \([SD]\) form relation (2) and (3) respectively as

\[ [S] = [NLBL][SD] \quad \text{(6)} \]

b) To obtain the summation of power injections \([S_i]\) at each node including losses add branch losses \([BL]\) to the equation (6) as

\[ [S_i] = [NIBP][SD] + [BL] \]

To find the loss in the branches of a network in sorted order, arrange the elements \([S]\) and \([V]\) in diagonal form as

\[ [BL] = \text{abs(diag}([SD]/[V])) \times \text{diag}([SD]/[V]) \]
Where \( \text{diag}(SD) = \) Arrangement of elements of \([S]\) in diagonal form to match the matrix dimension as

\[
[\text{SD}] =
\begin{bmatrix}
S_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & S_3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & S_4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & S_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & S_6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & Z_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & Z_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_9 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{10} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{11} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{12} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{13} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{14} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & Z_{15}
\end{bmatrix}
\]

Similarly assumed voltage matrix \([V]\) also

\[ [S_i] = [\text{NIBP}] [\text{SD}] + \text{abs(diag}([\text{SD}/[V]])^2)[ZD] \] \hspace{1cm} (7)

2.4 Calculation of current injection matrix \([I_i]\)

Injected branch-current matrix is the conjugate of the ratio of the injected powers to voltage can be expressed in matrix form as

\[ [I_i] = ([S_i]/[V_i])^* \] \hspace{1cm} (8)

The ‘/’ command indicates element by element division operation in matrix form as per MATLAB [15].

2.5 Calculation of nodal voltage matrix \([V_i]\)

Applying KVL directly to update the node voltage for the network shown in Figure 6 the voltage at \((k+1)\) node is equal to

\[ [V_{i+1}] = [V_i] - [I_i][ZD] \] \hspace{1cm} (9)

Equations (7), (8), and (9) are to be executed repeatedly until convergence is reached. The voltage mismatch at node can be expressed as

\[ [V^{n+1}] = [V^n] + [\Delta V^{n+1}] \] \hspace{1cm} (10)
## 2. COMPARATIVE ANALYSIS OF THE PROPOSED METHOD

The basic forward–backward techniques are analyzed as follows:

### Table 1. Comparison between methods [11] and [14]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Nodal currents: The current injection $I_i(k)$ is $I_i(k) = \frac{S_i(k)}{V(k)}$</td>
<td>Nodal currents: The current injection $I_i(k)$ is $I_i(k) = \frac{S_i(k)}{V(k)}$</td>
<td>Backward sweep: The power injection $S_i(k)$ is $[S_i] = [\text{NLBL}] [SD] + [BL]$ where $[\text{NLBL}]$ having 1’s and 0’s converts given nodal powers information into branch powers.</td>
</tr>
<tr>
<td>B</td>
<td>Backward sweep: Expression for branch currents $I_i^{(p)}(k + 1) = \sum_{k=1}^{\text{nodes}} I_i^{(p)}(k)$ where ‘In’ is nodal current</td>
<td>Bus-Injection to Branch-Current (BIBC): Branch currents in matrix form as $[\text{Branch currents}] = [\text{BIBC}] [\text{Nodal currents}]$ where $[\text{BIBC}]$ having 1’s and 0’s converts given nodal currents information into branch currents.</td>
<td>Injected branch-current matrix is the conjugate of the ratio of the injected powers to voltage $[I_i] = ([S_i]/[V_i])^*$ Where $[V_i]$ is the updated voltage</td>
</tr>
<tr>
<td>C</td>
<td>Forward sweep: Nodal voltages are computed in forward sweep as $V_i(k + 1) = V_i(k) - \sum_{k=1}^{\text{nodes}} Z(k + 1) I_i(k + 1)$</td>
<td>Branch-Current to Bus-Voltage (BCBV): Final voltage at bus using $[\text{BCBV}] [\text{BIBC}]$ and $[I_i]$ is $[V(k)] = [V(1)] - [\text{BCBV}] [\text{BIBC}] [I_i(k)]$</td>
<td>Nodal voltage $[V_i]$ matrix is $[V_i + 1] = [V_i] - [I_i][ZD]$</td>
</tr>
</tbody>
</table>

### Table 2. Performance for IEEE 15-node radial network

<table>
<thead>
<tr>
<th>Methods/Performance</th>
<th>Convergence</th>
<th>Iteration</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method [10]</td>
<td>0.0001</td>
<td>3</td>
<td>4KB</td>
</tr>
<tr>
<td>Method [11]</td>
<td>0.0001</td>
<td>3</td>
<td>3.70KB</td>
</tr>
<tr>
<td>Proposed method</td>
<td>0.0001</td>
<td>2</td>
<td>4.91KB</td>
</tr>
</tbody>
</table>


Table 3. Power flow solution for IEEE -15 bus network

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Magnitude</td>
<td>Angle</td>
<td>Magnitude</td>
</tr>
<tr>
<td>V1</td>
<td>1.0000</td>
<td>0.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>V2</td>
<td>0.9750</td>
<td>-0.0476</td>
<td>0.9750</td>
</tr>
<tr>
<td>V3</td>
<td>0.9743</td>
<td>0.0487</td>
<td>0.9743</td>
</tr>
<tr>
<td>V4</td>
<td>0.9624</td>
<td>0.0636</td>
<td>0.9624</td>
</tr>
<tr>
<td>V5</td>
<td>0.9601</td>
<td>-0.0704</td>
<td>0.9601</td>
</tr>
<tr>
<td>V6</td>
<td>0.9576</td>
<td>-0.0753</td>
<td>0.9577</td>
</tr>
<tr>
<td>V7</td>
<td>0.9559</td>
<td>0.0805</td>
<td>0.9560</td>
</tr>
<tr>
<td>V8</td>
<td>0.9551</td>
<td>-0.0830</td>
<td>0.9553</td>
</tr>
<tr>
<td>V9</td>
<td>0.9550</td>
<td>-0.0834</td>
<td>0.9552</td>
</tr>
<tr>
<td>V10</td>
<td>0.9610</td>
<td>-0.0662</td>
<td>0.9574</td>
</tr>
<tr>
<td>V11</td>
<td>0.9581</td>
<td>-0.0721</td>
<td>0.9456</td>
</tr>
<tr>
<td>V12</td>
<td>0.9723</td>
<td>-0.0547</td>
<td>0.9723</td>
</tr>
<tr>
<td>V13</td>
<td>0.9710</td>
<td>-0.0574</td>
<td>0.9710</td>
</tr>
<tr>
<td>V14</td>
<td>0.9694</td>
<td>-0.0606</td>
<td>0.9694</td>
</tr>
<tr>
<td>V15</td>
<td>0.9692</td>
<td>-0.0606</td>
<td>0.9692</td>
</tr>
</tbody>
</table>

4. RESULTS AND DISCUSSIONS
Using the network data in [16] the performance of the proposed algorithm is compared with the methods in [10] and [11]. However, for these data the NR and GS method do not converge. The performance results mentioned in Tables 2 and 3 were programmed in MATLAB Ver 7.0 software package installed in the PC having specification as: 512MB-RAM, Intel Pentium IV-Processor, 1.73GHz-Speed. The strength of the algorithm has been demonstrated by considering losses associated with branches in equation (7). The method is recommended based on the nodal voltage obtained from equation (10). Thus the proposed method has been found to be superior in accuracy, number of iterations and efficient as per the comparison and the results given in Tables 2 and 3.
5. CONCLUSION

A simple and powerful algorithm has been proposed for balanced radial distribution network to obtain power flow solution. It has been found from the cases presented that the proposed method has fast convergence characteristics when compared to existing methods. The algorithm is found to be robust in nature. The method can be easily extended to solve three phase networks also.

References
