A computationally efficient modeling method for large size water network analysis

Orazio Giustolisi, Daniele Laucelli, Luigi Berardi, Dragan A. Savić
Technical University of Bari, v. E. Orabona 4, 70125 Bari, Italy.
o.giustolisi@poliba.it, d.laucelli@poliba.it, l.berardi@poliba.it
University of Exeter, Centre for Water Systems, North Park Road, Exeter EX4 4QF, UK.
d.savic@ex.ac.uk

Abstract

Nowadays, unprecedented computing power of desktop personal computers and efficient computational methodologies, such as the global gradient algorithm (GGA), make large water distribution system modeling feasible. However, many network analysis applications, such as optimization models, require running numerous hydraulic simulations with modified input parameters. Therefore, a methodology that could reduce the computational burden of network analysis, and still provide the required model accuracy, is needed. This paper presents a matrix transformation approach to convert the classic GGA, which is implemented within the widely available freeware EPANET2 (Rossman, 2000), into a more computationally efficient enhanced global gradient algorithm (EGGA). The latter achieves the improved efficiency by reducing the size of the mathematical problem through the transformed topological representation of the original network model. By removing serial nodes and serial pipe-sections from the original topological representation, whilst preserving those elements both in energy and mass balance equations, EGGA provides a significant improvement to the model’s computational efficiency without forfeiting the hydraulic accuracy of the model. The computational efficiency and effectiveness of the EGGA approach is demonstrated on four examples of different real-life networks. Results show that the computational burden of the EGGA model is significantly lower than for its GGA counterpart particularly as the size of the network and/or number of service connections increases.

Introduction

A Water Distribution Network (WDN) may be viewed as a graph composed of edges and vertices, which correspond to pipes interconnected at nodes, respectively. Pipes, which are characterized by their length, diameter and hydraulic resistance, transfer water across the WDN and may contain other components and fittings, such as pumps, bends and valves. Pipes are interconnected through nodes in a specified topological configuration with all demands allocated at nodes (unless a node
represents a tank) for hydraulic modeling purposes. The flows and pressures in the WDN are
governed by a set of laws, the conservation of energy for each pipe and the conservation of mass for
each node (Savić and Banyard, 2011). The so-called steady-state network analysis problem is then
solved for a given set of boundary conditions (i.e., tank levels, nodal demands, pipe hydraulic
resistances, pump characteristics, minor losses, etc.) resulting in the pipe flows and nodal heads in
the WDN. The related mathematical problem is partly nonlinear (energy balance equations) and
partly linear (mass balance equations, provided that demands are fixed \textit{a priori}). The size of the
problem (i.e., the number of equations versus unknown states) is equal to the number of pipes
(edges in the topological representation) plus the number of nodes (vertices in the topological
representation).

The solution methodologies for the network analysis problem have been addressed by a number
of researchers in the past. After the local linearization method proposed by Cross (1936), the work
by Collins et al. (1978) gave rise to a number of global linearization techniques (i.e., characterized
by the simultaneous solution of all the network equations) (e.g. Martin and Peters, 1963; Shamir
and Howard, 1968; Epp and Fowler, 1970; Hamam and Brammeller, 1971; Kesavan and
Chandrashekar, 1972; Wood and Charles, 1972; Isaacs and Mills, 1980; Wood and Rayes, 1981;
Carpentier et al., 1987) culminating with the work by Todini and Pilati (1988). They developed the
Global Gradient Algorithm (GGA) that exhibits excellent convergence characteristics and is used in
the freely available EPANET2 software (Rossman, 2000). All these methods are iterative in nature
and solve a system of nonlinear equations whose size is a function of the topological representation
of the network.

As more and more water utilities adopt the approach of creating models from geographic
information systems, all-mains models containing tens or even hundreds of thousands of pipes and
nodes are being commonly built (Savić and Banyard, 2011). However, attempting to include each
individual component of a large system in a WDN model is not only an enormous undertaking by
itself, but could significantly impact on the computational efficiency of a model being used for real-
time control and management. Furthermore, optimization of WDNs, whether used for planning or
operational purposes, often requires many iterations to be performed, with each iteration involving
computationally expensive simulations. Therefore, a network hydraulic analysis methodology that
could reduce the computational burden, but still provide the required model accuracy is needed.

Giustolisi and Todini (2009) and Berardi et al. (2010) enhanced the GGA in order to account for
a uniformly distributed demand along pipes. Their work addressed the network calibration problem
by showing that the common approach of allowing demands to occur at network nodes only (even
though they are distributed along pipes in the real-life networks) might generate significant errors
due to problems with achieving energy conservation laws. Later on, Giustolisi (2010) and Giustolisi et al. (2011) further extended GGA and demonstrated that the new enhanced methodology (EGGA) can accommodate any pattern of connections along pipes. This also means that the one-to-one association of pipes and nodes to edges and vertices, respectively, is not maintained by EGGA as each edge corresponds to a series of interior pipe-sections and nodes. Thus, the size of the mathematical problem related to the WDN model becomes equal to the number of edges and vertices of the transformed topological representation which can be up to one order of magnitude lower in size than the original representation. For example, if the original WDN topology has \( n \) serial nodes (on interior pipe sections), the topological representation size can be reduced by \( n \) in terms of vertices and edges. Consequently, by removing serial nodes/pipe-sections from the topological representation of the original network model, EGGA can provide a significant improvement in computational efficiency whilst preserving all the hydraulic elements of the original energy and mass balance equations, thus the hydraulic accuracy of the model is not forfeited.

The new approach not only reduces the problem size but improves numerical stability of the solution procedure (Giustolisi, 2010) as the removal of serial nodes introduces a sort of regularization (Piller, 1985). This is particularly important for pipes with low flow rates that could cause numerical instability as the GGA solution requires the inversion of the matrix of the derivatives of pipe head loss functions (see the following section).

Consequently, EGGA is not a skeletonization approach with respect to serial nodes as, for example, in Walski et al. (2004; 2010) because the energy balance is not forfeited (Giustolisi, 2010). Rather, it is a comprehensive framework for WDN schematization with respect to serial nodes (Hamberg and Shamir, 1988). Once the nodes are eliminated from the topological representation of the model, each edge of the reduced model represents the original serial pipe sections. Thus, the original demands, hydraulic resistances, devices, are not removed from the model but are rather used to compute flow rates and head losses in the original pipe sections.

Thus, this paper presents a matrix transformation approach to convert the classic GGA into a more computationally efficient EGGA that could be easily implemented in any software utilizing the former approach. The following sections will introduce the matrix notation used, illustrate how the transformation can be easily implemented in the existing GGA-based code, and demonstrate the approach used by comparing the efficiency and accuracy of EGGA solutions to those obtained by GGA on large-size water distribution systems.
A WDN hydraulic model is based on solving the following two sets of equations, conservation of energy and conservation of mass at pipes and nodes, respectively:

\[
H_j - H_i + R_k Q_k|Q_k|^{\gamma - 1} + K_k^{ml} Q_k|Q_k| + \omega_k^2 r_k^p Q_k|Q_k|^{\gamma - 1} = \pm \alpha_k^2 H_k^p
\]

\[
\sum_i \pm Q_{is} = d_i
\]

(1)

where \(H_i\) and \(H_j\) = unknown nodal heads at the upstream (\(i\)th) and downstream (\(j\)th) terminal nodes of the \(k\)th pipe according to the assumed positive direction for the unknown flow rate \(Q_k\) from node \(i\) to \(j\); \(R_k\) = \(k\)th pipe hydraulic resistance; \(K_k^{ml}\) = minor loss (if any) along the \(k\)th pipe; \(\omega_k\) and \(r_k^p\), \(H_k^p\), \(\gamma_k\) = speed factor and the parameters of the pump system (if any) along the \(k\)th pipe; \(Q_{is}\) = flow rates of the \(s\) pipes joining at the \(i\)th node and \(d_i\) = fixed (i.e., known) demand at the \(i\)th node. The sign of \(Q_{is}\) depends on the assumed positive direction for the \(s\)th pipe with respect to flow entering the node. The sign of \(\omega_k\) depends on the installation direction of the pump with respect to the positive sign of the flow rate in the \(k\)th pipe.

Note that the known terms in (1) are moved to the right-hand side of the equations and the sign of \(H_k^p\) is related to the installation direction of the pump system.

If the \(i\)th node is a tank (i.e., the nodal head is assumed as known), the energy balance equation is modified by moving the known head \(H_0\) to the right-hand side as follows:

\[
H_j + R_k Q_k|Q_k|^{\gamma - 1} + K_k^{ml} Q_k|Q_k| + \omega_k^2 r_k^p Q_k|Q_k|^{\gamma - 1} = H_0 + \pm \alpha_k^2 H_k^p
\]

while the mass balance equation disappears because it is not possible to constrain both the head and demand at that node.

Consequently the model of a hydraulic network of \(n_p\) pipes, \(n_n\) demand nodes (i.e., internal nodes) and \(n_0\) tank nodes (i.e., known head levels) may be represented in a matrix form:

\[
\begin{align*}
\mathbf{A}_{p,p} Q_{p,1} + \mathbf{A}_{p,n} H_{n,1} &= -\mathbf{A}_{p,0} H_{0,1} \pm H_{p,1}^p \\
\mathbf{A}_{n,p} Q_{p,1} &= \mathbf{d}_{n,1}
\end{align*}
\]

(3)

where \(Q_{p,1} = [n_p, 1]\) column vector of unknown pipe flow rates; \(H_{n,1} = [n_n, 1]\) column vector of unknown nodal heads; \(H_{0,1} = [n_0, 1]\) column vector of known nodal heads; \(H_{p,1}^p = [n_p, 1]\) column vector of the static heads of pump systems; \(\mathbf{d}_{n,1} = [n_n, 1]\) column vector of demands lumped at nodes driving the simulation as they are fixed and assumed satisfied regardless of the pressure achieved at a node; \(\mathbf{A}_{p,n} = \mathbf{A}_{n,p}^T\) and \(\mathbf{A}_{p,0} = \) topological incidence sub-matrices obtained from the related representation of the hydraulic system whose size is \([n_p, n_n]\) and \([n_p, n_0]\), respectively. \(\mathbf{A}_{p,p} Q_{p,1}\) is the
column vector of pipe head losses containing the terms related to internal head loss of pump systems and minor losses together with uniformly distributed head losses as in (1).

The general GGA solution of system (3) can be obtained by solving iteratively the following equations (Todini and Pilati, 1988):

\[
\begin{align*}
B_{p,p}^{iter} &= \left( D_{p,p}^{iter} \right)^{-1} A_{p,p}^{iter} \\
F_{n,1}^{iter} &= A_{n,p}^{iter} \left( Q_{p,1}^{iter} - B_{p,p}^{iter} Q_{p,1}^{iter} \right) - A_{n,p}^{iter} \left( D_{p,p}^{iter} \right)^{-1} \left( A_{p,0}^{iter} H_{0,1}^{iter} m H_{p,1}^{iter} \right) - d_{n,1}^{iter} \\
H_{n,1}^{iter+1} &= \left( A_{n,p}^{iter} \left( D_{p,p}^{iter} \right)^{-1} A_{p,n}^{iter} \right)^{-1} F_{n,1}^{iter} \\
Q_{p,1}^{iter+1} &= \left( Q_{p,1}^{iter} - B_{p,p}^{iter} Q_{p,1}^{iter} \right) - \left( D_{p,p}^{iter} \right)^{-1} \left( A_{p,0}^{iter} H_{0,1}^{iter} m H_{p,1}^{iter} + A_{p,n}^{iter} H_{n,1}^{iter+1} \right)
\end{align*}
\]

where \( iter \) is a counter of the iterative solution algorithm, \( D_{p,p} \) is a diagonal matrix whose elements are the derivatives of the head loss function (including components such as minor losses, losses of pump systems and uniformly distributed losses) with respect to \( Q_{p,1} \).

The GGA is a nodal algorithm and requires the iterative solution of a linear symmetric system based on the sparse matrix \( (A_{n,p}^{iter} \left( D_{p,p}^{iter} \right)^{-1} A_{p,n}^{iter} \) having order \( n_n \) (i.e., the number of internal nodes equal to the number of vertices in the model’s topological representation).

From GGA to EGGA formulation

As explained in the introduction section, the EGGA reduces the size of the linear system in (4) by removing the number of serial nodes from the topological representation, ending up with a reduced number of nodes (further referred to as non-serial nodes, \( m_n \)). All the corresponding (serial) pipe segments are thus aggregated into a number of longer edges (\( m_p \)) in the EGGA topological representation. The reduced-size system results in a solution methodology that is computationally more efficient than GGA whilst preserving all the hydraulic elements determining the energy and mass balance of the original model. Consequently, there is no reduction in the hydraulic accuracy of EGGA with respect to the GGA model. It will be shown next how a transformation from the GGA to the EGGA approach can be achieved using two main transformation matrices.

The EGGA is formulated here without using the correction factor for pipe hydraulic resistance, which was adopted in Giustolisi (2010). The new formulation demonstrates the computational and numerical enhancements achieved by EGGA and emphasizes the differences between GGA and EGGA. In particular, it is shown the way the energy balance is preserved in the latter when serial nodes/pole-sections are removed from the network’s topological representation.
Assuming that a head loss monomial formulation with exponent \( n \) is used, and considering existing minor losses and pumps at each \( i \)th section of the \( k \)th pipe, the head loss in that pipe is calculated as follows (Giustolisi and Berardi; 2011):

\[
\Delta H_k = \sum_{i=1}^{m_{k,i} + 1} Q_{k,i} \left( R_{k,i} \left| Q_{k,i} \right|^{n-1} + K_{k,i}^{ml} \left| Q_{k,i} \right|^{n} + \omega_{k,i}^{2-n} r_{k,i} \left| Q_{k,i} \right|^{n-2} \right) + m \omega_{k,i}^{2-n} H_{k,i}^p
\]

(5)

where \( R_{k,i} \) = hydraulic resistances of the \( i \)th serial section; \( K_{k,i}^{ml} \) = coefficient of minor head losses of the \( i \)th serial section; \( \omega_{k,i}, H_{k,i}^p, r_{k,i}, \gamma_i \) = speed factor and three parameters of the curve of the pump installed in the \( i \)th serial pipe-section; \( m_k \) = number of serial nodes in the \( k \)th pipe. The new Figure 1 shows a schematic of the merging procedure.

**Figure 1.** A schematic representation of several pipes in series merged to create a single pipe

Consequently, EGGA can be obtained from GGA taking into account that \( H_{k,i}^p \) are known terms in the energy balance equations (i.e., can be put at the right-hand side as in the system (2)). The diagonal matrices \( A_{mp,mp}, D_{mp,mp} \) and \( B_{mp,mp} \), now of size \( m_p \), may be defined as follows:

\[
A_{mp,mp} (k,k) = \frac{1}{Q_k} \left[ \sum_{i=1}^{m_{k,i} + 1} Q_{k,i} \left( R_{k,i} \left| Q_{k,i} \right|^{n-1} + K_{k,i}^{ml} \left| Q_{k,i} \right|^{n} + \omega_{k,i}^{2-n} r_{k,i} \left| Q_{k,i} \right|^{n-2} \right) \right]
\]

\[
D_{mp,mp} (k,k) = \sum_{i=1}^{m_{k,i} + 1} n R_{k,i} \left| Q_{k,i} \right|^{n-1} + 2K_{k,i}^{ml} \left| Q_{k,i} \right|^{n} + \gamma_i \omega_{k,i}^{2-n} r_{k,i} \left| Q_{k,i} \right|^{n-2} \left( \sum_{i=1}^{m_{k,i} + 1} \frac{1}{p_{k,i}} \right)
\]

(6)

\[
B_{mp,mp} (k,k) = \frac{A_{mp,mp} (k,k)}{D_{mp,mp} (k,k)} = \frac{1}{Q_k} \sum_{i=1}^{m_{k,i} + 1} p_{k,i} h_{k,i} = \sum_{i=1}^{m_{k,i} + 1} y_{k,i}
\]

where \( h_{k,i}, p_{k,i} \) and \( y_{k,i} \) = scalar variables defined for the single pipe in the EPANET2 user manual (Rossman, 2000). The new subscripts in Eq. (6) refer to different elements from those in EPANET2 because the first is related to pipe \( k \) in EGGA and the second refers to the \( i \)th serial section contained in the \( k \)th pipe (while in the EPANET2 user manual \( h_{ij} \) and \( p_{ij} \) are the headloss and the inverse derivative of the headloss in the link between nodes \( i \) and \( j \)). The flows \( Q_{k,1} \) in the first pipe-section are related to WDN model flow rate \( Q_k \) and the total demand \( P_k \) at the serial nodes of the \( k \)th pipe as follows:

\[
Q_{k,i} = Q_k + \frac{P_k}{2}
\]

(7)

It is worth observing that such formulation assumes that the total demand of the interior nodes (\( P_k \)) is halved and allocated to the two terminal nodes of \( k \)th pipe, although a more general scheme of
demand lumping can be used as in Giustolisi (2010). Finally, it is also easy to demonstrate that EGGA reverts to the GGA representation without serial sections, i.e., for $m_k = 0$.

**Matrix transformations to enable EGGA formulation**

The starting point is the WDN model in (3), which is modified by means of the following matrices:

- $P_{mp,p} = \text{energy balance transformation matrix}$, whose size is $m_p$ (the number of EGGA pipes after aggregating the original serial sections into a single edge of its topological representation) by $p$ (total number of original pipes). Its entries in each row are either 0 or $\{1, -1\}$. Thus, the non-null values in each row are associated with identifiers of the serial-pipe sections of the original topology (GGA), which in EGGA compose the new pipes (rows) without serial nodes. The sign of non-null values indicate the direction (positive or negative) in the same way as in the original network representation.

- $N_{mp,n} = \text{mass balance transformation matrix}$, whose size is $m_p$ (the number of EGGA pipes after the aggregation operation) by $n$ (total number of original nodes). Its entries in each row are either 0 or 1. Thus, the non-null values in each row are associated with identifiers of the serial-nodes of the original topology (GGA). In other words, the EGGA model accounts for them via $N_{mp,n}$, but they are not present in the new topological representation.

This way, the energy balance equation in (3) is left multiplied by $P_{mp,p}$ matrix:

$$
P_{mp,p} A_{p,p} Q^{EGGA}_{mp,p} + P_{mp,p} A_{p,n} H_{n,i} = -P_{mp,p} A_{p,n} H_{0,i} \pm P_{mp,p} H_{p,1}$$

where

$$
Q^{EGGA}_{mp,1} = I_{mp,p} Q_{p,1} - \frac{1}{2} N_{mp,n} d_{n,1} = Q_{mp,1} - \frac{1}{2} N_{mp,n} d_{n,1}
$$

$Q^{EGGA}$ are the model flow rates in EGGA which are related to those of the first serial-pipe section ($I_{mp,p} Q_{p,1}$), as $I_{mp,p}$ is a matrix derived from the identity matrix of size $n_p$, by removing the rows corresponding to indices of the serial pipes (those excluded first) in the original topology. The second term on the right-hand side of Eq. (9) is the demand of the serial nodes (assuming a lumping coefficient equal to $\frac{1}{2}$), which is subtracted from $I_{mp,p} Q_{p,1}$ (i.e., flow of first pipe sections) as in Eq. (7). It is evident that for the pipes without serial nodes/links $Q_{mp,1}^{EGGA} = I_{mp,p} Q_{p,1}$ holds.

Consequently, the system (3) is modified to obtain:
\[ A_{mp,mp}^{EGGA} + E_{mn,mp}^{EGGA} H_{nn,1} = - E_{mp,0} H_{n,1} \pm P_{mp,p} H'_{p,1} \]

\[ E_{mn,mp}^{EGGA} = d_{m,1} + \frac{1}{2} |E_{mn,mp}| N_{mp,n} d_{n,1} \]  \hspace{1cm} (10)

where

\[ E_{mp,nn} = (P_{mp,p} A_{p,n}) \text{ with non zero columns} \]

\[ E_{mn,mp} = (E_{mp,nn})^T \]

\[ \text{diag}(A_{mp,1}) = \frac{P_{mp,p} A_{p,n} Q_{p,1}}{Q_{n,1}} \]  \hspace{1cm} (11)

In particular, \( E_{mp,nn} \) and \( E_{mp,0} \) = EGGA topological incidence sub-matrices of size \([m_p,m_n]\) and \([m_p,n_0]\), respectively, which are part of the general topological matrix \( \hat{E}_{mp,nn} = [E_{mp,nn} | E_{mp,0}] \) of size \([m_p,m_n+n_0]\). It is worth noting that the matrix product \( P_{mp,p} A_{p,n} \) provides a matrix of size \([m_p,n_n]\) whose columns contain all the elements equal to zero that are related to serial nodes. Those columns are eliminated to get \( E_{mp,nn} \). The vector \( d_{nn,1} \) contains demands at non-serial nodes in the original WDN topology.

**Modifications of the hydraulic solver**

In order to use the same solver for hydraulic steady-state computations of GGA also in EGGA, it is necessary to use \( E_{mp,nn}, E_{mn,mp} \) and \( E_{mp,0} \) instead of GGA topological matrices, \( A_{p,n}, A_{n,p} \) and \( A_{p,0} \), and to provide \( P_{mp,p} \) and \( N_{mp,n} \). Then, for each iteration of the solver \( Q_{mp,1}^{EGGA} \) are computed (as in Eq. 9) from the demand vector using the transformation matrices \( N_{mp,n} \) and \( |E_{mn,mp}|\):

\[ d^{\text{serial}}_{mn,1} = \frac{1}{2} |E_{mn,mp}| N_{mp,n} d_{n,1} \]  \hspace{1cm} (12)

where \( d^{\text{serial}}_{mn,1} \) = demand of serial nodes assigned at the pipe terminal nodes (non-serial nodes) by a lumping coefficient (e.g., \( \frac{1}{2} \), for further details on different lumping coefficient refer to Giustolisi, 2010). It is worth noting that the elements of \( d^{\text{serial}} \) are equal to zero for those pipes which do not contain serial nodes in the original network. The term \( N_{mp,n} d_{n,1} \) is the total demand of serial nodes and the related column contains all null values (see Eq. (A.8) in Appendix I).

Eq. (12) needs to be used just one time as the demands are fixed and do not change during iterations. At each iteration the diagonal elements of the matrices \( B_{mp,mp} \) and \( D_{mp,mp} \) (as in GGA \( B_{p,p} \) and \( D_{p,p} \)) are computed using \( P_{mp,p} \) and \( D_{p,1} \) (which is the column vector of the derivative of head losses along each original pipe section):
\[
\text{diag}(\mathbf{D}_{mp,1}) = \mathbf{P}_{mp,p} \mathbf{D}_{p,1} \\
\text{diag}(\mathbf{B}_{mp,1}) = \mathbf{P}_{mp,p} \mathbf{A}_{p,p} \mathbf{Q}_{p,1} - \mathbf{P}_{mp,p} \mathbf{D}_{p,1} \frac{1}{\mathbf{Q}_{mp,1}}
\]  \hspace{1cm} (13)

This way, the EGGA equations become:

\[
\mathbf{d}^{EGGA}_{mn,1} = \mathbf{d}_{mn,1} + \mathbf{d}^{serial}_{mn,1} \\
\mathbf{F}_{mn,1}^{iter} = \mathbf{E}_{mn,mp} \left( \mathbf{Q}_{mp,1}^{EGGA,iter} - \mathbf{B}_{mp,mp}^{EGGA,iter} \mathbf{Q}_{mp,1}^{EGGA,iter} \right) - \mathbf{E}_{mn,mp} \left( \mathbf{D}_{mp,mp}^{iter} \right)^{-1} \left( \mathbf{E}_{mn,0} \mathbf{H}_{0,1} m \mathbf{P}_{mp,p} \mathbf{H}_{p,1}^p \right) - \mathbf{d}^{EGGA}_{mn,1} \\
\mathbf{H}_{mn,1}^{iter+1} = \left( \mathbf{E}_{mn,mp} \left( \mathbf{D}_{mp,mp}^{iter} \right)^{-1} \mathbf{E}_{mn,mp} \right)^{-1} \mathbf{F}_{mn,1}^{iter} \\
\mathbf{Q}_{mp,1}^{EGGA,iter+1} = \left( \mathbf{Q}_{mp,1}^{EGGA,iter} - \mathbf{B}_{mp,mp}^{iter} \mathbf{Q}_{mp,1}^{EGGA,iter} \right) - \left( \mathbf{D}_{mp,mp}^{iter} \right)^{-1} \left( \mathbf{E}_{mn,0} \mathbf{H}_{0,1} m \mathbf{P}_{mp,p} \mathbf{H}_{p,1}^p + \mathbf{E}_{mn,0} \mathbf{H}_{mn,1}^{iter+1} \right) \hspace{1cm} (14)
\]

Finally, Eq. (7) or (9) are used to obtain \( \mathbf{I}_{mp,p} \mathbf{Q}_{p,1} \) of the first pipes from \( \mathbf{Q}_{mp,1}^{EGGA} \) at each iteration, while the other \( \mathbf{Q}_{k,i} \) can be efficiently computed using the internal mass balance equations and a proper indexing based on the matrix \( \mathbf{P}_{mp,p} \), because the serial demands are known. In a similar way the computation of the internal \( \mathbf{H}_{k,j} \) from \( \mathbf{Q}_{k,i} \) can be efficiently computed at the end of simulation as the internal head values are not needed during the iterative search. It is important to emphasize that the EGGA does not add significant computational burden when compared to GGA as the use of sparse matrix operations and indexing are very efficient in a standard developing environment, such as for example Matlab - MathWorks which has been used in this work.

Although EGGA has been presented here as a transformation of the GGA approach, as it is implemented in the most commonly used hydraulic software package EPANET2 (Rossman, 2000), the strategy can be used for any other algorithm. To aid the reader’s understanding of the methodology presented here, a detailed numerical example of the transformation process from GGA to EGGA has been given in Appendix I for a simple network in Figure A1.

Finally, it is worth noting that it is possible to aggregate \( h_{ij} \) and \( p_{ij} \) (the headloss and the inverse derivative of the headloss in the link between nodes \( i \) and \( j \), as given in the EPANET2 user manual) by consistently multiplying the matrix \( \mathbf{P}_{mp,p} \) by the column vectors obtained from \( h_{ij} \) and \( p_{ij} \) (see Eq. (6)).

**Generation of \( \mathbf{P}_{mp,p} \) and \( \mathbf{N}_{mp,n} \)**

This section describes an easy way to built the transformation matrices \( \mathbf{P}_{mp,p} \) and \( \mathbf{N}_{mp,n} \). The first step is to identify the interior/serial nodes. The task can be easily performed by analyzing the general topological matrix \( \mathbf{\bar{A}}_{pn} \). The identifier numbers of the serial nodes correspond the columns
having two non-zero values only because of the graph-theoretic properties of $\tilde{A}_{pn}$. Thus, the serial nodes are identified as those corresponding to the summation by columns of $|\tilde{A}_{pn}|$ equal to two. This identification is very quick as $\tilde{A}_{pn}$ is a sparse matrix.

Then, the serial nodes are traversed in sequence in order to identify the two joining pipes in the pipe table. By following the pipe order in the related table, the second found pipe is merged with the first. This way the original positive flow direction of the first pipe is preserved and becomes the positive direction of flow for the merged pipe. The flow directions of the subsequent pipes are stored as positive or negative according to their relationship to the positive direction of the merged pipe. Finally, in the table of pipes the first pipe becomes the merged pipe and the second pipe is marked as to be removed at the end of the process. Thus, the new terminal node of the first pipe is that of the second pipe which is not identified as serial.

The information about serial nodes and the identifiers of the serial pipes are stored sequentially in a separate matrix. This information can be then retrieved in the correct sequence to preserve the ordering of original pipe and node identifiers as merged pipes can contain more than two sections. In the Matlab environment used in the current implementation, the original pipe and node identifiers are stored as a cell column vector whose elements are two-row matrices of varying size. The number of columns of such a two-row matrix depends on the number of merged sections, which can be different among the resulting $m_p$ edges. Therefore, at the end of the process, the cell vector is sequentially read to generate $P_{mp,p}$ and $N_{mp,n}$ as each element contains the positions and sign of non-zero elements of the rows of the two matrices. It is important to note that the procedure is computationally efficient. For example, it takes less than two seconds for the largest original network used in the case studies (i.e., 14,814 pipes) to retrieve the information. Furthermore, $P_{mp,p}$ and $N_{mp,n}$ need to be built only once.

Finally, as EGGA preserves original accuracy of mass and energy balance of GGA, the internal flows of merged pipes are retrieved in order to compute correct head losses by Eq. (5) and EGGA matrices by Eq. (6). The flow rate of the first section of the merged pipes is first computed by:

$$Q_{mp,1} = Q_{EGGA}^{mp,1} + \frac{1}{2} N_{mp,n} d_{n,1}$$  \hspace{1cm} (15)

Simple sequential mass balance calculations along the merged pipes then allow computing internal flows $Q_{k,i}$ based on the original demands in the serial nodes. These flows are retrieved by a very quick indexing procedure using the position identifiers of non-zeros elements in each row of $P_{mp,p}$ and $N_{mp,n}$ and $d_{n,1}$.

Without using the EGGA methodology, the simulation problem reduces to lumping the demands of the serial nodes at the terminal nodes of merged pipes. That approach would preserve the mass
balance without accounting for the energy balance conservation, which is the case of the classical skeletonization procedure where Eq. (5) becomes:

\[
\Delta H_i = \left[ \sum_{i=1}^{m_i+1} R_{i,j} \right] Q_i |Q_i|^{-1} + \left[ \sum_{i=1}^{m_i+1} K_{i,j}^{ni} \right] Q_i |Q_i| + \left[ \sum_{i=1}^{m_i+1} \omega_j^{2r_j} r_{j,i} \right] Q_i |Q_i|^{-1} + \sum_{i=1}^{m_i+1} \omega_i Q_i H_{i,j}^p
\] (16)

Although this type of skeletonization procedure can be useful in many situations, it adds other computational uncertainties whose magnitude and relevance with respect to system analysis purposes is not known in advance. For example, when control devices and/or operating rules are present in the hydraulic model, the system behavior and the network dynamic response become sensitive to low changes in pressure heads. This in turn makes difficult preserving accuracy of the simulation model.

### Case studies

This section presents some numerical analyses to show the advantages of the EGGA modeling strategy over GGA on four real large-size networks. Due to page limits, relevant topologies and salient data (water demands, reservoirs, pumps, etc.) are not reported herein; the interested reader is referred to relevant published works:

- Apulian_2 (see Giustolisi and Laucelli, 2011);
- Exnet (see Giustolisi et al., 2008);
- BWSN (see Ostfeld et al., 2008);
- Town_C (Ostfeld et al., 2011)

The essential data describing original network topologies are reported in the first four columns of Tables 1 to 4. In particular, the first row of each table refers to the base configurations reported in original data and show, for example, that the smallest network (i.e., Town_C) comprises 396 nodes and 444 pipes, while the largest one (i.e. BWSN) comprises 12,515 nodes and 14,814 pipes. The third column (first row) also shows the number of serial nodes in the original network configurations.

As reported above, a large number of serial nodes/sections can strongly influence the computational efficiency of the hydraulic simulation model. To make this more pronounced and allow further comparison of the two computational approaches the original topologies of the four networks have been changed by adding a varying number of service connections along the original pipes, thus increasing the number of serial nodes and pipes in the networks. This procedure is aimed at emulating the availability of detailed data on service connections (e.g. as available from GIS). In particular, the fourth column of Tables 1 to 4 reports the number of service connections added per pipe, ranging from 1 to 10. The corresponding size of modified topologies in terms of the number of
pipes, nodes and serial nodes are reported in the first three columns of the same tables. As a result, 44 different network topologies have been analyzed whose size is up to 162,954 pipes and 160,655 nodes in the case of BWSN. Although evenly distributed connections represent a numerical abstraction, the random variation of nodal demands and pipe hydraulic resistances is equivalent to consider variable spacing between service connections when taking into account energy loss pattern.

It is known that convergence properties of the hydraulic simulation models can be affected by the boundary conditions (e.g., pipe hydraulic resistances and demands). Thus the proposed numerical tests have been performed by assuming 1,000 different network compositions in terms of pipe hydraulic resistances and demands in the service connections for each network topology (i.e., a total of 44,000 different networks).

In summary, such networks have been constructed as follows:

1. A number of service connections evenly distributed along the original pipes have been added, as detailed in the fourth column of Tables 1 to 4. The size of the resulting network topologies is reported in the first three columns of the same tables.

2. The unit hydraulic resistances of all pipes (in the new network topology) have been obtained by randomly multiplying the unit hydraulic resistance values of the original pipes by a factor in the range [0;10]. Thus, 1,000 unit hydraulic resistance samples for each pipe have been obtained using the Latin Hypercube sampling method (McKay et al., 1979).

3. Demands at service connection nodes are obtained by randomly subdividing the total demand, which was supplied by the original pipes, among the service connections. Also in this case, 1,000 demand scenarios are sampled for each network topology.

It is important to note that networks demands were only available at node level for Exnet, BWSN and Town_C. Thus, as a working hypothesis, the demand supplied along the original pipes was obtained by dividing the original nodal demand by the number of pipes joined in a node. Those demand values were then added together and divided among the end nodes for each pipe. Such a mathematically expedient has been adopted to preserve the total network demand and mass balance among pipes and nodes.

The performance of the EGGA model strategy is compared here with that of GGA. Tables 1 to 4 report a number of numerical indicators of EGGA and GGA simulation models in terms of average values over 1,000 simulation runs for each network topology. The number of iterations refers to the average value of index $\text{iter}$ of the core solver as reported in equations (4) and (14) for GGA and EGGA, respectively.
The columns labeled “CPU time Model” report the average duration of each simulation run. This duration is the CPU time required to run both the core solver and other calculations outside the solver, such as for example, initialization, matrix reordering and evaluation of the stopping criterion. Thus, columns named “unit CPU time Solver” and “total CPU time Solver” report the average duration of a single iteration and the average time taken to run the core solver (i.e., equations (4) for GGA and (14) for EGGA) as the product of the average number of iterations (# iterations) and the unit CPU time for a solver.

**Table 1.** EGGA vs. GGA computational performance: Apulian_2 network.

**Table 2.** EGGA vs. GGA computational performance: Exnet network.

**Table 3.** EGGA vs. GGA computational performance: BWSN network.

**Table 4.** EGGA vs. GGA computational performance: Town_C network.

Comparisons between EGGA and GGA in terms of the ratio between the respective average CPU times for the simulation as a whole and the average CPU run times for the core solver (for all the analyzed topologies) are plotted in Figures 2 and 3. The number of connections added per pipe, which is assumed to be an indicator of network complexity, is shown on the x-axis.

**Figure 2.** Total EGGA/GGA model CPU time ratios for all considered topologies.

**Figure 3.** Total EGGA/GGA solver CPU time ratios for all considered topologies.

All the tests were performed in the Matlab R2011a-64bit environment using a Notebook equipped with an Intel vPro i7 processor and Windows 7.0. The solution of the linear systems of equations (4) and (14) were performed using a direct method based on Cholesky factorization with the Approximate Minimum Degree Permutation being used as a reordering method. The simulation runs were performed using the accuracy based on the average error on mass and energy balance equations of system (3) or (10), equal to $10^{-7}$ (Giustolisi and Laucelli, 2011).
Discussion of results

Results reported in Tables 1 to 4 give rise to a number of observations about the effectiveness of using EGGA for large size networks; if compared to GGA they encompass both computational speed and algorithmic convergence aspects.

The most evident result is that the computational burden required for a single simulation by the EGGA model, in terms of average CPU time, is significantly lower than for its GGA counterpart as the number of service connections (added serial nodes) increases. Both Figures 2 and 3 show that for the large networks considered here, EGGA requires only about 10% of CPU time required by GGA, both in terms of the entire simulation and core solver.

The average number of iterations required by GGA and EGGA does not significantly change for three out of four case studies and remains roughly constant even when the number of service connections increases. The only exception is for the Town_C model, where the number of iterations required for convergence of GGA increases with connections. Such behavior is likely to be due to the combined effects of pumping stations with exponents of internal losses larger than 2 and the increased size of the model.

The CPU time taken for each iteration increases with the number of service connections for both models. Nonetheless, adding 10 connections per pipe, the unit CPU time required by the EGGA model increases about 3 times from that needed for the original configuration (i.e., without added connections). However, the unit CPU time used by GGA increases by 20 times on average. This confirms that the single iteration time required by the GGA solver is nearly linear with the size of the network (in terms of number of nodes). There are two opposite aspects which affect convergence of the EGGA model when multiple serial nodes are present. On the one hand, the condition number of the system of equations should not increase by adding serial nodes since the size of the linear problem does not change. On the other hand, the multiple summation of head loss terms in serial pipes (e.g., in Eqs. (6)) is likely to result in a larger condition number, especially when pipe hydraulic resistances are changed randomly as in the numerical examples here reported.

Figures 2 and 3 show that EGGA allows a significant reduction in CPU time even in the topologies without service connections added. Such a reduction is more evident as the number of original serial nodes increases. Moreover, when there are not many serial nodes (i.e., Apulian_2 network), the performance of EGGA is still comparable to that of GGA. However, for the largest networks considered, when only one service connection per pipe is added, the time taken by EGGA drops below 30% of that for GGA. The key point is that EGGA permits easy inclusion and treatment of service connection data within existing models as GIS based tools become more prevalent in hydraulic model construction.
From Figure 2 and 3, it is also evident that the computational efficiency achievable by EGGA is more pronounced as the size of the networks increases, regardless whether serial nodes are added or not. For example, the CPU time for the simulation of the original Exenet and BWSN networks is reduced on average to 68% and 63% of their GGA counterparts, respectively.

Figure 3 also shows that the improvements achieved by EGGA are mostly due to the time reduction for running the core solver, which shows the same trend of Figure 2 for all case studies.

It is also worth analyzing the CPU time taken for operations outside the core solver that involve initialization of variables and matrices, and computation of simulation performance indicators. These operations and run times depend on internal nodes in GGA (which increase with service connections) and non-serial nodes in EGGA (i.e. which remain unchanged). It is evident that such run times for GGA are always significantly larger than their EGGA counterparts in all analyzed cases.

**Numerical test for accuracy: EGGA VS. GGA**

As already discussed, the accuracy of EGGA and GGA is the same as they both solve the equivalent hydraulic problem in (3) and (10), although EGGA reduces the size of the linear system to be solved iteratively by the classical GGA approach. In this section a numerical test is performed in order to test the differences of heads and flows computed using EGGA and GGA for the same network configuration. To this purpose the Exnet and BWSN models have been as the largest ones. They have been simulated using the maximum accuracy allowed by the Matlab environment for computing the average error on mass and energy balances. In the case of Exnet the reached value of such a performance indicator was $10^{-25}$ and $10^{-23}$ for EGGA and GGA, respectively. In the case of BWSN the reached performance indicator was in both cases of the order of $10^{-18}$.

Figures 4 to 6 show the results for both networks in terms of nodal head and pipe flow differences between EGGA and GGA.

**Figure 4.** Head differences [m] between GGA and EGGA in Exnet for each node.

**Figure 5.** Flow differences [l/s] between GGA and EGGA in Exnet for each pipe.

**Figure 6.** Head differences [m] between GGA and EGGA in BWSN for each node.

**Figure 7.** Flow differences [l/s] between GGA and EGGA in BWSN for each pipe.

The average absolute values of the observed differences for nodal heads and flow rates are $[3.2 \times 10^{-9} \text{ m}, 8.1 \times 10^{-9} \text{ l/s}]$ in the Exnet case and $[4.7 \times 10^{-7} \text{ m}, 3.0 \times 10^{-6} \text{ l/s}]$ in the BWSN case. It can be observed that the differences are low and increase only slightly in the case of a larger network. This
confirms that they are related to numerical accuracy and not to the method implemented in EGGA. In fact, the main difference between EGGA and GGA resides in the dimension of the space of the linear system to be iteratively solved. It is worth noting that by decreasing this space in moving from the GGA to EGGA algorithm, the linear problem becomes better conditioned. This allows using a direct solution method with better accuracy. In fact, the accuracy for both algorithms is driven by the accuracy in solving the linear system and, consequently, depends on the condition number of such a problem.

**Conclusions**

This study presents a matrix-based transformation methodology needed to improve the numerical and computational behavior of the General Gradient Algorithm (GGA) for solving large water distribution models. The Enhanced GGA algorithm allows simplified topological representation of the simulation model whilst preserving the hydraulic accuracy, thus achieving superior computational performance when compared to GGA. This has been shown on an example of four large real-life networks with or without additional service connections included. In the most extreme case (i.e., with 10 service connections added to each pipe) the EGGA run times have been reduced to approximately 10% of those for GGA. This means that for a network of about 160,000 nodes the EGGA run is completed in 0.77 sec whereas the GGA run takes 6.74 sec. However, this improved computational efficiency does not come at the expense of the solution accuracy as EGGA preserves the energy and continuity mass balance equations.

The step-by-step derivation of the EGGA methodology from the GGA-based set of matrix equations has also been presented. A numerical example for a simple water distribution network has also been given in the Appendix I. Thus the new methodology can be easily implemented for any software code based on the GGA (or any other solution) algorithm.

**Appendix I**

An example of the transformation from GGA to EGGA is here reported. It is applied to the simple network in Figure A1.

**Figure A1.** Example network

Considering the network in Figure A1, the vectors of unknowns and boundary conditions of the simulation model problem are:
and the matrices of the model system in (2) are:

\[
A_{p,p} = \begin{bmatrix}
R_1 |Q|^{-1} & 0 & \cdots & 0 & 0 \\
0 & R_2 |Q|^{-1} & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & R_s |Q|^{-1} & 0 \\
0 & 0 & \cdots & 0 & R_s |Q|^{-1} \\
\end{bmatrix}
\]

\[
A_{p,s} = \left[ A_{p,p} \ M \ A_{p,0} \right] = \begin{bmatrix}
-1 & +1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & +1 & 0 & 0 & 0 & -1 \\
0 & 0 & -1 & 0 & 0 & +1 & 0 \\
0 & 0 & 0 & +1 & -1 & 0 & 0 \\
-1 & 0 & 0 & +1 & 0 & 0 & 0 \\
0 & +1 & 0 & 0 & 0 & 0 & -1 \\
1 & 0 & 0 & 0 & 0 & 0 & -1 \\
\end{bmatrix}
\]

Therefore the model system of equations in (2) is,

\[
\begin{bmatrix}
R_1 |Q|^{-1} & 0 & \cdots & 0 & 0 \\
0 & R_2 |Q|^{-1} & \cdots & 0 & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & \cdots & R_s |Q|^{-1} & 0 \\
0 & 0 & \cdots & 0 & R_s |Q|^{-1} \\
\end{bmatrix}
\begin{bmatrix}
Q_1 \\
Q_2 \\
\vdots \\
Q_s \\
\end{bmatrix}
= \begin{bmatrix}
-1 & +1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & +1 \\
0 & 0 & +1 & -1 & 0 & 0 \\
-1 & 0 & 0 & +1 & 0 & 0 \\
0 & +1 & 0 & 0 & 0 & -1 \\
1 & 0 & 0 & 0 & 0 & -1 \\
\end{bmatrix}
\begin{bmatrix}
H_1 \\
H_2 \\
\vdots \\
H_s \\
\end{bmatrix}
\]

\[
= \begin{bmatrix}
0 & 0 & 0 \\
+1 & 0 & 0 \\
-1 & +1 & 0 \\
0 & -1 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & -1 \\
\end{bmatrix}
\]

The transformation matrices from GGA to EGGA are,
\[
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\quad \quad
\begin{pmatrix}
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]  
(A.4)

Consequently,

\[
P_{ap,p}A_{pp}Q_{p,i} =
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
R_i|Q|^{-1} & 0 & ... & 0 & 0 \\
0 & R_i|Q|^{-1} & ... & 0 & 0 \\
... & ... & ... & ... & ... \\
0 & 0 & ... & R_i|Q|^{-1} & 0 \\
0 & 0 & ... & 0 & R_i|Q|^{-1}
\end{pmatrix}
\begin{pmatrix}
Q_i \\
Q_i \\
Q_i \\
Q_i \\
Q_i \\
Q_i \\
Q_i \\
Q_i \\
\end{pmatrix}
\]
(A.5)

\[
P_{mp,p}A_{pp}Q_{p,1}
\]
represents the head losses in the pipes of EGGA as arithmetical sum of those of the serial sections composing it.

In addition, the nodal head differences of EGGA are

\[
P_{ap,p}A_{p,p}H_{x,1} =
\begin{pmatrix}
1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
-1 & +1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & +1 & 0 \\
0 & 0 & +1 & -1 & 0
\end{pmatrix}
\begin{pmatrix}
H_i \\
H_i \\
H_i \\
H_i \\
\end{pmatrix}
_{x=5,1}
\]
(A.6)

\[
\begin{pmatrix}
-1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
H_i \\
H_i \\
H_i \\
H_i \\
\end{pmatrix}
_{ap=4,x=5}
\Rightarrow
\begin{pmatrix}
H_i \\
H_i \\
H_i \\
H_i \\
\end{pmatrix}
_{ap=4,x=5} = \begin{pmatrix}
-1 & 0 \\
0 & 1 \\
1 & 1 & 1 \\
\end{pmatrix}
_{ap=4,x=5}
\]

which are the differences of heads at the terminal nodes of non serial nodes.

As for the computation of \(Q_{mp,1}^{EGGA}\),
\[ Q^\text{EGGA}_{\text{mp},1} = I_{\text{mp},p} Q_{p,1} - \frac{1}{2} N_{\text{mp},d_{s,1}} \]

where \( I_{\text{mp},p} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}_{/\text{mp} \rightarrow \text{p},\text{p}}. \]

\[ Q^\text{EGGA}_{\text{mp},1} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}_{/\text{mp} \rightarrow \text{p},\text{p}} \]

while the right side term of the mass balance equation can be obtained by

\[ d_{mn,1} + \frac{1}{2} E_{mn,mp} N_{mp,d_{s,1}} \]

\[ (d_{i})_{mn,1} + \frac{1}{2} \begin{pmatrix}
0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 0
\end{pmatrix}_{/\text{mp} \rightarrow \text{p},\text{p}} \begin{pmatrix}
d_1 \\
d_2 \\
d_4 \\
d_5
\end{pmatrix}_{/\text{p},\text{s},\text{i}} = d_i + \frac{d_a + d_w + d_a + d_w}{2} \]

(A.8)

Notations

The following symbols are the most important:

- \( A_{p,n}, A_{n,p}, A_{p,0} \) = topological incidence sub-matrices in GGA;
- \( A_{p,p} \) = diagonal matrix in GGA;
- \( A_{mp,mp} \) = diagonal matrix in EGGA;
- \( B_{p,p} \) = diagonal matrix used in GGA;
- \( B_{mp,mp} \) = diagonal matrix used in EGGA;
- \( d_{n,1} \) = vector of nodal demands in GGA;
- \( d_{mn,1} \) = demands at non-serial nodes in the original WDN topology;
- \( d_{mn,1}^{\text{serial}} \) = demand of serial nodes in EGGA;
- \( d_{mn,1}^{\text{EGGA}} \) = demand vector in EGGA;
- \( D_{p,p} \) = diagonal matrix of derivative of head losses with respect to \( Q_{p,1} \);
- \( D_{mp,mp} \) = diagonal matrix of EGGA derived from \( D_{p,p} \);
- \( E_{mp,mp} \) = general topological matrix in EGGA;
- \( E_{mn,mp}, E_{mn,mp}, E_{mn,mp,0} \) = topological incidence sub-matrices in EGGA;
- \( F_{n,1} \) = temporary matrix used in GGA;
- \( F_{mn,1} \) = temporary matrix used in EGGA;
\( H_{k,j} \) = pipe hydraulic resistances of the \( j \)th serial nodes;  
\( H_{n,1} \) = vector of total network heads;  
\( H_{0,1} \) = vector tank head levels;  
\( K_{k,i}^{mi} \) = coefficient of minor head losses of the \( i \)th pipe-section;  
\( m_k \) = number of serial nodes in the \( k \)th pipe;  
\( m_n \) = number of nodes in EGGA;  
\( m_p \) = number of pipes in EGGA;  
\( N_{mp,n} \) = mass balance transformation matrix;  
\( P_{mp,p} \) = energy balance transformation matrix;  
\( R_{k,i} \) = pipe hydraulic resistances of the \( i \)th pipe-section;  
\( Q_{p,1}^{EGGA} \) = vector of pipe flow rates in EGGA;  
\( Q_{p,1} \) = vector of pipe flow rates in GGA;  
\( \omega_{k,i}, H_{k,i}^{p}, r_{k,i}, \gamma_{i} \) = speed factor and parameters of the pump curve of the \( i \)th pipe-section;  

Operators and Acronyms:  
EGGA = Enhanced Global Gradient Algorithm;  
GGA = Global Gradient Algorithm;  
WDN = Water Distribution Network.

References


LIST OF TABLES

Table 1. EGGA vs. GGA computational performance comparison: Apulian_2 network.
Table 2. EGGA vs. GGA computational performance comparison: Exnet network.
Table 3. EGGA vs. GGA computational performance comparison: BWSN network.
Table 4. EGGA vs. GGA computational performance comparison: Town_C network.
LIST OF FIGURES

Figure 1. A schematic representation of several pipes in series merged to create a single pipe.

Figure 2. Total EGGA/GGA model CPU time ratios for all considered topologies.

Figure 3. Total EGGA/GGA solver CPU time ratios for all considered topologies.

Figure 4. Head differences [m] between GGA and EGGA in Exnet for each node.

Figure 5. Flow differences [l/s] between GGA and EGGA in Exnet for each pipe.

Figure 6. Head differences [m] between GGA and EGGA in BWSN for each node.

Figure 7. Flow differences [l/s] between GGA and EGGA in BWSN for each pipe.

Figure A1. Example network.
<table>
<thead>
<tr>
<th>Apulian_2 topologies</th>
<th># iterations</th>
<th>CPU time Model [sec]</th>
<th>unit CPU time Solver [sec]</th>
<th>total CPU time Solver [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EGGA</td>
<td>GGA</td>
<td>EGGA</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1092</td>
<td>919</td>
<td>17</td>
<td>0</td>
<td>5.536</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.017262</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.002413</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.013361</td>
</tr>
<tr>
<td>2184</td>
<td>2011</td>
<td>1109</td>
<td>1</td>
<td>5.345</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.022767</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.003122</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.016687</td>
</tr>
<tr>
<td>3276</td>
<td>3103</td>
<td>2201</td>
<td>2</td>
<td>5.333</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.025131</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.003336</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.017788</td>
</tr>
<tr>
<td>4368</td>
<td>4195</td>
<td>3293</td>
<td>3</td>
<td>5.324</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.029129</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.003732</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.019868</td>
</tr>
<tr>
<td>5460</td>
<td>5287</td>
<td>4385</td>
<td>4</td>
<td>5.302</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.031742</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.003967</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.021035</td>
</tr>
<tr>
<td>6552</td>
<td>6379</td>
<td>5477</td>
<td>5</td>
<td>5.348</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.035051</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.004248</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.022720</td>
</tr>
<tr>
<td>7644</td>
<td>7471</td>
<td>6569</td>
<td>6</td>
<td>5.34</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.038334</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.004589</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.024505</td>
</tr>
<tr>
<td>8736</td>
<td>8563</td>
<td>7661</td>
<td>7</td>
<td>5.308</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.041827</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.004964</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.026347</td>
</tr>
<tr>
<td>9828</td>
<td>9655</td>
<td>8753</td>
<td>8</td>
<td>5.294</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.044131</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.005189</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.027469</td>
</tr>
<tr>
<td>10920</td>
<td>10747</td>
<td>9845</td>
<td>9</td>
<td>5.31</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.047027</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.005450</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.028942</td>
</tr>
<tr>
<td>12012</td>
<td>11839</td>
<td>10937</td>
<td>10</td>
<td>5.303</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.050728</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.005828</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.030904</td>
</tr>
<tr>
<td># pipes</td>
<td># nodes</td>
<td># serial nodes</td>
<td>service connections</td>
<td>Exnet topologies</td>
</tr>
<tr>
<td>---------</td>
<td>---------</td>
<td>----------------</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>1991</td>
<td>1466</td>
<td>678</td>
<td>0</td>
<td>Exnet</td>
</tr>
<tr>
<td>3982</td>
<td>3457</td>
<td>2669</td>
<td>1</td>
<td>Exnet</td>
</tr>
<tr>
<td>5973</td>
<td>5448</td>
<td>4660</td>
<td>2</td>
<td>Exnet</td>
</tr>
<tr>
<td>7964</td>
<td>7439</td>
<td>6651</td>
<td>3</td>
<td>Exnet</td>
</tr>
<tr>
<td>9955</td>
<td>9430</td>
<td>8642</td>
<td>4</td>
<td>Exnet</td>
</tr>
<tr>
<td>11946</td>
<td>11421</td>
<td>10633</td>
<td>5</td>
<td>Exnet</td>
</tr>
<tr>
<td>13937</td>
<td>13412</td>
<td>12624</td>
<td>6</td>
<td>Exnet</td>
</tr>
<tr>
<td>15928</td>
<td>15403</td>
<td>14615</td>
<td>7</td>
<td>Exnet</td>
</tr>
<tr>
<td>17919</td>
<td>17394</td>
<td>16606</td>
<td>8</td>
<td>Exnet</td>
</tr>
<tr>
<td>19910</td>
<td>19385</td>
<td>18597</td>
<td>9</td>
<td>Exnet</td>
</tr>
<tr>
<td>21901</td>
<td>21376</td>
<td>20588</td>
<td>10</td>
<td>Exnet</td>
</tr>
</tbody>
</table>
Table 3. EGGA vs. GGA computational performance comparison: BWSN network.

<table>
<thead>
<tr>
<th>BWSN topologies</th>
<th># iterations</th>
<th>CPU time Model [sec]</th>
<th>unit CPU time Solver [sec]</th>
<th>total CPU time Solver [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># pipes</td>
<td># nodes</td>
<td># serial nodes</td>
<td>service connections</td>
</tr>
<tr>
<td>14814</td>
<td>12515</td>
<td>4980</td>
<td>0</td>
<td>7.143</td>
</tr>
<tr>
<td>29628</td>
<td>27329</td>
<td>19794</td>
<td>1</td>
<td>6.887</td>
</tr>
<tr>
<td>44442</td>
<td>42143</td>
<td>34608</td>
<td>2</td>
<td>6.871</td>
</tr>
<tr>
<td>59256</td>
<td>56957</td>
<td>49422</td>
<td>3</td>
<td>6.834</td>
</tr>
<tr>
<td>74070</td>
<td>71771</td>
<td>64236</td>
<td>4</td>
<td>6.806</td>
</tr>
<tr>
<td>88884</td>
<td>86585</td>
<td>79050</td>
<td>5</td>
<td>6.778</td>
</tr>
<tr>
<td>103698</td>
<td>101399</td>
<td>93864</td>
<td>6</td>
<td>6.78</td>
</tr>
<tr>
<td>118512</td>
<td>116213</td>
<td>108678</td>
<td>7</td>
<td>6.781</td>
</tr>
<tr>
<td>133326</td>
<td>131027</td>
<td>123492</td>
<td>8</td>
<td>6.792</td>
</tr>
<tr>
<td>148140</td>
<td>145841</td>
<td>138306</td>
<td>9</td>
<td>6.757</td>
</tr>
<tr>
<td>162954</td>
<td>160655</td>
<td>153120</td>
<td>10</td>
<td>6.710</td>
</tr>
</tbody>
</table>
### Table 4. EGGA vs. GGA computational performance comparison: Town_C network.

<table>
<thead>
<tr>
<th>Town_C topologies</th>
<th># iterations</th>
<th>CPU time Model [sec]</th>
<th>unit CPU time Solver [sec]</th>
<th>total CPU time Solver [sec]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># pipes</td>
<td># nodes</td>
<td># serial nodes</td>
<td>service connections</td>
</tr>
<tr>
<td>444</td>
<td>396</td>
<td>143</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>888</td>
<td>840</td>
<td>587</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1332</td>
<td>1284</td>
<td>1031</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>1776</td>
<td>1728</td>
<td>1475</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2220</td>
<td>2172</td>
<td>1919</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>2664</td>
<td>2616</td>
<td>2363</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3108</td>
<td>3060</td>
<td>2807</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>3552</td>
<td>3504</td>
<td>3251</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>3996</td>
<td>3948</td>
<td>3695</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>4440</td>
<td>4392</td>
<td>4139</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>4884</td>
<td>4836</td>
<td>4583</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>
Figure 1. A schematic representation of several pipes in series merged to create a single pipe.

\[ d_A = q_A + \frac{1}{2} P_k \]

\[ d_B = q_B + \frac{1}{2} P_k \]

\[ P_k = \sum_{j=1}^{m_k} d_{k,j} \]
Figure 2. Total EGGA/GGA model CPU time ratios for all considered topologies.
Figure 3. Total EGGA/GGA solve CPU time ratios for all considered topologies.
Figure 4. Head differences [m] between GGA and EGGA in Exnet for each node.
Figure 5. Flow differences [l/s] between GGA and EGGA in Exnet for each pipe.
Figure 6. Head differences [m] between GGA and EGGA in BWSN for each node.
Figure 7. Flow differences [l/s] between GGA and EGGA in BWSN for each pipe.
Figure A1. Example network.