Independent Loops Search in Flow Networks Aiming for Well-Conditioned System of Equations

Jukka-Pekka Humaloja, Simo Ali-Löytty, Timo Hämäläinen and Seppo Pohjolainen

Abstract We approach the problem of choosing linearly independent loops in a pipeflow network as choosing the best-conditioned submatrix of a given larger matrix. We present some existing results of graph theory and submatrix selection problems, based on which we construct three heuristic algorithms for choosing the loops. The heuristics are tested on two pipeflow networks that differ significantly on the distribution of pipes and nodes in the network.

1 Introduction

In boiler design it is essential that we can estimate the flow rates in each of the pipes related to the boiler. By estimating the flow rates the pipes can be made sufficiently durable so that they withstand the flow, while the pipes should not be made needlessly durable so that the material costs can be held at a reasonable level. Thus, it is possible to deliver durable boilers at a competitive prize to the potential customers.

Jukka-Pekka Humaloja, e-mail: jukka-pekka.humaloja@tut.fi
Simo Ali-Löytty, e-mail: simo.ali-loytty@tut.fi
Timo Hämäläinen, e-mail: timo.t.hamalainen@tut.fi
Seppo Pohjolainen, e-mail: seppo.pohjolainen@tut.fi

Address for the authors
Tampere University of Technology, Department of Mathematics, P.O. Box 553, FIN-33101 Tampere, Finland

Acknowledgements This work was carried out in the research program Flexible Energy Systems (FLEXe) and supported by Tekes - the Finnish Funding Agency for Innovation. The aim of FLEXe is to create novel technological and business concepts enhancing the radial transition from the current energy systems towards sustainable systems. FLEXe consortium consists of 17 industrial partners and 10 research organizations. The program is coordinated by CLIC Innovation Ltd. www.clicinnovation.fi.

A very simple example of a boiler modeled as a pipeflow network is given in Fig. 1. There the network consists of $m = 12$ pipes and $n = 6$ nodes, and the heat surface is modeled by the seven adjacent pipes between the nodes 3 and 4. The behavior of the flow in such a network can be depicted by two simple principles - the first one being the continuity of flow on each node, i.e., flow in equals flow out, and the second one being that the pressure loss over any closed circuit has to be zero.

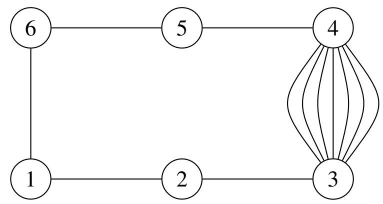

Fig. 1 A very simple example of a boiler modeled as a pipeflow network. In this case, the model consists of 12 pipes corresponding the 12 edges and 6 nodes corresponding the 6 vertices. The heating surface is modeled by the seven adjacent pipes between nodes 3 and 4

We obtain $n-1$ linearly independent linear equations from the continuity of flow, and therefore we need $m-n+1$ equations based on the pressure losses (see [4] for details on obtaining the equations). Even though the pressure loss equations are nonlinear with respect to the flow rates, we want to have as many equations as we have unknown variables so that we can solve the linearized version of the equations. As a whole the flow rates are solved iteratively by subsequently solving the linearized version of the equations by following the procedure depicted in [4]. In order to improve the convergence and numerical stability of the iterative procedure, we will present heuristics for improving the condition number of the set of linearized equations.

The structure of this paper is as follows. In Sect. 2 we consider the problem from a theoretical standpoint and present some existing results as a basis for our heuristics. The heuristics for choosing the pressure loss loops are presented in Sect. 3 where we also consider the effect of scaling on the condition number. In Sect. 4 we test our heuristics in two different pipeflow networks, and in Sect. 5 we conclude our work.

Throughout this paper we denote the number of pipes in a network by m and the number of nodes by n. The singular values of a matrix are denoted by σ_i and the condition number of the matrix is determined by the quotient of its largest and smallest singular values. We denote the inner product of two vectors x and y by $\langle x, y \rangle$, and the Euclidean (2-)norm $||\cdot||$ is induced by the inner product, i.e., $||x||^2 = \langle x, x \rangle$. The maximum ($\infty$-)norm is denoted by $||\cdot||_\infty$.

2 Theoretical Background

In this section we will consider some required background to understand our approach to the problem of choosing linearly independent pressure loss loops. The section is divided into two subsections considering graph theory and the problem of choosing the best-conditioned submatrix. As the reference of graph theory we use the recent book by Santamu [2], and in the submatrix selection problem our reference is the article by Šrámek et al. [3].

2.1 Concepts of Graph Theory

In this subsection we will review some required background to graph theory in order to present the concept of fundamental cycles. First of all, we call the minimum set of edges that connects all the vertices a *spanning tree*. There are $n-1$ edges in a spanning tree, and the remaining $m-n+1$ edges form the *cospanning tree*. When an edge from the cospanning tree is added to the spanning tree, a cycle is formed. By separately adding the vertices of the cospanning tree to the spanning tree we obtain a cycle for each of the vertices, and together these cycles form the set of *fundamental cycles*. As each fundamental cycle contains at least one unique edge, the set of fundamental cycles is independent under the ring sum operation. Similarly the set of vectors of $\mathbb{R}^m$ corresponding to the fundamental cycles are linearly independent in the usual sense. Note also that as the spanning tree is not unique, the set of fundamental cycles is not unique either.

Based on the above we can choose the pressure loss loops such that they form a set of fundamental cycles, which will make them linearly independent. However, we do not know which set of fundamental cycles is the most linearly independent. In our approach we aim to choose the most linearly independent cycles from the set of an excessive number of cycles by utilizing the ideas presented in the next subsection.

2.2 Choosing the Best-Conditioned Submatrix

Since we aim to choose the most linearly independent set of cycles from a larger set, our problem is similar to choosing the best-conditioned submatrix of a larger matrix. Assume that we have an $M \times m$ ($M > m$) matrix of which we aim to find the best-conditioned $m \times m$ submatrix. The problem is in fact shown to be NP-hard in [3], but the authors have also presented a heuristic to find a well-conditioned submatrix in (Mm^2) time. The heuristic is based on finding such a submatrix A whose rows are as orthogonal as possible. Thus, the heuristic tends to maximize $\mathrm{vol}(A) = \Pi_{i=1}^{m} \sigma_i$ whereas the condition number of A is defined as $\mathrm{cond}(A) = \sigma_{\max}/\sigma_{\min}$, and therefore the heuristic usually provides suboptimal solutions.

Consider the problem of finding a well-conditioned $m \times m$ submatrix B of an $M \times m$ matrix A, where $M > m$. The heuristic presented in [3] repeats the following steps m times:

1. On the i:th step, find the row a_j of A that has the largest norm. Choose the row as the i:th row of B, i.e., $b_i = a_j$.

2. For each row of A, Subtract its projection to b_i from itself, i.e., update the remaining rows a_k by $a_k = a_k - \langle a_k, b_i \rangle ||b_i||^{-2} b_i$.

Thus, on each step, the heuristic finds the row of A that has the largest orthogonal component with respect to the previously chosen rows. Therefore the rows of the submatrix B should eventually be rather orthogonal, and B should be fairly well-conditioned. We will be using this algorithm as a basis for our own heuristics for choosing the pressure loss loops.

3 Heuristics for Improving the Condition Number

In this section we consider three heuristics for choosing the pressure loss loops based on the previously presented algorithm form [3]. Before presenting the heuristics in Subsect. 3.2 we will consider improving the condition number by scaling the equations.

3.1 Scaling the Equations

Consider the linearized set of equations for the flow rates in the form $Ax = b$. The $m - n + 1$ rows of A corresponding to the pressure loss equations contain elements of the scale $10^3 - 10^5$, whereas the $n - 1$ continuity of flow equations consist of elements $-1, 0$ and 1. Thus, the matrix A may be badly scaled, and therefore we consider simple row scaling to improve its condition number.

A heuristic given in [1] suggests that the scaling matrix S should be chosen such that every row of SA has approximately the same ∞-norm. Thus, by choosing $S = \mathrm{diag}(||a_1||_\infty^{-1}, ||a_2||_\infty^{-1}, \ldots, ||a_m||_\infty^{-1})$ the ∞-norm of each row of SA is 1. As a comparison we will also consider normalizing the rows of A with respect to the Euclidean (2-)norm. The results of scaling in two different cases are presented in Table 1. The two cases are considered in more detail in Sect. 4.

Table 1 The effect of scaling on the condition number. The numbers represent the mean condition number and its standard deviation for 100 matrices generated from the same network. The second and third rows display the effect of normalization with respect to ∞ and 2-norms, and the first row shows the condition number of the matrix without scaling for reference.

	Case 1 $m = 618, n = 235$	Case 2 $m = 953, n = 78$
I	$(9.4 \pm 1.8) \cdot 10^5$	$(13.4 \pm 2.2) \cdot 10^5$
S_∞	$(15.0 \pm 3.0) \cdot 10^3$	$(15.6 \pm 4.2) \cdot 10^3$
S_2	$(12.2 \pm 2.7) \cdot 10^3$	$(7.7 \pm 2.4) \cdot 10^3$

The table shows that normalization with respect to both ∞- and 2-norms improve the condition number by a factor of 100. Furthermore, the table indicates that 2-norm actually provides better condition numbers than ∞-norm, which is somewhat unexpected based on the above heuristic.

3.2 Choosing the Pressure Loss Loops

In this subsection we present three heuristics for choosing the pressure loss loops. All the heuristics utilize the submatrix selection algorithm presented in [3]. In each of the heuristics we assume that we have generated an excessive number of loops and have a matrix A the rows of which correspond to these loops. Our task is to find the $m-n+1$ rows of A that together with the $n-1$ rows obtained from the continuity of flow equations result in the best-conditioned $m \times m$ matrix.

Heuristic 1: The first heuristic directly utilizes the algorithm presented in [3]. Denote the $n-1 \times m$ matrix corresponding to the continuity of flow equations by C. We begin by subtracting the projection of A to C from A, i.e., we update A by $A = A - (C^T(CC^T)^{-1}CA^T)^T$, which makes the rows of A orthogonal to the rows of C. Now we simply use the algorithm of [3] to choose the $m-n+1 \times m$ submatrix of A, which yields the choice of the pressure loss loops.

Heuristic 2: In the second heuristic we assume that we have an initial $m \times m$ matrix A_0 that can be created by choosing a set of fundamental circuits as the pressure loss loops. After the initialization, the heuristic is described as follows.

1. Find the row index j of A_0 that satisfies
$$j = \max_{n \le j \le m} \sum_{k=1, k\neq j}^{m} |\langle a_{0j}||a_{0j}||^{-1}, a_{0k}||a_{0k}||^{-1}\rangle|$$
and remove the row a_{0j} from A_0. Denote $\tilde{A}_0 = A_0 \setminus a_{0j}$.
2. Compute the norm of the orthogonal component of a_{0j} to the rows of $\tilde{A}_0$, i.e., $r = ||a_{0j}^T - \tilde{A}_0^T(\tilde{A}_0\tilde{A}_0^T)^{-1}\tilde{A}_0 a_{0j}^T||$.
3. Find the row a_i of A that has norm-wise the largest orthogonal component to the rows of $\tilde{A}_0$.
4. If $||a_i^T - \tilde{A}_0^T(\tilde{A}_0\tilde{A}_0^T)^{-1}\tilde{A}_0 a_i^T|| > r$, swap a_{0j} and a_i and go to step 2, otherwise stop.

That is, as long as we can find a row of A that is more orthogonal to A_0 than the least orthogonal row of A_0, we swap the rows.

Heuristic 3: The third heuristic is essentially a combination of the first two heuristics. First, we generated an initial matrix A_0 as in the previous heuristic. Then compute

$$\sum_{k=1, k\neq j}^{m} |\langle a_{0j}||a_{0j}||^{-1}, a_{0k}||a_{0k}||^{-1}\rangle| \text{ for } n \le j \le m. \tag{1}$$

Next, find N largest of these values, where N is predetermined, and move the corresponding rows from A_0 to A. Denote $\tilde{A}_0 = A_0 \setminus A_{0,moved}$, and update A by

$A = A - (\tilde{A}_0^T(\tilde{A}_0\tilde{A}_0^T)^{-1}\tilde{A}_0A^T)^T$. Finally, use the algorithm presented in [3] to find a well-conditioned $N \times m$ submatrix of A that yields the choice of the N loops that replace the ones that were removed earlier.

4 Test Cases and Results

We will test the heuristics on the two cases we inspected the effect of scaling on. The cases are chosen due to their different distributions of pipes and nodes. We will perform the choosing of loops 100 times for both test cases, and initially we will generate approximately $10(m-n)$ loops. The parameter N for Heuristic 3 is chosen based on the mean value of the quantities computed in Eq. (1). Additionally, it should be noted that since row normalization with respect to 2-norm was found effective on Subsect. 3.1, we will do that in the test cases as well.

4.1 Case 1: 618 pipes, 235 nodes

In Case 1 the pipeflow network consists of 618 pipes and 235 nodes, i.e., the number of nodes is relatively high. The results on the test case are shown in Table 2 where we see that while Heuristic 1 provides the best-conditioned matrix, it also requires the most iterations and computational time. Heuristic 2 performs also rather well as it yields almost as good results as Heuristic 1 while it requires the least time and iterations. Heuristic 3 does clearly the worst as it is worse than Heuristic 2 on all the aspects. Regardless, it should be noted that all the heuristics yield clearly better results than choosing a matrix randomly as seen by comparing the two bottom rows of the table. Note also that the $\mathrm{cond}(A_0)$ for Heuristic 1 is left blank as the heuristic does not use any initial matrix, and the standard deviation on the number of iterations is not displayed as the heuristic always requires $m-n+1$ iterations.

Table 2 Results on Case 1. The numbers represent the mean values and standard deviations of the number of iterations, computational time, condition number of the $m \times m$ submatrix A and condition number of the initial $m \times m$ matrix A_0.

	Heuristic 1	Heuristic 2	Heuristic 3
iterations	384	51.0 ± 31.8	216.7 ± 36.7
time (s)	30.6 ± 1.2	8.0 ± 4.8	19.0 ± 3.5
cond(A)	6905 ± 4228	7119 ± 3910	9332 ± 4437
cond(A_0)	-	12312 ± 3083	12312 ± 3083

4.2 Case 2: 953 pipes, 78 nodes

In Case 2 the pipeflow network consist of 953 pipes and 78 nodes, i.e., the number of nodes is relatively low. In this case Heuristic 1 provides clearly the best-conditioned matrices but it also requires clearly the most computational time and iterations. Heuristic 2 performs only a few iterations, which explains the short computational time and only the slight improvement from the initial $\mathrm{cond}(A_0)$. Heuristic 3 similarly does relatively few iterations, but oddly the resulting matrix has on average worse condition number than the initial matrix. However, it should be noted that the heuristic aims to increase the orthogonality of the matrix, which may be achieved even if the condition number is increased.

Table 3 Results on Case 2. The numbers represent the mean values and standard deviations of the number of iterations, computational time, condition number of the $m \times m$ submatrix A and condition number of the initial $m \times m$ matrix A_0.

	Heuristic 1	Heuristic 2	Heuristic 3
iterations	876	2.33 ± 2.62	82.57 ± 9.98
time (s)	258.9 ± 7.9	2.1 ± 1.6	27.1 ± 3.2
cond(A)	1976 ± 462	7813 ± 2161	8454 ± 3470
cond(A_0)	-	7852 ± 2152	7852 ± 2152

5 Conclusions

We presented three heuristics for choosing linearly independent loops in a pipeflow network. The heuristics were tested on two cases with differently distributed pipes and nodes. The first heuristic provided the best results but also required the most time, whereas the second heuristic was the fastest one but could not always provide as good results as Heuristic 1. The third heuristic was clearly the worst of the three.

References

1. Golub, G.H., Van Loan, C.F.: Matrix Computations, 4th Edition. The Johns Hopkins University Press, Baltimore (2013).
2. Santamu, S.R.: Graph Theory with Algorithms and its Applications: In Applied Science and Technology. Springer India, New Delhi (2013)
3. Šrámek, R., Fisher, B., Vicari, E., Widmayer, P.: Optimal Transitions for Targeted Protein Quantification: Best Conditioned Submatrix Selection. In: Hung Q. Nqo (Ed.) COCOON 2009, LNCS 5609, pp. 287-296. Springer-Verlag, Berlin Heidelberg (2009)
4. Stephenson, D.: Pipeflow Analysis, 1st Edition. Elsevier Science, New York (1984)