Modeling of background leakages and inertia phenomena in drinking water distribution networks

Camille Chambon

▶ To cite this version:

Camille Chambon. Modeling of background leakages and inertia phenomena in drinking water distribution networks. Mathematics [math]. HESAM Université, 2023. English. NNT : 2023HESAC011 . tel-04176925

HAL Id: tel-04176925
https://hal.inrae.fr/tel-04176925

Submitted on 3 Aug 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0 International License

École doctorale Sciences des Métiers de l'Ingénieur
Laboratoire de Modélisation Mathématique et Numérique

THÈSE

présentée par : **Camille CHAMBON**
soutenue le : **10 juillet 2023**

pour obtenir le grade de : **Docteur d'HESAM Université**

préparée au : **Conservatoire national des arts et métiers**

Discipline : **Mathématiques appliquées et applications des mathématiques**

Spécialité : **Sciences pour l'ingénieur spécialité Mathématiques appliquées**

Modélisation des fuites diffuses et des phénomènes d'inertie dans les réseaux de distribution d'eau potable

THÈSE co-dirigée par :

M. MORTAZAVI Iraj Professeur des Universités, Cnam
et
M. PILLER Olivier Directeur de Recherche, INRAE

Jury:

Mme Fatiha NEJJARI	Associate Professor, Vice-Rector for teaching and students, CS2AC, UPC	Rapporteure
Mme Lisl WEYNANS	Maîtresse de Conférences HDR, IMB, Université de Bordeaux	Rapporteure
M. Thierry HORSIN	Professeur des Universités, M2N, Cnam	Président
M. Anthony BEAUDOIN	Maître de Conférences HDR, Pprime, Université de Poitiers	Examinateur
M. Jochen DEUERLEIN	Chercheur Principal, 3S Consult GmbH	Examinateur
M. Iraj MORTAZAVI	Professeur des Universités, M2N, Cnam	Co-directeur
M. Olivier PILLER	Directeur de Recherche, ETTIS, INRAE	Co-directeur

T
H
È
S
E

Affidavit

Je soussigné / soussignée, Camille CHAMBON, déclare par la présente que le travail présenté dans ce manuscrit est mon propre travail, réalisé sous la co-direction scientifique d'Iraj MORTAZAVI et d'Olivier PILLER, dans le respect des principes d'honnêteté, d'intégrité et de responsabilité inhérents à la mission de recherche. Les travaux de recherche et la rédaction de ce manuscrit ont été réalisés dans le respect de la charte nationale de déontologie des métiers de la recherche. Ce travail n'a pas été précédemment soumis en France ou à l'étranger dans une version identique ou similaire à un organisme examinateur.

Fait à Pessac, le 8 mai 2023

Affidavit

I, undersigned, Camille CHAMBON, hereby declare that the work presented in this manuscript is my own work, carried out under the scientific co-direction of Iraj MORTAZAVI and Olivier PILLER, in accordance with the principles of honesty, integrity and responsibility inherent to the research mission. The research work and the writing of this manuscript have been carried out in compliance with the French charter for Research Integrity. This work has not been submitted previously either in France or abroad in the same or in a similar version to any other examination body.

Pessac, on Mai 8, 2023

Acknowledgments

First and foremost I am extremely grateful to my supervisors, Professor Iraj MORTAZAVI and Research Director Olivier PILLER for their invaluable advice, continuous support, and patience during my PhD study. Their immense knowledge and plentiful experience have encouraged me in all the time of my academic research and daily life.

I would like to extend my sincere thanks to Associate Professors Fatiha NEJJARI and Lisl WEYNANS, for their meticulous proofreading of my manuscript, and to all the members of my defense's Jury – including Professor Thierry HORSIN and Doctor / Senior Project Engineer Jochen DEUERLEIN – for their relevant questions and discussion.

My gratitude extends to ERDF, the Nouvelle Aquitaine region, the Loire-Bretagne and Adour-Garonne water agencies, and the ARS Aquitaine for the funding opportunity to undertake my studies in the ETTIS (ex-ETBX) Research Unit of the INRAE (ex-IRSTEA) Aqua Department.

I would like to express my sincere gratitude to the members of my steering committee – Research Director Olivier LE MAITRE, Professor Elena DI BERNARDINO, and Associate Professors Lucie ROULEAU and Anthony BEAUDOIN – for their insightful comments and suggestions.

I would also like to thank Doctor / ROC's Project Manager Denis GILBERT, and Engineers Alain HUSSON, Eddy RENAUD and Sandrine SABATIE for their technical support on my study.

I am deeply grateful to Doctor David STEFFELBAUER, author of the OOPNET framework, for allowing me to use OOPNET for free in the scope of my thesis.

I would like to thank all the members in the ETTIS Research Unit and GPIE team. It is their kind help and support that have made my study and life in Bordeaux's area a wonderful time.

Finally, I would like to express my gratitude to my parents, partner and family. Without their tremendous understanding and encouragement in the past few years, it would be impossible for me to complete my study.

Résumé

L'objectif de cette thèse est de modéliser des réseaux de distribution d'eau potable sujets à des fuites diffuses et à des phénomènes d'inertie. Les pressions dans les réseaux doivent être suffisantes pour que tous les consommateurs aient de l'eau avec une bonne qualité de service. Cependant, pour limiter les fuites diffuses, ces pressions ne doivent pas être excessives. Un élément clé pour résoudre ce problème d'optimisation est de modéliser avec précision la dépendance des fuites diffuses à la pression. Nous proposons donc dans cette thèse plusieurs nouveaux modèles de fuites diffuses qui prennent en compte le gradient de pression le long des conduites. Nous montrons, à travers plusieurs expérimentations numériques sur des réseaux théoriques et réels, la supériorité de nos modèles par rapport à ceux de l'état de l'art. Notre approche permet également d'identifier les points hauts isolés en cas de pression insuffisante, et les parties les plus fuyardes des tronçons. Après validation de nos modèles en régime permanent, nous explorons la faisabilité de les intégrer dans un nouveau simulateur transitoire-lent qui décrit les phénomènes d'inertie. Ces phénomènes apparaissent par exemple lorsque les demandes des utilisateurs ou les hauteurs des réservoirs varient rapidement, des pompes sont démarrées, ou quand des vannes s'ouvrent ou se ferment en moins d'une minute. Nous observons des différences significatives entre les résultats de notre modèle transitoire-lent et ceux d'un simulateur pseudo-transitoire qui néglige les phénomènes d'inertie. Nous mettons aussi en évidence un accroissement important de la raideur du système à résoudre lorsque des fuites diffuses dépendant de la pression sont modélisées. Enfin, nous introduisons le calage des paramètres de fuite à partir des données expérimentales collectées lors du projet de Renouvellement Orienté des Conduites (ROC). Tous nos développements sont parties intégrantes d'un cadriciel collaboratif dédié à la modélisation des réseaux d'eau.

Mots-clés : réseau de distribution d'eau potable, fuite diffuse, modélisation dépendant de la pression, phénomène d'inertie, analyse hydraulique

Abstract

The purpose of this thesis is to model water distribution networks (WDNs) subject to background leakage outflows and inertia phenomena. Pressures in WDNs must be high enough for all consumers to have water with a good quality of service, but low enough to limit background leakages. A key element to solve this optimization problem is to model accurately the dependence of background leakages to pressures. For this purpose, we propose in this thesis several new background leakage models that take into account the gradient of pressure along the pipes. We show, through numerical experimentation on both theoretical and real networks, the superiority of our models when compared to the state-of-the-art ones. Also, our approach allows the simulation of high-lying nodes in case of insufficient pressures, and the identification of the leakiest parts of the pipes. Once our models are validated in steady-state, we explore the feasibility of integrating them into a new rigid water column (RWC) simulator that takes into account inertia phenomena. These phenomena appear, e.g., when users' demands or heads at tanks vary quickly, pumps are started, or valves are opening or closing in less than a minute. We observe significant differences between the results of our RWC and the ones of an extended-period simulator (EPS) that neglects inertia phenomena. We also highlight the increase of stiffness due to the integration of pressure-dependent outflows in the slow-transient equations. Finally, we initiate the calibration of the leakage parameters from the experimental data collected during the Oriented Renewal of Pipes (ROC) project. All our developments are integrated into a collaborative framework dedicated to WDNs modeling.

Keywords: water distribution network (WDN), background leakage, pressure-dependent model (PDM), inertia phenomena, hydraulic analysis

Contents

List of Tables

List of Figures

Résumé substantiel en français

Introduction

Motivations

Les réseaux de distribution d'eau potable

Les réseaux de distribution d'eau potable sont des infrastructures composées de canalisations, pompes, vannes, réservoirs et autres équipements hydrauliques. Ils permettent d'acheminer l'eau potable des stations de traitement jusqu'aux consommateurs [28, p. 1].

En amont des réseaux, l'eau non-traitée est prélevée dans des réservoirs souterrains, des rivières ou des lacs, puis traitée pour la rendre potable. En aval des réseaux, les eaux usées sont collectées, assainies, traitées, et finalement rejetées propres (mais non-potables) dans les milieux naturels [109].

La topologie des réseaux de distributions dépend de l'agencement des rues, de la topographie, du type de zone (c.-à-d. urbaine ou rurale) et de l'emplacement des installations de traitement et de stockage [28, p. 19]. Les canalisations sont généralement disposées en grille, ce qui permet à l'eau de circuler dans des boucles interconnectées. Ces boucles permettent une redondance qui évite qu'une partie des réseaux ne soit plus alimentée en cas de panne (rupture de canalisation, coupure de courant, etc.) et évite la stagnation de l'eau [21].

Les réseaux sont conçus pour répondre aux demandes en eau domestiques, commerciales, industrielles et pour lutter contre les incendies, à tout moment et avec une pression suffisante. Cependant, pour limiter les coûts de fonctionnement et les fuites, cette pression ne doit pas être excessive [109].

Les réseaux d'eau sont des infrastructures critiques, complexes et interconnectées d'une importance vitale pour la vie et le bien-être des humains. Ainsi, il est essentiel de développer des solutions abordables, qui permettant de maintenir une bonne qualité de service, et de garantir la sécurité des consommateurs et la pérennité des réseaux [124].

Les fuites diffuses

Les êtres humains ne pourraient pas survivre sans eau potable. La population mondiale ne cesse de croître [110], les deux tiers de cette population mondiale (c.-à-d. 4 milliards d'habitants) subissent déjà des pénuries graves d'eau potable pendant au moins un mois par an [104], et le réchauffement climatique et l'évolution de la demande en eau potable accentuera le risque de sécheresse, même en Europe [98]. Ainsi, il primordial de préserver au maximum les ressources actuelles en eau potable.

[136] a estimé que 30 % de l'eau potable est perdue à cause des fuites dans les réseaux. De plus, ces fuites entraînent une dégradation de la qualité de service [1] et un gaspillage des ressources énergétiques [24]. Par conséquent, les fuites doivent être limitées autant que possible.

Les fuites apparaissent avec le vieillissement des réseaux. Il existe deux catégories de fuites: les casses (aussi appelées "fuites rapportées"), et les fuites diffuses (ou "fuites non-rapportées") [136]. Les casses représentent des pertes importantes en eau, mais sont généralement rapidement détectées, localisées et réparées. À l'inverse, les fuites diffuses sont de petits écoulements provenant des joints, des raccords et de minces fissures le long des conduites. Les fuites diffuses sont trop faibles pour être localisées par des instruments de mesures ; cependant, elles engendrent des pertes d'eau en continu, souvent pendant de longues durées, et contribuent donc fortement aux pertes en eau dans les réseaux [91].

Les phénomènes d'inertie

Selon la première loi du mouvement de Newton [115], tout système résiste au changement lorsqu'une force s'y applique, pendant une période de temps donnée. Cette résistance est appelée "inertie".

L'inertie apparaît dans un réseau d'eau lorsque les demandes des consommateurs varient, lorsqu'une vanne est ouverte ou fermée, lors de la mise en marche et ou de l'arrêt d'une pompe, lors du remplissage ou vidage de réservoirs, etc. La variation des vitesses d'écoulement est alors ralentie par des phénomènes d'inertie [165, p. 578].

Les phénomènes d'inertie doivent être étudiés pour mieux analyser les réseaux [83]. En particulier, la prise en compte des phénomènes d'inertie permet d'optimiser la conception des réseaux [41, 72], et de réduire les coûts de fonctionnement [24, 90], les fuites [39, 85, 161, 169] et les risques de défaillance (c.-à-d. les risques de pannes) [79].

Modélisation mathématique

La modélisation mathématique consiste à définir les hypothèses, les équations et les contraintes qui caractérisent un modèle, c'est-à-dire d'une représentation simplifiée d'un système ou d'un processus [40]. A partir d'un jeu de paramètres et de données, l'exécution d'un modèle sur un ordinateur doit permettre d'obtenir des résultats fiables et interprétables. Lorsque le système d'équations décrit par le modèle mathématique ne peut être résolu analytiquement, un solveur est alors nécessaire pour fournir la meilleure approximation numérique possible des solutions. Les solveurs peuvent être optimisés pour accélérer la simulation (par ex., en parallélisant les calculs). Enfin, les simulateurs sont souvent utilisés via une interface graphique, ou intégrés à des infrastructure logicielles plus larges.

La modélisation et la simulation des réseaux d'eau sont des processus itératifs : identifier les besoins et les objectifs, convertir tous les composants du réseau en nœuds et tronçons interconnectés, décrire mathématiquement le comportement de ces composants, résoudre les équations, et afficher les solutions sur des cartes ou sous forme de tableaux [165, p. 10]). Dans un réseau, les interconnexions des nœuds et des tronçons sont représentées au moyen d'un graphe orienté, dans lequel les nœuds sont les sommets, les tronçons sont les arêtes, et l'orientation des arêtes correspond au sens des écoulements [15]. Le comportement des composants du réseau tient compte:

1. des propriétés physique de l'eau (c.-à-d. densité, viscosité, compressibilité et pression de vapeur),

2. de la pression statique et de la dynamique (c.-à-d. vitesse et régime d'écoulement) de l'eau,
3. des pertes d'énergie liées aux forces de friction et aux turbulences, et de l'énergie produite par les pompes,
4. des lois de conservation de la masse et de l'énergie,
5. et des lois de transport, de mélange, et de réactions chimiques.

Les lois de conservation de la masse et de l'énergie assurent la cohérence de l'ensemble du système [165, p. 49-50] : dans une conduite, la masse de fluide entrante doit être égale à la masse de fluide sortante (conservation de masse), et la différence d'énergie entre deux points d'un réseau doit être la même quel que soit le chemin suivi (conservation de l'énergie, d'après le principe de Bernoulli [12]). Pour résoudre les équations d'équilibre, le graphe du réseau est d'abord converti en une matrice d'incidence possédant une ligne par nœud et une colonne par tronçon, et constituée de 0, $+1$ et de -1. Pour chaque élément d'une colonne de la matrice, "$+1$" signifie que le tronçon est orienté "sortant" du nœud, "-1" que le tronçon est orienté "entrant" dans le nœud, et 0 que le tronçon n'est pas connecté au nœud [15]. Des solveurs non linéaires permettent ensuite de résoudre le système matriciel, et de trouver les débits dans les tronçons et les charges aux nœuds. En régime permanent (c.-à-d., qui ne dépend pas du temps), les techniques les plus populaires sont les méthodes de Newton et de Levenberg-Marquardt (LM). La méthode de Newton est quadratique mais la solution initiale ne doit pas trop éloignée de la solution optimale ; la méthode LM est plus robuste mais sa convergence plus lente [165, p. 662]. En régime transitoire (c.-à-d., qui dépend du temps), la méthode la plus utilisée est celle des caractéristiques (MOC), qui convertit les deux équations aux dérivées partielles (PDEs) de continuité et de quantité de mouvement en quatre équations différentielles ordinaires (ODEs); les ODEs sont ensuite résolues numériquement par différences finies [165, p. 583].

Les réseaux d'eau sont généralement simulés lorsqu'il n'est pas possible de mener certaines expérimentations ou campagnes de mesures physiques, ou lors de la conception ou de l'extension d'un réseau [165, p. 4]. En effet, une bonne conception et l'utilisation d'équipements appropriés permettent de limiter les coûts de construction et de fonctionnement, en agissant par exemple sur le nombre initial de canalisations, la fréquence des opérations de maintenance, la consommation énergétique des pompes et des vannes, et le nombre de capteurs à installer [2, 60]. En raison de leur topologie complexe, de leur grande taille (il n'est pas rare qu'un réseau alimente des centaines de milliers de consommateurs [165, p. 6]) et de leur agrandissement fréquent, la modélisation et la simulation efficaces des réseaux sont fondamentales.

Le projet ROC

Cette thèse a été financée par les agences de l'eau Loire-Bretagne et Adour-Garonne et par l'Agence Régionale de Santé (ARS) de la région Nouvelle-Aquitaine, dans le cadre du projet de recherche de Renouvellement Orienté des Canalisations (ROC). L'objectif de ce projet était de mener des recherches multidisciplinaires (sciences de l'ingénieur, mathématiques appliquées, sciences humaines et sociales), pour développer des outils et méthodes centrés sur la compréhension et la réduction des fuites diffuses et sur la qualité de l'eau.

Au cours du projet ROC, des données expérimentales ont été mesurées par [52], pour évaluer les niveaux de fuites dans plusieurs réseaux. L'un des apports de cette thèse est le développement de modèles de fuite et de méthodes de calage de ces modèles à partir des données mesurées au cours du projet ROC. Le calage d'un modèle est le processus au cours duquel les paramètres du modèle sont ajustés de façon à ce que le modèle génère après calage des résultats proches des valeurs mesurées [80, p. 221].

Des modèles bien calés permettent d'obtenir des résultats fiables et de simuler des scénarios extrêmes (pic de demandes, utilisation de bornes à incendies, casses de conduites, etc.) [94, 164]. En particulier, le calage d'un modèle peut être utilisé pour détecter et estimer le niveau de fuite dans les réseaux [101, 161, 171]. Il est donc très important de caler les modèles de réseaux d'eau avec précision.

État de l'art

Modélisation des réseaux de distribution d'eau potable

Parmi les logiciels de simulations des réseaux d'eau, il existe les logiciels Porteau [131], EPANET [140, 141] et OOPNET [147, 148].

Porteau [131] est une boîte à outils logicielle pour l'analyse des réseaux de distribution développée par l'Institut national de recherche pour l'agriculture, l'alimentation et l'environnement[1] (INRAE). Porteau est codé en langage Java pour l'interface graphique, et en C++ pour les moteurs de calcul Zomayet, Opointe et Quality :

- Zomayet [122] permet le lancement de simulations déterministes en régime pseudo-transitoire (ou quasi-stationnaire). À chaque pas de temps de la simulation, les équations hydrauliques en régime permanent (c.-à-d. le bilan massique aux nœuds jonctions et la conservation de l'énergie dans les tronçons) sont résolues avec une variante de la méthode de Newton, et les nouvelles charges aux nœuds réservoirs sont calculées avec une formule de quadrature. Le module Zomayet implémente un "modèle dirigé par la demande" (DDM) pour les consommations des utilisateurs ; dans les modèles DDMs, la demande des utilisateurs est supposée satisfaite quelle que soit la pression aux nœuds,
- le module Opointe ([125]) est un modèle stochastique qui permet de prédire les valeurs maximales des débits dans les tronçons et les valeurs minimales des pressions aux nœuds lors des pics de demandes,
- le module Quality résout les équations d'advection-réaction; il permet notamment de déterminer l'âge de l'eau, de remonter la trace et de prédire le devenir et les concentrations de composés chimiques (chlore, par ex.).

Porteau possède son propre format de données basé sur le langage XML[2], mais l'import et l'export aux formats EPANET, Excel et SIG sont également possibles. Porteau a été utilisé dans de nombreux projets de recherche, pour concevoir des plans de réhabilitation [51], évaluer la résilience de réseaux [134], étudier le transport de contaminants [16], etc.

EPANET [140, 141] est l'outil d'évaluation des réseaux de distribution d'eau potable développé par l'Environmental Protection Agency (EPA) des États-Unis. Le moteur de calcul d'EPANET est codé en langage C, et son interface graphique en Pascal Objet. Comme le logiciel Porteau, EPANET permet d'analyser le fonctionnement hydraulique des réseaux en régime pseudo-transitoire, en résolvant les équations hydrauliques avec une méthode de type Newton. EPANET possède également un module dédié à la qualité de l'eau. Depuis peu, EPANET modélise les consommations des usagers en tenant

[1] `https://www.inrae.fr/`

[2] XML : eXtensible Markup Language

compte de la pression dans les réseaux [141].

Enfin, OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) est une API (Application Programming Interface) Python à destination des étudiants, ingénieurs et chercheurs qui souhaitent réaliser des simulations hydrauliques, appliquer un pré ou post-traitement aux données d'entrée ou de sortie, et rendre les résultats de simulation d'EPANET plus lisibles [147]. Récemment, un effort important a été entrepris pour rendre OOPNET plus flexible et davantage centré sur l'utilisateur, au moyen d'une approche de conception pilotée par le domaine (DDD) [148]. OOPNET est donc bien adapté au prototypage, à l'intégration de nouveaux processus physiques et au test de nouvelles hypothèses.

Sa gratuité, l'accès libre à son code source, sa programmation orienté-objet, et sa conception favorisant son développement collaboratif par la communauté des utilisateurs et modélisateurs de réseaux d'eau, font d'OOPNET la solution la plus adaptée actuellement pour le prototypage, l'intégration de nouveaux processus hydrauliques, et le test de nouvelles hypothèses. Nous avons donc décidé d'utiliser et d'étendre OOPNET dans le cadre de cette thèse.

Modélisation des fuites diffuses

Les fuites diffuses ne peuvent être localisées. Une solution pour les réduire consiste à remplacer les canalisations les plus fuyardes par des neuves, mais cela coûte très cher et peut entraîner une longue interruption de service. Plusieurs travaux cherchent à optimiser les stratégies de réhabilitation [3, 70], et à prioriser le remplacement des canalisations [53, 100, 108].

Avec un coût nettement inférieur au remplacement des canalisations, et sans aucune interruption de service, il est possible de réduire les fuites diffuses au moyen d'un contrôle intelligent des pressions [86]. Mais, pour être efficace, cette approche nécessite au préalable une modélisation précise des fuites diffuses dépendant de la pression [55, 57, 101].

Au cours des dernières décennies, plusieurs modèles ont été développés pour simuler les fuites diffuses dans les réseaux. Certains de ces modèles tiennent compte de la dépendance des fuites diffuses à la pression, mais ils négligent le gradient de pression le long des tronçons [49, 55] et la perte de quantité de mouvement axiale due aux fuites [45, 81]. Cependant, la prise en compte du gradient de pression et de la perte de moment axial permettrait de décrire plus fidèlement les fuites diffuses, et de mieux estimer le niveau et le type des fuites lors du calage.

Simulation des phénomènes d'inertie

Les logiciels Porteau et Epanet implémentent chacun leur simulateur pseudo-transitoire, dans lesquels, à chaque pas de temps, les équations d'équilibre hydraulique sont d'abord résolues en régime permanent, puis le niveau de l'eau dans les réservoirs est mis-à-jour avec une formule de quadrature. Ces simulateurs sont rapides en termes de temps de calcul, et permettent ainsi de réaliser des simulations longues et sur de grands réseaux. Cependant, ces simulateurs négligent les effets inertiels, ce qui fausse les résultats lorsque les débits et les charges varient à une échelle de temps d'une minute ou moins.

Pour simuler les phénomènes inertiels ainsi que les effets liés à la viscosité et la compressibilité de l'eau, les simulateurs de coups de bélier (aussi appelés simulateurs "transitoires-rapides") sont très efficaces. Ils permettent de prédire les surcharges dans les réseaux, et ainsi d'éviter des casses de conduites et d'équipements. Ces conditions extrêmes peuvent se produire par exemple lors d'une panne de courant, lorsque des vannes d'urgence sont actionnées, ou lorsque des bornes à incendie sont utilisées [50, 83, 87]. Cependant, les simulateurs de coups de bélier sont trop lents en termes de temps de calcul pour permettre la simulation de grands réseaux sur de longues périodes de temps [111].

A mi-chemin entre les simulateurs pseudo-transitoires et les simulateurs de coup de bélier, les simulateurs transitoires-lents négligent la compressibilité et les effets visqueux, mais décrivent bien les phénomènes d'inertie liés aux variations de débit à une échelle de temps de l'ordre de la minute [112]. Ces variations relativement lentes peuvent être observées lorsque la demande des utilisateurs fluctue, ou lorsque des vannes ou des pompes sont activées progressivement. Parmi les applications possibles, les simulateurs transitoires-lents permettent de modéliser le processus de vidange dans une canalisation à l'aide d'air sous pression [25], de localiser les fuites et de surveiller la qualité de l'eau [34, 157], et de contrôler la pression [89]. Les simulateurs transitoires-lents sont un bon compromis entre la rapidité des simulateurs pseudo-transitoires et la précision des simulateurs de coups de bélier [111].

Plusieurs modèles et simulateurs transitoires-lents ont été développés par le passé (par exemple, [81, 112, 119, 144]). Cependant, aucun d'entre eux n'intègre explicitement à la fois les phénomènes d'inertie et des fuites diffuses dépendant de la pression.

Principaux résultats

L'objectif principal de cette thèse est le développement d'un modèle plus précis des fuites diffuses dans les réseaux d'eau. Nous définissons 5 sous-objectifs : (1) développer un simulateur Python en régime permanent qui intègre un modèle de consommation tenant compte de la pression, (2) développer de nouveaux modèles de fuites diffuses qui prennent en compte le gradient de pression le long des tronçons, (3) améliorer les modèles pour qu'ils simulent correctement les réseaux comportant des points-hauts, (4) améliorer les modèles pour prendre en compte les phénomènes d'inertie, et (5) résoudre le problème inverse du calage des paramètres de fuite. Chacun de ces sous-objectifs est traité dans un chapitre dédié de la thèse.

Dans le chapitre 1, nous montrons que le premier sous-objectif (développer un simulateur Python en régime permanent qui intègre un modèle de consommation tenant compte de la pression) est clairement atteint. Nous testons notre simulateur sur des réseaux de grande tailles (c.-à-d. comportant jusqu'à 19 647 conduites et 17 986 nœuds), et pour différents niveaux de demandes des utilisateurs. Lorsque les demandes des utilisateurs sont pleinement satisfaites, notre simulateur est presque aussi rapide qu'EPANET. Nous intégrons notre simulateur au framework de modélisation OOPNET. Ainsi, nous obtenons un outil fonctionnel, efficace, flexible et maîtrisé pour simuler les consommations dépendant de la pression. Qui plus est, notre simulateur peut être réutilisé facilement par l'ensemble de la communauté scientifique, et permet l'étude de nouveaux processus hydrauliques.

Le chapitre 2 explique le travail réalisé pour atteindre notre deuxième sous-objectif. Nous proposons un nouveau modèle de référence qui discrétise récursivement les tronçons en sous-tronçons jusqu'à ce que la ligne de piézométrique le long de chaque tronçon converge. Nous développons également trois autres nouveaux modèles, par raffinements successifs d'un modèle pré-existant et considéré comme étant l'état de l'art avant cette thèse ; ces nouveaux modèles intègrent explicitement la variation des débits le long des tronçons fuyards. Tous nos modèles de fuites diffuses sont ensuite intégrés au simulateur Python développé dans le chapitre 1. Des tests numériques sur un unique tronçons puis sur un réseau réel montrent la supériorité de nos nouveaux modèles par rapport au modèle préexistant. Le choix entre nos nouveaux modèles dépend de l'exhaustivité des données mesurées (plus le modèle est complexe, plus il nécessite de données pour être calé), et/ou du niveau de précision souhaité (les modèles plus précis sont également plus gourmands en temps de calcul).

Notre chapitre 3 présente les méthodes misent en œuvre pour atteindre notre troisième sous-objectif (c.-à-.d, étendre les modèles de fuites développés au chapitre 2 pour qu'ils tiennent compte des points-hauts). Ainsi, nos modèles sont maintenant capables d'identifier précisément les parties des tronçons qui sont alimentées et celles qui ne le sont pas. Les tests numériques sur plusieurs petits réseaux dédiés et sur un réseau réel dans lequel des points-hauts ont été ajoutés montrent de meilleures prédictions des débits de fuite et des lignes piézométriques à l'échelle des tronçons. Nous rencontrons ensuite quelques difficultés à déterminer les débits exacts dans les parties d'un réseau qui sont en amont d'un point-haut situé dans une branche du réseau. Cependant, une simulation réalisée en post-traitement, en mettant les demandes des utilisateurs et les débits de fuite à zéro dans toutes les parties déconnectées du réseau, permet d'obtenir de bonnes approximations.

Le chapitre 4 montre une atteinte partielle de notre quatrième sous-objectif (c.-à-d., étendre les modèles développés dans les chapitres précédents pour qu'ils prennent en compte les phénomènes d'inertie). Nous développons un nouveau simulateur transitoire-lent qui néglige la compressibilité de l'eau mais tient compte des phénomènes d'inertie liés aux variations des débits et des charges à une échelle de temps de l'ordre de la minute. Nous montrons comment intégrer le modèle de consommation dépendant de la pression développé au chapitre 1. Par contre, nous ne parvenons pas à simuler des fuites diffuses dépendant de la pression en régime transitoire-lent. En effet, la méthode de pénalité employée pour contraindre les consommations requiert que les débits sortants rapportés aux nœuds puissent s'exprimer sous forme de fonctions inversibles, ce qui n'est plus le cas dans les modèles de fuites diffuses des chapitres 2 et 3. Nos tests numériques sur un seul tronçon fuyard permettent d'observer des phénomènes d'inertie significatifs. Cependant, sur un secteur d'un réseau réel, la méthode de pénalité utilisée ne permet pas de contraindre suffisamment les consommations pour qu'elles restent toujours en inférieures ou égales aux demandes, et le θ-schéma implémenté manque de stabilité.

Enfin, le chapitre 5 décrit les premières étapes nécessaires pour atteindre le cinquième sous-objectif: résoudre le problème inverse pour la calibration des modèles de fuites développés dans le chapitre 2. Nous expliquons analytiquement une méthode de calibration permettant d'estimer les paramètres de fuite associés à chaque tronçon. Nous fournissons également une première approximation numérique des paramètres de fuite à l'échelle du réseau. Les données expérimentales utilisées pour le calage des modèles proviennent de campagnes de mesures réalisées dans le cadre du projet ROC (Renouvellement Orienté des Conduites).

Principales contributions

L'objectif principal de la thèse (c.-à-d., développer un modèle plus précis des fuites diffuses dans les réseaux d'eau) est atteint. Nous apportons ainsi de nouvelles connaissances et une meilleure compréhension des fuites dépendant de la pression et des phénomènes d'inertie dans les réseaux. En particulier, nous montrons l'intérêt de prendre en compte le gradient de pression le long des tronçons pour une meilleure prédiction des fuites diffuses, des lignes piézométriques et pour identifier les points-hauts et les tronçons partiellement alimentés. Notre résultats confirment également la difficulté et l'importance de considérer les phénomènes d'inertie pour simuler correctement les événements et processus se déroulant à l'échelle de la minute (variation de la demande des utilisateurs, ouverture et fermeture des vannes, mise en marche/arrêt des pompes, etc.). De nouveaux simulateurs pseudo-transitoire et transitoire-lent développés en langage Python incluent tous nos travaux de recherche ; ces simulateurs sont intégrés au framework de modélisation OOPNET, afin de permettre une réutilisation et une extension plus faciles de nos travaux par la communauté. Enfin, cette thèse contribue au développement d'outils d'aide à la décision, indispensables au choix des meilleures stratégies de maintenance opérationnelle et de réhabilitation des réseaux, tout en tendant à réduire les pertes en eau. Les publications réalisées durant cette thèse sont listées à la fin de l'introduction générale (page 37).

General introduction

Motivation

Water distribution networks

Water distribution networks (WDNs) are infrastructures composed of pipes, pumps, valves, storage facilities and other hydraulic appurtenances. They aim to transport drinking water from treatment plants to consumers [28, p. 1].

At upstream of WDNs, the raw water is withdrawn from underground reservoirs, rivers or lakes, and purified in treatment plants. At downstream, the wastewater is collected from the consumers and treated, and the resulting clean (but non-potable) water is finally discharged to rivers, lakes or estuaries [109].

The configurations of WDNs depend on street patterns, topography, type of area (i.e., urban or rural), and location of the treatment and storage facilities [28, p. 19]. Pipes are generally arranged in grid layout, which permits water to circulate in interconnected loops. These loops allow redundancy that prevents parts of the networks to become unsupplied in case of breakdown (e.g., pipe burst, pump failure) and avoid water stagnation [21].

WDNs are designed to satisfy the domestic, commercial, industrial and fire-fighting demands, at any time and with enough pressure. However, to limit maintenance, leakages and functioning costs, this pressure should not be excessive [109].

WDNs are critical, complex and interconnected infrastructures of vital importance for human live and welfare. Thus, affordable solutions are needed to maintain their quality of service, security and sustainability [124].

Background leakages

Human beings could not survive without drinking water. The world-wide population is continuously growing [110], two-thirds of the global population (i.e., 4 billion people) already endure severe water scarcity during at least one month per year [104], and the global warming and evolution of water demand will accentuate drought risk, even in Europe [98]. Thus, it is essential to preserve drinking water as much as possible.

[136] estimated that 30 % of the drinking water is lost because of leakages in WDNs. Also, leakages in WDNs lead to undermined service quality [1] and wasted energy resources [24]. Thus, leakages have to be reduced.

Leakages appear with the aging of WDNs. They are generally classified into bursts (or “reported”) and background (or “unreported”) leakages [136]. Bursts represent large water outflows that can be quickly detected, located and repaired. Conversely, background leakages are slight outflows from joints, fittings, and thin cracks, and are too small to be located by physical measurements; however, they run continuously, often for a long time, and contribute greatly to water losses [91].

Inertia phenomena

According to the first Newton’s law of motion [115], any system resists to change when a force applies to it over a given period of time. This resistance is called “inertia”.

Inertia appears in a WDN when its state is changing (i.e., “transiting”): variation of users’ demands, opening and closure of valves, switching on and off of pumps, filling or emptying of reservoirs, etc. In all these cases, the variation of the flow velocities is slow down by inertia phenomena [165, p. 578].

Inertia phenomena must be investigated for better analysis of WDNs [83]. In particular, taking into account these phenomena permits to optimize the design of WDNs [41, 72], and to reduce functioning costs [24, 90], leakages [39, 85, 161, 169], and system failures [79].

Mathematical modeling

Mathematical modeling consists in defining the assumptions, the governing equations and the constraints of a model, which is a simplified representation of a system or a process [40]. The run of a mathematical model on a computer from specific parameters and data should permit to obtain meaningful results. When the system of equations described by the mathematical model cannot be solved analytically, a numerical solver is needed to provide the best possible approximation of the solutions. Solvers can be optimized to quicken the simulations (through, e.g., parallel computing). Finally, simulators are often integrated into larger software solutions, used through a graphical user interface (GUI), or integrated into a framework.

The modeling and simulation of WDNs are iterative processes: identify the needs and the purposes, convert all network components to nodes and links, describe mathematically the components’ behavior, solve the equations, and display the solutions on maps or as tabular output [165, p. 10]). In a WDN, the interconnections of the nodes and links are represented by a directed graph, in which the nodes are the vertices, the links are the edges, and the signs of the flow rates indicate the direction of the flows [15]. The behaviors of the nodes and links are described according to:

- fluid properties (i.e., density, viscosity, compressibility and vapor pressure),
- fluid static (i.e., static pressure) and dynamics (i.e., velocity and flow regime),
- energy losses (i.e., friction and turbulence) and gains (i.e., pumps),
- mass and energy conservation laws,
- and transport, mixing and chemical reactions laws.

The mass and energy conservation laws insure the consistency of the entire system [165, p. 49-50]: the fluid mass entering any pipe must be equal to the mass leaving the pipe (mass conservation), and the difference in energy between two points of a WDN must be the same regardless of the path that is taken (energy conservation, from Bernoulli’s principle [12]). To solve the equations, the graph of a WDN is first converted to an incidence matrix that has a row for every node and a column for every link; for each column of the matrix, the nonzero row entries “$+1$” and “-1” respectively indicate the nodes that begin and end an edge [15]. Then, non-linear solvers are used to find the flow rates traversing the pipes and the heads at nodes. In steady-state, popular technics are the Newton and the Levenberg-Marquardt (LM) methods. The Newton’s method is quadratic but the trial solution needs to be close to the optimum; the LM method is more robust but its convergence slower [165, p. 662]. For transient equations, the most popular approach is the method of characteristics (MOC), which converts the two partial differential equations (PDEs) of continuity and momentum into four ordinary differential equations (ODEs) that are solved numerically using finite difference techniques [165, p. 583].

In WDNs, simulations are commonly performed when it is not practical to conduct physical experimentation in the real system, or for the purpose of evaluating a system before it is actually built or extended [165, p. 4]. Like so, appropriate design and equipment can be chosen to limit the construction and functioning costs, by reducing for example the initial number of pipes, the frequency of maintenance operations, the energy consumption of pumps and valves, and the number of installed sensors [2, 60]. Because of their complex topology, frequent growth, and large size (it is not uncommon for a system to supply hundreds of thousands of consumers [165, p. 6]), effective modeling and simulation of WDNs are fundamental.

The ROC project

This PhD thesis was funded by the Loire-Bretagne and Adour-Garonne' water agencies and by the Regional Health Agency (ARS) of the Aquitaine region, within the scope of the Oriented Renewal of Pipes (ROC) research project. The goal of this project was to carry out multidisciplinary (i.e., engineering sciences, applied mathematics, human and social sciences) research to develop tools and methods focused on the understanding and reduction of background leakages and on water quality.

During the ROC project, experimental data were measured by [52], to assess in particular the leakages in several WDNs. One of the contributions of this thesis is the development of leakage models and the calibration of these models from the data measured during the ROC project. Model calibration is the process whereby the parameters of a model are adjusted to obtain a satisfactory agreement between model-generated results and measured variables [80, p. 221].

Well calibrated WDN models are able to produce meaningful results and simulate extreme scenarios (e.g., peak demands, fire flows, burst, etc.) [94, 164]. In particular, model calibration can be used to detect and estimate the level of leakages in WDNs [101, 161, 171]. Thus, it is very important to calibrate accurately the models.

State of the art

Water distribution networks modeling

Among other famous WDNs software solutions, there are Porteau [131], EPANET [140, 141], and OOPNET [147, 148].

Porteau [131] is the Object-Oriented programming hydraulic toolkit for water distribution system analysis developed by the French National Research Institute for Agriculture, Food and Environment[3] (INRAE). It is coded with the Java programming language for the Graphical User Interface (GUI), and in C++ for the calculation engines Zomayet, Opointe and Quality.

- Zomayet [122] allows the run of deterministic extended-period (a.k.a., quasi-steady) simulations (EPSs). At each time step of an EPS, the steady-state hydraulic equations (i.e., mass balance at junctions and energy conservation in pipes) are solved with a Newton's method variant, and the new heads at tanks are computed with a quadrature formula. The Zomayet module implements a "demand-driven model" (DDM) of users' consumptions; in DDM models, the users' demands is supposed satisfied whatever the pressure at nodes.
- Opointe ([125]) implements a stochastic model for peak period that predicts the upper limits for link flow rates and the lower limits for the nodal pressures.

[3] `https://www.inrae.fr/en`

- Quality solves the advection-reaction equations to determine water age, source tracking and fate of waterborne component concentration (e.g., free chlorine).

Porteau has its own XML[4]-based data format, but import/export from/to EPANET, Excel and GIS formats is possible too. Porteau has been used in several research projects, for rehabilitation design [51], resilience assessment [134], transport of contaminant [16], etc.

EPANET [140, 141] is the water distribution network evaluation tool developed by the United States Environmental Protection Agency (EPA). The computational engine of EPANET is implemented with the C programming language, and its user interface in Object Pascal. As the Porteau software, EPANET allows the run of water quality and EPS simulations, solving the hydraulic equations with a Newton-like method. The modeling of pressure-dependent users' consumptions was added to EPANET recently [141].

OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) is a Python API (Application Programming Interface) for students, engineers and researchers who want to run hydraulic simulations in high-level way, apply pre and post-processing on EPANET's input or output data, and make the simulation results from EPANET more readable [147]. Since recently, an important collaborative effort is made for OOPNET to become more flexible, user-focus and efficient through a domain-driven design (DDD) approach [148].

Because OOPNET is free, open-source, object-oriented, and designed to enhance collaborative effort from the community of programmers and users of WDN software, it is currently the best suited solution for prototyping, and to integrate new physical processes and test new hypotheses. We thereby decided to reuse and extend it during this thesis.

Background leakages modeling

Background leakages cannot be located and repaired. A solution to reduce them consists in replacing the leakiest pipes by new ones, but it is expensive and could lead to long interruptions of service. Several authors already tried to optimize the rehabilitation strategies [3, 70], and to prioritize pipe replacement [53, 100, 108].

With a lower cost than pipe replacement, and without any service interruption, it is possible to reduce background leakages through a smart control of the pressures [86]. But, to be efficient, this approach needs accurate modeling of pressure-dependent background leakages [55, 57, 101].

In last decades, several models have been proposed to simulate background leakages in WDNs. Some of these models consider the dependence of leakages to pressure, but they still neglect the gradient of pressure along the pipes [49, 55] and the loss of axial momentum due to background leakages [45, 81]. However, taking into account this gradient of pressure and loss of axis momentum would permit to model background leakages more accurately, and to better estimate the level and the type of the leakages through model calibration.

Simulation of inertia phenomena

Extended period simulators (EPSs) are computationally quick and permit to run long-term simulations of large WDNs. However, EPS simulators neglect inertial effects, skewing the results when the

[4] XML: eXtensible Markup Language

flows and heads change at a time scale of one minute or less.

To capture inertial phenomena and viscous and compressibility effects, the water hammer (a.k.a., "unsteady-compressible" or "fast-transient") simulators are very effective. These simulators are well suited for surge analysis to protect the system from excessive transient conditions, which can happen when a power failure occurs, emergency valves are operated, or fire hydrants are used [50, 83, 87]. However, the water hammer simulators are too slow for running long simulations on large WDNs [111].

Between EPS and water hammer simulators, the rigid water column (RWC) (a.k.a., "unsteady-incompressible" or "slow-transient") simulators neglect compressibility and viscous effects, but describe well inertia phenomena related to flow variations at time scale of one minute [112]. These relatively slow variations can be observed when the users' demands fluctuates, or when valves and/or pumps are activated. Among other applications, RWC simulators permit to model the emptying process in a pipeline using pressurized air [25], to locate leaks and monitor water quality [34, 157], and to control the pressure [89]. RWC simulators are good compromise between the accuracy of water hammer simulators and the computational efficiency of EPS simulators [111].

Several rigid water column (RWC) solutions already exist (e.g., [81, 112, 119, 144]). However, none of them integrate explicitly both inertia phenomena and pressure-dependent background leakage outflows.

Objective and outline of the thesis

The main goal of this thesis is to **develop a more accurate model of background leakage outflows**. To do so, we define 5 sub-objectives: (1) implement a pressure-dependent model (PDM) of users' consumptions in a quick steady-state Python simulator, (2) develop new models of background leakage outflows that take into account the gradient of pressure along the pipes, (3) extend our models to simulate high-lying nodes, (4) extend our models to consider inertia phenomena, and (5) solve the inverse problem of leakage parameter calibration. These sub-objectives correspond to the chapters of this thesis, which are outlined in the subsections below.

By decomposing the main objective into several sub-objectives, the research strategy adopted in this thesis follows an iterative, incremental and adaptive development process [93] close to the ones of the Agile methods [22]. This approach permits to identify and break down more easily the barriers that occur when modeling such intricate physical processes. Also, after each objective is reached, we will implement regression-tests to ensure that previous developments still perform correctly after integrating the new ones [170].

Chapters' overview

In chapter 1, we will develop a new Python simulator that implements a state-of-the-art pressure-dependent model (PDM) of users' consumptions based on the Wagner's pressure-outflow relationship (POR). We will then test our simulator on large WDNs. The goal of this chapter is to develop a mastered, stable and robust Python simulator, from which we can explore new complex processes such as background leakage outflows and inertia phenomena.

In chapter 2, we will develop new models of background leakage outflows that take into account the gradient of pressure along the pipes. We will implement our new models as an extension of the simulator of PDM users' consumptions developed in chapter 1. We will test our models on one theoretical and one real leaky network.

The chapter 3 will propose a method to extend the simulator developed in chapter 2 so it can deal with high-lying nodes and partly-supplied pipes. This corresponds to a special case met in chapter 2, which we choose to present in a separate chapter. The method will be tested on several dedicated theoretical networks, and on an adapted real one that includes high-lying nodes.

Chapter 4 will aim to develop a new RWC simulator that describes inertia phenomena, pressure-dependent users' consumptions, and pressure-driven background leakage outflows. We will then compare the results of the RWC simulator with the ones of an EPS simulator that models the same processes excepting inertia phenomena.

Finally, chapter 5 will propose methods for the calibration of the leakage parameters of the leakage models developed in previous chapters, from the experimental data collected during the ROC project. This chapter is an introduction to inverse modeling (i.e., the finding of the causal factors that produced a set of observations).

Related publications

The publications made during this thesis are:

- C. Chambon, O. Piller, and I. Mortazavi. Assessing Background Leakage Models in WDNs. In *17th International Computing & Control for the Water Industry Conference*, page 2, Sept. 2019. URL https://hal.inrae.fr/hal-02610102.
- C. Chambon, O. Piller, and I. Mortazavi. A new slow transient pressure-dependent model to simulate background leakages and inertia in water distribution networks. In *WATER LOSS 2022*, Prague, 2022.
- D. Steffelbauer, O. Piller, C. Chambon, and E. Abraham. Towards a novel multi-purpose simulation software of water distribution systems in Python. In *14th International Conference on Hydroinformatics, Water INFLUENCE Water INFormatic soLUtions and opEN problems in the cycle from Clouds to ocEan*, page 4, Bucharest, Romania, July 2022.
- C. Chambon, O. Piller, and I. Mortazavi. Modeling of pressure-dependent background leakages in water distribution networks. *Mathematics and Computers in Simulation.*. To revise.

Chapter 1

Pressure-dependent users' consumptions

Abstract

Modeling of pressure-dependent user's consumption is mandatory to simulate accurately the hydraulics of water distribution networks (WDNs). Several software solutions already exist for this purpose, but none of them actually permits the easy integration and test of new physical processes. In this first chapter, we propose a new Python simulator that implements a state-of-the-art pressure-dependent model (PDM) of users' consumptions based on the Wagner's pressure-outflow relationship (POR). We tested our simulator on 8 large and complex WDNs, composed of up to 19647 pipes and 17986 nodes, for different levels of users' demands. The results show similar precision and efficiency as the ones obtained by the authors of the original model with their MATLAB implementation. Moreover, in case of fully satisfied users' demands, our simulator provides same results as EPANET in comparable computational times. Finally, our simulator is integrated into the collaborative Python framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis); thus, it can be reused and/or extended by a large community of WDN modelers. The work in this chapter was a preliminary step before the modeling and testing of new processes such as background leakage outflows.

Keywords:
water distribution network (WDN), pressure-dependent model (PDM), user's consumption, Python programming language, numerical modeling

1.1 Introduction

1.1.1 Users' consumptions in WDNs

Users of water distribution networks (WDNs) are households, schools, hospitals, businesses, food-processing and pharmaceutical industries, fire departments, etc. Each user has its own needs of water that can vary over time. On a specific period, these needs correspond to a "demand", expressed as an outflow rate. From this demand, and depending on the capacities of the WDN, the amount of water a user actually consumes is called its "consumption". A user's consumption is between 0 (i.e., no water is provided to the user) and the demand (i.e., the demand of the user is fully satisfied). In the last three decades, the worldwide amount of drinking water consumed daily by households and public services is estimated to $1.28 \times 10^9\,\mathrm{m}^3$, which represents 510,995 Olympic-size swimming pools of 2,500 m^3, and 11 percent of total freshwater withdrawals [139]. In France, this amount is of $1.50 \times 10^7\,\mathrm{m}^3$ per day, which leads to an average domestic consumption of 148 l per day and inhabitant [92].

The first goal of WDNs is to satisfy the demand of the users. To fulfill these needs, WDNs must be well sized at their building, and then extended according to new needs (e.g., population growing, installation of new facilities). Pumps are used to maintain enough pressure to carry water from treatment plant or natural source to users. Also, valves are used to control the pressure, and to divide large networks in several district metered areas (DMA) that are easier to monitor with the help of sensors. Appropriate design and equipment permit to limit the construction and functioning costs, by reducing for example the initial number of pipes, maintenance operations, energy consumption of pumps and valves, number of installed sensors and amount of water losses due to leakages [2, 29, 60, 165]. Pressure-driven models (PDMs) of user's consumption are mandatory to optimize the choices of decision-makers [10, 86, 96, 97, 106].

1.1.2 Pressure-driven modeling of user's consumption

One approach to model user's consumption is by assuming that the demand of the users is always satisfied. These models are called "demand-driven models" (DDMs); they neglect unsatisfied user's demand due to insufficient pressure. Nevertheless, it is implemented in several water distribution system modeling software, such as EPANET ([United State] Environmental Protection Agency Network Evaluation Tool) [140] 2.0, its object-oriented C++ interface OOTEN (Object-Oriented Toolkit for EPANET) [158], and Porteau (object-oriented programming hydraulic toolkit for water distribution system analysis developed by Cemagref/IRSTEA/INRAE) [131].

Conversely, the PDM models permit to simulate users' consumptions that depend on the pressure. Different pressure-outflow relationships (POR) are used in these models [46, 61, 130, 152, 162]. The Wagner's POR [162] was proved to perform best against data [145]. This POR is implemented, for example, in EPANET 2.2 [141], in the EPANET compatible Python package WNTR (Water Network Tool for Resilience) [88], and in the MATLAB simulator developed by [43].

EPANET is not object-oriented, and its PDM model of users' consumptions was not operational at the start of this PhD. The Python package WNTR was designed to simulate and analyze resilience of water distribution networks under disaster scenarios, and its software architecture does not permit to extend it easily, in particular to add new physical processes like background leakages. The MATLAB simulator from [43] is efficient and works fine, but modifying it requires the purchasing of an expensive MATLAB license. Thus, there is currently no simulator of PDM consumption that is easy and/or cheap to extend.

1.1.3 The Python language

Python is a free, multi-platform, high-level and object-oriented language [99]. Its core philosophy emphasizes readability, making the codes written in Python easy to reuse and extend, even by people who are not software developers. Many efficient libraries for scientific computing are available in Python [19, 63, 74, 103, 160], and Python is today one of the most popular programming languages[1].

Big efforts have already been made for the simulation of WDNs with Python. For example, [168] allows the runs of transient simulations using an adapted method of characteristics. For extended-period simulations (EPS), the EPANET based package WNTR (Water Network Tool for Resilience) [88], and the object-oriented framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147] are stable and active solutions.

1.1.4 The framework OOPNET

OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) is a convenient Python API (Application Programming Interface) of EPANET, for students, engineers and researchers who want to run hydraulic simulations in high-level way, apply pre and post-processing

[1] `https://survey.stackoverflow.co/2022/`, `https://www.jetbrains.com/lp/devecosystem-2021/`, `https://octoverse.github.com/2022/top-programming-languages`, `https://pypl.github.io/PYPL.html`, `https://spectrum.ieee.org/top-programming-languages-2022`

on input or output data of EPANET, and make the simulation results from EPANET more readable [147].

OOPNET is based on several state-of-the-art Python libraries for scientific computing [63, 68, 71, 74, 103]. It permits, in particular, to parse and convert EPANET's input file to Python objects, run an EPANET simulation through a command line interface, and parse EPANET's output files and convert them to Python objects. It is free, open-source, object-oriented, and designed to enhance collaborative effort from the community of programmers and users of WDN software. Moreover, its authors and contributors currently work on making it more flexible, user-focus and efficient through a domain-driven design (DDD) approach [148]. Thus, OOPNET is well suited for prototyping and integrating new physical processes or test new hypotheses.

1.1.5 Hypothesis, objectives and research strategy

The consumption model proposed by [43] seems the most effective one to us. However, it needs MATLAB environment and it is not easy to extend. Thus, we need to implement another simulator, to test new physical processes in a more convenient way and share our developments with a wider community of users and programmers.

We think that using the Python language and the framework OOPNET is today the best solution for developing such a simulator and reaching our goal. We believe that these implementation choices make it possible to reproduce the results obtained by [43] with their MATLAB simulator. Moreover, this would extend in a non-regressive way the current capabilities of OOPNET, which by now permits DDM simulations only, through the run of EPANET's executable.

In this chapter, we will first describe the steady-state equations of the simulator, and the method to solve them, including several numerical enhancements to deal with potential sources of instabilities. Then, we will explain the Python implementation, the networks to simulate, and the tests and metrics proposed for validation. Finally, we will present our results and discuss them.

1.2 Methods

Hereafter, since we do not model any other type of demand and consumption, we will call "user's demand" just "demand", and "user's consumption" simply "consumption".

In all the networks that we will simulate, the kinetic energy (a.k.a., velocity head) is negligible compared to the pressure-head (a.k.a., internal pressure energy). The head h will then be calculated as:

$$h = u + p, \tag{1.1}$$

with u the elevation (in m) and p the pressure-head [165, p. 29]. h and p, as well as friction head-losses, will be expressed in $\mathrm{mH_2O}$ (metres water column), which are consistent to m. Also, all flow rates, demand and consumption will be expressed in $\mathrm{l\,s^{-1}}$, rather than $\mathrm{m^3\,s^{-1}}$, to avoid problems of stability due to machine precision. Indeed, [131] showed that these instabilities could appear when some flow rates are very close to 0.

In absence of other indications, scalar parameters and variables will be denoted in italic (e.g., x), vectors in italic bold (e.g., $\boldsymbol{v}$), matrices in italic bold upper-case (e.g., $\boldsymbol{M}$), scalar functions in upright (e.g., $\mathrm{f}(x)$), vector functions in upright bold (e.g., $\mathbf{f}(\boldsymbol{v})$), matrix functions in upright bold upper-case (e.g., $\mathbf{M}(\boldsymbol{v})$), and sets in blackboard style (e.g., $\mathbb{R}$).

1.2.1 Friction head-loss

In each pipe, we use the Hazen-Williams formulation [167] to compute the friction head-loss. To do so, denoting $\varnothing_p$ the inside diameter (in mm) of the pipe and $c_{\scriptscriptstyle HW} \in \mathbb{R}_*^+$ its roughness coefficient (unit-less; for flow rates in $\mathrm{m}^3\,\mathrm{s}^{-1}$), and supposing the pipe is cylindrical, we first compute the friction coefficient as

$$f = \frac{10.67}{(1000\,c_{\scriptscriptstyle HW})^{\gamma_{\scriptscriptstyle HW}}\ (\varnothing_p/1000)^{4.87}}. \tag{1.2}$$

Then, we compute the Hazen-Williams friction head-loss along the full length ℓ (in m) of the pipe with the function:

$$\xi_{\mathrm{f}}(q) = f\,\ell\,q\,|q|^{\gamma_{\scriptscriptstyle HW}-1}, \tag{1.3}$$

where $q \in \mathbb{R}$ is the algebraic flow rate (in $\mathrm{l\,s^{-1}}$) traversing the pipe, and $\gamma_{\scriptscriptstyle HW} = 1.852$ is the Hazen-Williams exponent (unit-less).

1.2.2 Consumption

At each junction node, we use the Wagner's POR [162] to compute the consumption. To do so, denoting $d \in \mathbb{R}^+$ the user's demand (in $\mathrm{l\,s^{-1}}$) located at the junction node, $p \in \mathbb{R}^+$ the pressure-head (in $\mathrm{mH_2O}$), and $p_m \in \mathbb{R}^+$ and $p_s \in \mathbb{R}_*^+$ the fixed minimum and service pressure-heads (in $\mathrm{mH_2O}$) such that $p_s > p_m$, we compute the fraction of pressure-head (unit-less) as:

$$\mathrm{z}(p) = \frac{p - p_m}{p_s - p_m}, \tag{1.4}$$

and the consumption as:

$$\mathrm{c}(p) = \begin{cases} 0 & \text{if } p \leq p_m \\ d\sqrt{\mathrm{z}(p)} & \text{if } p_m < p < p_s \\ d & \text{if } p \geq p_s. \end{cases} \tag{1.5}$$

Usually, minimum and service pressure-heads are chosen respectively equal to 0 and $20\,\mathrm{mH_2O}$ [35, 43, 55, 57, 130].

1.2.3 Equations of equilibrium in a WDN

For any WDN, we denote n_p the number of cylindrical and longitudinal pipes of lengths $\boldsymbol{\ell} = (\ell_1, \dots, \ell_{n_p})^T \in \mathbb{R}^{n_p}$, n_j the number of junction nodes at which the heads are unknown, n_t and n_r respectively the number of tank and reservoir nodes at which the heads are known and fixed, $n_0 = n_t + n_r$ the total number of source nodes, and $n_N = n_j + n_0$ the total number of nodes. Also, we denote $\boldsymbol{q} = (q_1, \dots, q_{n_p})^T \in \mathbb{R}^{n_p}$ and $\boldsymbol{h} = (h_1, \dots, h_{n_j})^T \in \mathbb{R}^{n_j}$ the unknown flow rates in pipes and heads at junctions, $\boldsymbol{h_t} = (h_{t,1}, \dots, h_{t,n_t})^T \in \mathbb{R}^{n_t}$ and $\boldsymbol{h_r} = (h_{r,1}, \dots, h_{r,n_r})^T \in \mathbb{R}^{n_r}$ the known and fixed heads at tanks and at reservoirs, $\boldsymbol{h_0} = (\boldsymbol{h_t}^T \mid \boldsymbol{h_r}^T)^T \in \mathbb{R}^{n_0}$ and $\boldsymbol{h_N} = (\boldsymbol{h}^T \mid \boldsymbol{h_0}^T)^T \in \mathbb{R}^{n_N}$ the heads at all source nodes and at all nodes, $\boldsymbol{p} = \left(p_1, \ldots, p_{n_j}\right)^T \in \mathbb{R}^{n_j}$ and $\boldsymbol{l_t} = \left(l_1, \ldots, l_{n_t}\right)^T \in \mathbb{R}^{n_t}$ the pressure-heads at junctions and the water levels at tanks, and $\boldsymbol{u} = \left(u_1, \ldots, u_{n_j}\right)^T \in \mathbb{R}^{n_j}$, $\boldsymbol{u_t} = \left(u_1, \ldots, u_{n_t}\right)^T \in \mathbb{R}^{n_t}$ and $\boldsymbol{u_r} = \left(u_1, \ldots, u_{n_r}\right)^T \in \mathbb{R}^{n_r}$ respectively the elevations at junctions, the elevations at tank bottoms (i.e., where water level is zero), and the elevations of water surface in reservoirs. Then, we have the relations:

$$\boldsymbol{p} = \boldsymbol{h} - \boldsymbol{u} \tag{1.6}$$

at junctions,

$$\boldsymbol{l_t} = \boldsymbol{h_t} - \boldsymbol{u_t} \tag{1.7}$$

at tanks, and

$$\boldsymbol{h_r} = \boldsymbol{u_r} \tag{1.8}$$

at reservoirs. Next, we denote the $n_j \times n_p$ junction-pipe, the $n_t \times n_p$ tank-pipe, and the $n_r \times n_p$ reservoir-pipe incidence matrices as respectively $\boldsymbol{A}$, $\boldsymbol{A_t}$ and $\boldsymbol{A_r}$, the $n_0 \times n_p$ source-pipe incidence matrix as $\boldsymbol{A_0} = \left(\boldsymbol{A_t^T} \mid \boldsymbol{A_r^T}\right)^T$, the $n_N \times n_p$ node-pipe incidence matrix as $\boldsymbol{A_N} = \left(\boldsymbol{A}^T \mid \boldsymbol{A_0^T}\right)^T$, and $\boldsymbol{\xi_f}(\boldsymbol{q}) = \left(\xi_{\mathrm{f},1}(q_1), \ldots, \xi_{\mathrm{f},\mathrm{n_p}}(q_{n_p})\right)^T$ the vector function of $\mathbb{R}^{n_p}$ to $\mathbb{R}^{n_p}$ to compute the friction head-losses in pipes. $\forall k \in \{1, \ldots, n_p\}$, we compute the friction head-loss $\xi_{f,k}$ along the full pipe k by deriving eq. (1.3), such that:

$$\xi_{\mathrm{f},k}(q_k) = f_k\, \ell_k\, q_k \left|q_k\right|^{\gamma_{HW}-1}. \tag{1.9}$$

Finally, we denote $\boldsymbol{d} = \left(d_1, \ldots, d_{n_j}\right)^T \in \mathbb{R}^{n_j}$ the fixed demand at junctions, and $\mathbf{c}(\boldsymbol{h}) = \left(\mathrm{c}_1(h_1), \ldots, \mathrm{c}_{\mathrm{n_j}}(h_{n_j})\right)^T$ the vector function of $\mathbb{R}^{n_j}$ to $\mathbb{R}^{n_j}$ to compute the consumptions at junctions. Then, $\forall i \in \{1, \ldots, n_j\}$, we compute the consumption $\mathrm{c_i}(h_i)$ by deriving eq. (1.5), such that:

$$\mathrm{c_i}(h_i) = \begin{cases} 0 & \text{if } h_i \leq u_i + p_m \\ d_i \sqrt{\mathrm{z}(h_i - u_i)} & \text{if } u_i + p_m < h_i < u_i + p_s \\ d_i & \text{if } h_i \geq u_i + p_s. \end{cases} \tag{1.10}$$

To find the unknown flow rates $\boldsymbol{q}$ and heads $\boldsymbol{h}$ in the WDN at steady-state, we solve the non-linear system of equations:

$$\boldsymbol{\rho}(\boldsymbol{q}, \boldsymbol{h}) = \begin{pmatrix} \boldsymbol{\xi_f}(\boldsymbol{q}) - \boldsymbol{A}^T\boldsymbol{h} - \boldsymbol{A_0^T}\boldsymbol{h_0} \\ -\boldsymbol{A}\,\boldsymbol{q} - \mathbf{c}(\boldsymbol{h}) \end{pmatrix} = \boldsymbol{0}, \tag{1.11}$$

where

$$\boldsymbol{\xi_f}(\boldsymbol{q}) - \boldsymbol{A}^T\boldsymbol{h} - \boldsymbol{A_0^T}\boldsymbol{h_0} = \boldsymbol{\rho_e} \tag{1.12}$$

are the energy residuals in pipes, and

$$-\boldsymbol{A}\,\boldsymbol{q} - \mathbf{c}(\boldsymbol{h}) = \boldsymbol{\rho_m} \tag{1.13}$$

are the mass residuals at junctions. Then, the $\boldsymbol{h}$ found satisfies the principle of the conservation of energy, corresponding, in any pipe k of the network, to the Bernoulli's equation:

$$u_i + p_i = u_j + p_j + \xi_{f,k}, \tag{1.14}$$

where $\xi_{f,k}$ is the friction head-loss along k, and $\{u_i, u_j\}$ and $\{p_i, p_j\}$ are respectively the elevations and the pressure-heads at the nodes i and j located at the extremities of k; the kinetic energy is not represented in eq. (1.14) because it is negligible [165, p. 29]. Also, the flow rates $\boldsymbol{q}$ satisfy the conservation of the mass at nodes, derived from the Kirchhoff's first law: "the algebraic sum of the branch currents towards any node of an electric network is zero" [75].

1.2.4 Newton's method

We solve the system (1.11) with a Newton's method including a system reduction.

1.2.4.1 Newton's method

First, we define the Jacobian matrix function of the system (1.11) as:

$$\mathbf{J} = \begin{pmatrix} \mathbf{J_{11}} & \mathbf{J_{12}} \\ \mathbf{J_{21}} & \mathbf{J_{22}} \end{pmatrix} \tag{1.15}$$

where

$$\mathbf{J_{11}} = \frac{\boldsymbol{\partial \xi_f}}{\boldsymbol{\partial q}} \quad , \quad \mathbf{J_{12}} = -\boldsymbol{A}^T \quad , \quad \mathbf{J_{21}} = -\boldsymbol{A} \quad \text{and} \quad \mathbf{J_{22}} = -\frac{\boldsymbol{\partial \mathrm{c}}}{\boldsymbol{\partial h}}. \tag{1.16}$$

Then, denoting:

- $\boldsymbol{q}^{(0)}$ the initial guesses of flow rates in pipes, and $\boldsymbol{h}^{(0)}$ the ones of heads at junctions,
- and $\boldsymbol{J}^{(m)} = \mathbf{J}(\boldsymbol{q}^{(m)}, \boldsymbol{h}^{(m)})$ the Jacobian matrix of $\boldsymbol{\rho}^{(m)} = \boldsymbol{\rho}(\boldsymbol{q}^{(m)}, \boldsymbol{h}^{(m)})$ at iteration m,

and supposing that $\boldsymbol{J}^{(m)}$ is invertible, the Newton's method consists in computing, at each iteration $m = 0, 1, 2, \ldots$, the flow rates $\boldsymbol{q}^{(m+1)}$ and heads $\boldsymbol{h}^{(m+1)}$ as

$$\begin{pmatrix} \boldsymbol{q}^{(m+1)} \\ \boldsymbol{h}^{(m+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{q}^{(m)} \\ \boldsymbol{h}^{(m)} \end{pmatrix} - \left(\boldsymbol{J}^{(m)}\right)^{-1} \begin{pmatrix} \boldsymbol{\rho_e}^{(m)} \\ \boldsymbol{\rho_m}^{(m)} \end{pmatrix}, \tag{1.17}$$

until the differences between two successive iterates become less than a given tolerance.

1.2.4.2 System reduction

In eq. (1.17), computing the iterates $\boldsymbol{q}^{(m+1)}$ and $\boldsymbol{h}^{(m+1)}$ by inverting numerically the potentially very large matrix $\boldsymbol{J}^{(m)} \in \mathbb{R}^{(n_p+n_j)\times(n_p+n_j)}$ can take much computational time. Thus, we rather look for the descent directions on $\boldsymbol{q}^{(m)}$ and $\boldsymbol{h}^{(m)}$, defined respectively as

$$\boldsymbol{\delta_q}^{(m)} = \boldsymbol{q}^{(m+1)} - \boldsymbol{q}^{(m)} \qquad \text{and} \qquad \boldsymbol{\delta_h}^{(m)} = \boldsymbol{h}^{(m+1)} - \boldsymbol{h}^{(m)}, \tag{1.18}$$

and which satisfy the linear system:

$$\left(-\boldsymbol{J}^{(m)}\right) \begin{pmatrix} \boldsymbol{\delta_q}^{(m)} \\ \boldsymbol{\delta_h}^{(m)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\rho_e}^{(m)} \\ \boldsymbol{\rho_m}^{(m)} \end{pmatrix}, \tag{1.19}$$

or, in developed form:

$$\begin{pmatrix} \mathbf{J_{11}}^{(m)} \boldsymbol{\delta_q}^{(m)} + \mathbf{J_{12}}^{(m)} \boldsymbol{\delta_h}^{(m)} \\ \mathbf{J_{21}}^{(m)} \boldsymbol{\delta_q}^{(m)} + \mathbf{J_{22}}^{(m)} \boldsymbol{\delta_h}^{(m)} \end{pmatrix} = - \begin{pmatrix} \boldsymbol{\rho_e}^{(m)} \\ \boldsymbol{\rho_m}^{(m)} \end{pmatrix}. \tag{1.20}$$

To solve the system (1.20), we first use its first row to express $\boldsymbol{\delta_q}^{(m)}$ in function of $\boldsymbol{\delta_h}^{(m)}$:

$$\boldsymbol{\delta_q}^{(m)} = -\left(\boldsymbol{J_{11}}^{(m)}\right)^{-1} \left(\boldsymbol{J_{12}}^{(m)} \boldsymbol{\delta_h}^{(m)} + \boldsymbol{\rho_e}^{(m)}\right), \tag{1.21}$$

assuming that the diagonal matrix $\boldsymbol{J_{11}^{(m)}} \in \mathbb{R}^{n_p \times n_p}$ is invertible. Next, we substitute (1.21) into the second equation of the system (1.20) to obtain:

$$\boldsymbol{S}^{(m)}\, \boldsymbol{\delta_h^{(m)}} = \boldsymbol{\rho_m^{(m)}} - \boldsymbol{J_{21}^{(m)}} \left(\boldsymbol{J_{11}^{(m)}}\right)^{-1} \boldsymbol{\rho_e^{(m)}}, \tag{1.22}$$

where $\boldsymbol{S}^{(m)} \in \mathbb{R}^{n_j \times n_j}$ is the Schur complement of the block $-\boldsymbol{J_{11}^{(m)}}$ of the matrix $-\boldsymbol{J}^{(m)}$, and is defined as:

$$\boldsymbol{S}^{(m)} = \boldsymbol{J_{21}^{(m)}} \left(\boldsymbol{J_{11}^{(m)}}\right)^{-1} \boldsymbol{J_{12}^{(m)}} - \boldsymbol{J_{22}^{(m)}}. \tag{1.23}$$

In (1.23), $\boldsymbol{J_{21}^{(m)}}(\boldsymbol{J_{11}^{(m)}})^{-1}\boldsymbol{J_{12}^{(m)}}$ is symmetric and $\mathbf{J_{22}}$ is diagonal. Also, supposing that none of the flow rates in vector $\boldsymbol{q}^{(m)}$ are zero, then both $\boldsymbol{J_{21}^{(m)}}(\boldsymbol{J_{11}^{(m)}})^{-1}\boldsymbol{J_{12}^{(m)}}$ and $-\mathbf{J_{22}}$ are positive-definite. Thus, $\boldsymbol{S}^{(m)}$ is symmetric and positive-definite, and we can use a Cholesky factorization to compute $\boldsymbol{\delta_h^{(m)}}$ of eq. (1.22). Once $\boldsymbol{\delta_h^{(m)}}$ is calculated, we can easily compute $\boldsymbol{\delta_q^{(m)}}$ from eq. (1.21) because $\boldsymbol{J_{11}^{(m)}}$ is diagonal and supposed invertible.

Finally, after computing both $\boldsymbol{\delta_h^{(m)}}$ and $\boldsymbol{\delta_q^{(m)}}$, we can replace

$$-\left(\boldsymbol{J}^{(m)}\right)^{-1} \begin{pmatrix} \boldsymbol{\rho_e^{(m)}} \\ \boldsymbol{\rho_m^{(m)}} \end{pmatrix} \qquad \text{by} \qquad \begin{pmatrix} \boldsymbol{\delta_q^{(m)}} \\ \boldsymbol{\delta_h^{(m)}} \end{pmatrix}$$

in eq. (1.17), and compute the new iterates $\boldsymbol{q}^{(m+1)}$ and $\boldsymbol{h}^{(m+1)}$ as

$$\begin{pmatrix} \boldsymbol{q}^{(m+1)} \\ \boldsymbol{h}^{(m+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{q}^{(m)} \\ \boldsymbol{h}^{(m)} \end{pmatrix} + \begin{pmatrix} \boldsymbol{\delta_q^{(m)}} \\ \boldsymbol{\delta_h^{(m)}} \end{pmatrix}. \tag{1.24}$$

1.2.4.3 Initial guesses

Similarly to [43] and $\forall k \in \{1, \ldots, n_p\}$, the initial guess of the flow rate in the pipe k is chosen consistent with a velocity $v_k = 0.5\,\mathrm{m\,s^{-1}}$:

$$q_k^{(0)} = v_k\, \pi \frac{\varnothing_{p,k}^2}{4} \times 1000. \tag{1.25}$$

Also, $\forall i \in \{1, \ldots, n_j\}$, we set the initial guess of the head at i as:

$$h_i^{(0)} = p_m + u_i + \frac{p_s - p_m}{5}, \tag{1.26}$$

using the same notation as in sections 1.2.2 and 1.2.3.

1.2.4.4 Convergence criterion

To test for the convergence of scheme (1.24), we use a stop criterion that is almost the same as the one already used by [43]. Indeed, we consider that the convergence is reached as soon as, $\forall \boldsymbol{y} \in \{\boldsymbol{q}, \boldsymbol{h}\}$,

$$\begin{cases} \dfrac{\left\|\boldsymbol{y}^{(m+1)} - \boldsymbol{y}^{(m)}\right\|_\infty}{\left\|\boldsymbol{y}^{(m+1)}\right\|_\infty} \leq 10^{-6} & \text{if } \left\|\boldsymbol{y}^{(m+1)}\right\|_\infty \geq 10^{-6} \\ \left\|\boldsymbol{y}^{(m+1)} - \boldsymbol{y}^{(m)}\right\|_\infty \leq 10^{-6} & \text{otherwise.} \end{cases} \tag{1.27}$$

The only difference with [43] is that our version of the criterion also handles the case where $\left\|\boldsymbol{y}^{(m+1)} - \boldsymbol{y}^{(m)}\right\|_\infty \leq 10^{-6}$. We discuss briefly on the choice of the criterion in appendix A.2.

1.2.5 Sources of instabilities

Different sources of instabilities can hinder the convergence of the Newton's method described at section 1.2.4. We present them below, along with numerical enhancements to deal with them. These numerical enhancements are of primary importance to ensure the convergence of the Newton's method in many common situations. However, to preserve readability we choose to describe them in appendix A.1.

Remark: the numerical enhancements presented in sections 1.2.5.1 to 1.2.5.3 are the same as the ones implemented in the Porteau software [131].

1.2.5.1 Pipes with zero flow rate

At each iteration of the Newton's method and for each pipe $k \in \{1, \ldots, n_p\}$, the element $\left[\boldsymbol{J_{11}}\right]_{kk}$ of the Jacobian matrix is equal to the derivative of the friction head-loss, $d\xi_{f,k}/dq_k$, which can be computed as:

$$\frac{\mathrm{d}\xi_{\mathrm{f},k}}{\mathrm{d}q_k}(q_k) = \gamma_{\scriptscriptstyle HW}\, f_k\, \ell_k\, |q_k|^{\gamma_{\scriptscriptstyle HW}-1}. \tag{1.28}$$

But eq. (1.28) gives 0 for $q_k = 0$. Thus, since we need to compute the inverse of $\left[\boldsymbol{J_{11}}\right]_{kk}$ at each iteration (see eqs. (1.20), (1.21) and (1.23)), the use of eq. (1.28) to compute $d\xi_{f,k}/dq_k$ could lead to a division by zero. To prevent such error, we choose, as [122], to regularize the friction head-loss and its derivative for q_k close to 0, by respectively a cubic and a quadratic polynomial. Full description of this regularization is presented in appendix A.1.1.

1.2.5.2 Junction nodes with pressure-head close to the minimum or the service pressure-head

At each iteration of the Newton's method and for each pipe $k \in \{1, \ldots, n_p\}$, $[\boldsymbol{J_{22}}]_{ii}$ is equal to dc_i/dh_i, which can be computed as:

$$\frac{\mathrm{dc}_i}{\mathrm{d}h_i}(h_i) = \left(\frac{\mathrm{dc}_i}{\mathrm{d}p_i}\frac{\mathrm{dp}_i}{\mathrm{d}h_i}\right)(h_i)\,, \tag{1.29}$$

where, $\forall h_i \in \mathbb{R}^+$,

$$\frac{\mathrm{dp}_i}{\mathrm{d}h_i}(h_i) = 1, \tag{1.30}$$

and

$$\frac{\mathrm{dc}_i}{\mathrm{d}p_i}(p_i) = \begin{cases} d_i \dfrac{1}{p_s - p_m}\dfrac{1}{2\sqrt{\mathrm{z}(p_i)}} = \dfrac{1}{2\,(p_s - p_m)}\dfrac{\mathrm{c}_i(p_i)}{\mathrm{z}(p_i)} & \text{if } p_m < p_i < p_s \\ 0 & \text{otherwise,} \end{cases} \tag{1.31}$$

with $\mathrm{z}(p_i)$ and $\mathrm{c}_i(p_i)$ defined by eqs. (1.4) and (1.5). However, eq. (1.31) is undefined at $p_i = p_m$ and discontinuous at $p_i = p_s$. Thus, using eq. (1.31) to compute dc_i/dp_i could lead to convergence failure when p_i is close to p_m or p_s. To make eq. (1.31) continuous $\forall p_i \in \mathbb{R}$, we choose, as [130], to regularize the consumption and its derivative by respectively a cubic and a quadratic polynomial, when p_i is close to p_m and p_s. Full description of these regularizations is presented in appendix A.1.2.

1.2.5.3 Initial guesses far from the solution and/or Jacobian with (sub-)linear growth

The bottleneck of the Newton's method is the solving of the system (1.22). But the method can need numerous iterations if the initial guesses of the flow rates and heads at junctions are far from the solutions at equilibrium. In this case, the simulation of large WDNs can become unfeasible in reasonable computational time. More generally, the convergence is not anymore guaranteed if the Jacobian of the non-linear system of equations (1.11) does not have a super-linear growth [133]. Thus, to reduce the number of iterations needed by the Newton's method and to guarantee its convergence for any network configuration, we choose to reuse the damping method proposed by [43]. Succinct description of this method is presented in appendix A.1.3.

1.2.5.4 WDN with highly contrasted values of flow rates and/or heads

At each iteration of the Newton's method, the magnitude of the elements in the Jacobian matrix can vary strongly from one part of the WDN to another, according to the demands, the tank levels, the use of pumps and/or valves, etc. In this case, using the Jacobian matrix $\boldsymbol{J}$ as it can cause instabilities and increase significantly the number of iterations needed to converge. To limit the difference of magnitude order between all elements of $\boldsymbol{J}$, we choose to extend the preconditioning method initially proposed by [42]. Indeed, in [42], the method dealt with consumptions that do not depend on the pressures; thus, we extend it so it can now deal with pressure-dependent consumptions. Full description of our extend version is presented in appendix A.1.4. Using this preconditioning method, the solution algorithm becomes a quasi-Newton method.

1.2.6 Implementation and framework

To implement our Python simulator, we use the Python programming language[2] through its Anaconda distribution[3]. In addition to the Python's standard library[4], we use several third-party libraries:

- NetworkX [63], for the creation and manipulation of the graphed data structures of the WDNs,
- NumPy [65], for the creation and manipulation of the vectors and basic linear algebra operations (e.g., computation of vector norms),
- SciPy [160], to find the coefficients of the polynomials defined in appendices A.1.1 and A.1.2[5], and to handle sparse matrices[6],
- Scikit-Sparse[7], which itself interfaces CHOLMOD [19], to compute the Cholesky decomposition of the Schur complement matrix,
- XArray [71], to store the outputs in multi-dimensional labeled arrays,
- Pandas [103], to convert the outputs to tabular data structures, to post-process them, and to write them to comma-separated values (CSV) files,
- Matplotlib [74], to plot the outputs,
- and Sphinx[8], to extract the documentation from the source code, and generate an HTML file

[2] https://docs.python.org/3/

[3] https://www.anaconda.com/

[4] https://docs.python.org/3/library/

[5] https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.root.html

[6] https://docs.scipy.org/doc/scipy/reference/sparse.html

[7] https://scikit-sparse.readthedocs.io/

[8] https://www.sphinx-doc.org/

from it.

Our Python simulator is integrated into the framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147]. We use OOPNET in particular to parse and convert the EPANET's input and output files to Python objects.

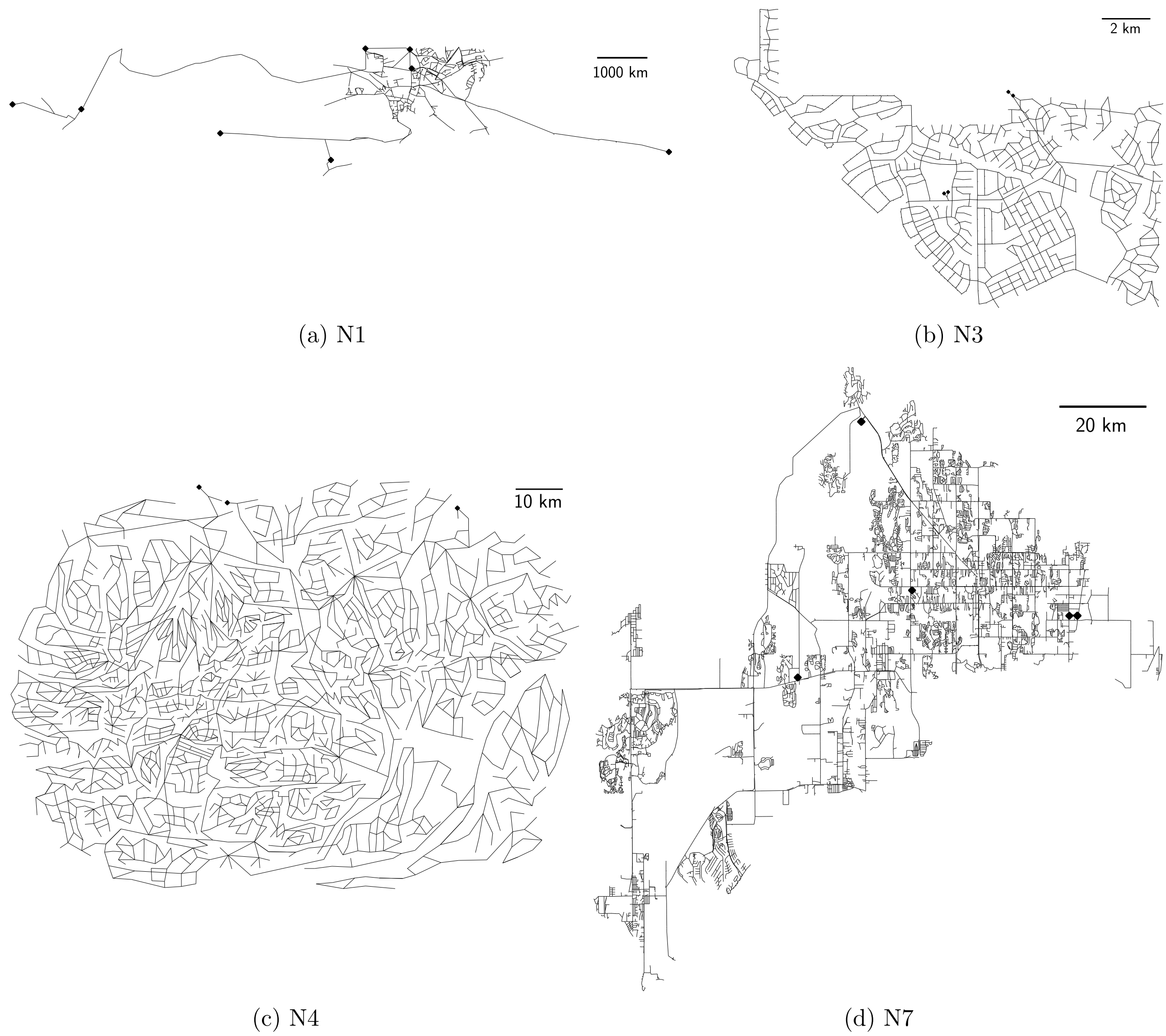
1.2.7 Simulated networks

To test our Python simulator, we simulate the set of networks $\mathbb{S}^{net} = \{\mathrm{N1}, \ldots, \mathrm{N8}\}$, which numbers of nodes and pipes are presented in table 1.1. EPANET's input files of networks $\{\mathrm{N1}, \mathrm{N3}, \mathrm{N4}, \mathrm{N7}\}$ are freely downloadable from the American Society of Civil Engineers (ASCE)'s library[9]; they are plotted in fig. 1.1. Other networks $\{\mathrm{N2}, \mathrm{N5}, \mathrm{N6}, \mathrm{N8}\}$ cannot be plotted here, because of ownership and/or security concerns.

Table 1.1: Number of pipes n_p, junctions n_j and reservoirs n_r in the networks $\{\mathrm{N1}, \ldots, \mathrm{N8}\}$ used to test our Python simulator.

Id	n_p	n_j	n_r
N_1	934	848	8
N_2	1,118	1,039	2
N_3	1,976	1,770	4
N_4	2,465	1,890	3
N_5	2,508	2,443	2
N_6	8,584	8,392	2
N_7	14,830	12,523	7
N_8	19,647	17,971	15

[9] www.ascelibrary.org



(a) N1

(b) N3

(c) N4

(d) N7

Figure 1.1: Networks $\{N1, N3, N4, N7\}$ used to test our Python simulator. Reservoirs are represented by ◆.

1.2.8 Tests and metrics

1.2.8.1 Functioning and performances of the Python simulator

To verify the good functioning of the Python simulator, we simulate the set of networks $\mathbb{S}^{net}$ presented in section 1.2.7, multiplying the peak demand by a multiplier $\mu_d \in \mathbb{S}^{\mu_d} = \{1, 2, 3, 5\}$. Like so, we test the Python simulator for different levels of demand satisfaction (in %), defined for each multiplier μ_d and at any junction $i \in \{1, \ldots, n_j\}$ as:

$$d_{s,\mu_d,i} = \frac{c_{\mu_d,i}}{d_{\mu_d,i}} \times 100. \tag{1.32}$$

For all these simulations, the minimum and service pressure-heads at all junctions are chosen respectively equal to 0 and $20\,\mathrm{mH_2O}$.

From the outputs of the simulations, we first check that the ∞-norm of the residuals is less than 10^{-3} (in $\mathrm{l\,s^{-1}}$ if the greatest absolute value is a mass residual, or in $\mathrm{mH_2O}$ if it is an energy one). If so, we then assume that the solver reaches the equilibrium state with enough precision.

Next, we check that the cumulated demand and consumption (in $\mathrm{l\,s^{-1}}$), defined for each demand multiplier $\mu_d \in \mathbb{S}^{\mu_d}$ as respectively:

$$d_{\mu_d}^{cumul} = \sum_{j \in \mathbb{S}^{net}} \left(\sum_{i \in \left\{1,\ldots,n_i^j\right\}} d_{\mu_d,i}^{j} \right) \qquad \text{and} \qquad c_{\mu_d}^{cumul} = \sum_{j \in \mathbb{S}^{net}} \left(\sum_{i \in \left\{1,\ldots,n_i^j\right\}} c_{\mu_d,i}^{j} \right) \tag{1.33}$$

both increase with μ_d. Also, we check that the cumulated demand increases quicker than the cumulated consumption, and that the global demand satisfaction, defined for each demand multiplier $\mu_d \in \mathbb{S}^{\mu_d}$ as:

$$d_{s,\mu_d}^{glob} = \frac{c_{\mu_d}^{cumul}}{d_{\mu_d}^{cumul}} \times 100, \tag{1.34}$$

decreases with μ_d.

Next, to assess the efficiency of the Python simulator and of the Newton's method, we compute, for each network $j \in \mathbb{S}^{net}$, the mean central processing unit (CPU) time elapsed during all runs with all $\mu_d \in \mathbb{S}^{\mu_d}$, first considering the whole simulations (i.e., pre-processing, solver and post-processing):

$$\overline{t_{CPU}^{simu,j}} = \frac{1}{\mathrm{card}(\mathbb{S}^{\mu_d})} \sum_{\mu_d \in \mathbb{S}^{\mu_d}} t_{CPU,\mu_d}^{simu,j}, \tag{1.35}$$

and then considering only the times needed by the solver to reach convergence:

$$\overline{t_{CPU}^{solv,j}} = \frac{1}{\mathrm{card}(\mathbb{S}^{\mu_d})} \sum_{\mu_d \in \mathbb{S}^{\mu_d}} t_{CPU,\mu_d}^{solv,j}, \tag{1.36}$$

where $\mathrm{card}(\mathbb{S}^{\mu_d})$ is the cardinality of the set $\mathbb{S}^{\mu_d}$. We expect logically the elapsed CPU times to increase with the size of the networks. Also, since the critical part of the solver (i.e., the Cholesky decomposition of the Schur complement matrix) uses the optimized CHOLMOD compiled library [19], we expect that the times needed by the solver increases slower than the ones needed for the whole simulations when larger networks are simulated. To check this point, we compute, for each network $j \in \mathbb{S}^{net}$, the ratio of time needed by the solver when compared to the times needed by the whole simulations, defined as:

$$\delta\,\overline{t_{CPU}^{solv,j}} = \frac{\overline{t_{CPU}^{solv,j}}}{\overline{t_{CPU}^{simu,j}}} \times 100. \tag{1.37}$$

We implement several numerical enhancements to deal with potential sources of instabilities (see section 1.2.5 and appendix A.1). To quantify the gain of these enhancements, we simulate each network $j \in \mathbb{S}^{net}$ for each demand multiplier $\mu_d \in \mathbb{S}^{\mu_d}$, first with no numerical enhancement, and next adding successively each numerical enhancement. Hereafter, we denote this set of enabled enhancements as $\mathbb{S}^{enh} = \{\mathrm{None}, \mathrm{Head\text{-}loss}, +\ \mathrm{Consumption}, +\ \mathrm{Damping}, \mathrm{Preconditioning}\}$ ("+" means "in addition to all previously enabled enhancement(s)"). We then compute, for each run, the convergence order of the Newton's method when it converges. To do so, for each enabled enhancement(s) $i \in \mathbb{S}^{enh}$, each network $j \in \mathbb{S}^{net}$ and each demand multiplier $\mu_d \in \mathbb{S}^{\mu_d}$, denoting $\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m)} \in \mathbb{R}^{n_p+n_j}$ the residuals at the iteration m of the Newton's method, we compute the difference between the ∞-norm of $\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m)}$ and $\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m+1)}$, $\forall m \in \left\{0, \dots, m_{c,\mu_d}^{i,j}\right\}$, where $m_{c,\mu_d}^{i,j}$ is the number of iterations needed to reach convergence, as:

$$\varepsilon_{\mu_d}^{i,j\,(m)} = \|\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m)}\|_\infty - \|\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m+1)}\|_\infty. \tag{1.38}$$

The convergence order $\eta_{\mu_d}^{i,j}$ then satisfies the relation:

$$\varepsilon^{(m)} = a_{\mu_d}^{i,j}\, m^{\eta_{\mu_d}^{i,j}}, \tag{1.39}$$

with $a_{\mu_d}^{i,j}$ a constant. After rewriting eq. (1.39) in log - log scale as:

$$\log(\varepsilon_{\mu_d}^{i,j\,(m)}) = \log\left(a_{\mu_d}^{i,j}\right) + \eta_{\mu_d}^{i,j} \log(m), \tag{1.40}$$

we then compute $\eta_{\mu_d}^{i,j}$ by linear regression of eq. (1.40).

The $\varepsilon_{\mu_d}^{i,j\,(m)}$ computed by eq. (1.38) can sometimes be negative, especially at $m = 1$ or at an iteration that follows a damping correction. Thus, to avoid error in the computation of $\eta_{\mu_d}^{i,j}$ through eq. (1.40), we do not consider these negative $\varepsilon_{\mu_d}^{i,j\,(m)}$, and we use as previous iteration the last iteration m' such that $\varepsilon_{\mu_d}^{i,j\,(m')} > 0$. Equation (1.38) then becomes:

$$\varepsilon_{\mu_d}^{i,j\,(m)} = \|\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m')}\|_\infty - \|\boldsymbol{\rho}_{\boldsymbol{\mu_d}}^{\boldsymbol{i,j}\,(m+1)}\|_\infty. \tag{1.41}$$

Finally, from all $\eta_{\mu_d}^{i,j}$, we compute, for each enabled enhancement(s) $i \in \mathbb{S}^{enh}$, the mean convergence order as:

$$\overline{\eta^i} = \frac{1}{\operatorname{card}(\mathbb{S}^{net})\operatorname{card}(\mathbb{S}^{\mu_d})} \sum_{j \in \mathbb{S}^{net}} \sum_{\mu_d \in \mathbb{S}^{\mu_d}} \eta_{\mu_d}^{i,j}. \tag{1.42}$$

1.2.8.2 Comparison of the Python simulator with the MATLAB one

To validate the results of the Python simulator, we compare the consumptions computed by the Python simulator with the ones computed by [43] running their MATLAB simulator, for the same sets of networks $\mathbb{S}^{net}$ and demand multipliers $\mathbb{S}^{\mu_d}$, and same minimum and service pressure-heads at all junctions: $p_m = 0$ and $p_s = 20\,\mathrm{mH_2O}$. Then, we assume that the consumptions from both Python and MATLAB simulators are the same as soon as they all differ by less than $10^{-2}\,\mathrm{l\,s^{-1}}$. Supposing that the average flow rate in the tested networks is equal to $10\,\mathrm{l\,s^{-1}}$, then $10^{-2}\,\mathrm{l\,s^{-1}}$ represents a precision of ~ 1 ‰, which corresponds to $1/10^{\text{th}}$ of the best precision expected when measuring flow rates physically in a WDN [78, p. 3].

Remark: the MATLAB simulator of [43] implements the numerical enhancements described in appendices A.1.1 to A.1.3.

To compare the performances of the Python simulator with the ones of the MATLAB simulator, we compute, in a similar way as in section 1.2.8.1 but for each network $j \in \mathbb{S}^{net}$, the mean convergence orders associated to each simulator, as:

$$\overline{\eta^j} = \frac{1}{\operatorname{card}(\mathbb{S}^{\mu_d})} \sum_{\mu_d \in \mathbb{S}^{\mu_d}} \eta_{\mu_d}^{j}, \tag{1.43}$$

where $\eta^j_{\mu_d}$ is the convergence order obtained when simulating the network j with the demand multiplier μ_d.

Also, to check that the orders of convergence are not biased by greater numbers of damping corrections for one of the simulators, we compute, for each network $j \in \mathbb{S}^{net}$, the mean numbers of damping corrections associated to each simulator as:

$$\overline{n^j_{corr}} = \frac{1}{\operatorname{card}(\mathbb{S}^{\mu_d})} \sum_{\mu_d \in \mathbb{S}^{\mu_d}} n^j_{corr,\mu_d}, \tag{1.44}$$

where n^j_{corr,μ_d} is the number of damping corrections obtained when simulating the network j with the demand multiplier μ_d. Since the damping algorithm implemented in both simulators is the same, we expect the simulators to need (more or less) the same number of damping corrections.

Finally, it is not possible to compare the process times elapsed using each simulator, because the MATLAB simulator was run by [43] on a different machine than the one we use to run our Python simulator.

1.2.8.3 Comparison of the Python simulator with EPANET

To check for the non-regression of our Python simulator, we simulate the same set of networks $\mathbb{S}^{net}$ with EPANET, but increasing sufficiently the heads at the source nodes $\boldsymbol{h_0}$, and reducing enough the service pressure-head p_s, so that the demand becomes satisfied at every junction. Then, we check, at each junction of each network, that the consumption computed by the Python simulator and the one computed by EPANET differ by less than $10^{-2}\,\mathrm{l\,s^{-1}}$.

Finally, to compare the performances of the Python simulator with the ones of the EPANET simulator, we compute, for both simulators and for each network, the ∞-norms of the residuals at convergence, along with the CPU times elapsed running each simulator on the same machine: an Intel Core i9 with 32 GB of memory. For these comparisons to be relevant, we set the "ACCURACY" parameter of EPANET to the same value as the one used in the convergence criterion of the Python simulator eq. (1.27), that is 10^{-6} (unit-less). The "ACCURACY" parameter is used by the EPANET solver to determine when the convergence is reached. Indeed, according to the EPANET documentation [141, p. 147]: "The trials end when the sum of all flow changes from the previous solution divided by the total flow in all links is less than this number".

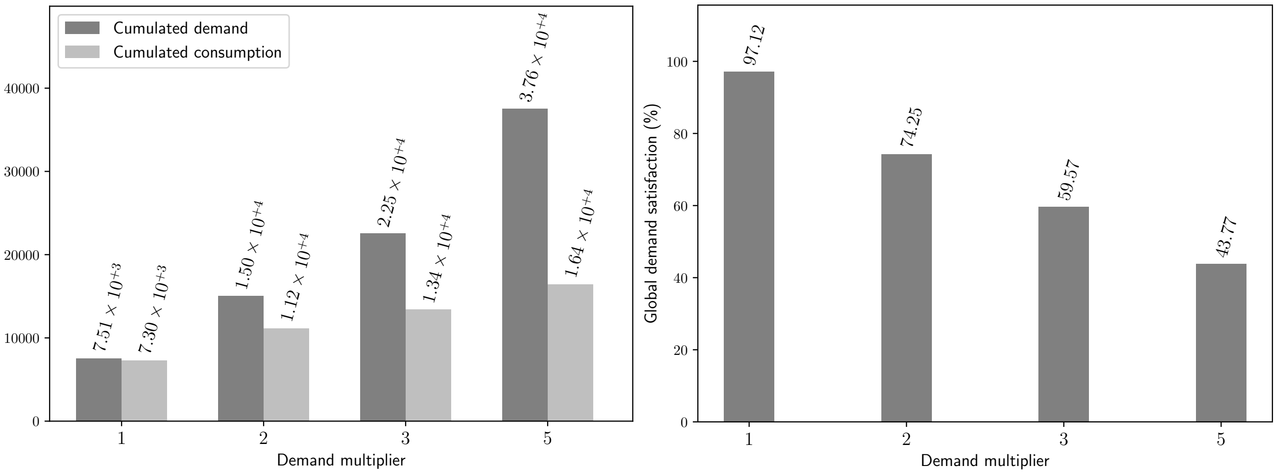
1.3 Results and discussion

This section presents the results obtained when running the tests explained in section 1.2.8.

1.3.1 Functioning and performances of the Python simulator

When simulating the set of networks for each demand multiplier, the maximal ∞-norm of all residuals is equal to 5.45×10^{-5} (in $\mathrm{l\,s^{-1}}$ or $\mathrm{mH_2O}$), which is much less than 10^{-3}. Thus, the solver reaches the equilibrium with enough precision.

We clearly observe, as expected, that the cumulated demand and consumption increase with the demand multiplier μ_d, and that the cumulated demand increases quicker than the cumulated consumption (fig. 1.2a). This last observation is confirmed when computing the global demand satisfaction (fig. 1.2b), which decreases when μ_d increases. This functioning is completely realistic.

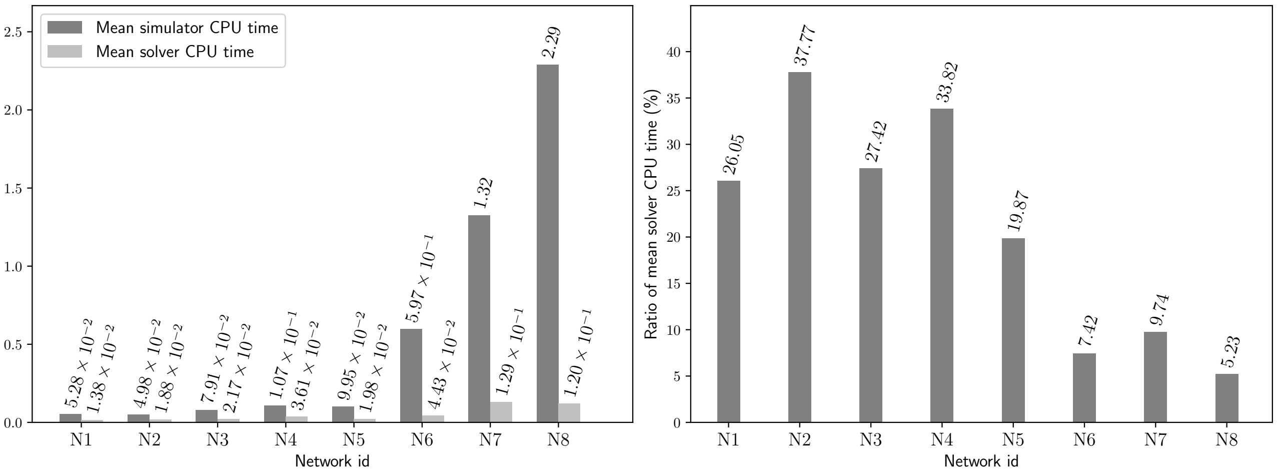


(a) Cumulated demand and consumption (in $\mathrm{l\,s^{-1}}$)

(b) Global demand satisfaction (in %)

Figure 1.2: Cumulated demand and consumption (fig. 1.2a), and global demand satisfaction (fig. 1.2b), simulated in networks $\{\mathrm{N1}, \ldots, \mathrm{N8}\}$ with the Python simulator, for each demand multiplier $\mu_d \in \{1, 2, 3, 5\}$.

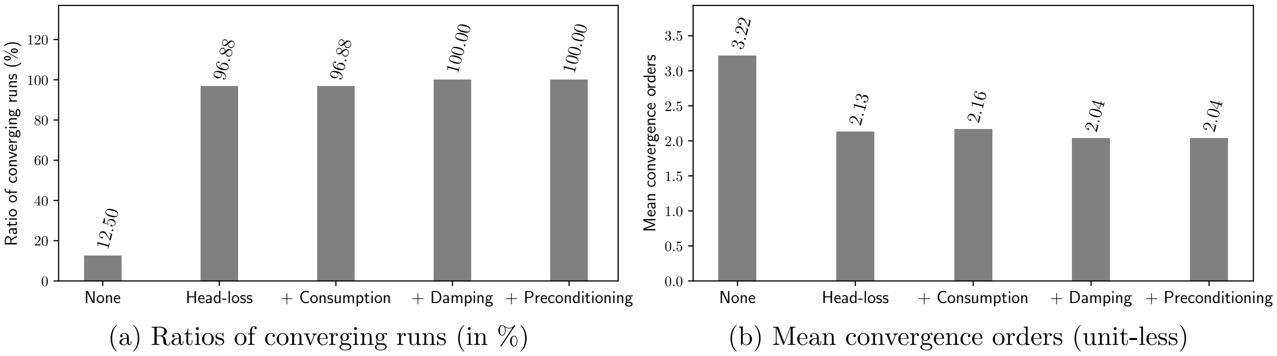
As logically expected, the elapsed CPU times of both simulator and solver increase globally with the size of the networks, and the elapsed time of the solver increases slower (fig. 1.3a). This is confirmed when computing the ratio of time needed by the solver when compared to the one needed for the whole simulator (fig. 1.3b).



(a) Mean simulator and solver CPU times (in s) (b) Ratio of mean CPU time used by the solver (in %)

Figure 1.3: Mean simulator and solver elapsed CPU times (fig. 1.3a), and ratio of mean solver elapsed CPU time when compared to mean simulator elapsed CPU time (fig. 1.3b).

Figures 1.4a and 1.4b show the ratios of converging runs (in %) and the mean convergence orders when simulating all networks for all demand multipliers, first with no numerical enhancement, and then adding successively each numerical enhancement. When no numerical enhancement is enabled (bars "None"), only 12.50 % of the simulations converge, with an order of convergence equal to 3.22. With just friction head-loss regularization (bars "Head-loss"), 96.88 % of the simulations converge, with an order of convergence equal to 2.13. With friction head-loss and consumption regularization (bars "+ Consumption"), still 96.88 % of the simulations converge, with an order of convergence equal to 2.16. With friction head-loss regularization, consumption regularization and damping correction (bars "+ Damping"), then 100 % of the simulations converge, with an order of convergence equal to 2.04. Finally, with all numerical enhancements enabled (i.e., friction head-loss regularization, consumption regularization, damping correction and preconditioning; bars "+ Preconditioning"), still 100 % of the simulations converge with an order of convergence of 2.04. The "not-averaged" orders of convergence per network and demand multiplier are presented in appendix A.3.



(a) Ratios of converging runs (in %) (b) Mean convergence orders (unit-less)

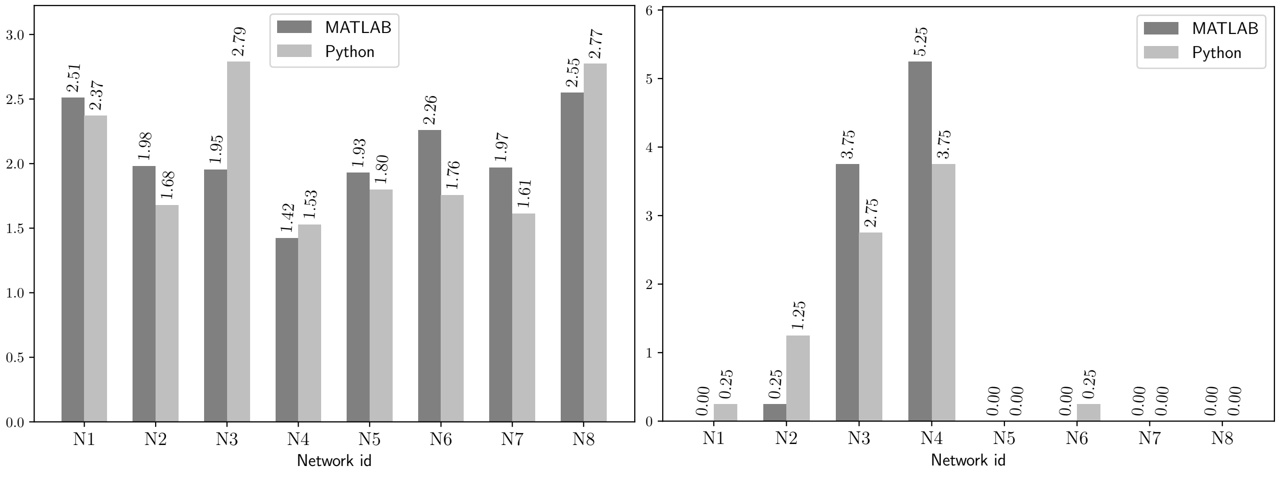
Figure 1.4: Ratio of converging runs (fig. 1.4a) and mean convergence orders (fig. 1.4b) when simulating all networks {N1, . . . , N8} with the Python simulator, enabling successively each numerical enhancement.

Globally, these metrics show that for most of the configurations (i.e., one network combined with one demand multiplier), we need to regularize the friction head-loss to reach convergence. Also, friction head-loss regularization, consumption regularization and damping correction must be enabled for the simulation of all configurations to reach convergence. If we consider only the networks and demand multipliers for which all simulations converge, then the mean convergence order is better when none of the numerical enhancements are enabled (i.e., > 3). Even so, the mean convergence order remains very good (i.e., > 2) when a part or all of the numerical enhancements are enabled. Adding preconditioning when all other numerical enhancements are already enabled does not show any improvement in these test cases. However, this does not mean that there would be no difference for other networks and configurations. Thus, since we do not observe computational overhead, we decide, for future simulations, to systematically enable all numerical enhancements, including the preconditioning.

1.3.2 Comparison of the Python simulator with the MATLAB one

When simulating each network for each demand multiplier, the maximal difference between the consumptions computed with the MATLAB and the Python simulators is equal to $9.72 \times 10^{-3}\,\mathrm{l\,s^{-1}}$. This difference is not significant. Thus, both simulators give the same result.

Globally, both Python and MATLAB simulators show mean convergence orders that are slightly less than 2 (fig. 1.5a). The mean numbers of damping corrections for each simulated network show that both simulators need corrections for almost the same networks, and that the numbers of corrections are comparable (fig. 1.5b). Finally, we see clearly that the size of the networks is not correlated with the number of needed corrections. Thus, even if there are smaller than $\{\mathrm{N5}, \ldots, \mathrm{N8}\}$, the networks $\{\mathrm{N3}, \ldots, \mathrm{N4}\}$ seem more complex to solve.



(a) Mean convergence orders

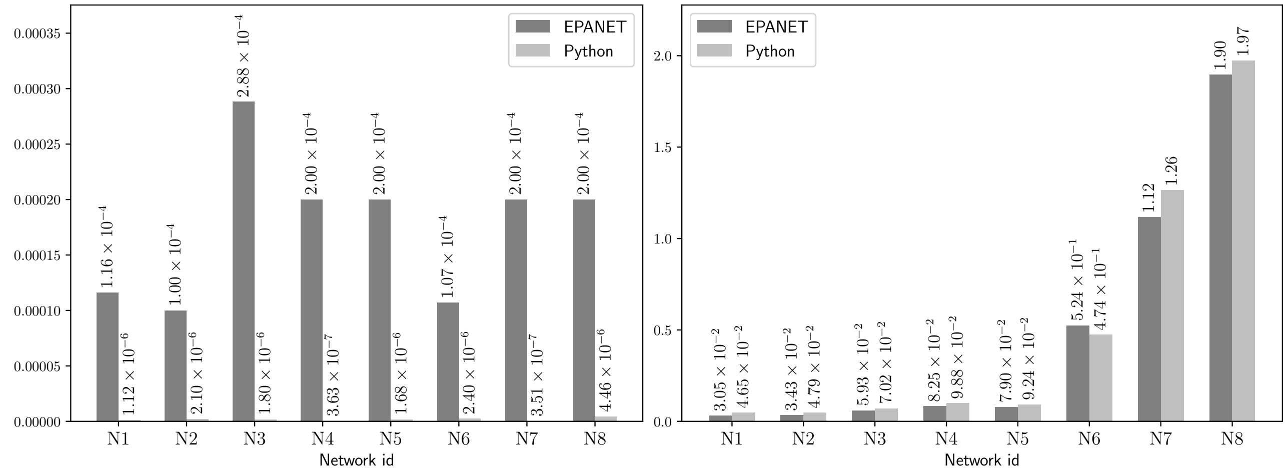
(b) Mean numbers of damping corrections

Figure 1.5: Mean convergence orders (fig. 1.5a) and numbers of needed damping corrections (fig. 1.5b), simulating each network $\{\mathrm{N1}, \ldots, \mathrm{N8}\}$ with the Python and MatLab simulators.

1.3.3 Comparison of the Python simulator with EPANET

We find that increasing the heads at sources $\boldsymbol{h_0}$ by $10\,\mathrm{mH_2O}$ and reducing the service pressure-head p_s to $1\,\mathrm{mH_2O}$ leads to fully satisfied demand at every junction of every network. With such values of $\boldsymbol{h_0}$ and p_s, the maximal difference between the consumptions computed by the Python simulator and the ones computed by EPANET is equal to $5.00 \times 10^{-5}\,\mathrm{l\,s^{-1}}$. Thus, the Python simulator gives the same results as EPANET when the demands are fully satisfied.

Even with an “ACCURACY” parameter equal to 10^{-6}, the ∞-norms of the residuals at convergence of EPANET are always greater than the ones of the Python simulator (fig. 1.6a). Thus, the Python simulator is more accurate than EPANET. We believe that this difference is due to EPANET convergence criterion, which considers only the differences between iterates of flow rates while the criterion of the Python simulator considers the differences between iterates of both flow rates and heads.



(a) Infinity norms of residuals (in $\mathrm{l\,s^{-1}}$ or $\mathrm{mH_2O}$)

(b) Simulator CPU times (in s)

Figure 1.6: Infinity norm of residuals (fig. 1.6a) and simulator elapsed CPU time (fig. 1.6b), simulating each network $\{\mathrm{N1}, \dots, \mathrm{N8}\}$ with EPANET and the Python simulator.

As expected, the elapsed CPU times of the Python simulator are slightly greater than the ones of EPANET (fig. 1.6b). This is normal because EPANET is coded entirely with the C programming language, which is compiled and thus more efficient than the Python language. But these differences remain small, probably because the CHOLMOD [19] library used by the Python simulator to compute the Cholesky decomposition of the Schur complement matrix is probably more optimized than the method used in EPANET, which is based on an older work from [48].

1.4 Conclusions

In this chapter, we designed and implemented a new Python simulator to model pressure-dependent users’ consumptions in water distribution networks (WDNs). Our simulator produces same results as the ones computed by the state-of-the-art MATLAB simulator developed by [43]. Extensive numerical experiments on 8 complex, large and realistic networks, composed of up to 19,647 pipes and 17,986 nodes, showed similar precision and efficiency for both simulators. Moreover, the performances of our simulator are close to those of EPANET in case of fully satisfied demand. Thus, we fully reached our objective.

We now have an efficient and flexible simulator of pressure-dependent user's consumption, integrated into the generic and collaborative framework OOPNET [147] dedicated to WDNs modeling and analysis. This is the first Python simulator that includes all numerical enhancements from [42, 43, 122, 130]. These enhancements were mandatory for the simulation of all tested network configurations to reach convergence. Among these enhancements, the damping correction will allow the study of more complex models, especially those for which the Jacobian of the system to solve can have sub-linear growth.

Thus, from this new mastered, stable and robust Python simulator, we can now think about studying trickier physical processes. Indeed, the complete set of tests, driven to assess the good functioning of the simulator, as well as the native properties of the Python language to make code extension easier, are good insurance to reach the next objectives of this PhD limiting the risk of software regression.

The modeling of valves and pumps belongs to the important processes that could be later integrated into our simulator. But our next goal will be rather to take into account pressure-dependent background leakage outflow rates, which are also critical processes to simulate.

Highlights

- A new simulator of pressure-dependent user's consumption in water distribution network
- Coded with Python, and based on a state-of-the-art model implemented in MATLAB
- Numerical experiments on networks composed of up to 19,647 pipes and 17,986 nodes
- Similar precision and efficiency as the ones of MATLAB and EPANET simulators
- Preliminary step for integration of new processes and first contribution to OOPNET

Chapter 2

Pressure-dependent background leakages

Abstract

In last decades, several mathematical models have been proposed to simulate background leakages in water distribution networks (WDNs). Some of these models already consider the dependence of leakages to pressure, but they still neglect the gradient of pressure along the pipes. In this chapter, new models to take into account this gradient are presented. One of them computes reference background leakage outflow rates, using a recursive algorithm that discretizes the pipes into sub-pipes until the hydraulic grade line (HGL) along each pipe converges. The other new models consist in gradually refining a state-of-the-art one. All models are then integrated into the simulator of pressure-dependent users' consumptions already developed in chapter 1, and are tested and compared on both a single leaky pipe and a WDN derived from a real leaky network. The results of this comparison show clearly the better estimations obtained from our new models of background leakage outflows when compared to the state-of-the-art one. Accurate leakage models are essential to estimate the level of leakages and, more generally, the good working order of WDNs. Thus, our new background leakage models will help in taking the best decisions for optimal functioning and rehabilitation of the WDNs. Moreover, our recursive discretization approach could be reused for other applications in WDNs, or derived to more general fields of applied mathematics and scientific computation.

Keywords:
water distribution network (WDN), background leakage, pressure-dependent model (PDM), recursive discretization method

Remark: This chapter is the adaptation of a research paper submitted to the *Mathematics and Computers in Simulation* journal[1], entitled "Modeling of pressure-dependent background leakages in water distribution networks".

2.1 Introduction

2.1.1 Leakages and background leakages in water distribution networks

When water distribution networks (WDNs) age, leakages appear, causing significant water losses. Among the leakages, some are too small to be detected by traditional acoustic equipments; they are called background leakages, or diffuse outflow rates. Even if they are small, background leakages run continuously, often for a long time, and thus contribute greatly to water losses [91]. Because drinking water is a limited resource that has to be preserved as much as possible, civil engineers and scientists work, since several decades, on detecting and reducing leakages. To do so, they always need for more accurate mathematical models and efficient simulation tools [136].

Currently, the best way to detect leakages in a WDN is to use a dual modeling approach that consists in transforming the tenuous variations of pressures induced by the leakages to equivalent but clearer flow rate variations [149]. Then, once these leakages are located, operators can repair or replace the concerned components.

Background leakages cannot be located and repaired. A solution consists in replacing all the pipes

[1] `https://www.sciencedirect.com/journal/mathematics-and-computers-in-simulation`

of the leaky sectors; but it is very expensive and would lead to a long interruption of service. Thus, many authors try to optimize the rehabilitation strategies [3, 70], and to prioritize pipe replacement [53, 100, 108].

With a much lower cost than pipe replacement, and without any service interruption, it is possible to reduce background leakages through a smart control of the pressures [86]. But, to be efficient, this control approach needs to model background leakages that depend on the pressures [55, 57, 101].

2.1.2 Modeling of pressure-dependent background leakages

[49] first proposed a pressure-dependent model (PDM) of the background leakages in water distribution networks, derived from the Torricelli's law, supposing that there is no gradient of pressure along the pipes. Like so, the lineic (i.e., per length unit) leakage outflow rate in each pipe is independent of the position along the pipe, and is computed from an average value of the pressure-head into each pipe.

Next, [55] reused the model of [49], combining it with a PDM model of users' consumptions based on the Wagner's pressure-outflow relationship (POR) [162]. Thereby, [55] run steady-state simulations of leaky WDNs derived from real networks.

Due to background leakages, the flow rates at the starts of the pipes are different than the ones at their ends. However, in the equations of [55], the computation of the friction head-loss along each pipe considers a unique value for the flow rate, then neglecting a significant loss of axial momentum along the pipe. Thus, to correct that issue, [45] proposed to add an extra resistance term in the equation of the conservation of energy.

Using a different approach, [81] proposed a slow-transient (a.k.a., unsteady-incompressible, or rigid water column) model to simulate background leakages and inertia phenomena in WDNs subject to quick variations of flows and heads. There, conversely to [45], the friction head-losses are computed by integrating the variation of the flow rates due to background leakages. But, as [45], leakages are simplified to be independent of the pressure in pipes.

All previous models of background leakages have their own advantages and drawbacks. Indeed, the models from [45] and [81] both consider the variations of the flow rates along the pipes, but they neglect the dependence of leakages to pressure. The model of [55] simulates leakages that depend on the pressure, but it neglects the variation of the flow rate along each pipe.

The model from [55] has already been tested by [57], and calibrated by [9, 101]. Thus, we choose it as the state-of-the-art one. However, this model supposes leakage outflow rates that are independent of the gradients of pressure along the pipes, and computes the friction head-losses without integrating the variation of the flow rate along each pipe.

2.1.3 Hypothesis, objectives and research strategy

We believe that taking into account the gradient of pressure along the pipes permits to model the background leakages in the WDNs more accurately, and that integrating the variation of the flow rates along the pipes prevents from neglecting a significant loss of axial momentum. Thus, our first objective is to propose new steady-state PDM models of background leakages that consider the gradients of pressure and integrate the variation of the flow rates along the pipes. This way, civil engineers and WDN managers could benefit from more accurate models to simulate WDNs.

Also, like many other authors (e.g., [5, 27, 36, 43, 62, 154]), we think that WDNs modeling needs the use of rigorous numerical methods. Thus, our second objective is to implement our new models in a generic, efficient and easy to diffuse way, so they could be easily adapted and reused for other applications in WDNs, or even a source of inspiration for more general fields of applied mathematics and scientific computation.

To achieve these two goals, we will first describe the background leakage models at the pipe scale. Then, we will extended them to the WDN scale, and adapt the Newton's method proposed in chapter 1 to solve the steady-state equations at equilibrium. Next, we will explain how we check and compare the background leakage models to each other, and how we implement our new developments. Finally, we will present our results and discuss them.

2.2 Methods

This section first presents, at the pipe scale, the state of art model from [55] and our new models of pressure-dependent background leakages. Next, it extends the models to the WDN scale, integrating them into the equations of equilibrium already defined in chapter 1. Finally, it describes the networks and method we used to check and compare the models between each other. In absence of other indications, we will adopt the same notation as the ones of chapter 1, and the same unit conventions (see section 1.2).

2.2.1 Models of background leakages at the pipe scale

This section describes the background leakage models at the scale of a leaky pipe of length ℓ.

2.2.1.1 Lineic background leakage outflow rate

We denote $x \in [0, \ell]$ the position along the leaky pipe, and p_0 and p_ℓ the pressure-heads (in $\mathrm{mH_2O}$) respectively at $x = 0$ and $x = \ell$. Then, assuming that p_0 and p_ℓ are known, [49] computes the approximated average pressure-head in the pipe as:

$$\widetilde{p} = \frac{p_0 + p_\ell}{2}. \tag{2.1}$$

$\widetilde{p}$ is considered "approximated" because it is an approximation of the exact average pressure-head defined as:

$$\overline{p} = \frac{1}{\ell} \int_0^\ell \mathrm{p}^{\mathrm{theo}}(x)\,\mathrm{d}x\,, \tag{2.2}$$

where $\mathrm{p}^{\mathrm{theo}}(x)$ is the continuous theoretical function permitting to compute the exact pressure-head at any position $x \in [0, \ell]$. Next, [49] uses the Torricelli's equation to compute an approximated lineic background leakage outflow rate (in $\mathrm{l\,s^{-1}\,m^{-1}}$) along the full pipe as:

$$\mathrm{q_{LL}}(\widetilde{p}) = \beta_L \left([\widetilde{p}]^+\right)^{\alpha_L}, \tag{2.3}$$

where α_L corresponds to the type of leakage (unit-less), β_L represents the level of degradation of the pipe (in $\mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$), and $[\widetilde{p}]^+$ refers to the positive-part of $\widetilde{p}$. Equation (2.3) can be used to model both local and background leakage outflow rates. For background leakages, $0.5 < \alpha_L \leq 2.5$ [91, 102], and $10^{-7} \leq \beta_L \leq 10^{-1}$ [9, 101]. We will denote hereafter $\widetilde{q_{LL}}$ the value calculated by the function (2.3) from $\widetilde{p}$.

2.2.1.2 Theoretical model of background leakage, flow rate and friction head-loss

We suppose that the theoretical continuous function $\mathrm{p}^{\mathrm{theo}}(x)$ permitting to compute the exact pressure-head at any position $x \in [0, \ell]$ is perfectly known. Then, we can extend eq. (2.3) to

$$\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x) = \beta_L \left(\left[\mathrm{p}^{\mathrm{theo}}(x) \right]^{+} \right)^{\alpha_L} . \tag{2.4}$$

Also, we suppose that the flow rate at the middle of the pipe, denoted $q_{0.5}$, is known too. Then, using eq. (2.4), we can compute the flow rate at any $x \in [0, \ell]$ as:

$$\mathrm{q}^{\mathrm{theo}}(x) = q_{0.5} - \int_{\ell/2}^{x} \mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(y)\, \mathrm{d}y\, . \tag{2.5}$$

Remark: denoting q_0 and q_ℓ the flow rates at respectively $x = 0$ and $x = \ell$, and supposing that either q_0 or q_ℓ is known, then we could also define $\mathrm{q}^{\mathrm{theo}}(x)$ as a function of q_0 or q_ℓ rather than a function of $q_{0.5}$. But the use of $q_{0.5}$ leads to symmetrical and more generic formulation of $\mathrm{q}^{\mathrm{theo}}(x)$. We therefore chose to define $\mathrm{q}^{\mathrm{theo}}(x)$ as a function of $q_{0.5}$ (see eq. (2.5)).

Finally, we can extend the Hazen-Williams model [167] to take into account the varying flow rate along the pipe, and compute the friction head-loss from position 0 to position $x \in [0, \ell]$ as:

$$\xi_{\mathrm{f}}^{\mathrm{theo}}(x) = f \int_{0}^{x} \mathrm{q}^{\mathrm{theo}}(y) |\mathrm{q}^{\mathrm{theo}}(y)|^{\gamma_{HW}-1}\, \mathrm{d}y\, . \tag{2.6}$$

Unfortunately, it is very difficult to determine the function $\mathrm{p}^{\mathrm{theo}}(x)$, because it would need many experimental measures. As well, the functions $\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x)$, $\mathrm{q}^{\mathrm{theo}}(x)$ and $\xi_{\mathrm{f}}^{\mathrm{theo}}(x)$ cannot be found easily. Thus, we propose in next sections 2.2.1.3 and 2.2.1.4 to approximate the continuous functions $\{\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x), \mathrm{q}^{\mathrm{theo}}(x), \xi_{\mathrm{f}}^{\mathrm{theo}}(x)\}$ by different mathematical models, supposing that only p_0, p_ℓ and $q_{0.5}$ are known.

2.2.1.3 State-of-the-art background leakage model

We choose as state-of-the-art background leakage model the one initially proposed by [55], and already validated by [55, 57]. In this model, denoted hereafter M0, [55] suppose that the lineic leakage outflow rate and the flow rate are invariant along the pipe.

To construct M0, [55] reused the function (2.3) from [49]. Then, $\forall x \in [0, \ell]$, they computed the lineic background leakage outflow rate at x as:

$$\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}(x) = \widetilde{q_{LL}}, \tag{2.7}$$

and the flow rate as:

$$\mathrm{q}^{\mathrm{M0}}(x) = q_{0.5}\, . \tag{2.8}$$

Finally, [55] used the Hazen-Williams model [167] to compute the friction head-loss from position 0 to position $x \in [0, \ell]$ as:

$$\xi_{\mathrm{f}}^{\mathrm{M0}}(x) = \int_0^x \varphi^{\mathrm{M0}}(y)\,\mathrm{d}y\,, \tag{2.9}$$

where $\varphi^{\mathrm{M0}}(y)$ is the unitary friction head-loss function, defined, $\forall y \in [0, \ell]$, as:

$$\varphi^{\mathrm{M0}}(y) = f \mathrm{q}^{\mathrm{M0}}(y) |\mathrm{q}^{\mathrm{M0}}(y)|^{\gamma_{HW}-1} = f q_{0.5} |q_{0.5}|^{\gamma_{HW}-1}. \tag{2.10}$$

Thus, eq. (2.9) can be simplified to:

$$\xi_{\mathrm{f}}^{\mathrm{M0}}(x) = f q_{0.5}\, |q_{0.5}|^{\gamma_{HW}-1}\, x. \tag{2.11}$$

2.2.1.4 Reference model from recursive discretization

We propose here a new model based on the recursive discretization of the pipe into sub-pipes, until the difference between the hydraulic grade lines (HGLs) along the pipe computed at two consecutive iterations becomes small enough. In this new model, the background leakage outflow rates, the flow rates and the friction head-losses in the initial undiscretized pipe and in each discretized sub-pipe are computed with the functions of the model M0 (see section 2.2.1.3). Each iteration of the discretization algorithm corresponds to a level of discretization of the pipe.

The functions (2.7) and (2.8) are piecewise constant per sub-pipe. They lead to a good numerical approximation of the continuous theoretical functions (2.4) and (2.5) providing that the pipe is discretized in enough sub-pipes; a proof of this statement is proposed in appendix B.1. Since $\xi_{\mathrm{f}}^{\mathrm{M0}}$ depends linearly on q^{M0}, using $\xi_{\mathrm{f}}^{\mathrm{M0}}$ in each sub-pipe permits to approximate the continuous theoretical function $\xi_{\mathrm{f}}^{\mathrm{theo}}$ as well.

We will now describe the discretization algorithm through a simple example, iteration by iteration. To do so, we first denote q_x, p_x and h_x respectively the flow rate, the pressure-head and the head at the position x along the pipe, $\forall x \in [0, \ell]$. Also, $\forall \{x_1, x_2\} \in [0, \ell] \times [0, \ell]$, $x_1 \leq x_2$, we denote:

$$\widetilde{p_{x_1 x_2}} = \frac{p_{x_1} + p_{x_2}}{2}$$

and

$$\widetilde{q_{LLx_1x_2}} = \beta_L \big([\widetilde{p_{x_1x_2}}]^+ \big)^{\alpha_L}$$

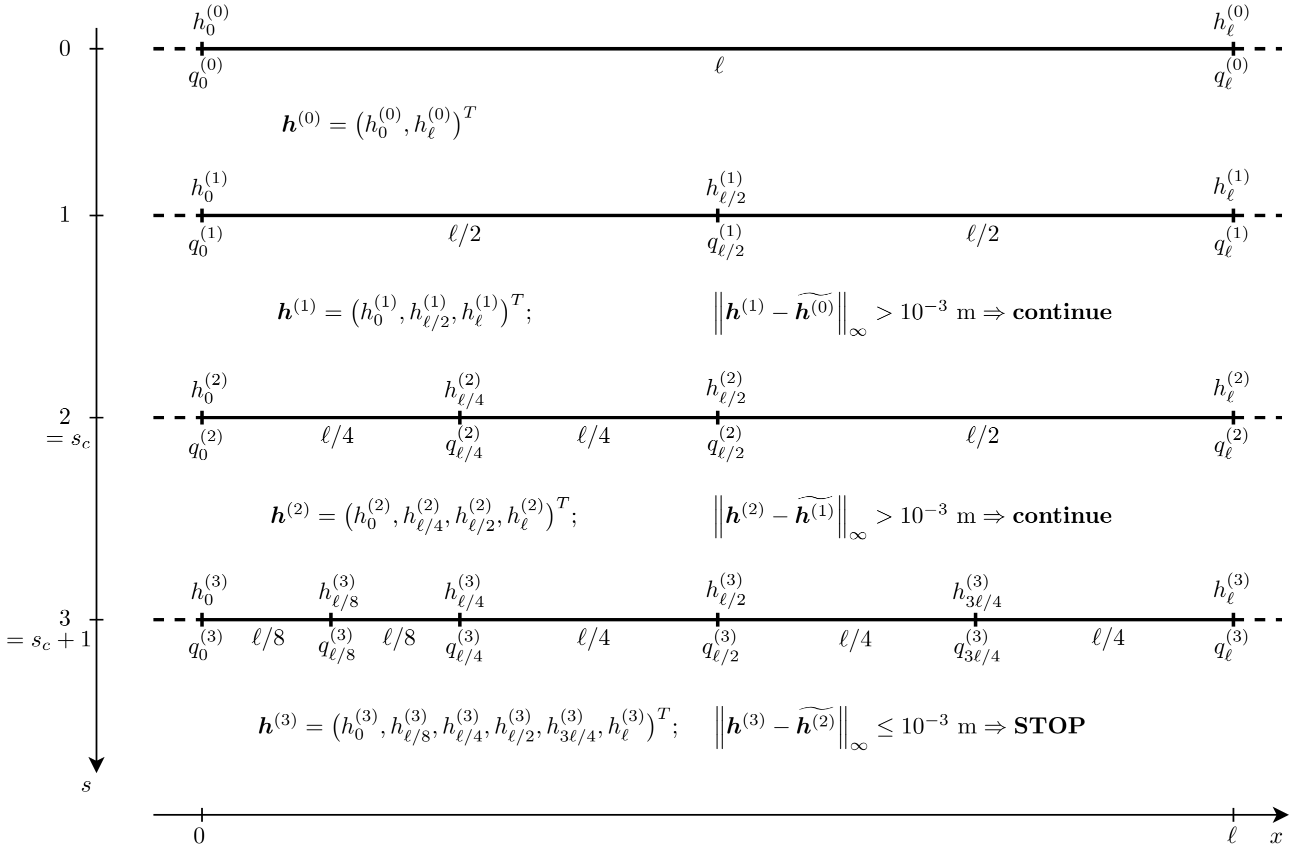
respectively the approximated average pressure-head and the approximated lineic background leakage outflow rate in the interval $[x_1, x_2]$. Finally, we define:

$$q_{L[x_1,x_2]} = \widetilde{q_{LLx_1x_2}}\,(x_2 - x_1)$$

and

$$\xi_{f[x_1,x_2]} = \xi_{\mathrm{f}}^{\mathrm{M0}}(x_2) - \xi_{\mathrm{f}}^{\mathrm{M0}}(x_1)$$

respectively the background leakage outflow rate and the friction head-loss cumulated from x_1 to x_2. We explain in next paragraphs what is done at each iteration of the discretization algorithm, following the example in fig. 2.1.



$h_0^{(0)}$
$h_\ell^{(0)}$
0
$q_0^{(0)}$
ℓ
$q_\ell^{(0)}$
$\boldsymbol{h}^{(0)} = \left(h_0^{(0)}, h_\ell^{(0)}\right)^T$
1
$h_0^{(1)}$ $h_{\ell/2}^{(1)}$ $h_\ell^{(1)}$
$q_0^{(1)}$ $\ell/2$ $q_{\ell/2}^{(1)}$ $\ell/2$ $q_\ell^{(1)}$
$\boldsymbol{h}^{(1)} = \left(h_0^{(1)}, h_{\ell/2}^{(1)}, h_\ell^{(1)}\right)^T;$
$\left\|\boldsymbol{h}^{(1)} - \widetilde{\boldsymbol{h}^{(0)}}\right\|_\infty > 10^{-3}$ m ⇒ **continue**
2 $= s_c$
$h_0^{(2)}$ $h_{\ell/4}^{(2)}$ $h_{\ell/2}^{(2)}$ $h_\ell^{(2)}$
$q_0^{(2)}$ $\ell/4$ $q_{\ell/4}^{(2)}$ $\ell/4$ $q_{\ell/2}^{(2)}$ $\ell/2$ $q_\ell^{(2)}$
$\boldsymbol{h}^{(2)} = \left(h_0^{(2)}, h_{\ell/4}^{(2)}, h_{\ell/2}^{(2)}, h_\ell^{(2)}\right)^T;$
$\left\|\boldsymbol{h}^{(2)} - \widetilde{\boldsymbol{h}^{(1)}}\right\|_\infty > 10^{-3}$ m ⇒ **continue**
3 $= s_c + 1$
$h_0^{(3)}$ $h_{\ell/8}^{(3)}$ $h_{\ell/4}^{(3)}$ $h_{\ell/2}^{(3)}$ $h_{3\ell/4}^{(3)}$ $h_\ell^{(3)}$
$q_0^{(3)}$ $\ell/8$ $q_{\ell/8}^{(3)}$ $\ell/8$ $q_{\ell/4}^{(3)}$ $\ell/4$ $q_{\ell/2}^{(3)}$ $\ell/4$ $q_{3\ell/4}^{(3)}$ $\ell/4$ $q_\ell^{(3)}$
$\boldsymbol{h}^{(3)} = \left(h_0^{(3)}, h_{\ell/8}^{(3)}, h_{\ell/4}^{(3)}, h_{\ell/2}^{(3)}, h_{3\ell/4}^{(3)}, h_\ell^{(3)}\right)^T;$
$\left\|\boldsymbol{h}^{(3)} - \widetilde{\boldsymbol{h}^{(2)}}\right\|_\infty \leq 10^{-3}$ m ⇒ **STOP**
s
0
ℓ
x

Figure 2.1: Illustration to describe the functioning of the discretization algorithm when applied on a leaky pipe.

At iteration 0 (i.e., initial state), the conservation of energy and mass along the undiscretized pipe leads to the system of equations:

$$\begin{pmatrix} h_0^{(0)} - h_\ell^{(0)} - \xi_{f[0,\ell]}^{(0)} \\ q_0^{(0)} - q_\ell^{(0)} - q_{L[0,\ell]}^{(0)} \end{pmatrix} = \boldsymbol{0}. \tag{2.12}$$

Then, the discretization algorithm solves the system (2.12) to compute the HGL $\boldsymbol{h}^{(0)} = \left(h_0^{(0)}, h_\ell^{(0)}\right)^T$ at positions $\boldsymbol{x}^{(0)} = \left(0, \ell\right)^T$, and goes to iteration 1.

At iteration 1, the algorithm discretizes the pipe in 2 sub-pipes of equal length $\ell/2$. Except for their length, the new sub-pipes have exactly the same characteristics as the initial undiscretized pipe: same diameter, roughness, leakage type and level of degradation. The conservation of energy and mass along the sub-pipes leads to the system of equations:

$$\begin{pmatrix} h_0^{(1)} - h_{\ell/2}^{(1)} - \xi_{f[0,\ell/2]}^{(1)} \\ h_{\ell/2}^{(1)} - h_{\ell}^{(1)} - \xi_{f[\ell/2,\ell]}^{(1)} \\ \\ q_0^{(1)} - q_{\ell/2}^{(1)} - q_{L[0,\ell/2]}^{(1)} \\ q_{\ell/2}^{(1)} - q_{\ell}^{(1)} - q_{L[\ell/2,\ell]}^{(1)} \end{pmatrix} = \boldsymbol{0}. \tag{2.13}$$

The discretization algorithm solves the system (2.13) to compute $\boldsymbol{h}^{(1)} = \left(h_0^{(1)}, h_{\ell/2}^{(1)}, h_{\ell}^{(1)}\right)^T$ at positions $\boldsymbol{x}^{(1)} = \left(0, \ell/2, \ell\right)^T$. Then, it tests the convergence with the criterion:

$$\|\boldsymbol{h}^{(1)} - \widetilde{\boldsymbol{h}^{(0)}}\|_\infty \leq 10^{-3}\,\mathrm{mH_2O}, \tag{2.14}$$

where the vector $\widetilde{\boldsymbol{h}^{(0)}}$ contains the values of $\boldsymbol{h}^{(0)}$ interpolated at the positions $\boldsymbol{x}^{(1)}$ using a Piecewise Cubic Hermite Interpolating Polynomial (PCHIP).

Supposing that the average pressure-head in the tested networks is equal to $30\,\mathrm{mH_2O}$, then the absolute tolerance $10^{-3}\,\mathrm{mH_2O}$ represents a precision of 3 ‱, which is ~ 3 times less than the best precision expected when measuring pressures physically in a WDN (i.e., ~ 1 ‰ [44, p. 1]). In the example of fig. 2.1, the criterion (2.14) is satisfied for the second sub-pipe but not for the first one; then, the algorithm continues to the next iteration.

At iteration 2, the algorithm discretizes only the first sub-pipe (i.e., the sub-pipe where the convergence criterion (2.14) was not satisfied) in two sub-pipes of equal length $\ell/4$. The conservation of energy and mass along the 3 resulting sub-pipes leads to:

$$\begin{pmatrix} h_0^{(2)} - h_{\ell/4}^{(2)} - \xi_{f[0,\ell/4]}^{(2)} \\ h_{\ell/4}^{(2)} - h_{\ell/2}^{(2)} - \xi_{f[\ell/4,\ell/2]}^{(2)} \\ h_{\ell/2}^{(2)} - h_{\ell}^{(2)} - \xi_{f[\ell/2,\ell]}^{(2)} \\ \\ q_0^{(2)} - q_{\ell/4}^{(2)} - q_{L[0,\ell/4]}^{(2)} \\ q_{\ell/4}^{(2)} - q_{\ell/2}^{(2)} - q_{L[\ell/4,\ell/2]}^{(2)} \\ q_{\ell/2}^{(2)} - q_{\ell}^{(2)} - q_{L[\ell/2,\ell]}^{(2)} \end{pmatrix} = \boldsymbol{0}. \tag{2.15}$$

The algorithm solves the system (2.15) to compute $\boldsymbol{h}^{(2)} = (h_0^{(2)}, h_{\ell/4}^{(2)}, h_{\ell/2}^{(2)}, h_{\ell}^{(2)})^T$ at positions $\boldsymbol{x}^{(2)} = (0, \ell/4, \ell/2, \ell)^T$. Then, it tests the convergence as:

$$\|\boldsymbol{h}^{(2)} - \widetilde{\boldsymbol{h}^{(1)}}\|_\infty \leq 10^{-3}\,\mathrm{mH_2O}, \tag{2.16}$$

where the vector $\widetilde{\boldsymbol{h}^{(1)}}$ contains the values of $\boldsymbol{h}^{(1)}$ interpolated at the positions $\boldsymbol{x}^{(2)}$ using the same method as at iteration 1. The criterion (2.16) is not satisfied for the first and the third sub-pipes; then, the algorithm continues to the next iteration.

At iteration 3, the algorithm discretizes the first sub-pipe in two sub-pipes of length $\ell/8$, and the third sub-pipe in two sub-pipes of length $\ell/4$. The conservation of energy and mass along the 5 resulting sub-pipes leads to:

$$\begin{pmatrix} h_0^{(3)} - h_{\ell/8}^{(3)} - \xi_{f[0,\ell/8]}^{(3)} \\ h_{\ell/8}^{(3)} - h_{\ell/4}^{(3)} - \xi_{f[\ell/8,\ell/4]}^{(3)} \\ h_{\ell/4}^{(3)} - h_{\ell/2}^{(3)} - \xi_{f[\ell/4,\ell/2]}^{(3)} \\ h_{\ell/2}^{(3)} - h_{3\ell/4}^{(3)} - \xi_{f[\ell/2,3\ell/4]}^{(3)} \\ h_{3\ell/4}^{(3)} - h_{\ell}^{(3)} - \xi_{f[3\ell/4,\ell]}^{(3)} \\ \\ q_0^{(3)} - q_{\ell/8}^{(3)} - q_{L[0,\ell/8]}^{(3)} \\ q_{\ell/8}^{(3)} - q_{\ell/4}^{(3)} - q_{L[\ell/8,\ell/4]}^{(3)} \\ q_{\ell/4}^{(3)} - q_{\ell/2}^{(3)} - q_{L[\ell/4,\ell/2]}^{(3)} \\ q_{\ell/2}^{(3)} - q_{3\ell/4}^{(3)} - q_{L[\ell/2,3\ell/4]}^{(3)} \\ q_{3\ell/4}^{(3)} - q_{\ell}^{(3)} - q_{L[3\ell/4,\ell]}^{(3)} \end{pmatrix} = \boldsymbol{0}. \tag{2.17}$$

The algorithm solves the system (2.17) to compute $\boldsymbol{h}^{(3)} = \left(h_0^{(3)}, h_{\ell/8}^{(3)}, h_{\ell/4}^{(3)}, h_{\ell/2}^{(3)}, h_{3\ell/4}^{(3)}, h_{\ell}^{(3)}\right)^T$ at positions $\boldsymbol{x}^{(3)} = \left(0, \ell/8, \ell/4, \ell/2, 3\ell/4, \ell\right)^T$. Then, it tests the convergence with the criterion:

$$\|\boldsymbol{h}^{(1)} - \widetilde{\boldsymbol{h}^{(2)}}\|_\infty \leq 10^{-3}\,\mathrm{mH_2O}, \tag{2.18}$$

where the vector $\widetilde{\boldsymbol{h}^{(2)}}$ contains the values of $\boldsymbol{h}^{(2)}$ interpolated at the positions $\boldsymbol{x}^{(3)}$ using the same method as at iterations 1 and 2. The criterion (2.18) is satisfied; then, the algorithm stops. The differences between the HGL computed at iterations 2 and 3 are less than $10^{-3}\,\mathrm{mH_2O}$. Thus, the discretization algorithm converged at iteration $s_c = 2$.

At iteration $s_c + 1 = 3$, we computed the values of h_x and q_x $\forall x \in \{0, \ell/8, \ell/4, \ell/2, 3\ell/4, \ell\}$. From each h_x, we can also compute the lineic background leakage outflow rate q_{LLx} as:

$$q_{LLx} = \beta_L\left([h_x - u_x]^+\right)^{\alpha_L}, \tag{2.19}$$

where u_x is the elevation at the position x along the undiscretized pipe. Finally, from all values of q_{LLx}, q_x and h_x, we can determine by PCHIP interpolation the functions $\mathrm{q_{LL}}(x)$, $\mathrm{q}(x)$ and $\xi_\mathrm{f}(x)$, $x \in [0, \ell]$. Since these functions compute values that are very close to the ones we would obtain with the theoretical model of section 2.2.1.2, we will consider them as the reference ones, and denote them $\mathrm{q}_\mathrm{LL}^\mathrm{Ref}(x)$, $\mathrm{q}^\mathrm{Ref}(x)$ and $\xi_\mathrm{f}^\mathrm{Ref}(x)$ hereafter.

The convergence criterion, defined at iterations 1, 2 and 3 by respectively eqs. (2.14), (2.16) and (2.18), can be generalized to any iteration $s > 0$ as:

$$\|\boldsymbol{h}^{(s-1)} - \widetilde{\boldsymbol{h}^{(s)}}\|_\infty \leq 10^{-3}\,\mathrm{mH_2O}, \tag{2.20}$$

where the vector $\widetilde{\boldsymbol{h}^{(s-1)}}$ contains the PCHIP interpolation of $\boldsymbol{h}^{(s-1)}$ at the positions $\boldsymbol{x}^{(s)}$.

2.2.1.5 Lineic leakage outflow rate invariant along the pipe but affine flow rate

We propose here a new model, denoted M1, obtained by refining the state-of-the-art one M0. Indeed, we consider in M1, as for model M0, that the lineic leakage outflow rate is invariant along the pipe; but we now also suppose, conversely to M0, that for M1 the flow rate is affine along the pipe, as initially proposed by [81].

To do so, denoting

$$\mathrm{q}_{\mathrm{LL}}^{\mathrm{M1}}(x) = q_{LL}^{\mathrm{M1}} = \widetilde{q_{LL}}, \tag{2.21}$$

we compute the flow rate at any $x \in [0, \ell]$ as:

$$\mathrm{q}^{\mathrm{M1}}(x) = q_{0.5} - q_{LL}^{\mathrm{M1}} \left(x - \frac{\ell}{2} \right). \tag{2.22}$$

Then, denoting $q_0^{\mathrm{M1}} = \mathrm{q}^{\mathrm{M1}}(0)$, we compute the friction head-loss from position 0 to position $x \in [0, \ell]$ as:

$$\xi_{\mathrm{f}}^{\mathrm{M1}}(x) = \begin{cases} \dfrac{1}{(\gamma_{HW} + 1)\, q_{LL}^{\mathrm{M1}}} \left(q_0^{\mathrm{M1}} \varphi_0^{\mathrm{M1}} - \mathrm{q}^{\mathrm{M1}}(x) \varphi^{\mathrm{M1}}(x) \right) & \text{if } q_{LL}^{\mathrm{M1}} \neq 0 \\ \text{eq. (2.11)} & \text{otherwise,} \end{cases} \tag{2.23}$$

where:

$$\varphi^{\mathrm{M1}}(x) = f \mathrm{q}^{\mathrm{M1}}(x) |\mathrm{q}^{\mathrm{M1}}(x)|^{\gamma_{HW} - 1} \tag{2.24}$$

and

$$\varphi_0^{\mathrm{M1}} = \varphi^{\mathrm{M1}}(0).$$

Equation (2.23) can be simplified to:

$$\xi_{\mathrm{f}}^{\mathrm{M1}}(x) = \begin{cases} \dfrac{f}{(\gamma_{HW} + 1)\, q_{LL}^{\mathrm{M1}}} \left(|q_0^{\mathrm{M1}}|^{\gamma_{HW} + 1} - |\mathrm{q}^{\mathrm{M1}}(x)|^{\gamma_{HW} + 1} \right) & \text{if } q_{LL}^{\mathrm{M1}} \neq 0 \\ \text{eq. (2.11)} & \text{otherwise.} \end{cases} \tag{2.25}$$

2.2.1.6 Affine lineic leakage outflow rate

The next new model, denoted M2, is a refining of model M1. Indeed, in M2, we now consider that the lineic leakage outflow rate is affine along the pipe. M2 is the first background leakage model that takes into account the gradient of pressure along the pipe without discretization.

To define M2, we first denote $q_{LL0} = \beta_L([p_0]^+)^{\alpha_L}$ and $q_{LL\ell} = \beta_L([p_\ell]^+)^{\alpha_L}$, and we choose $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}(0) = q_{LL0}$ and $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}(\ell) = q_{LL\ell}$. Next, we compute the lineic leakage outflow rate at any $x \in [0, \ell]$ by linear interpolation as:

$$\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}(x) = \frac{q_{LL\ell} - q_{LL0}}{\ell} x + q_{LL0}. \tag{2.26}$$

Subsequently, denoting $\widehat{\mathrm{x}}(x) = (x + \ell/2)/2$ and $\widehat{\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}}(x) = \mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}} \circ \widehat{\mathrm{x}}(x)$, we compute the flow rate at any $x \in [0, \ell]$ as:

$$\mathrm{q}^{\mathrm{M2}}(x) = q_{0.5} - \widehat{\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}}(x) \left(x - \frac{\ell}{2} \right). \tag{2.27}$$

Finally, we compute the friction head-loss from position 0 to position $x \in [0, \ell]$ using a Newton-Cotes formula of degree 2, as:

$$\xi_{\mathrm{f}}^{\mathrm{M2}}(x) = \frac{x}{6}\Big(\varphi^{\mathrm{M2}}(0) + 4\,\varphi^{\mathrm{M2}}(x/2) + \varphi^{\mathrm{M2}}(x)\Big), \tag{2.28}$$

where:

$$\varphi^{\mathrm{M2}}(y) = f\mathrm{q}^{\mathrm{M2}}(y)|\mathrm{q}^{\mathrm{M2}}(y)|^{\gamma_{HW}-1}. \tag{2.29}$$

A Newton-Cotes formula of degree 2 allows the exact integration of polynomials of degree 3. Thus, it is accurate enough to model the slope (i.e., 1st derivative estimate) and curvature (i.e., 2nd derivative estimate) of the friction head-loss function.

We need to use a quadrature formula to integrate the unitary friction head-loss function (2.29) because there does not exist any elementary antiderivative of (2.29). A proof of this statement is given in appendix B.2.

2.2.1.7 Pseudo-quadratic lineic leakage outflow rate

Finally, the last new model that we propose and denote M3 is a refining of model M2. Indeed, in M3, we consider, as in M2, that the lineic leakage outflow rate depends on $q_{LL0} = \beta_L\big([p_0]^+\big)^{\alpha_L}$ and $q_{LL\ell} = \beta_L\big([p_\ell]^+\big)^{\alpha_L}$; but, conversely to M2, M3 now also depends on $\widetilde{q_{LL}} = \beta_L\big([(p_0 + p_\ell)/2]^+\big)^{\alpha_L}$. Like so, model M3 takes into account the gradient of pressure along the pipe, using the three values of lineic leakage outflow rates q_{LL0}, $q_{LL\ell}$ and $\widetilde{q_{LL}}$.

To define M3, we denote $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(x)$ the quadratic polynomial defined $\forall x \in [0, \ell]$ and such that $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(0) = q_{LL0}$, $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(\ell/2) = \widetilde{q_{LL}}$ and $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(\ell) = q_{LL\ell}$. Then, after identification of the coefficients, we have:

$$\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(x) = \frac{2\,(q_{LL0} - 2\,\widetilde{q_{LL}} + q_{LL\ell})}{\ell^2}\,x^2 - \frac{3\,q_{LL0} - 4\,\widetilde{q_{LL}} + q_{LL\ell}}{\ell}\,x + q_{LL0}. \tag{2.30}$$

Next, simply replacing $\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(y)$ by $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(y)$ in eq. (2.5), we compute the flow rate at any $x \in [0, \ell]$ as:

$$\mathrm{q}^{\mathrm{M3}}(x) = q_{0.5} + \left(\frac{\ell}{2} - x\right)\left(\frac{2\,(q_{LL0} - 2\,\widetilde{q_{LL}} + q_{LL\ell})}{3\,\ell^2}\left(x^2 + x\frac{\ell}{2} + \frac{\ell^2}{4}\right)\right. \tag{2.31}$$

$$\left. - \frac{3\,q_{LL0} - 4\,\widetilde{q_{LL}} + q_{LL\ell}}{2\,\ell}\left(x + \frac{\ell}{2}\right) + q_{LL0}\right). \tag{2.32}$$

Finally, using the same method as for model M2, we compute the friction head-loss from position 0 to position $x \in [0, \ell]$ as:

$$\xi_{\mathrm{f}}^{\mathrm{M3}}(x) = \frac{x}{6}\Big(\varphi^{\mathrm{M3}}(0) + 4\,\varphi^{\mathrm{M3}}(x/2) + \varphi^{\mathrm{M3}}(x)\Big), \tag{2.33}$$

where

$$\varphi^{\mathrm{M3}}(y) = f\mathrm{q}^{\mathrm{M3}}(y)|\mathrm{q}^{\mathrm{M3}}(y)|^{\gamma_{HW}-1}. \tag{2.34}$$

Remark: the use of $\widetilde{q_{LL}}$ to compute $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(x)$ does not make $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(x)$ of a full higher degree compared to $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}(x)$. Indeed, all of $\{q_{LL0}, \widetilde{q_{LL}}, q_{LL\ell}\}$ are computed using only the two pressure-heads $\{p_0, p_\ell\}$. Hence the qualifying of pseudo-quadratic for M3.

2.2.2 Modeling of background leakages at WDN scale

This section shows how to integrate the leakage models presented in sections 2.2.1.3 to 2.2.1.7 into the equations of equilibrium defined in chapter 1.

2.2.2.1 Equations of equilibrium in a WDN

We reuse the notation from section 1.2. However, rather than flow rates $\boldsymbol{q}$ at indeterminate positions along the pipes (this position didn't matter in section 1.2 because $\boldsymbol{q}$ was constant along the pipes), the unknown flow rates are now the ones at the middle of the pipes, denoted $\boldsymbol{q_{0.5}} = \left(q_{0.5,1}, \dots, q_{0.5,n_p}\right)^T \in \mathbb{R}^{n_p}$. Also, we now denote the positive and the negative parts of the incidence matrix $\boldsymbol{A}$ as respectively $\boldsymbol{A}^+$ and $\boldsymbol{A}^-$, and $\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0})$, $\mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0})$ and $\boldsymbol{\xi}_\mathbf{f}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0})$ the vector functions of $\mathbb{R}^{n_p \times n_N}$ to $\mathbb{R}^{n_p}$, defined as $\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) = \boldsymbol{q_0}$, $\mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) = \boldsymbol{q_\ell}$ and $\boldsymbol{\xi}_\mathbf{f}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) = \boldsymbol{\xi_f}$. $\forall k \in \{1, \dots, n_p\}$, $q_{0,k} = \mathrm{q}_k(0)$, $q_{\ell,k} = \mathrm{q}_k(\ell_k)$ and $\xi_{f,k} = \xi_{\mathrm{f},k}(\ell_k)$ are computed using one of the models $\{\mathrm{M0}, \dots, \mathrm{M3}\}$ described in sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7. $\boldsymbol{h_N}$ represent the vector of heads at all nodes of the network.

Then, to find the unknown middle flow rates in pipes $\boldsymbol{q_{0.5}}$ and heads at junctions $\boldsymbol{h}$ in the WDN at steady-state, we now need to solve the non-linear system of equations:

$$\boldsymbol{\rho}(\boldsymbol{q_{0.5}}, \boldsymbol{h}) = \begin{pmatrix} \boldsymbol{\xi}_\mathbf{f}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \boldsymbol{A}^T\boldsymbol{h} - \boldsymbol{A_0}^T\boldsymbol{h_0} \\ \boldsymbol{A}^- \, \mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \boldsymbol{A}^+ \, \mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \mathbf{c}(\boldsymbol{h}) \end{pmatrix} = \boldsymbol{0}, \tag{2.35}$$

where $\boldsymbol{A_0}$ is the incidence matrix reduced to all source nodes of the network, $\boldsymbol{h_0}$ is the vector of heads at sources, and $\mathbf{c}(\boldsymbol{h})$ is the vector function to compute the users' consumptions at all junctions using the Wagner's POR (all these notations were already defined in chapter 1). Also,

$$\boldsymbol{\xi}_\mathbf{f}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \boldsymbol{A}^T\boldsymbol{h} - \boldsymbol{A_0}^T\boldsymbol{h_0} = \boldsymbol{\rho_e} \tag{2.36}$$

represent the energy residuals in pipes, and

$$\boldsymbol{A}^- \, \mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \boldsymbol{A}^+ \, \mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \mathbf{c}(\boldsymbol{h}) = \boldsymbol{\rho_m} \tag{2.37}$$

the mass residuals at junctions. The use of matrices $\{\boldsymbol{A}^+, \boldsymbol{A}^-\}$ and flow rates $\{\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}), \mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0})\}$, needed to ensure the mass conservation at junctions when simulating PDM background leakages, represents a new formulation of the equations of equilibrium.

2.2.2.2 Newton's method

We use the same Newton's method as in chapter 1. However, the sub-blocks $\{\mathbf{J_{11}}, \mathbf{J_{12}}, \mathbf{J_{21}}, \mathbf{J_{22}}\}$ of the Jacobian matrix function $\mathbf{J}$ of the system (2.35), defined in chapter 1 by (1.16), are now defined as:

$$\begin{aligned} \mathbf{J_{11}} &= \frac{\partial \boldsymbol{\xi}_\mathbf{f}}{\partial \boldsymbol{q_{0.5}}} & \mathbf{J_{12}} &= \frac{\partial \boldsymbol{\xi}_\mathbf{f}}{\partial \boldsymbol{h}} - \boldsymbol{A}^T \\ \mathbf{J_{21}} &= -\left(\boldsymbol{A}^+ - \boldsymbol{A}^-\right) = -\boldsymbol{A} & \mathbf{J_{22}} &= \boldsymbol{A}^- \frac{\partial \mathbf{q_\ell}}{\partial \boldsymbol{h}} - \boldsymbol{A}^+ \frac{\partial \mathbf{q_0}}{\partial \boldsymbol{h}} - \frac{\partial \mathbf{c}}{\partial \boldsymbol{h}}. \end{aligned} \tag{2.38}$$

Also, contrary to chapter 1, it is not possible here to use a Cholesky factorization to compute the descent direction $\boldsymbol{\delta}_{\boldsymbol{h}}^{(m)}$ on the heads at junctions $\boldsymbol{h}^{(m)}$ at each Newton iteration m (see eq. (1.22) in section 1.2.4.2), because the Schur complement $\boldsymbol{S}^{(m)}$ (defined by eq. (1.23)) is not symmetric anymore. Thus, the most suitable method to compute $\boldsymbol{\delta}_{\boldsymbol{h}}^{(m)}$ is now to use a direct sparse LU factorization, as long as $\boldsymbol{S}^{(m)}$ is invertible.

2.2.3 Extension of the discretization algorithm to the WDN scale

At section 2.2.1.4, we proposed the new reference model "Ref", based on the recursive discretization of a pipe into sub-pipes until the difference between the HGLs of two consecutive discretization levels becomes small enough.

To extend the algorithm to the scale of a whole WDN, we simply apply, at each iteration $s > 0$ of the discretization algorithm, the following procedure:

1. iterate over each pipe $k \in \{1, \ldots, n_p\}$ of the initial undiscretized WDN, to look if k needs, according to the convergence criterion (2.20), to be discretized more,
2. for each (sub-)pipe that needs to be (re-)discretized: replace the (sub-)pipe by 2 (sub-)sub-pipes connected by a new intermediate junction node; intermediate junction nodes have no demand, and their elevations are computed by linear interpolation from the elevations at the extremities of the (sub-)pipes,
3. and solve the whole new discretized WDN at once to obtain the new flow rates and HGL along each pipe k,

until the convergence criterion (2.20) becomes satisfied $\forall k \in \{1, \ldots, n_p\}$. Then, at the last iteration, the heads computed at the extremities of the undiscretized pipes are the reference heads. Also, in each pipe k, the flow rates along k obtained at the last iteration permit to compute, by PCHIP interpolation, the reference flow rate at the middle of k.

2.2.4 Sources of instabilities

As in chapter 1, different sources of instabilities can hinder the convergence of the Newton's method described in section 2.2.2.2. We present them below, along with numerical enhancements to overcome them. These numerical enhancements are fully described in appendices A.1 and B.3.

2.2.4.1 Pipes with zero flow rate

As in chapter 1, we need to compute $\left(\boldsymbol{J_{11}}\right)^{-1}$ at each iteration of the Newton's method. Each element on the diagonal of $\boldsymbol{J_{11}}$ corresponds to the inverse of the derivative of the friction head-loss. For all models, the friction head-loss function along the whole pipe k is defined as:

$$\xi_{\mathrm{f},k}(\ell_k) = f_k \int_0^{\ell_k} \mathrm{q}_k(x) |\mathrm{q}_k(x)|^{\gamma_{HW}-1} \, \mathrm{d}x \,, \tag{2.39}$$

and its derivative as:

$$\frac{\mathrm{d}\xi_{\mathrm{f},k}}{\mathrm{d}q_{0.5,k}}(\ell_k) = \gamma_{HW} \, f_k \int_0^{\ell_k} \frac{\mathrm{dq}_k}{\mathrm{d}q_{0.5,k}}(x) \cdot |\mathrm{q}_k(x)|^{\gamma_{HW}-1} \, \mathrm{d}x \,. \tag{2.40}$$

Thus, eq. (2.40) is 0 if $\mathrm{q}_k(x) = 0 \ \forall x \in [0, \ell_k]$. In this case, we will have a division by zero error. To avoid this error, we choose,

- for model M0: to reuse, as in chapter 1, the cubic regularization of the friction head-loss function initially proposed by [122] for flow rate close to 0; this regularization is fully described in appendix A.1.1,
- for model $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$: to reuse the preconditioning method initially proposed by [42], as explained in appendix B.3.2. When the derivatives are modified by preconditioning, the Newton's method used in current chapter (see section 2.2.2.2) then becomes a quasi-Newton method.

2.2.4.2 Junction nodes with pressure-head close to the minimum or service pressure-head

The derivatives of users' consumptions $\boldsymbol{\partial \mathbf{c}/\partial h}$, which appear in the sub-block $\mathbf{J_{22}}$ (see (2.38)), are discontinuous at $p = p_m$ and $p = p_s$. Thus, to avoid problem of convergence, we reuse, as in chapter 1, the regularization method described in appendix A.1.2 and initially proposed by [130].

2.2.4.3 Leaky pipe with pressure close to zero

For each leaky pipe k, we need to compute the derivative of the lineic leakage outflow rate:

$$\mathrm{q}_{\mathrm{LL},k}(p_k) = \beta_{L,k} \left([p_k]^+ \right)^{\alpha_{L,k}} \tag{2.41}$$

for all pressure-head p_k in $\{p_{0,k}, \widetilde{p_k}, p_{\ell,k}\}$, as:

$$\frac{\mathrm{d}\mathrm{q}_{\mathrm{LL},k}}{\mathrm{d}p_k}(p_k) = \begin{cases} \alpha_{L,k}\, \beta_{L,k}\, {p_k}^{\alpha_{L,k}-1} & \text{if } p_k > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.42}$$

(see eq. (2.3) for the definition of parameters $\alpha_{L,k}$ and $\beta_{L,k}$). However, (2.42) is discontinuous at $p_k = 0$, $\forall \alpha_{L,k} \in]0.5, 1[$. Thus, using eq. (2.42) as it could lead to convergence errors when p_k is close to 0 and $\alpha_{L,k} \in]0.5, 1]$. To avoid these errors, we implement a new cubic regularization of eq. (2.41) and quadratic regularization of eq. (2.42), to apply when p_k is close to 0. These regularizations are described in appendix B.3.1.

2.2.4.4 Initial guesses far from the solution and/or Jacobian with (sub-)linear growth

To avoid numerous iterations due to initial guesses of flow rates in pipes and heads at junctions far from the solutions at equilibrium, and to guarantee the convergence of the Newton's method even if the non-linear system of equations (2.35) does not have a super-linear growth [133], we choose to reuse the damping algorithm initially proposed by [43], extending it so it now takes pressure-dependent background leakages into account too. Full description of this extended version is presented in appendix B.3.3.

2.2.5 Checking and comparison

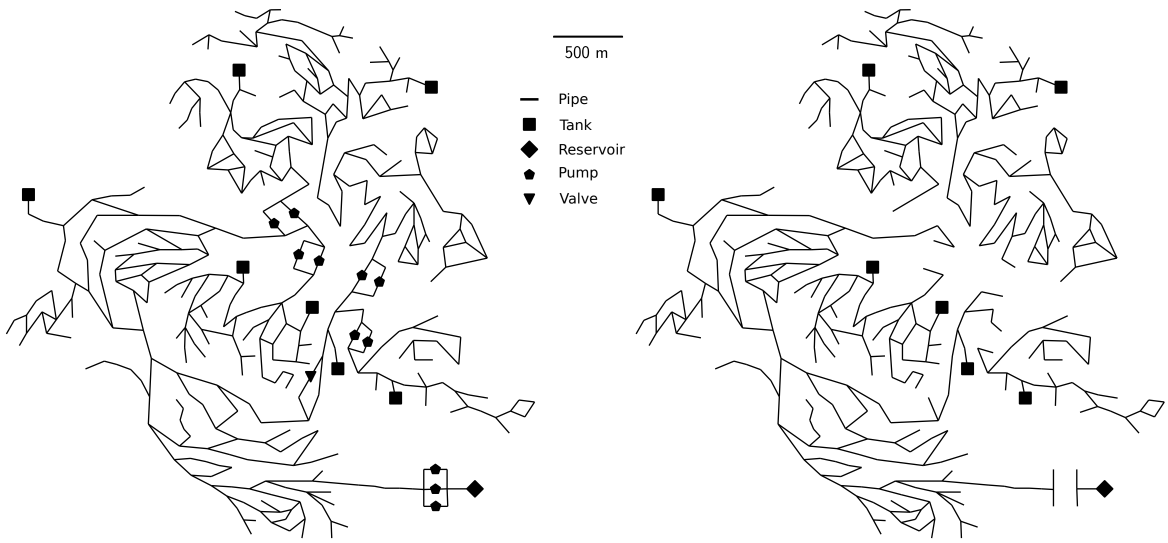
In this section, we describe the test cases used to check and compare the models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ and Ref with each other. We also propose a method to estimate the convergence order of the discretization algorithm used in model Ref (see section 2.2.1.4). Finally, we show how to compute the total background leakage outflow rate in each pipe of a WDN.

2.2.5.1 Test cases

The first test case aims at checking for the good functioning and stability of the models for a very simple network. Thereby, this test consists in simulating a single leaky pipe of length $\ell = 1{,}500\,\mathrm{m}$, connecting one tank to one junction, both located at the ground level (fig. 2.2). The junction has a demand $d = 10\,\mathrm{l\,s^{-1}}$, and the tank has a fixed head $h_t = 10\,\mathrm{mH_2O}$. The pipe has a diameter $\varnothing_p = 200\,\mathrm{mm}$, a Hazen-Williams friction coefficient $c_{\scriptscriptstyle HW} = 120$, background leakages of type $\alpha_L = 1.5$, and a level of degradation $\beta_L = 10^{-3}\,\mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$. These values of $\{\alpha_L, \beta_L\}$ correspond to a very high level of leakages; they are chosen to magnify the differences between the leakage models, and to check the robustness of the solution algorithm. Finally, this first test case explores the potential of models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ to adjust the model Ref, using a classical calibration method that is described in appendix B.4.

Figure 2.2: Single leaky pipe of length $\ell = 1{,}500\,\mathrm{m}$, connecting a tank ■ to a junction ●.

Next, through a second test case, we propose to compare the models with each other for a bigger network derived from a real leaky WDN. To do so, we simulate a simplified version of the network C-Town already used by [57, 120] (fig. 2.3a). In this simplified version, we set the users' demands $\boldsymbol{d}$ to half of the peak demands, which leads to an average demand $\overline{d} = 0.35\,\mathrm{l\,s^{-1}}$. For background leakages parameters $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\}$, we use the same values as in [57], such that $\alpha_{L,k} = 0.9$ and $\beta_{L,k} \in \{1, 2, 4\} \times 10^{-5}\,\mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$, $\forall k \in \{1, \ldots, n_p\}$. Finally, we choose initial levels of water in tanks that are sufficient for pumps and valve to be all closed; thus, all pumps and valve can be removed, as represented in fig. 2.3b.



(a) With equipment (11 pumps and 1 valve)

(b) Without equipment

Figure 2.3: Water distribution network C-Town from [120], composed of 432 pipes, 7 tanks and 1 reservoir, with (fig. 2.3a) and without (fig. 2.3b) equipment.

For all simulations, we use, as in chapter 1, a minimum pressure-head $p_m = 0\,\mathrm{mH_2O}$ and a service pressure-head $p_s = 20\,\mathrm{mH_2O}$. The machine used to run the simulations is an Intel Core i9 with 32 GB of memory.

2.2.5.2 Order of convergence

To quantify the efficiency of the discretization algorithm used in model Ref (see section 2.2.1.4), we compute its order of convergence. To do so, we denote $\boldsymbol{h}_{\boldsymbol{N}}^{(s_c+1)}$ the vector of heads at all (intermediate-)nodes at the iteration $s_c + 1$ of the discretization algorithm. Also, $\forall s \in \{1, \dots, s_c\}$, we denote $\widetilde{\widetilde{\widetilde{\boldsymbol{h}}}}_{\boldsymbol{N}}^{(s)}$ the vector that contains the values of $\boldsymbol{h}_{\boldsymbol{N}}^{(s)}$ interpolated at the positions $\boldsymbol{x}^{(s_c+1)}$. Then, denoting $\varepsilon^{(s)} = \|\boldsymbol{h}_{\boldsymbol{N}}^{(s_c+1)} - \widetilde{\widetilde{\widetilde{\boldsymbol{h}}}}_{\boldsymbol{N}}^{(s)}\|_\infty$ the infinity norm between $\boldsymbol{h}_{\boldsymbol{N}}^{(s_c+1)}$ and $\widetilde{\widetilde{\widetilde{\boldsymbol{h}}}}_{\boldsymbol{N}}^{(s)}$, the order of convergence η of the discretization algorithm is defined by the relation:

$$\varepsilon^{(s)} = a\, s^{\eta}, \tag{2.43}$$

with a a constant. To estimate η, we first compute $\varepsilon^{(s)}$ $\forall s \in \{1, \dots, s_c\}$; next we rewrite eq. (2.43) in log-log scale as

$$\log(\varepsilon^{(s)}) = \log(a) + \eta \log(s), \tag{2.44}$$

and we compute η by linear regression.

2.2.5.3 Derived leakage outflow rate

Models $\{\mathrm{M0}, \dots, \mathrm{M3}\}$ and Ref permit to approximate the functions $\{\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}, \mathrm{q}^{\mathrm{theo}}, \xi_{\mathrm{f}}^{\mathrm{theo}}\}$ (defined by eqs. (2.4) to (2.6)). But, to obtain a more explicit indicator of the level of background leakages in each pipe, we also compute, for each model, the total leakage outflow rate:

$$q_L = |q_\ell - q_0|, \tag{2.45}$$

where q_0 and q_ℓ are the flow rates at respectively the start and the end of the pipe.

When simulating the simplified network C-Town (fig. 2.3b), we then obtain the vector of background leakage outflow rates $\boldsymbol{q_L} = \left(q_{L,1}, \ldots, q_{L,n_p}\right)^T \in \mathbb{R}^{n_p}$. To compare the models at the global scale, we compute the p-norm of $\boldsymbol{q_L}$, $p \in \{1, 2, \infty\}$. We suppose that the difference between the q_L or p-norm computed with two different models becomes significant when it is greater than $10^{-2}\ \mathrm{l\,s^{-1}}$. Supposing that the average background leakage outflow rate in the tested networks is equal to $1\,\mathrm{l\,s^{-1}}$, then $10^{-2}\,\mathrm{l\,s^{-1}}$ represents a precision of 1 %, which corresponds to the best precision expected when measuring flow rates physically in a WDN [78, p. 3].

2.2.6 Implementation and framework

We extend the implementation of the Python simulator developed in chapter 1, including each background leakage model. We also adapt existing numerical enhancements and add new ones, to deal with all sources of instabilities (see section 2.2.4). Also, we implement our discretization algorithm at the WDN scale (see section 2.2.3), and the method to calibrate the leakage parameters in a single pipe (see appendix B.4).

At each Newton iteration m, to compute the descent direction $\boldsymbol{\delta_h}^{(m)}$ on $\boldsymbol{h}^{(m)}$, we now solve the linear system:

$$\boldsymbol{S}^{(m)}\,\boldsymbol{\delta_h}^{(m)} = \boldsymbol{\rho_m}^{(m)} - \boldsymbol{J_{21}}^{(m)} \left(\boldsymbol{J_{11}}^{(m)}\right)^{-1} \boldsymbol{\rho_e}^{(m)} \tag{2.46}$$

using the sparse solver UMFPACK (Unsymmetric MultiFrontal PACKage) [32] through its SciPy interface[2]. As a reminder, in eq. (2.46), $\boldsymbol{\rho_m}^{(m)}$ are the vector of mass residuals at junctions, $\boldsymbol{\rho_e}^{(m)}$ the vector of energy residuals in pipes, $\boldsymbol{J_{21}}^{(m)}$ and $\boldsymbol{J_{11}}^{(m)}$ are sub-blocks of the Jacobian matrix computed by eq. (2.38), and $\boldsymbol{S}^{(m)}$ is the Schur complement of the Jacobian matrix. We cannot use a Cholesky factorization to solve eq. (2.46) because $\boldsymbol{S}^{(m)}$ is not symmetric section 2.2.2.2 for more explanation). We also use the SciPy library [160]:

- to find the coefficients of the polynomials defined in appendix B.3.1 for the regularization of the lineic leakage outflow rates[3],
- for the PCHIP-interpolation[4] of the HGLs in the discretization algorithm,
- and to find the calibrated leakage parameters, solving[5] the non-linear systems defined in appendix B.4.

As in chapter 1, we make use of the convenient functionalities provided by the Python framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147, 148]. In particular, we use OOPNET to parse and convert the EPANET [141]'s input files to Python objects.

[2] https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.linalg.spsolve.html

[3] https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.root.html

[4] https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.PchipInterpolator.html

[5] https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html

2.3 Results and discussion

We present here the results obtained when running the test cases presented in section 2.2.5.1. For each test case, we first check the functioning of the discretization algorithm. Next, we discuss the execution times of each model. Finally, we compare the models with each other.

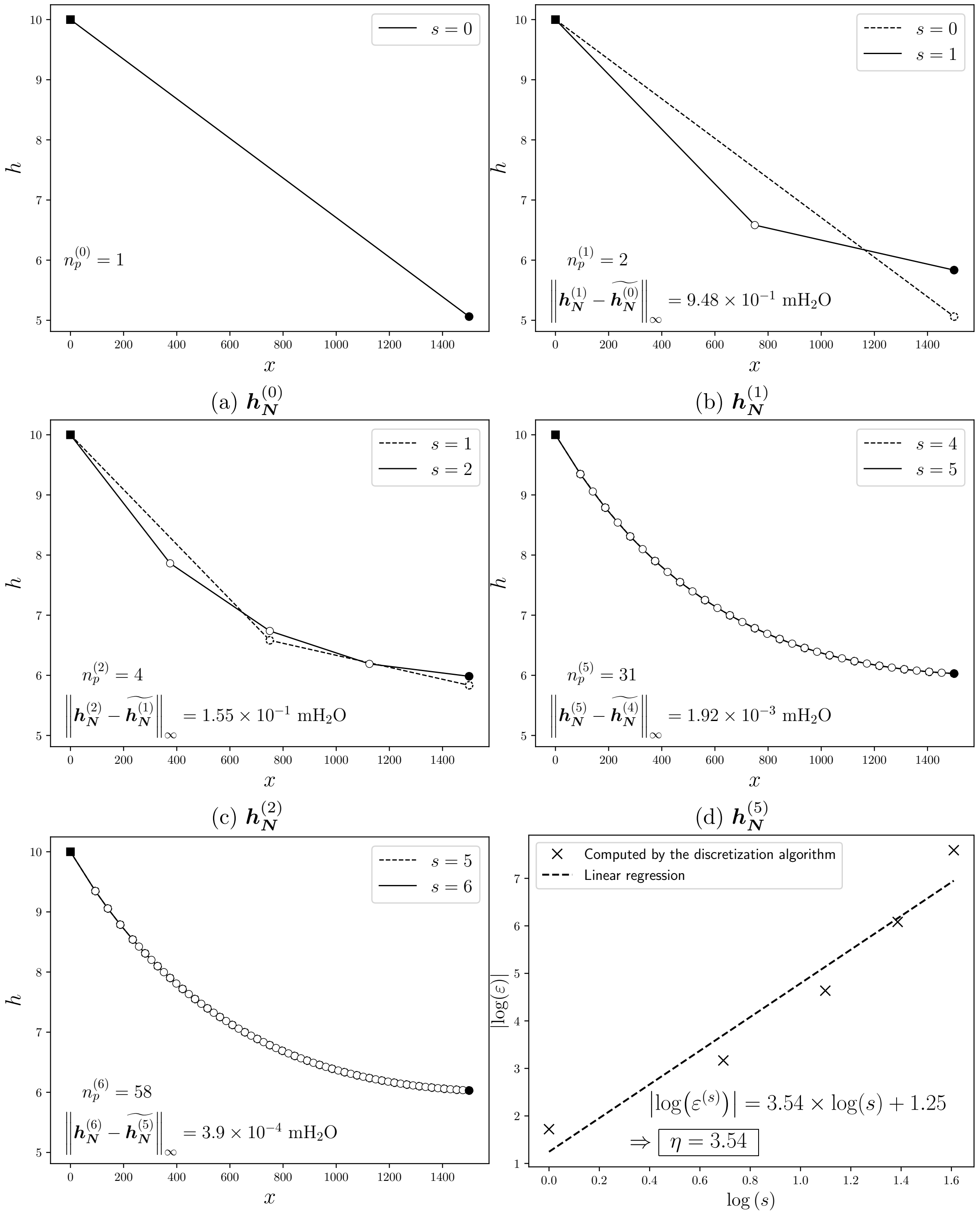
For the first test case only (i.e., the single leaky pipe), we also compare the results of the models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ after the calibration of their leakage parameters using the method described in appendix B.4, supposing that the measured data needed for the calibration are equal to the results of the Ref model. Like so, we explore the potential of models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ to adjust model Ref.

2.3.1 First test case: single leaky pipe

This first test case, that corresponds to the simulation of the single leaky pipe of fig. 2.2, is a useful and simple benchmark to verify the accuracy and stability of the proposed models.

2.3.1.1 Discretization algorithm

For this test case, the discretization algorithm needs 6 iterations to find the reference HGL $\boldsymbol{h}_{\boldsymbol{N}}^{\mathrm{Ref}} = \boldsymbol{h}_{\boldsymbol{N}}^{(s_c)} = \boldsymbol{h}_{\boldsymbol{N}}^{(5)}$ (fig. 2.4; s_c: iteration at which the algorithm reaches convergence). We can see that new sub-pipes and intermediate junction nodes are created at each iteration (figs. 2.4a to 2.4e). Also, as expected, the discretization is irregular. For example, at iteration 6 the number of sub-pipes is equal to 58 (fig. 2.4e) while at iteration 5 it is equal to 31 (fig. 2.4d). This signifies that 2 of the 31 sub-pipes from iteration 5 have not been rediscretized because they did not need it.



(a) $\boldsymbol{h}_{\boldsymbol{N}}^{(0)}$

(b) $\boldsymbol{h}_{\boldsymbol{N}}^{(1)}$

(c) $\boldsymbol{h}_{\boldsymbol{N}}^{(2)}$

(d) $\boldsymbol{h}_{\boldsymbol{N}}^{(5)}$

(e) $\boldsymbol{h}_{\boldsymbol{N}}^{(6)}$

(f) Convergence order η of the algorithm

Figure 2.4: Evolution of the hydraulic grade line (HGL) $\boldsymbol{h}_{\boldsymbol{N}}^{(s)}$ (figs. 2.4a to 2.4e), and convergence order η (fig. 2.4f), when applying the discretization algorithm of model Ref to the single leaky pipe of fig. 2.2. In figs. 2.4a to 2.4e: h, head (in mH$_2$O); x, position along the pipe (in m); s, iteration of the discretization algorithm; $n_p^{(s)}$, number of pipe or sub-pipes at iteration s. $\widetilde{\boldsymbol{h}_{\boldsymbol{N}}^{(s)}}$: HGL at iteration s, interpolated at positions $\boldsymbol{x}^{(s+1)}$. In fig. 2.4f: $\varepsilon^{(s)}$, infinity norm between the heads at iteration s and the ones at iteration 6.

The high number (i.e. 31) of needed sub-pipes to reach convergence (fig. 2.4d) can be explained by the strong values of the leakage parameters used. The order of convergence of the algorithm is equal to 3.54 (fig. 2.4f), which is more than cubic and so very good. The CPU time elapsed during the discretisation process is equal to 330 ms. This is quick, but normal when simulating a single pipe.

Replacing the use of model M0 by the one of M1, M2 or M3 in the discretization algorithm leads to the same HGL at convergence. Thus, the algorithm remains stable whatever the model we use, and all models M0, M1, M2 and M3 give equivalent results providing that the pipe is discretized in enough sub-pipes. This stability was expected because all the functions used in the models are continuous.

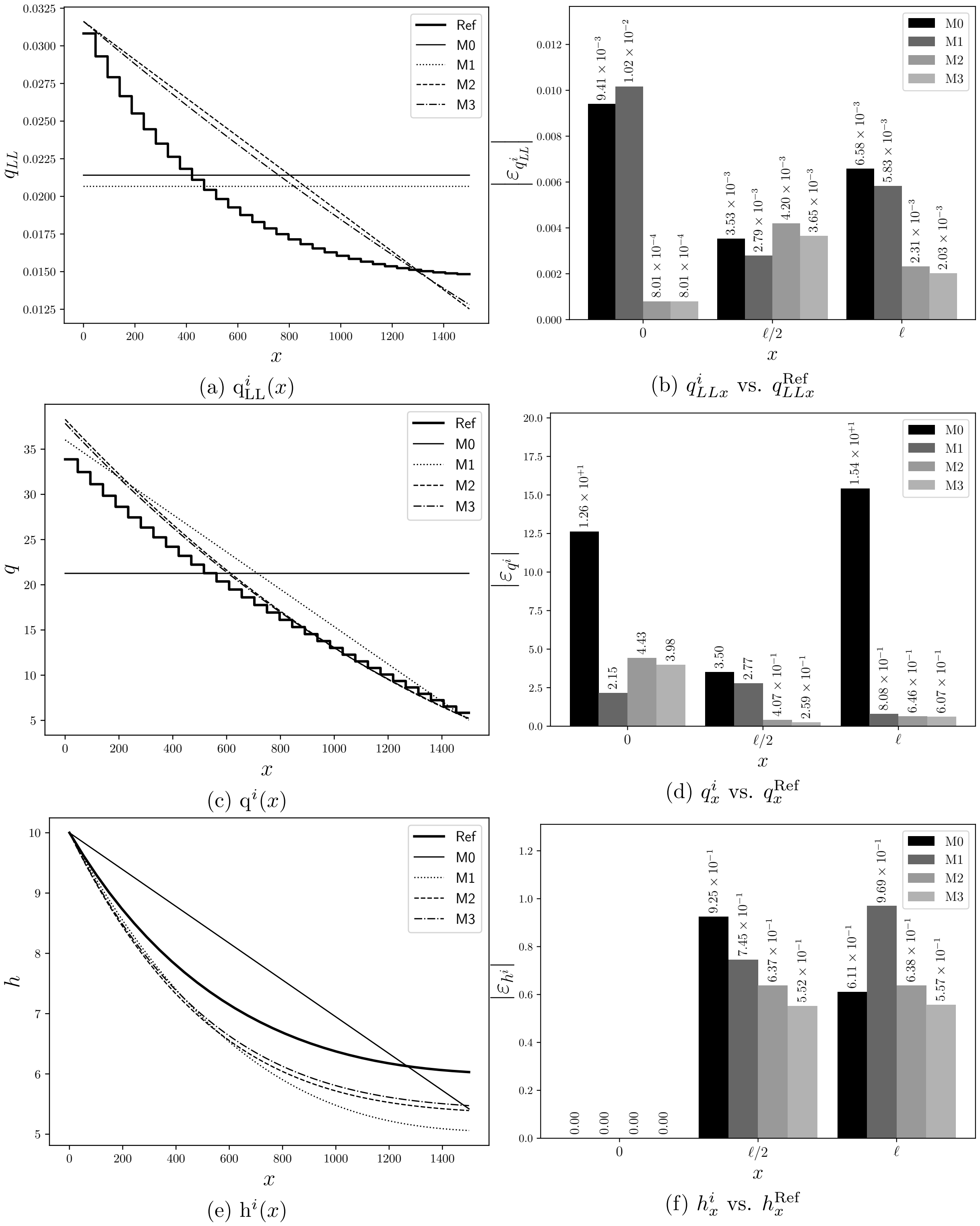
2.3.1.2 Execution times

The simulation of the leaky pipe lasts respectively 25 ms with M0, 35 ms with M1, 50 ms with M2, 66 ms with M3, and 367 ms with Ref. Thus, we can clearly see that our new models M1, M2, M3 and Ref need more time than the state-of-the-art one, but this is normal because the equations in our new models are more complex. Also, we can see that the increase of computational time is relative to the complexity of the models.

Model Ref is naturally much more time consuming than other models because it consists in running internally 6 times the model M0, on more and more discretized networks. Moreover, it needs PCHIP interpolation between each of these internal runs. The difference between the elapsed time of the discretization algorithm (330 ms; see section 2.3.1.1), and the total elapsed time of the model Ref (367 ms), is equal to $367 - 330 = 37\,\text{ms}$ ($\approx 10\ \%$ of the total elapsed time), and corresponds to the post-processing needed to aggregate the results at the undiscretized pipe's scale. Finally, the simulation of the leaky pipe with model Ref remains less than half of a second, which is an acceptable time given that we use the interpreted language Python.

2.3.1.3 Comparison of the models before calibration

The drawn profiles of functions $\{\mathrm{q}_{\mathrm{LL}}^{i}(x), \mathrm{q}^{i}(x), \mathrm{h}^{i}(x)\}$ ($\mathrm{q}_{\mathrm{LL}}^{i}$: lineic leakage outflow rates, q^{i}: flow rates, h^{i}: heads), $i \in \mathrm{Ref} \cup \{\mathrm{M0}, \dots, \mathrm{M3}\}$, are consistent with the degrees of each model (figs. 2.5a, 2.5c and 2.5e). Indeed, since we use model M0 in the discretization algorithm, it is normal that $\mathrm{q}_{\mathrm{LL}}^{\mathrm{Ref}}(x)$ and $\mathrm{q}^{\mathrm{Ref}}(x)$ are step functions. Also, as expected, $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}(x)$ and $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M1}}(x)$ are constant along the full pipe, $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}(x)$ is affine, and $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M3}}(x)$ is slightly convex. Finally, for all model i in $\mathrm{Ref} \cup \{\mathrm{M0}, \dots, \mathrm{M3}\}$, $\mathrm{h}^{i}(x)$ logically starts from the fixed head at tank $h_t = 10\,\mathrm{mH_2O}$. Globally, we can see that the higher the degree of the model is, the better the curves from $\{\mathrm{M0}, \dots, \mathrm{M3}\}$ fit the ones of model Ref.



(a) $\mathrm{q}^i_{\mathrm{LL}}(x)$

(b) q^i_{LLx} vs. q^{Ref}_{LLx}

(c) $\mathrm{q}^i(x)$

(d) q^i_x vs. q^{Ref}_x

(e) $\mathrm{h}^i(x)$

(f) h^i_x vs. h^{Ref}_x

Figure 2.5: Lineic leakage outflow rates q^i_{LL} (in $\mathrm{l\,s^{-1}\,m^{-1}}$), flow rates q^i (in $\mathrm{l\,s^{-1}}$) and heads h^i (in $\mathrm{mH_2O}$) along the leaky pipe of fig. 2.2, for each model $i \in \mathrm{Ref} \cup \{\mathrm{M0}, \ldots, \mathrm{M3}\}$. Figures 2.5a, 2.5c and 2.5e show the profiles of the functions. Figures 2.5b, 2.5d and 2.5f show the absolute errors between models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ and Ref, at positions $x \in \{0, \ell/2, \ell\}$, $\ell = 1{,}500\,\mathrm{m}$.

The absolute errors on $\{q^i_{LLx}, q^i_x, h^i_x\}$ when compared to $\{q^{\text{Ref}}_{LLx}, q^{\text{Ref}}_x, h^{\text{Ref}}_x\}$, $x \in \{0, \ell/2, \ell\}$, are globally smaller for the new models $\{\text{M1}, \text{M2}, \text{M3}\}$ than for the state-of-the-art model M0 (figs. 2.5b, 2.5d and 2.5f). For example, from model M0 to models M1, M2 and M3, the errors on $q^i_{LL\ell}$ (i.e., lineic leakage outflow rate at the end of the pipe) decrease (in absolute value) by respectively 11.30 %, 64.80 % and 69.20 %, and the ones on $h^i_{\ell/2}$ (i.e., head at the middle of the pipe) by respectively 19.40 %, 31.10 % and 40.30 %. Also, model M3 gives generally better results than M2, which gives itself better results than M1. This order is consistent with the increasing complexity of $\{\text{M0}, \ldots, \text{M3}\}$.

2.3.1.4 Comparison of the models after calibration

We present here the results of the models $\{\text{M0}, \ldots, \text{M3}\}$ after the calibration of their leakage parameters. We compare these results against the ones of the model Ref.

Globally, the calibrated leakage types α^i_L, $i \in \{\text{M2}, \text{M3}\}$ (α_L cannot be calibrated for $\{\text{M0}, \text{M1}\}$), are very close to the uncalibrated α_L. Conversely, the calibrated degradation levels β^i_L, $i \in \{\text{M0}, \ldots, \text{M3}\}$, differ much from the uncalibrated β_L (fig. 2.6). Indeed, the relative errors on α_L when compared to α^i_L are equal to ~ 2 ‱, while the relative errors on β_L when compared to β^i_L are all more than 22 %. Thus, we can say for this test case that β_L is much more sensitive than α_L to the leakage model in use.

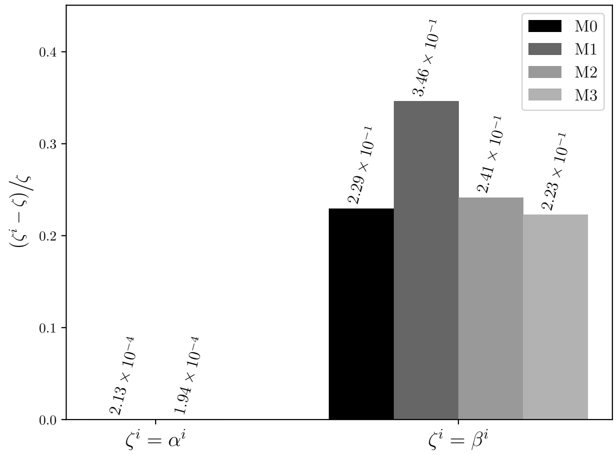
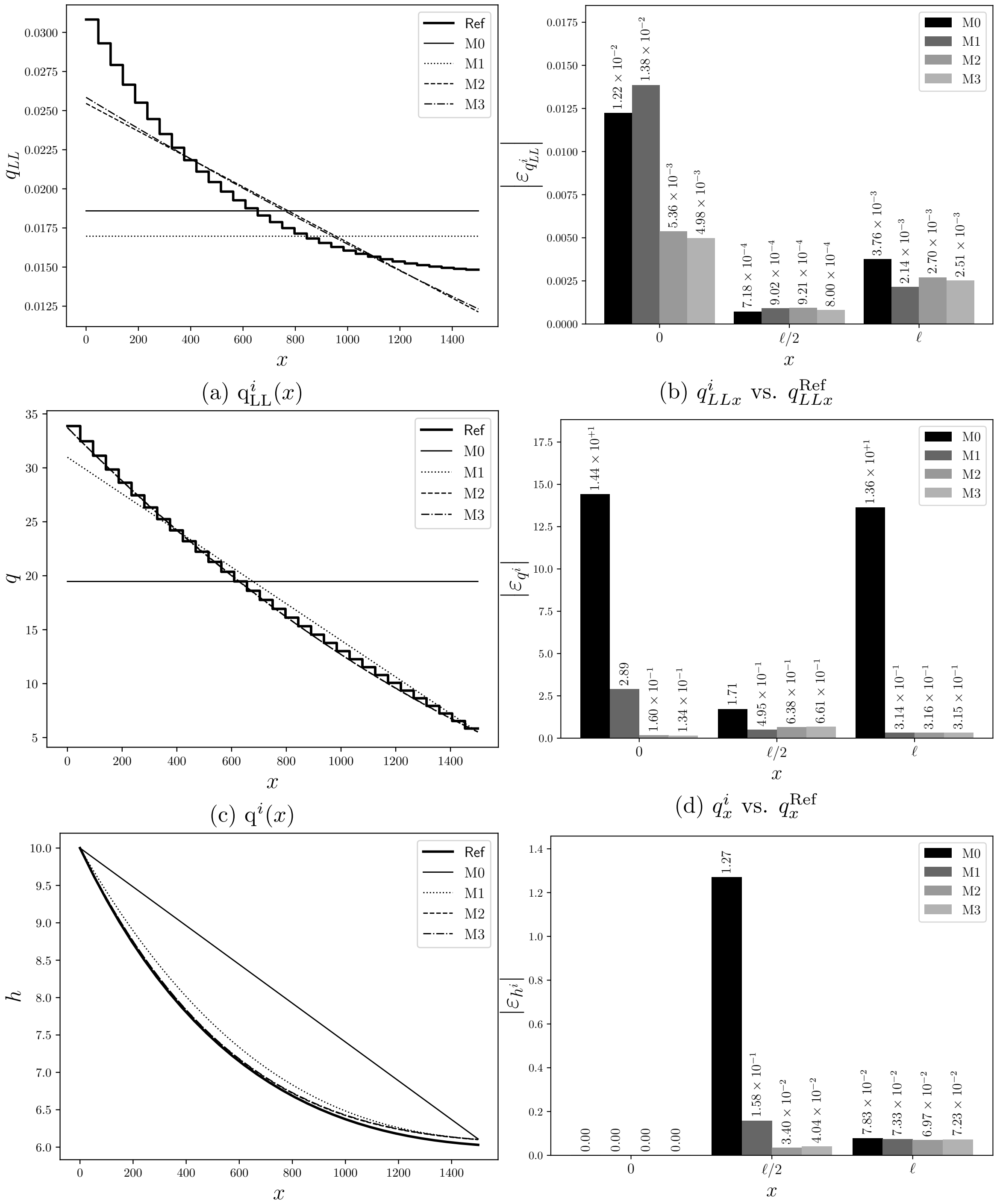


Figure 2.6: Relative difference (unit-less) between the uncalibrated leakage parameters $\zeta \in \{\alpha_L, \beta_L\}$, and the calibrated ones $\zeta^i \in \{\alpha^i_L, \beta^i_L\}$ obtained from the flows and heads computed by the model Ref, for the single leaky pipe of fig. 2.2. For α^i_L: $i \in \{\text{M2}, \text{M3}\}$ (not possible to calibrate parameter α_L for models $\{\text{M0}, \text{M1}\}$). For β^i_L: $i \in \{\text{M0}, \ldots, \text{M3}\}$.

When using the models $\{\text{M0}, \ldots, \text{M3}\}$ with the calibrated leakage parameters, the drawn profiles of the functions $\{\mathrm{q}^i_{\mathrm{LL}}(x), \mathrm{q}^i(x), \mathrm{h}^i(x)\}$ then follow the same tendencies as before calibration. But, as expected, the curves of the calibrated functions now fit more closely the reference ones (figs. 2.7a, 2.7c and 2.7e). In particular, $\mathrm{h}^i(x)$ now passes (or almost passes) through h^{Ref}_ℓ $\forall i \in \{\text{M0}, \ldots, \text{M3}\}$ (fig. 2.7e). This gain in accuracy is even more visible for models $\{\text{M2}, \text{M3}\}$, for which both leakage parameters α_L and β_L are calibrated. Indeed, $\mathrm{q}^{\text{M2}}(x)$ and $\mathrm{q}^{\text{M3}}(x)$ now pass through q^{Ref}_0 and q^{Ref}_ℓ(fig. 2.7c).



(a) $\mathrm{q}_{\mathrm{LL}}^{i}(x)$

(b) q_{LLx}^{i} vs. q_{LLx}^{Ref}

(c) $\mathrm{q}^{i}(x)$

(d) q_{x}^{i} vs. q_{x}^{Ref}

(e) $\mathrm{h}^{i}(x)$

(f) h_{x}^{i} vs. h_{x}^{Ref}

Figure 2.7: Lineic leakage outflow rates q_{LL}^{i} (in $\mathrm{l\,s^{-1}\,m^{-1}}$), flow rates q^{i} (in $\mathrm{l\,s^{-1}}$) and heads h^{i} (in $\mathrm{mH_2O}$) along the leaky pipe of fig. 2.2, for each model $i \in \mathrm{Ref} \cup \{\mathrm{M0}, \dots, \mathrm{M3}\}$. $\{\mathrm{M0}, \dots, \mathrm{M3}\}$ used calibrated leakage parameters. Figures 2.7a, 2.7c and 2.7e show the profiles of the functions. Figures 2.7b, 2.7d and 2.7f show the absolute errors between models $\{\mathrm{M0}, \dots, \mathrm{M3}\}$ and model Ref, at positions $x \in \{0, \ell/2, \ell\}$, $\ell = 1{,}500\,\mathrm{m}$.

As expected, the calibration of the leakage parameters leads to bigger reductions of the absolute errors on q_0 for models $\{M2, M3\}$ than for models $\{M0, M1\}$ (fig. 2.7d). Likewise, the calibration drives to stronger reductions of the errors on $h_{\ell/2}$ for $\{M1, M2, M3\}$ than for M0 (fig. 2.7f). This last result is particularly interesting since it demonstrates that, unlike the existing model M0 from [55], our new models compute after calibration much better estimations of head at intermediate positions along the pipe (here at $x = \ell/2$) than before calibration. Thus, using our new models $\{M1, M2, M3\}$ could probably reduce significantly the number of measuring points and sensors needed to get an exhaustive and accurate HGL along the pipe, especially once the models are calibrated.

Finally, for each of $\{M0, \ldots, M3\}$, when we gain precision by means of calibration for some variables at some locations, we also loose some precision on other variables and at other locations (e.g., q^i_{LLx} in fig. 2.7b). This is unavoidable since, even if more accurate, our new models remain approximations of the theoretical model.

2.3.2 Second test case: simplified network C-Town

This second test case is proposed in order to validate the functioning, accuracy and stability of our models when simulating a larger and more realistic network. For this purpose, we choose the network of fig. 2.3b, which includes most of the critical parameters needed for validation while still remaining compact enough to perform quick simulations.

2.3.2.1 Discretization algorithm

For the network of fig. 2.3b, our discretization algorithm needs 4 iterations and 12.99 s of CPU time to find the reference HGL $\boldsymbol{h}_{\boldsymbol{N}}^{\mathrm{Ref}} = \boldsymbol{h}_{\boldsymbol{N}}^{(s_c)} = \boldsymbol{h}_{\boldsymbol{N}}^{(3)}$ (s_c: iteration at which the algorithm reaches convergence). This longer elapsed time is explained by the high number of pipes and sub-pipes (i.e., $n_p^{(m)}$ at iteration m) to simulate when the network is discretized. Indeed, the discretized networks consist of $n_p^{(1)} = 819$, $n_p^{(2)} = 1590$, $n_p^{(3)} = 1824$ and $n_p^{(4)} = 1832$ pipes and sub-pipes at respectively iterations 1, 2, 3 and 4.

Also, none of $\{n_p^{(1)}, \ldots, n_p^{(4)}\}$ is a multiple of the initial number of pipes $n_p^{(0)} = 432$ in the undiscretized network, which means that the discretization is irregular (i.e., some pipes are more discretized than some others) at each iteration. As a reminder, this irregular discretization prevents from over-discretization of pipes presenting low gradients of pressures, and permits to speed-up the computations.

The order of convergence of the discretization algorithm is equal to 3.31 (fig. 2.8). This means that our discretization algorithm remains efficient even when applied to a real network with many pipes.

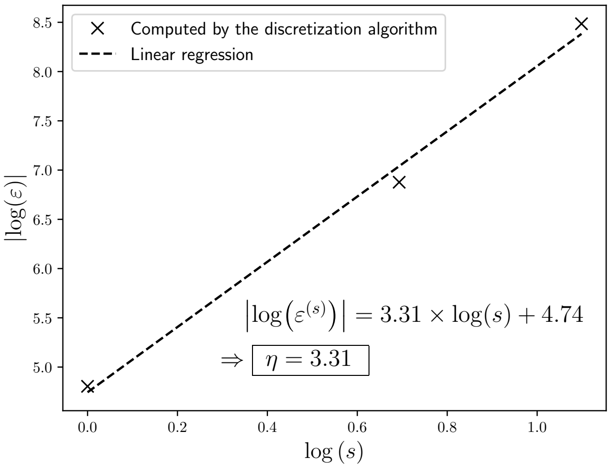


Figure 2.8: Computation of the convergence order η of the discretization algorithm when applied to simplified network C-Town (fig. 2.3b). $s \in \{1, \dots, s_c\}$: iteration of the discretization algorithm ($s = s_c$: convergence). $\varepsilon^{(s)}$: infinity norm between the hydraulic grade lines (HGLs) at s and the one at $s_c + 1$.

Finally, as for the first test case, replacing model M0 by M1, M2 or M3 in the discretization algorithm leads to the same HGL. Thus, the algorithm remains stable whatever the model we use, providing that the pipes are discretized in enough sub-pipes. Once again, this stability was expected because all the functions used in the models are continuous.

2.3.2.2 Execution times

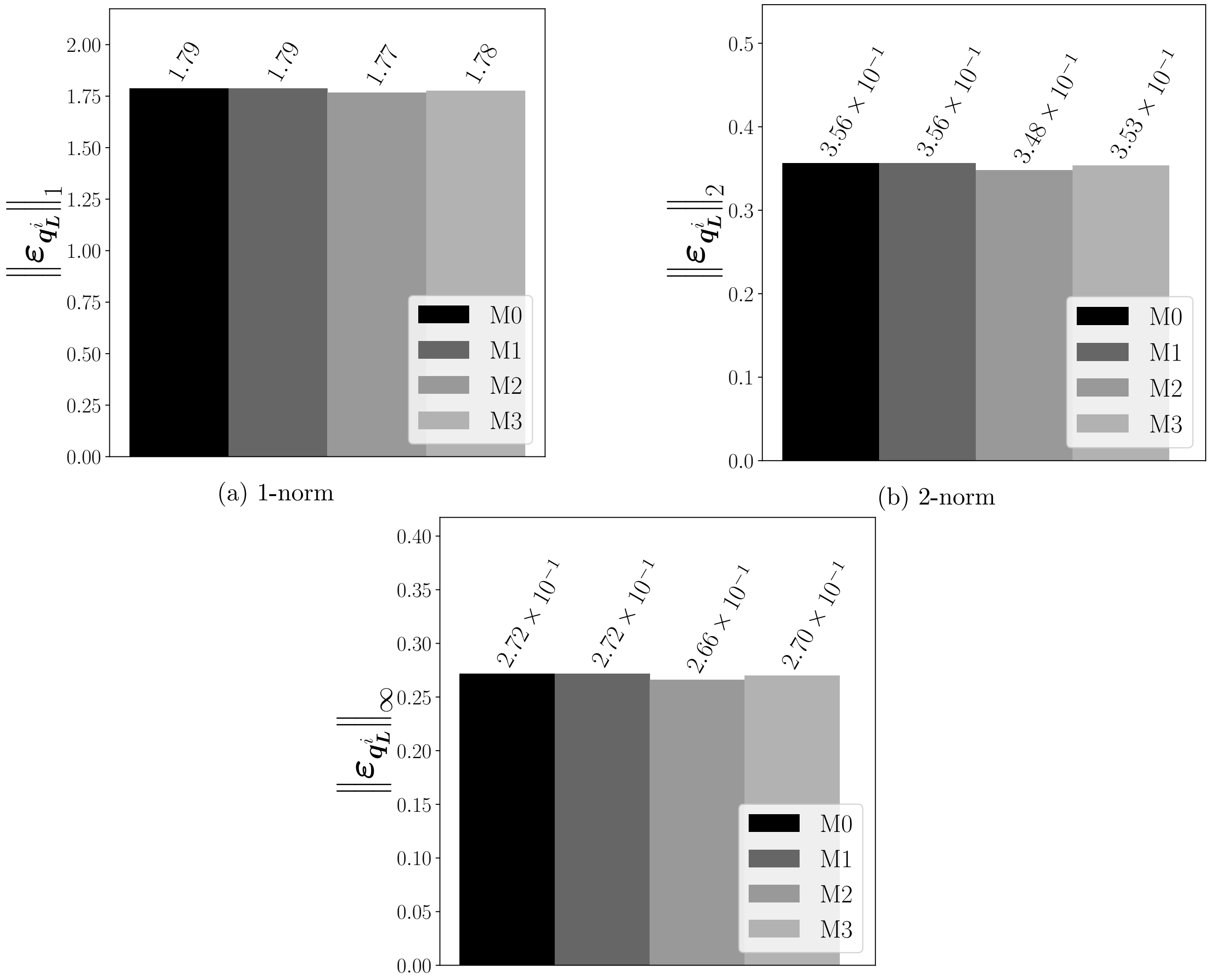
The simulation of the simplified network C-Town lasts respectively 0.19 s with M0, 0.32 s with M1, 0.41 s with M2, 0.49 s with M3, and 16.18 s with Ref. As in the first test case, we can clearly see that our new models, especially the Ref one, need more time than M0. But this is normal because our models are more complex and accurate.

The difference between the elapsed time of the discretization algorithm (12.99 s), and the total elapsed time of the model Ref (16.18 s), is equal to $16.18 - 12.99 = 3.19$ s; it represents ≈ 20 % of the total elapsed time, and corresponds to the post-processing needed to aggregate the outputs from the sub-pipe to the pipe scales. For current test case, the elapsed time due to post-processing is proportionally two times greater than the one found in the first test case (i.e., ≈ 10 %). This can be explained by the greater number of iterations needed in the first test case, and by the higher number of pipes in the second test case; indeed, the post-processing is done sequentially pipe after pipe.

Nevertheless, all these execution times remain acceptable, and could be probably reduced by re-implementing the simulator using a compiled language, and/or parallelizing the post-processing at the end of the discretization algorithm. However, for some applications like sensitivity analysis, for which many runs are needed, the models M0, M1, M2 and M3 should be preferred to model Ref in terms of rapidity.

2.3.2.3 Comparison of the models

The 1-norms of the errors between the leakage outflow rates computed with models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ and the ones computed with model Ref vary from $1.77\,\mathrm{l\,s^{-1}}$ to $1.79\,\mathrm{l\,s^{-1}}$ (fig. 2.9a). The 2 and ∞-norms of these errors are respectively around $0.35\,\mathrm{l\,s^{-1}}$ and $0.27\,\mathrm{l\,s^{-1}}$ (figs. 2.9b and 2.9c). Thus, the leakage outflow rates computed by models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ are very close to each other, but they are also significantly different from the ones computed with model Ref.



(a) 1-norm

(b) 2-norm

(c) ∞-norm

Figure 2.9: p-norm, $p \in \{1, 2, \infty\}$, of the absolute errors $\boldsymbol{\varepsilon}$ between the leakage outflow rates $\boldsymbol{q_L^i}$ (in $\mathrm{l\,s^{-1}}$) computed from each model $i \in \{\mathrm{M0}, \ldots, \mathrm{M3}\}$ and the ones computed with model Ref, in the simplified network C-Town of fig. 2.3b.

The small differences between models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ can be explained by the characteristics of the simplified network C-Town. Indeed, this network has low deltas of elevation along its pipes: a median equals to $3.06\,\mathrm{m}$, and an IPR (inter percentile range) between percentiles 10^{th} and 90^{th} equals to 13.71 m. Since the pressures are directly correlated to the elevations (i.e., $p = h - u$), the deltas of pressure induced by these deltas of elevation are low too. Thus, we believe that the gradients of pressure in C-Town is enough for seeing differences between models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ and Ref, but barely enough to observe differences between $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$. Still, supposing that the pressures and the demands remain the same during all a day, the differences between M0 and M3 represent $0.02 \times 3{,}600 \times 24 = 1.73\,\mathrm{m}^3$ per day, which already shows a little improvement from M0 to M3. Moreover, we think, by analogy with the first test case, that the leakage outflow rates computed from models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ would show more difference with the use for each of them of specifically calibrated leakage parameters. But we do not have, at the time of this study, the measured data that would permit us to calibrate these parameters.

Nevertheless, the differences between models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ and model Ref are quite significant. For example, supposing that the pressures and the demands remain the same during all a day, then these differences represent $1.78 \times 3{,}600 \times 24 = 153.79\,\mathrm{m}^3$ per day, and thus more than the daily consumption of 1,000 inhabitants in France [92]. Therefore, our new model Ref represents a real gain in accuracy comparing to other models, and so even on a network with normal levels of leakages and low deltas of elevation along its pipes.

2.4 Conclusions

In this chapter, we demonstrated, through two complementary test cases, the interest in using more complex pressure-dependent models (PDMs) of background leakages in water distribution networks (WDNs). This statement relies on the comparison of a state-of-the-art model to new more accurate models. The new models were developed from successive original refinements of the state-of-the-art one, or through the recursive discretization of the pipes into sub-pipes until the hydraulic grade line (HGL) along each pipe converges. The choice between our new models depends on the exhaustiveness of the measured data needed to calibrate them, and/or on the level of accuracy to reach. In particular, the new model based on a recursive discretization algorithm of the pipes leads to reference background leakage outflow rates, flow rates and friction head-losses that are very close to the ones we would obtain if we knew the exact pressures all along the pipes. This algorithm could be easily reused to test other WDN models, or adapt to any graph structured model from different fields of applied mathematics and scientific computation.

Despite their complexity, our new models can be easily integrated into existing software programs, providing that their solvers allow to deal with possibly large and ill-conditioned systems. For this purpose, we implemented an efficient Newton's method including several numerical enhancements. Thereby, our solver permits to simulate large networks in an acceptable time (deeper explanations of the numerical enhancements are presented in appendix B.3). Also, all our code is part of the Python collaborative framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147, 148], and will be available on the GitHub public repository[6] of the project soon. This way, we are convinced that our work, which represents a significant step forward from the previous state of knowledge, will be easily profitable to the whole community of WDN modelers.

By taking into account the gradient of pressure-heads along the pipes, our new models predict more accurately the background leakage outflow rates than does the state-of-the-art model, especially

[6] `https://github.com/oopnet`

in pipes presenting strong levels of leakages. Moreover, we believe that this gain of accuracy is strongly correlated to the deltas of elevations and to the friction head-losses along the pipes. Indeed, important variations of elevations and friction head-losses make the pressure-heads along the pipes to vary significantly too, and then should increase the benefit in using our models. In some situations, our new models could even permit to identify partly-unsupplied pipes or high-lying nodes, by detecting zero pressure locations along the pipes. We will focus on this point in the next chapter.

The characterization of the level of leakages in WDNs is of prime importance for operators and decision makers. Two of our new models take explicitly into account the pressure-heads at both extremities of each pipe. Thus, these two models could permit, after some further work, the calibration of both leakage types and degradation levels of pipes, and represent a new step toward more reliable decision support tools (e.g., for choosing the pipes that have to be repaired and/or replaced first according to their criticality). Nevertheless, this calibration would need exhaustive physical measurements in contrasted real leaky networks. The fifth chapter of this manuscript will propose some calibration strategies and discuss about existing barriers.

Highlights

- New models with more physical insights of losses in water distribution networks
- Integrating pressure gradient along pipes leads to more accurate leakage models
- Reference hydraulic grade lines and leakage outflow computed by discretization
- Other new gradually refined models are compared with existing and reference ones
- New method and models give better predictions without much computational overhead

Chapter 3

High-lying nodes and partly-supplied pipes

Abstract

Several authors already tackled the problem of high-lying nodes in water distribution networks (WDNs). Some other proposed solutions for pressure-dependent modeling (PDM) of background leakages. But none of them achieved to simulate both high-lying nodes and PDM background leakages at once. Thus, we propose in this chapter a new method, based on adapted and enhanced versions of the background leakage models and discretization algorithm implemented in the previous chapter, to compute accurately PDM background leakage outflow rates and user-consumptions in leaky networks with high-lying nodes. The new developments extend the simulator of previous chapter, and are tested on several dedicated networks. Different metrics and profiles are produced to assess the efficiency of our solution and its better realism. Results show more accurate user-consumptions, leakage outflow rates and hydraulic grade lines (HGLs) than with the previous simulator, despite the difficulty to find the exact flow rates in upstream parts of the network. Finally, our new simulator is integrated into a collaborative Python framework, to make it easily reusable by the community. Simulators integrating both high-lying nodes and background leakages are essential for asset management and to analyze the resilience of deficient networks. Thus, our solution will permit scientists and operators of WDNs to improve their understanding of the networks and to anticipate their rehabilitation. Moreover, the genericity of our algorithms make them valuable to study new processes or to test other modeling approaches.

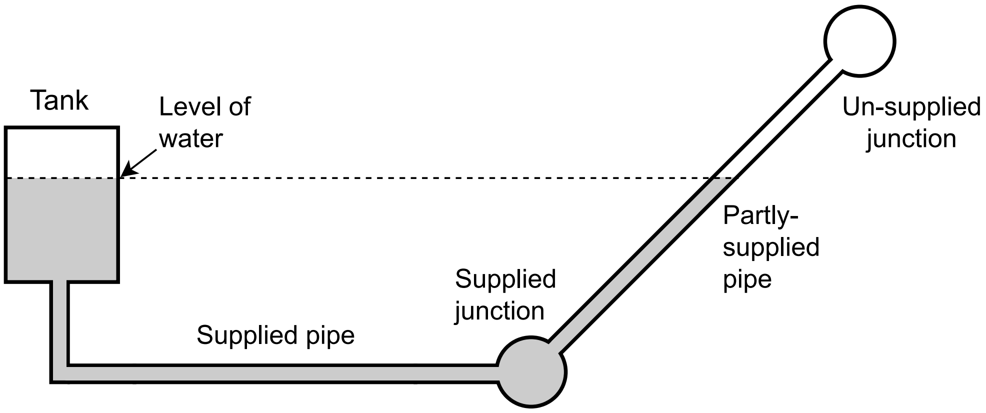
Keywords:
water distribution network (WDN), background leakage, pressure-dependent model (PDM), recursive discretization method, high-lying node

3.1 Introduction

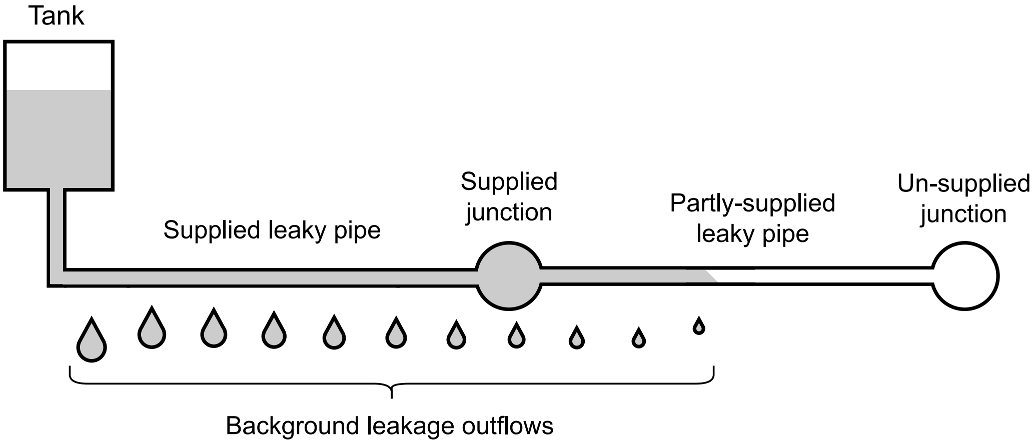
3.1.1 Problem description

In chapter 2, we introduced the modeling of background leakage outflow rates that depend on the pressure-head along the pipes. Then, each node of the studied networks was either partly or fully supplied, because the pressure-head was positive everywhere.

However, for some network configurations, the pressure-heads at some nodes and in some parts of the pipes can become "negative". By negative, we mean that the piezometric-head is lower than the elevation. We then talk of "high-lying" nodes and "partly-supplied" pipes. Such situations can occur for example when the elevation becomes upper than the piezometric head from some position along a pipe, and that no pump is working to increase enough the pressure-head and permitting the water to reach a node (fig. 3.1a). An other case is when the leakage outflow rates, or the user-consumptions (or any other source of outflow rate), are too strong for the upstream flow rate to supply a downstream part of the network (fig. 3.1b). In both cases, there should neither be any user's consumption at the high-lying nodes, nor background leakage outflow rate in the unsupplied parts of the pipes. Otherwise, this leads to overestimation of user-consumptions and background leakage outflow rates.



(a) Side view of a network where the junction at the right extremity is unsupplied because of its elevation. We suppose here, for better readability, that the friction head-losses are negligible. Thus, the pipe on the right becomes unsupplied as soon as the elevation becomes upper than the level of the water inside the tank.



(b) Side view of a leaky network where the junction at the right extremity is unsupplied because of background leakage outflows: the flow rate leaving the tank is not sufficient to "supply" the background leakage outflow rates until the junction. The size of the drops decreases along the pipes because the leakages depend on the pressure, which naturally decreases because of friction head-losses.

Figure 3.1: Two cases where a node can be unsupplied in water distribution networks.

3.1.2 State-of-the-art

First, [129] proposed an energy minimization algorithm, based on the work from [23], to prevent user-consumptions at elevated junctions where the pressure-head becomes negative. Next, [128] extended this algorithm to better predict the hydraulic parameters of a network section supplied via a junction with negative pressure; to do so, they introduced an additional constraint to the pressure-dependent model (PDM)'s equations, which acts as a dummy pressure sustaining valves that reduces the flow rate to zero in deficient pipes. Then, [127] enhanced the algorithm to simulate control valves. Recently, [135] developed and implemented an active set method to speed-up and make the algorithm more robust.

The algorithm developed by [127, 135] is quick and efficient, even for deficient networks [17]. Moreover, it could be derived to high-lying nodes with negative pressures. But their approach cannot handle background leakage outflow rates, since these outflow rates are computed from several pressure-heads along the pipes. Indeed, their method needs to express either everything as functions of the flow rates, or everything as functions of heads. In other words, all functions (i.e., user-consumptions, friction head-losses, leakage outflow rates, etc.) have to be invertible. This is not the case anymore as soon as we add PDM background leakages to the hydraulic equations.

Using a different approach, [54] simulated both PDM user-consumptions and background leakage outflow rates. Their model of background leakages is the one proposed by [55], and described and denoted M0 in chapter 2. Their algorithm prevents from user-consumptions at unsupplied junctions, and compute background leakage outflow rates that are realistic if the level of leakage is not too strong and if the pipes have low delta of elevation (i.e., gentle slope). It is already widely deployed and reused through the software WDNetXL (integrated system for Water Distribution Network analysis, planning and management distributed as ExceL® add-ins) [95]. However, it cannot properly take into account the leakage outflow rates from partly-supplied pipes, because in each pipe their model calculates a flow rate that does not vary along the pipe.

Accurate simulation of high-lying nodes and PDM background leakage outflow rates is essential to model the resilience of critical infrastructures in case of extreme events or disasters [6, 59, 132], to analyze the impact of intermittent water distribution [33, 77], to calibrate water distribution networks (WDNs) [151] and optimize the hydrants and valves control for fire incidents management [114], and to analyze energy recovery and leakage reduction potential from asset management data [14]. However, existing solutions are not refined enough to distinguish parts of pipes that must contribute to leakage outflow rate from other parts, in networks that include high-lying node(s) or subject to background leakage outflow rates.

3.1.3 Hypothesis and objectives

We think that the background leakage models and the discretization algorithm developed in chapter 2 could be extended to find which nodes are high-lying and which pipes are partly-supplied, and to determine the part of the pipes that is supplied and the one that is not. Thus, our first objective is to make these extensions.

Also, we think that the discretization algorithm developed in chapter 2 can be enhanced to speed-up the computations, and that all developments should be realized in accordance with professional standards. Thus, our second objective is to improve the discretization algorithm without increasing its complexity, to maintain its proper functioning for the applications of chapter 2 (i.e., non-regression testing), and to make all codes easily available to the community.

To achieve these two objectives, we will first describe the modifications brought to the model equations and to their derivatives. Then, we will explain how to simulate networks composed of high-lying nodes and partly-supplied pipes, using an enhanced version of the discretization algorithm developed in chapter 2. Next, we will show how to measure the gain of performances in the enhanced discretization algorithm, and how to compare the results obtained with and without taking account of the high-lying nodes and partly-supplied pipes. Finally, we will present the results obtained from the simulation of several dedicated WDNs, and discuss them.

3.2 Methods

This section first presents the modifications to bring to the equations of models $\{\mathrm{M1},\mathrm{M2},\mathrm{M3}\}$ defined in section 2.2.1 and of their derivatives, so they can simulate partly-supplied pipes with high-lying node(s) at their extremities. Next, it explains how to enhance and extend the discretization algorithm developed in chapter 2 to simulate a WDN that contains such kind of pipes and high-lying nodes. Finally, it presents the WDNs used for testing, and the computed metrics.

In absence of other indications, we use the same notation and unit convention as in chapters 1 and 2. In particular, scalar parameters and variables are denoted in italic (e.g., x), vectors in italic bold (e.g., $\boldsymbol{v}$), matrices in italic bold upper-case (e.g., $\boldsymbol{M}$), scalar functions in upright (e.g., $\mathrm{f}(x)$), vector functions in upright bold (e.g., $\mathbf{f}(\boldsymbol{v})$), matrix functions in upright bold upper-case (e.g., $\mathbf{M}(\boldsymbol{v})$), and sets in blackboard style (e.g., $\mathbb{R}$).

3.2.1 Modification of the equations of models M1, M2 and M3

To compute the flow rates $\boldsymbol{q_{0.5}}$ at the middle of the pipes and the heads $\boldsymbol{h}$ at the junction nodes in any leaky WDN, we need to solve the steady-state equilibrium equations, already defined in chapter 2 as:

$$\begin{pmatrix} \boldsymbol{\xi}_{\mathbf{f}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0}) - \boldsymbol{A}^T\boldsymbol{h} - \boldsymbol{A_0}^T\boldsymbol{h_0} \\ \boldsymbol{A}^-\,\mathbf{q}_{\boldsymbol{\ell}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0}) - \boldsymbol{A}^+\,\mathbf{q_0}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0}) - \mathbf{c}(\boldsymbol{h}) \end{pmatrix} = \boldsymbol{0},$$

where:

- $\boldsymbol{\xi}_{\mathbf{f}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$ is the vector function to calculate the friction head-losses in pipes,
- $\boldsymbol{A}$ is the junctions-pipes incidence matrix, $\boldsymbol{A_0}$ the sources-pipes incidence matrix and $\boldsymbol{h_0}$ is the fixed heads at source nodes (i.e., at tanks and reservoirs),
- $\boldsymbol{A}^+$ and $\boldsymbol{A}^-$ are respectively the positive and the negative parts of $\boldsymbol{A}$,
- $\mathbf{q_0}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$ and $\mathbf{q}_{\boldsymbol{\ell}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$ are the vector functions to compute the flow rates at respectively the start and the end of the pipes,
- and $\mathbf{c}(\boldsymbol{h})$ is the vector function to compute users' consumptions at junctions using the Wagner's pressure-outflow rate relation (POR).

$\boldsymbol{\xi}_{\mathbf{f}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$, $\mathbf{q_0}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$ and $\mathbf{q}_{\boldsymbol{\ell}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$ were defined for models $\{\mathrm{M0},\ldots,\mathrm{M3}\}$ at sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7. $\mathbf{c}(\boldsymbol{h})$ was defined for any junction by eq. (1.10).

In chapter 2, the function $\boldsymbol{\xi}_{\mathbf{f}}(\boldsymbol{q_{0.5}},\boldsymbol{h},\boldsymbol{h_0})$ permitted, for models $\{\mathrm{M0},\ldots,\mathrm{M3}\}$ and for each pipe k of the network, to compute the friction head-loss in k as:

$$\xi_{\mathrm{f},k}(\ell_k) = f_k \int_0^{\ell_k} \mathrm{q}_k(x)|\mathrm{q}_k(x)|^{\gamma_{HW}-1}\,\mathrm{d}x\,.$$

ℓ_k is the length of k and $\mathrm{q}_k(x)$ is the function to compute the flow rate along k. $\mathrm{q}_k(x)$ was defined

- for model M0 as:

  $$\mathrm{q}_k(x) = q_{0.5,k}$$

  where $q_{0.5}$ is the flow rate at the middle of k,
- and for models $\{\mathrm{M1},\mathrm{M2},\mathrm{M3}\}$ as:

  $$\mathrm{q}_k(x) = q_{0.5,k} - \int_{\ell_k/2}^{x} \mathrm{q}_{\mathrm{LL},k}(y)\,\mathrm{d}y\,,$$

where $\mathrm{q}_{\mathrm{LL},k}(y)$ is the function to compute the background lineic leakage outflow rate at any position y along k, defined by eqs. (2.21), (2.26) and (2.30).

In partly-supplied pipes, the flow rate necessarily becomes 0 at some position along the pipes. However, the flow rate computed by M0 is constant along k. Thus, M0 does not permit to simulate partly-supplied pipes. This limitation of model M0 also appears in the computation of the flow rates at pipe extremities by functions $\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O})$ and $\mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O})$, since $\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O}) = \mathbf{q_\ell}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O}) = \boldsymbol{q_{0.5}}$ for M0.

Conversely, the flow rate computed by $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$ varies along k according to background leakage outflow rate; thus, it is possible to simulate partly-supplied pipes with models $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$, but only after some adaptions. Indeed, in chapter 2, the function $\mathrm{q}_k(x)$ computed the flow rate at the start of each pipe k as:

$$q_{0,k} = \begin{cases} q_{0.5,k} + \displaystyle\int_0^{\ell_k} \mathrm{q_{LL}}_k(y)\,\mathrm{d}y & \text{if } q_{0.5,k} \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.1}$$

and the flow rate at the end of k as:

$$q_{\ell,k} = \begin{cases} q_{0.5,k} - \displaystyle\int_0^{\ell_k} \mathrm{q_{LL}}_k(y)\,\mathrm{d}y & \text{if } q_{0.5,k} \leq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

We can clearly see in eqs. (3.1) and (3.2) that the amount of leakage outflow rate considered to compute the flow rates at the start and the end of k depended only on the sign of the flow rate $q_{0.5,k}$ at the middle of k. However, the simulation of partly-supplied pipes needs $q_{0,k}$ to consider only the leakage outflow rates from the part of the pipe where the flow rate is *positive*, and $q_{\ell,k}$ to consider only the leakage outflow rates from the part of the pipe where the flow rate is *negative*. Thus, to simulate such kind of pipes, we will first find the position $\widetilde{x_{q=0,k}}$, if it exists, where the flow rate becomes 0 and changes of sign. From this position, we will then correct the eqs. (3.1) and (3.2), and show how it permits to also compute more accurate friction head-losses with models M2 and M3. Finally, we will explain how to compute the new derivatives of flow rates at pipe extremities and friction head-losses with models $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$, taking into account $\widetilde{x_{q=0,k}}$.

3.2.1.1 Determination of the position where the flow rate becomes 0 and changes of sign for models $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$

Paragraphs below explain how to compute the position $\widetilde{x_{q=0}}$ where the flow rate becomes 0, for each model $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$; the last paragraph defines the corrected values of q_0 and q_ℓ. We omit the subscript k for better readability.

Model M1: the function to compute the flow rate along a leaky pipe of length ℓ was defined in section 2.2.1.5, $\forall x \in [0, \ell]$, as:

$$\mathrm{q}^{\mathrm{M1}}(x) = q_{0.5} - \widetilde{q_{LL}} \left(x - \frac{\ell}{2} \right),$$

where $q_{0.5}$ is the flow rate at the middle of the pipe, and $\widetilde{q_{LL}}$ is the average lineic leakage outflow rate at any position along the pipe. $\widetilde{q_{LL}}$ was computed as:

$$\widetilde{q_{LL}} = \mathrm{q_{LL}}(\widetilde{p}) = \beta_L \left([\widetilde{p}]^+ \right)^{\alpha_L},$$

α_L and β_L are the leakage parameters (respectively the type of leakage and the level of degradation of the pipe), and $\widetilde{p}$ is the approximated average pressure-head in the pipe, computed as:

$$\widetilde{p} = \frac{p_0 + p_\ell}{2};$$

p_0 and p_ℓ are the pressure-heads at the start and end of the pipe, and $[...]^+$ is the positive-part function. Then, the root of $\mathrm{q}^{\mathrm{M1}}(x)$ is such that:

$$r^{\mathrm{M1}} = \frac{q_{0.5}}{\widetilde{q_{LL}}} + \frac{\ell}{2}. \tag{3.3}$$

Thus, if $0 < r^{\mathrm{M1}} < \ell$, the position where the flow rate becomes 0 and changes of sign is:

$$x^{\mathrm{M1}}_{q=0} = \frac{q_{0.5}}{\widetilde{q_{LL}}} + \frac{\ell}{2}. \tag{3.4}$$

Otherwise, if $r^{\mathrm{M1}} \leq 0$ or $r^{\mathrm{M1}} \geq \ell$, then the flow rate never changes of sign.

<u>Model M2</u>: the flow rate along the leaky pipe was computed in section 2.2.1.6 as:

$$\mathrm{q}^{\mathrm{M2}}(x) = q_{0.5} - \widehat{\mathrm{q}^{\mathrm{M2}}_{\mathrm{LL}}}(x)\left(x - \frac{\ell}{2}\right),$$

and the lineic leakage outflow rate $\widehat{\mathrm{q}^{\mathrm{M2}}_{\mathrm{LL}}}(x)$ as:

$$\widehat{\mathrm{q}^{\mathrm{M2}}_{\mathrm{LL}}}(x) = \frac{q_{LL\ell} - q_{LL0}}{\ell}\,\frac{x + \ell/2}{2} + q_{LL0}.$$

q_{LL0} and $q_{LL\ell}$ are the lineic leakage outflow rates at respectively the start and the end of the pipe, and are equal to:

$$q_{LL0} = \beta_L\left([p_0]^+\right)^{\alpha_L} \quad \text{and} \quad q_{LL\ell} = \beta_L\left([p_\ell]^+\right)^{\alpha_L}$$

(see section 2.2.1.6 for deeper explanation). Thus, the real roots of $\mathrm{q}^{\mathrm{M2}}(x)$, if they exist, are also real roots of the polynomial:

$$\mathrm{P}(x) = a^{\mathrm{M2}}_2\, x^2 + a^{\mathrm{M2}}_1\, x + a^{\mathrm{M2}}_0, \tag{3.5}$$

where

$$a^{\mathrm{M2}}_0 = q_{0.5} + \frac{\ell\,(3\,q_{LL0} + q_{LL\ell})}{8}, \quad a^{\mathrm{M2}}_1 = -q_{LL0}, \quad \text{and} \quad a^{\mathrm{M2}}_2 = \frac{q_{LL0} - q_{LL\ell}}{2\,\ell}. \tag{3.6}$$

If $q_{LL0} = q_{LL\ell}$, then only one real root exists; it can be computed, along with the position where the flow rate changes of sign, as for model M1 (see eqs. (3.3) and (3.4)). Otherwise (i.e., $q_{LL0} \neq q_{LL\ell}$), then eq. (3.5) has one or two roots, which can be calculated analytically using the quadratic and Viète's formulas [166, p. 1479]. For better clarity, the different steps of the method are explained in appendix C.1.1. After this calculation, if we found only one real root r^{M2}, and if $0 < r^{\mathrm{M2}} < \ell$, then the position along the pipe where the flow rate changes of sign is simply equal to r^{M2}:

$$x^{\mathrm{M2}}_{q=0} = r^{\mathrm{M2}}. \tag{3.7}$$

Otherwise, if eq. (3.5) has two real roots $\left\{r_i^{\mathrm{M2}},\ i \in \{1,2\}\right\}$, then the position is:

$$x_{q=0}^{\mathrm{M2}} = \begin{cases} \min\limits_{i\in\{1,2\}} \left(\left\{r_i^{\mathrm{M2}} \mid r_i^{\mathrm{M2}} \in \left]0,\ell\right[\ \right\}\right) & \text{if } q_{0.5} \geq 0, \\ \max\limits_{i\in\{1,2\}} \left(\left\{r_i^{\mathrm{M2}} \mid r_i^{\mathrm{M2}} \in \left]0,\ell\right[\ \right\}\right) & \text{otherwise.} \end{cases} \tag{3.8}$$

Finally, if eq. (3.5) has no real root, or if none of its real roots belong to $]0,\ell[$, then the flow rate never changes of sign.

Model M3: using the same notation as for models M1 and M2, the flow rate along the leaky pipe was computed in section 2.2.1.7 as:

$$\begin{aligned} \mathrm{q}^{\mathrm{M3}}(x) = q_{0.5} + \left(\frac{\ell}{2} - x\right) \Bigg(& \frac{2\left(q_{LL0} - 2\,\widetilde{q_{LL}} + q_{LL\ell}\right)}{3\,\ell^2} \left(x^2 + x\frac{\ell}{2} + \frac{\ell^2}{4}\right) \\ & - \frac{3\,q_{LL0} - 4\,\widetilde{q_{LL}} + q_{LL\ell}}{2\,\ell} \left(x + \frac{\ell}{2}\right) + q_{LL0} \Bigg). \end{aligned}$$

The roots of $\mathrm{q}^{\mathrm{M3}}(x)$, if they exists, are also roots of the polynomial:

$$\mathrm{P}(x) = a_3^{\mathrm{M3}}\, x^3 + a_2^{\mathrm{M3}}\, x^2 + a_1^{\mathrm{M3}}\, x + a_0^{\mathrm{M3}}, \tag{3.9}$$

where

$$a_3^{\mathrm{M3}} = -\frac{2\left(q_{LL0} - 2\,\widetilde{q_{LL}} + q_{LL\ell}\right)}{3\,\ell^2} \qquad a_2^{\mathrm{M3}} = \frac{3\,q_{LL0} - 4\,\widetilde{q_{LL}} + q_{LL\ell}}{2\,\ell} \tag{3.10}$$

$$a_1^{\mathrm{M3}} = -q_{LL0} \qquad a_0^{\mathrm{M3}} = q_{0.5} + \ell\, \frac{5\,q_{LL0} + 8\,\widetilde{q_{LL}} - q_{LL\ell}}{24}. \tag{3.11}$$

If $\widetilde{q_{LL}} = \left(q_{LL0} + q_{LL\ell}\right)/2$, then the root and the position can be obtained as for model M2 (see eqs. (3.7) and (3.8) and appendix C.1.1). Otherwise (i.e., $\widetilde{q_{LL}} \neq \left(q_{LL0} + q_{LL\ell}\right)/2$), then the root(s) of eq. (3.9) can be calculated analytically with the Cardano's formula [166, p. 364-365]. For better clarity, the different steps of the method are explained in appendix C.1.2. After this calculation, if we found only one real root r^{M3}, and if $0 < r^{\mathrm{M3}} < \ell$, then the position along the pipe where the flow rate changes of sign is:

$$x_{q=0}^{\mathrm{M3}} = r^{\mathrm{M3}}. \tag{3.12}$$

Otherwise, if we found three real roots $\left\{r_i^{\mathrm{M3}},\ i \in \{1,2,3\}\right\}$, then the position is:

$$x_{q=0}^{\mathrm{M3}} = \begin{cases} \min\limits_{i\in\{1,2,3\}} \left(\left\{r_i^{\mathrm{M3}} \mid r_i^{\mathrm{M3}} \in \left]0,\ell\right[\ \right\}\right) & \text{if } q_{0.5} \geq 0, \\ \max\limits_{i\in\{1,2,3\}} \left(\left\{r_i^{\mathrm{M3}} \mid r_i^{\mathrm{M3}} \in \left]0,\ell\right[\ \right\}\right) & \text{otherwise.} \end{cases} \tag{3.13}$$

Finally, if eq. (3.9) has no real root, or if none of its real roots belong to $]0,\ell[$, then the flow rate never changes of sign along the pipe.

$\widetilde{x_{q=0}}$ for any of $\{\mathrm{M1},\mathrm{M2},\mathrm{M3}\}$: for any model of $\{\mathrm{M1},\mathrm{M2},\mathrm{M3}\}$, the position $x_{q=0}$ computed above becomes ℓ if the flow rate is everywhere positive along the pipe, and 0 if it is everywhere negative. Thus, we can define the position $\widetilde{x_{q=0}}$ at which the flow rate becomes 0 and changes of sign as:

$$\widetilde{x_{q=0}} = \begin{cases} x_{q=0} & \text{if the flow rate changes of sign along the pipe,} \\ l & \text{if the flow rate is \textit{positive} everywhere along the pipe,} \\ 0 & \text{if the flow rate is \textit{negative} everywhere along the pipe.} \end{cases} \tag{3.14}$$

Finally, the corrected flow rates at the start and the end of the pipe can now be computed as respectively:

$$q_0 = q_{0.5} + \int_0^{\widetilde{x_{q=0}}} \mathrm{q}_{\mathrm{LL}k}(y)\,\mathrm{d}y \tag{3.15}$$

and

$$q_\ell = q_{0.5} - \int_{\widetilde{x_{q=0}}}^{\ell} \mathrm{q}_{\mathrm{LL}k}(y)\,\mathrm{d}y\,. \tag{3.16}$$

3.2.1.2 More accurate friction head-losses using models M2 and M3

In the previous sub-section, we computed the exact position $\widetilde{x_{q=0}}$, if it exists, where the flow rate becomes 0 and changes of sign. We will now use this position to compute more accurate friction head-losses for models M2 and M3.

For models M2 and M3, the friction head-loss is computed along any pipe with a Newton-Cotes formula of degree 2, as:

$$\xi_f = \frac{\ell}{6}\Big(\varphi(0) + 4\,\varphi(\ell/2) + \varphi(\ell)\Big), \tag{3.17}$$

where $\varphi(x)$ is the unitary friction head-loss function, defined, $\forall x \in [0,\ell]$, as

$$\varphi(x) = f\,\mathrm{q}(x)|\mathrm{q}(x)|^{\gamma_{HW}-1}. \tag{3.18}$$

In eq. (3.18), $\mathrm{q}(x)$ represents the function to compute the flow rate along the pipe, defined by eq. (2.27) for M2, and by eq. (2.31) for M3, and $\gamma_{HW} = 1.852$. For model M2, $\mathrm{q}(x)$ is quadratic, and so $\varphi(x) \propto x^{3.704}$. For model M3, $\mathrm{q}(x)$ is cubic, therefore $\varphi(x) \propto x^{5.556}$. Thus, for both models M2 and M3, $\varphi(x)$ has a minimum between 0 and ℓ in any pipe where the flow rate changes of sign. However, the Newton-Cotes formula of degree 2 allows the exact integration of polynomials of degree 3. Thus, if the flow rate changes of sign along the pipe, the Newton-Cotes formula used in eq. (3.17) is not accurate enough to model the slope (i.e., 1st derivative estimate) and the curvature (i.e., 2nd derivative estimate) of the friction head-loss function everywhere along the pipe. Hence, to compute ξ_f accurately in such pipes, we need to integrate eq. (3.18) by sub-part: over the sub-part where the flow rate is positive (ξ_f^+), and over the one where the flow rate is negative (ξ_f^-), using the position $\widetilde{x_{q=0}}$ computed above. This way, eq. (3.17) becomes:

$$\begin{aligned}\xi_f = \xi_f^+ + \xi_f^- = {} & \frac{\widetilde{x_{q=0}}}{6}\left(\varphi(0) + 4\,\varphi\!\left(\frac{\widetilde{x_{q=0}}}{2}\right) + \varphi\!\left(\widetilde{x_{q=0}}\right)\right) \\ & + \frac{\ell - \widetilde{x_{q=0}}}{6}\left(\varphi\!\left(\widetilde{x_{q=0}}\right) + 4\,\varphi\!\left(\frac{\widetilde{x_{q=0}} + \ell}{2}\right) + \varphi(\ell)\right). \end{aligned} \tag{3.19}$$

3.2.1.3 More accurate derivatives of flow rates and friction head-losses for models $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$

Using the position $\widetilde{x_{q=0}}$ permits to also compute more accurate derivatives of flow rates and friction head-losses for models $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$.

Indeed, the Jacobian of mass residuals at junctions with respect to heads at junctions is defined as:

$$\frac{\partial \boldsymbol{\rho_m}}{\partial \boldsymbol{h}} = \boldsymbol{A}^{-}\,\frac{\partial \mathbf{q_\ell}}{\partial \boldsymbol{h}} - \boldsymbol{A}^{+}\,\frac{\partial \mathbf{q_0}}{\partial \boldsymbol{h}} - \frac{\partial \mathbf{c}}{\partial \boldsymbol{h}}. \tag{3.20}$$

In chapter 2, the vector functions $\partial \mathbf{q_0}/\partial \boldsymbol{h}$ and $\partial \mathbf{q_\ell}/\partial \boldsymbol{h}$ computed, for any pipe k and any junction i of the network, the derivatives of the flow rate at the start and end of k as respectively:

$$\frac{\partial \mathrm{q_0}_k}{\partial h_i} = \begin{cases} \displaystyle\int_0^{\ell_k} \frac{\partial \mathrm{q_{LL}}_k}{\partial h_i}(x)\,\mathrm{d}x & \text{if } k \in \mathbb{P}_i^+ \text{ and } q_{0.5,k} \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.21}$$

and

$$\frac{\partial \mathrm{q}_{\ell k}}{\partial h_i} = \begin{cases} \displaystyle -\int_0^{\ell_k} \frac{\partial \mathrm{q_{LL}}_k}{\partial h_i}(x)\,\mathrm{d}x & \text{if } k \in \mathbb{P}_i^- \text{ and } q_{0.5,k} \leq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.22}$$

where $\mathrm{q_{LL}}_k$ is the function to compute the background lineic leakage outflow rate along k using any of the models $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$ (see eqs. (2.21), (2.26) and (2.30)), and $\mathbb{P}_i^+$ and $\mathbb{P}_i^-$ are the sets of pipes respectively leaving and entering the junction i. In current chapter, integrating $\partial \mathrm{q_{LL}}_k/\partial h_i$ according to the sign of the flow rate along k, we now have:

$$\frac{\partial \mathrm{q_0}_k}{\partial h_i} = \begin{cases} \displaystyle\int_0^{\widetilde{x_{q=0,k}}} \frac{\partial \mathrm{q_{LL}}_k}{\partial h_i}(x)\,\mathrm{d}x & \text{if } k \in \mathbb{P}_i^+ \text{ and } q_k(x) \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.23}$$

and

$$\frac{\partial \mathrm{q}_{\ell k}}{\partial h_i} = \begin{cases} \displaystyle -\int_{\widetilde{x_{q=0,k}}}^{\ell_k} \frac{\partial \mathrm{q_{LL}}_k}{\partial h_i}(x)\,\mathrm{d}x & \text{if } k \in \mathbb{P}_i^- \text{ and } q_k(x) \leq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.24}$$

Using eqs. (3.23) and (3.24) rather than eqs. (3.21) and (3.22) permits to compute more precise derivatives of the flow rates.

Also, the Jacobian of energy residuals in pipes with respect to the heads at junctions is defined as:

$$\frac{\partial \boldsymbol{\rho_e}}{\partial \boldsymbol{h}} = \frac{\partial \boldsymbol{\xi}_{\mathbf{f}}}{\partial \boldsymbol{h}} - \boldsymbol{A}^T, \tag{3.25}$$

In chapter 2, the vector function $\partial \boldsymbol{\xi}_{\mathbf{f}}/\partial \boldsymbol{h}$ computed, for any pipe k and any junction i of the network, the derivatives of the friction head-loss in k as:

$$\frac{\partial \xi_{\mathrm{f},k}}{\partial h_i} = \begin{cases} \displaystyle f_k \int_0^{\ell_k} \frac{\partial \varphi_k}{\partial h_i}(y)\,\mathrm{d}y & \text{if } \left\{k \in \mathbb{P}_i^+ \text{ and } q_{0.5,k} \geq 0\right\} \text{ or } \left\{k \in \mathbb{P}_i^- \text{ and } q_{0.5,k} \leq 0\right\}, \\ 0 & \text{otherwise,} \end{cases} \tag{3.26}$$

where φ_k is the function to compute the unitary friction head-loss along k using any of the models $\{\text{M1}, \text{M2}, \text{M3}\}$ (see eqs. (2.24), (2.29) and (2.34)). In current chapter, integrating $\partial\varphi_k/\partial h_i$ according to the sign of the flow rate along k leads to:

$$\frac{\partial \xi_{\text{f},k}}{\partial h_i} = \begin{cases} f_k \displaystyle\int_0^{\widetilde{x_{q=0,k}}} \frac{\partial \varphi_k}{\partial h_i}(y)\,\mathrm{d}y & \text{if } k \in \mathbb{P}_i^+ \text{ and } q_k(x) \geq 0, \\ f_k \displaystyle\int_{\widetilde{x_{q=0,k}}}^{\ell_k} \frac{\partial \varphi_k}{\partial h_i}(y)\,\mathrm{d}y & \text{if } k \in \mathbb{P}_i^- \text{ and } q_k(x) \leq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.27}$$

Equation (3.27) calculates more correct derivative of the friction head-loss than eq. (3.26).

Computing more accurate derivatives of flow rates and friction head-losses should permit the Newton's method defined in section 2.2.2.2 to converge in fewer iterations and improve its stability.

3.2.2 Discretization of partly-supplied pipes

In previous sub-section, we saw how to determine the pipes where the flow rates become 0. These pipes are partly-supplied if there is an high-lying node with negative of zero pressure-head at exactly one of their extremities. When the pressure-head is negative or zero at both extremities of a pipe, then the pipe is fully-unsupplied. Finally, when the flow rates becomes 0 in a pipe that does not have any extremity with zero or negative pressure, then it means that flows are entering from both extremities of the pipe, and the pipe is fully-supplied.

However, in each pipe, models $\{\text{M1}, \text{M2}, \text{M3}\}$ still integrate the lineic leakage outflow rate along the full pipe, whatever the status (i.e., fully-supplied, fully-unsupplied or partly-supplied) of the pipe. This can result in over-leakage outflow rates. Thus, to prevent these excessive outflow rates, we reuse the discretization algorithm developed in section 2.2.1.4, splitting each partly-supplied pipe until convergence. Like so, at the convergence of the algorithm, each partly-supplied pipe of the undiscretized network is described by a set of fully supplied and fully unsupplied sub-pipes, in which the leakage models $\{\text{M1}, \text{M2}, \text{M3}\}$ can compute the background leakage outflow rates accurately. Finally, along each partly-supplied discretized pipe, the position of the last supplied (i.e., no high-lying) intermediate junction is a good approximation of the position from where the pressure-head becomes negative, and thus from where the pipe is not supplied anymore. Then, from this position, the lineic leakage outflow rate is not integrated, and therefore does not contribute to the global leakage outflow rate of the pipe.

The model M0 alone does not permit to find the position along the pipes, if these positions exist, where the flow rates become 0, because the function of M0 that computes the flow rate is constant along the pipes. However, if used in the discretization algorithm, the model M0 should be able to find these positions iteratively, and compute accurate background leakage outflow rates as models $\{\text{M1}, \text{M2}, \text{M3}\}$.

3.2.3 Numerical enhancements of the discretization algorithm

High-lying nodes and strong levels of leakages can induce a significant variation of pressure-heads along the pipes. Since the convergence criterion of the discretization algorithm is based, at each iteration, on the difference between the heads at junctions computed at the current and at the previous iterations (see section 2.2.1.4), these high variations of pressure-heads along the pipes can increase considerably the number of iterations needed to reach convergence. Thus, we propose here two numerical enhancements to speed up the discretization algorithm. The version of the discretization algorithm that includes these enhancements will be called hereafter “enhanced discretization algorithm”.

3.2.3.1 Test of convergence

At each iteration s of the discretization algorithm, once the discretized network has been simulated, we check for the convergence of the algorithm by comparing, along each discretized pipe, the heads at (intermediate-)nodes computed at the previous iteration, $\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}$, with the heads at (intermediate-)nodes computed at the current iteration, $\boldsymbol{h}_{\boldsymbol{N}}^{(s)}$. Since the number of intermediate nodes increases from the previous to the current iteration, the sizes of vectors $\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}$ and $\boldsymbol{h}_{\boldsymbol{N}}^{(s)}$ are different, and thus cannot be compared with each other as it. Then, we proposed, in section 2.2.1.4 of chapter 2, to compute the values of $\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}$ at the positions of $\boldsymbol{h}_{\boldsymbol{N}}^{(s)}$ by Piecewise Cubic Hermite Interpolating Polynomial (PCHIP) interpolation, to obtain the interpolated heads $\widetilde{\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}}$. However, the use of a PCHIP interpolator is time consuming and can bring some numerical error.

Thus, to accelerate the convergence test, we propose to not use PCHIP interpolation anymore, but to rather compute $\widetilde{\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}}$ analytically, from $\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}$ and from the flow rates $\boldsymbol{q_{0.5}}^{(s-1)}$ obtained at the previous iteration. To do so, we first compute, in each (sub-)pipe k of the network at iteration $s-1$ that has been discretized in two sub-pipes at iteration s, the friction head-loss until the middle of k, as:

$$\xi_{f,\ell_k/2}^{(s-1)} = \xi_{\mathrm{f}}\left(\frac{\ell_k^{(s-1)}}{2}, q_{0.5,k}^{(s-1)}, h_{0,k}^{(s-1)}, h_{\ell,k}^{(s-1)}\right), \tag{3.28}$$

where $\ell_k^{(s-1)}$, $q_{0.5,k}^{(s-1)}$, $h_{0,k}^{(s-1)}$ and $h_{\ell,k}^{(s-1)}$ are respectively the length, the flow rate, the start head and the end head of k at iteration $s-1$, and ξ_{f} is the function to compute the friction head-loss along k with any leakage model (see sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7). Then, we can compute the head at the middle of k at iteration $s-1$ as:

$$h_{\ell_k/2}^{(s-1)} = h_{0,k}^{(s-1)} - \xi_{f,\ell_k/2}^{(s-1)}. \tag{3.29}$$

Finally, we obtain $\widetilde{\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}}$ by inserting all $h_{\ell_k/2}^{(s-1)}$ at their right positions into $\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}$.

3.2.3.2 Smart initialization of heads and flows at each iteration of the discretization algorithm

In section 2.2.1.4, at each iteration s of the discretization algorithm and for each (sub-)pipe k, we choose the initial guess of the flow rate at the middle of k, ${q_{0.5,k}^{init}}^{(s)}$, to be consistent with a velocity of $0.5\,\mathrm{m\,s^{-1}}$. Also, we set the initial head at any junction i, $h_i^{init(s)}$, equal to $p_m + u_i + (p_s - p_m)/5$, where p_m and p_s are the minimum and the service pressure-heads, and u_i is the elevation at the junction.

However, the values of $\boldsymbol{q_{0.5}}^{(s)}$ and $\boldsymbol{h}^{(s)}$ at equilibrium at iteration s do not change much from the values $\boldsymbol{q_{0.5}}^{(s-1)}$ and $\boldsymbol{h}^{(s-1)}$ at equilibrium at iteration $s-1$, and are anyway closer to each other than to values of ${q_{0.5,k}^{init}}^{(s)}$ and $h_i^{init(s)}$ computed empirically as in the previous paragraph and section 2.2.1.4.

Thus, to speed up the simulation at each iteration s of the discretization algorithm, we choose hereafter to rather compute the flow rates $\boldsymbol{q_{0.5}^{init^{(s)}}}$ and the heads $\boldsymbol{h}^{\boldsymbol{init}^{(s)}}$ from the values of $\boldsymbol{q_{0.5}}^{(s-1)}$ and $\boldsymbol{h}^{(s-1)}$.

$\boldsymbol{h}^{\boldsymbol{init}^{(s)}}$ is equal to the $\widetilde{\boldsymbol{h}_{\boldsymbol{N}}^{(s-1)}}$ that we already computed at section 3.2.3.1 (see section 3.2.3.1 to know how to compute it). To compute $\boldsymbol{q_{0.5}^{init^{(s)}}}$, we first calculate, in each (sub-)pipe k of the network at iteration $s-1$ that has been discretized in two sub-pipes at iteration s, the flow rate at positions $x^{(s-1)} \in \left\{\ell_k^{(s-1)}/4, 3\,\ell_k^{(s-1)}/4\right\}$ of k, as:

$$q_x^{(s-1)} = \mathrm{q}\Big(x^{(s-1)}, q_{0.5,k}^{(s-1)}, h_{0,k}^{(s-1)}, h_{\ell,k}^{(s-1)}\Big)\,, \tag{3.30}$$

where $\ell_k^{(s-1)}$, $q_{0.5,k}^{(s-1)}$, $h_{0,k}^{(s-1)}$ and $h_{\ell,k}^{(s-1)}$ are respectively the length, the flow rate, the start head and the end head of k at iteration $s-1$, and q is the function to compute the flow rate along k with any leakage model (see sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7). Then, to obtain $\boldsymbol{q_{0.5}^{init^{(s)}}}$, we just need to insert all $q_x^{(s-1)}$ at their right positions into $\boldsymbol{q_{0.5}}^{(s-1)}$.

3.2.4 Unsupplied node(s) and pipe(s) at downstream of high-lying node(s)

Once the Newton's method (section 2.2.2.2) converged to find the middle flow rates $\boldsymbol{q_{0.5}}$ and the heads at junctions $\boldsymbol{h}$ at equilibrium, the pressure-heads at junctions and the background leakage outflow rates in pipes are consistent with $\boldsymbol{q_{0.5}}$ and $\boldsymbol{h}$. However, in a network composed of high-lying node(s) or/and partly-supplied pipe(s), some parts of the network can become disconnected from any of the sources, while the pressure-heads in these parts remain positive and the flow rates remain not zero to fulfill users' demands and leakage outflow rates. Thus, in these disconnected parts, all flow rates (i.e., flows at middle of pipes, user-consumptions and leakage outflow rates) and heads are fictitious, and should not be considered in the final result. This artifact is due to the absence of constraint other than heads at sources and users' demands at junctions.

Having non-zero flows in the disconnected parts of the network can lead to extra upstream flows, and then to a slightly excessive friction head-losses in some parts of the network. However, setting the users' demands and the lineic leakage outflow rates to 0 in all the disconnected parts of the network, and running a post-processing simulation should permit to obtain accurate enough upstream flow rates and heads for most applications.

Finally, the flow rates and the heads at junctions in the disconnected parts of the network cannot be determined mathematically, because they are, by definition, not connected to any source nodes. Thus, for physical consistency matter, we choose to arbitrary set the heads at the disconnected junctions to the value of their elevations. Like so, none of the junctions have negative pressure-head, and the pressure-head at disconnected junctions is always 0. Also, to conserve the mass and energy in the whole system, we set the flow rates and the background leakage outflow rates in the disconnected pipes to 0.

3.2.5 Implementation

We extend the implementation of the Python simulator and of the discretization algorithm already developed in chapter 2, integrating the new developments of sections 3.2.1 to 3.2.4, and we deployed our new simulator through the Python framework OOPNET [147, 148].

We still use the sparse solver UMFPACK (Unsymmetric MultiFrontal PACKage) [32] to compute the descent direction on $\boldsymbol{h}^{(m)}$ at each iteration m of the Newton's method (section 2.2.2.2). But, to check for the convergence of the enhanced discretization algorithm, we now compute analytically the previous heads at new intermediate junctions (see section 3.2.3.1). Thus, we do not need anymore PCHIP interpolator during the discretization process, which should lead to a significant gain in terms of computational time.

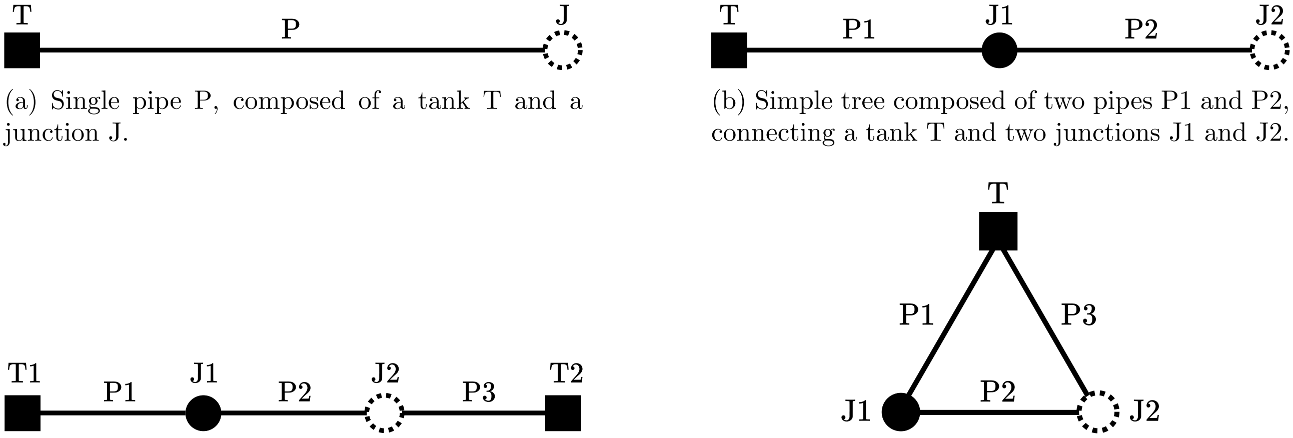
At each iteration of the Newton's method, we raise an exception that stops the simulation with a message of error as soon as the derivative of a friction head-loss with respect to flow rate becomes negative. That way, we check that the simulator always computes realistic derivatives. Indeed, in any pipe, a decrease of the friction head-loss when the flow rate increases is not physically realist.

3.2.6 Test networks and metrics

In this section, we propose several networks and metrics to check that our method allows the simulation of high-lying nodes and partly-supplied pipes, and that the numerical enhancements implemented speed up the computations.

3.2.6.1 Simple networks with high-lying node

We first simulate the simple leaky networks of fig. 3.2. These networks permit to check the good functioning of the models and to validate the method in an iterative and reproducible manner. They allow early detection, identification and corrections of anomalies, which would be much more difficult if we simulated directly a large real network.



(a) Single pipe P, composed of a tank T and a junction J.

(b) Simple tree composed of two pipes P1 and P2, connecting a tank T and two junctions J1 and J2.

(c) Path between two tanks T1 and T2, composed of three pipes P1, P2 and P3, and two junctions J1 and J2.

(d) Simple loop composed of three pipes P1, P2 and P3, connecting a tank T and two junctions J1 and J2.

Figure 3.2: Top view (i.e., the elevation is not shown) of the simple networks to simulate. Tanks are represented by ■, junctions at zero elevation by ●, and high-lying nodes by ◌.

The network in fig. 3.2a is a single pipe of length 1,500 m, connecting a tank to a junction. The one in fig. 3.2b is a simple tree, made of two pipes of length 750 m each, and two junctions. The network in fig. 3.2c is a path between two tanks and two junctions, composed of three pipes of length 500 m each. The network in fig. 3.2d is a simple loop, composed of three pipes of length 500 m each, a tank, and two junctions.

In each network of fig. 3.2, there is one high-lying node of elevation 15 m (◌ on the figure). The networks of figs. 3.2b to 3.2d have one additional junction of elevation 0 m. The head at all tanks of all networks is $10\,\mathrm{mH_2O}$.

The user's demand at junctions is $10\,\mathrm{l\,s^{-1}}$ in all networks. The pipes have a diameter of 200 mm, and an Hazen-Williams roughness coefficient of 120 (unit-less). The background leakages are described by the exponent $\alpha_L = 0.9$ and the coefficient $\beta_L = 2 \times 10^{-5}\,\mathrm{l\,s^{-1}\,m^{-\alpha_L - 1}}$, which correspond to a usual level of background leakage outflow rate [55, 57]. The minimum and service pressure-heads are respectively 0 and $20\,\mathrm{mH_2O}$, which are also values commonly used in the literature [35, 43, 130].

Since the two tanks in the network of fig. 3.2c have the same head, this network is equivalent to the one of fig. 3.2d. Thus, the networks of figs. 3.2c and 3.2d should produce exactly the same results.

3.2.6.2 Network C-Town

Next, we simulate the larger leaky network C-Town already used in chapter 2 (see section 2.2.5.1). This network contains no equipment, but still permits to validate our method in a more comprehensive way, and to check the reliability and the performance of our algorithms. Our version of C-Town is derived from the one of [57, 120] (see fig. 2.3a).

We first use our C-Town version to test the smart initialization of flows and heads at each iteration of the enhanced discretization algorithm (see section 3.2.3.2). Next, we modify the network to make it contains high-lying nodes; to do so, in each sector of the network composed of a tank, we choose a junction randomly, and lift it so its elevation becomes 5 m higher than the head at the tank (fig. 3.3).

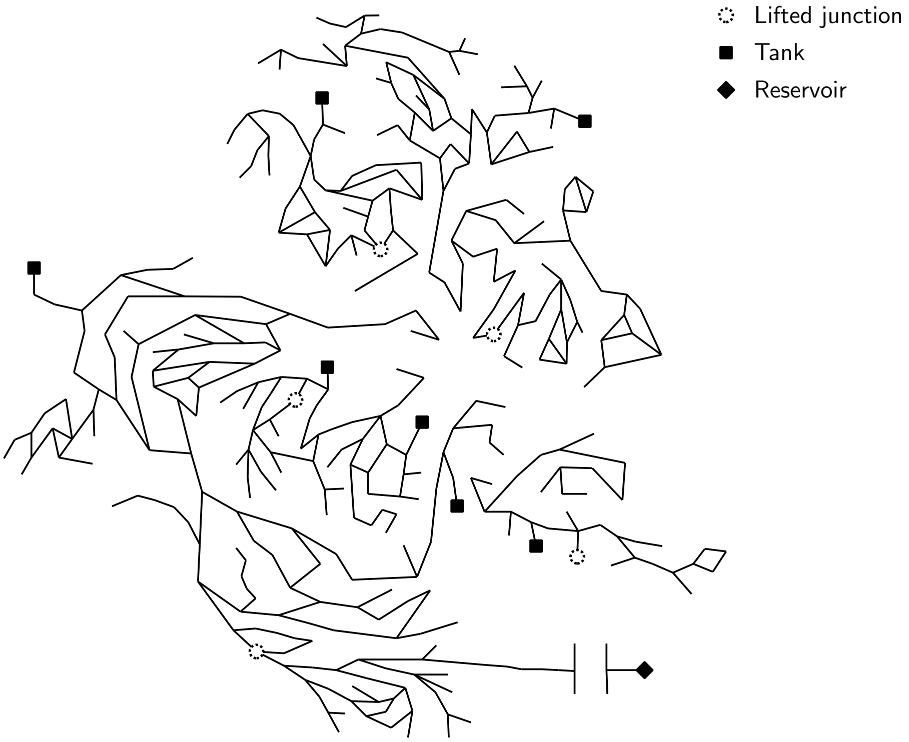


Figure 3.3: Top view (i.e., the elevation is not shown) of the leaky network C-Town ([57, 120]), without equipment, but with one lifted junction in each sector containing a tank.

3.2.6.3 Metrics

To assess the benefit of the smart initialization of flows and heads at each iteration of the discretization algorithm, we measure the CPU time elapsed (in s) during the whole discretization process, with each leakage model $i \in \{\mathrm{M0}, \ldots, \mathrm{M3}\}$, first without and next with smart initialization. We then obtain, for each leakage model, a couple of elapsed CPU times $\{t_{CPU,i}^{without}, t_{CPU,i}^{with}\}$, from which we can compute the relative gain of elapsed CPU time, $\delta_{CPU,i}$ (in %), as:

$$\delta_{CPU,i} = t_{CPU,i}^{with}/t_{CPU,i}^{without} \times 100 - 100. \tag{3.31}$$

Next, to check the convergence and equivalence of all leakage models when applying the enhanced discretization algorithm to networks with high-lying node(s) (i.e., networks of figs. 3.2 and 3.3), we compare the user-consumptions at each junction and the leakage outflow rates in each pipe, obtained at the end of the discretization process for each network and with all leakage models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$, and we check that they differ by less than $10^{-2}\,\mathrm{l\,s^{-1}}$. For each network, we also determine the absolute tolerance to use in the convergence criteriin eq. (2.20) for all leakage models to become equivalent. Finally, we check that we obtain the same results for the path between two tanks (fig. 3.2c) and the simple loop (fig. 3.2d).

Then, for each network of figs. 3.2a to 3.2c, we plot the profiles along pipes of the lineic leakage outflow rates obtained with each leakage model, the profiles of elevations, and the hydraulic grade lines (HGLs). Also, we compare, for each network of figs. 3.2 and 3.3, the number of iterations $s_{c,i} + 1$, the number of sub-pipes $n_{p,i}^{(s_c+1)}$, and the elapsed CPU time $t_{CPU,i}$ (in s), obtained when applying the enhanced discretization algorithm with each leakage model $i \in \{\text{M0}, \ldots, \text{M3}\}$. Finally, we check that the metrics associated to the path between two tanks (fig. 3.2c) and the simple loop (fig. 3.2d) are identical.

Next, to compare the results obtained from the enhanced discretization algorithm to the ones obtained without discretization, we plot the profiles of the lineic leakage outflow rates along the pipes of the networks of figs. 3.2a to 3.2c, using the leakage model M2 (i.e., lineic leakage outflow rate affine along each pipe; see section 2.2.1.6), and the HGLs. We choose the model M2 to better see the differences between the results obtained before and after discretization. Then, we check for the consistency of the profiles of lineic leakage outflow rates with the ones of elevations, and for the coherence of the locations along the pipes where the pressure-head becomes 0. Likewise, we plot the state of the network C-Town with lifted junctions (fig. 3.3) after application of the algorithm, to check that the simulated unsupplied junctions, partly-supplied pipes and unsupplied pipes are the expected ones.

Then, to quantify the better accuracy of the enhanced discretization algorithm, we compare for each network the global demand satisfaction d_s^{glob} (in %), computed as:

$$d_s^{glob} = \frac{\sum_i c_i}{\sum_i d_i} \times 100, \; i \in \{1, \ldots, n_j\}, \tag{3.32}$$

and obtained before ($d_{s,bef}^{glob}$) and after ($d_{s,aft}^{glob}$) discretization, using the leakage model M2. Also, we compute the absolute difference between these global demand satisfactions (in %), as:

$$\Delta d_s^{glob} = \left| d_{s,bef}^{glob} - d_{s,aft}^{glob} \right|. \tag{3.33}$$

Finally, we compare for each network the total leakage outflow rate q_L^{tot} (in $\text{l}\,\text{s}^{-1}$), computed as:

$$q_L^{tot} = \sum_k q_{L,k}, \; k \in \{1, \ldots, n_p\}, \tag{3.34}$$

obtained before ($q_{L,bef}^{tot}$) and after ($q_{L,aft}^{tot}$) discretization, and we compute the relative difference between these total leakage outflow rates (in %) as:

$$\delta q_L^{tot} = \frac{q_{L,aft}^{tot} - q_{L,bef}^{tot}}{q_{L,bef}^{tot}} \times 100. \tag{3.35}$$

3.3 Results and discussion

This section presents the results obtained when running the tests explained in section 3.2.6.

3.3.1 Smart initialization of flow rates and heads in the discretization algorithm

At each iteration of the enhanced discretization algorithm, the smart initialization of flows and heads, by the flows and heads obtained at the convergence of the previous iteration (see section 3.2.3.2), permits to reduce significantly the CPU time elapsed during the whole process. We observe this gain with all leakage models, from −1 % with model M0, to −10 % with model M3 (table 3.1).

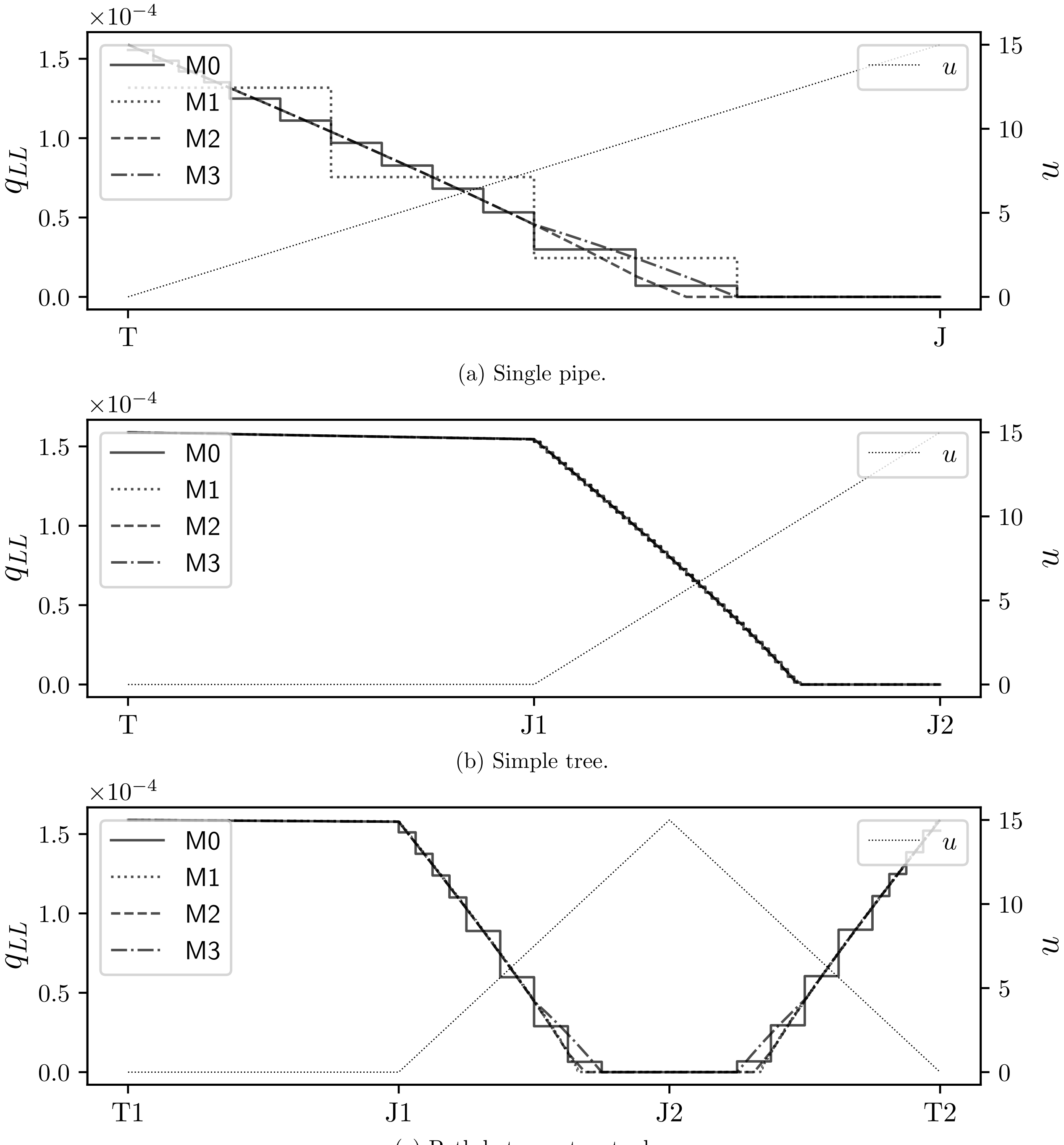
Table 3.1: CPU times elapsed (in s) when applying the enhanced discretization algorithm on network C-Town (fig. 2.3), using each model of background leakages $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ (sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7), without ($t_{CPU}^{without}$) and with (t_{CPU}^{with}) smart initialization. Values in column "Relative gain" are the relative differences (in %) between the CPU times elapsed without smart initialization and the ones with smart initialization (e.g., -9 means that smart initialization permits to reduce the CPU time elapsed by 9 %).

Leakage model	Without smart initialization ($t_{CPU}^{without}$)	With smart initialization (t_{CPU}^{with})	Relative gain (%) ($= t_{CPU}^{with}/t_{CPU}^{without} \times 100 - 100$)
0	66	65	-1
1	55	52	-5
2	45	41	-9
3	46	42	-10

3.3.2 Equivalence of leakage models

Each leakage model, when used in the enhanced discretization algorithm, permits to simulate the simple networks and the network C-Town with high-lying nodes. Also, all models lead to users' consumptions and leakage outflow rates that differ by less than $10^{-2}\,\mathrm{l\,s^{-1}}$, as soon as the pipes are discretized in enough sub-pipes. To obtain a sufficient discretization of pipes while keeping the computational times acceptable, we need to set the absolute tolerance used in the convergence criterion of the discretization algorithm (eq. (2.20)) to $10^{-6}\,\mathrm{mH_2O}$ for the single pipe (fig. 3.2a) and the simple tree (fig. 3.2b), $10^{-5}\,\mathrm{mH_2O}$ for the path between two tanks (fig. 3.2c) and the simple loop (fig. 3.2d), and $10^{-3}\,\mathrm{mH_2O}$ for the network C-Town (fig. 3.3). As expected, we obtain exactly the same results for the path between two tanks and the simple loop.

The profiles of the lineic leakage outflow rates along the single pipe and the path between two tanks show significant differences between the leakage models (figs. 3.4a and 3.4c); these differences are due to the contrasting numbers of sub-pipes obtained at the end of the discretization process with each model (tables 3.2a and 3.2c, column $n_p^{(s_c+1)}$). The profiles along the simple tree look closer between leakage models (fig. 3.4b), despite the same discrepancy in the number of sub-pipes (table 3.2b); in this case the high numbers of sub-pipes hide the differences on the plot.



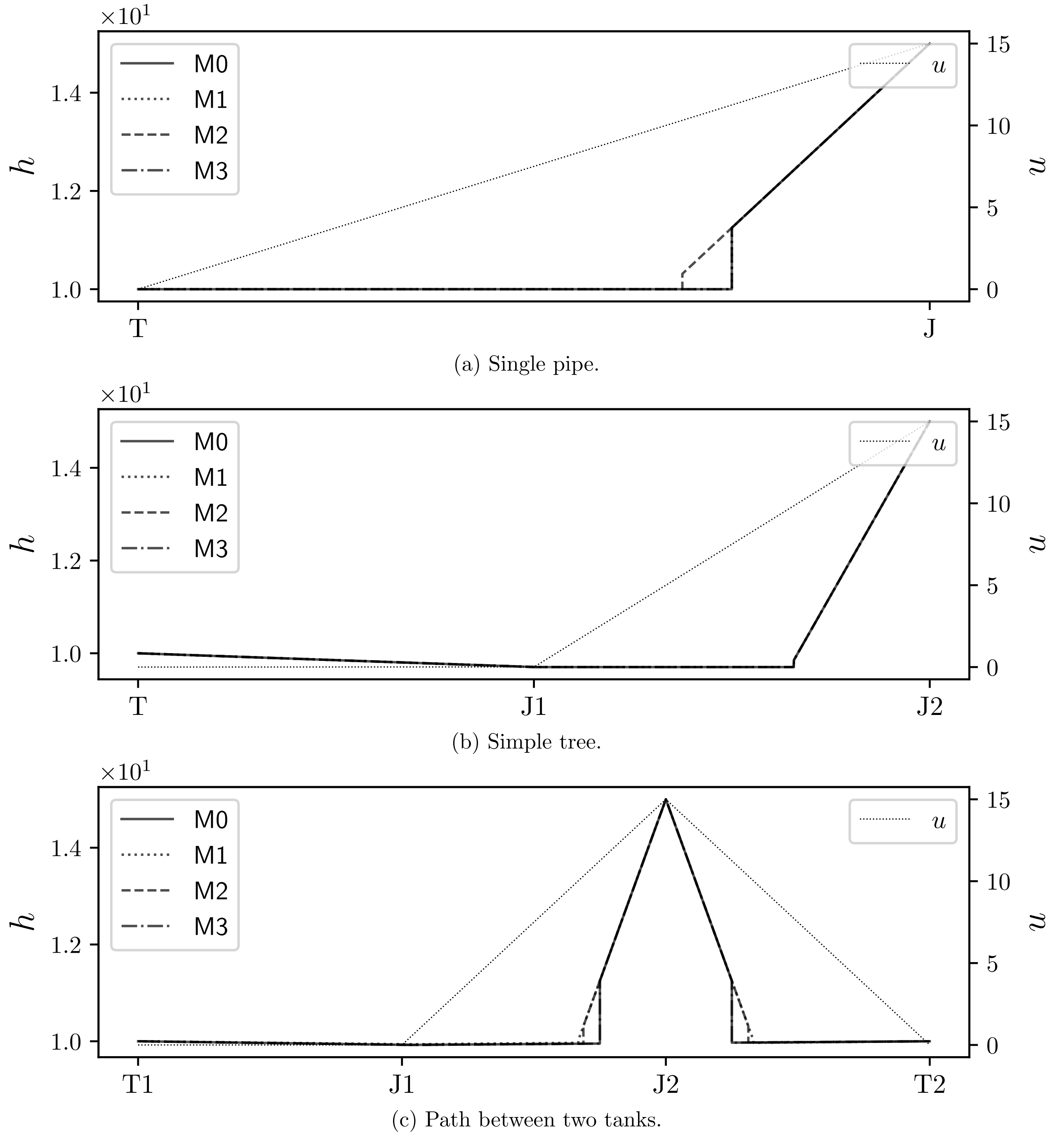
(a) Single pipe.

(b) Simple tree.

(c) Path between two tanks.

Figure 3.4: Profiles of lineic leakage outflow rates (q_{LL}, in $\mathrm{l\,s^{-1}\,m^{-1}}$) along the pipe(s) of the simple networks (fig. 3.2), for each leakage model $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ (sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7), after applying the enhanced discretization algorithm. Lines u show the profiles of elevation (in m).

The hydraulic grade lines (HGLs) in the single tree are graphically the same for all models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ (fig. 3.5b). For the single pipe and the path between two tanks, the HGLs look also the same, except near the positions where the slopes change strongly (figs. 3.5a and 3.5c). These positions correspond to the limits after which the pressure-heads become zero; then, after these positions, the HGLs just follow the elevations along the pipes, as explained in section 3.2.4. We can also observe these differences between the models on the profiles of lineic leakage outflow rates in figs. 3.4a and 3.4c. Finally, in all other parts of the networks and for all models, the HGLs are very close because the friction head-loss functions all have a degree at least $\gamma_{\scriptscriptstyle HW} = 1.852$; thus, after some discretizations, the differences between the models are not anymore visible graphically.



(a) Single pipe.

(b) Simple tree.

(c) Path between two tanks.

Figure 3.5: Hydraulic grade lines (in mH_2O) along the pipe(s) of the simple networks (fig. 3.2), for each leakage model $\{M0, \ldots, M3\}$ (sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7), after applying the enhanced discretization algorithm. Lines u show the profiles of elevation (in m).

For all networks and all leakage models, the number of iterations of the enhanced discretization algorithm is between 2 and 8 (tables 3.2a to 3.2e, column $s_c + 1$). The leakage model M3 is the one that needs the least numbers of sub-pipes, whatever the simulated network. Next, M1 needs the fewest number of sub-pipes for the single pipe and the simple tree (tables 3.2a and 3.2b), but its numbers explode for the path between two tanks, the simple loop and network C-Town (tables 3.2c to 3.2e). Finally, model M0 needs globally lesser sub-pipes than M2.

Table 3.2: Number of iterations ($s_c\!+\!1$), number of sub-pipes ($n_p^{(s_c+1)}$), and elapsed CPU times (t_{CPU}, in s) when applying the enhanced discretization algorithm to the simple networks and to network C-Town with high-lying node(s) (respectively fig. 3.2 and fig. 3.3), using each of background leakage models $\{\mathrm{M0},\ldots,\mathrm{M3}\}$ (sections 2.2.1.3 and 2.2.1.5 to 2.2.1.7).

(a) Single pipe

Model	$s_c\!+\!1$	$n_p^{(s_c+1)}$	t_{CPU}
0	5	14	0.16
1	2	4	0.09
2	4	15	0.20
3	2	4	0.13

(b) Simple tree

Model	$s_c\!+\!1$	$n_p^{(s_c+1)}$	t_{CPU}
0	7	192	0.68
1	6	112	0.41
2	6	122	0.50
3	5	61	0.35

(c) Path between two tanks

Model	$s_c\!+\!1$	$n_p^{(s_c+1)}$	t_{CPU}
0	4	36	0.17
1	8	760	5.08
2	4	38	0.25
3	2	11	0.18

(d) Simple loop

Model	$s_c\!+\!1$	$n_p^{(s_c+1)}$	t_{CPU}
0	4	36	0.17
1	8	760	5.02
2	4	38	0.26
3	2	11	0.18

(e) Network C-Town

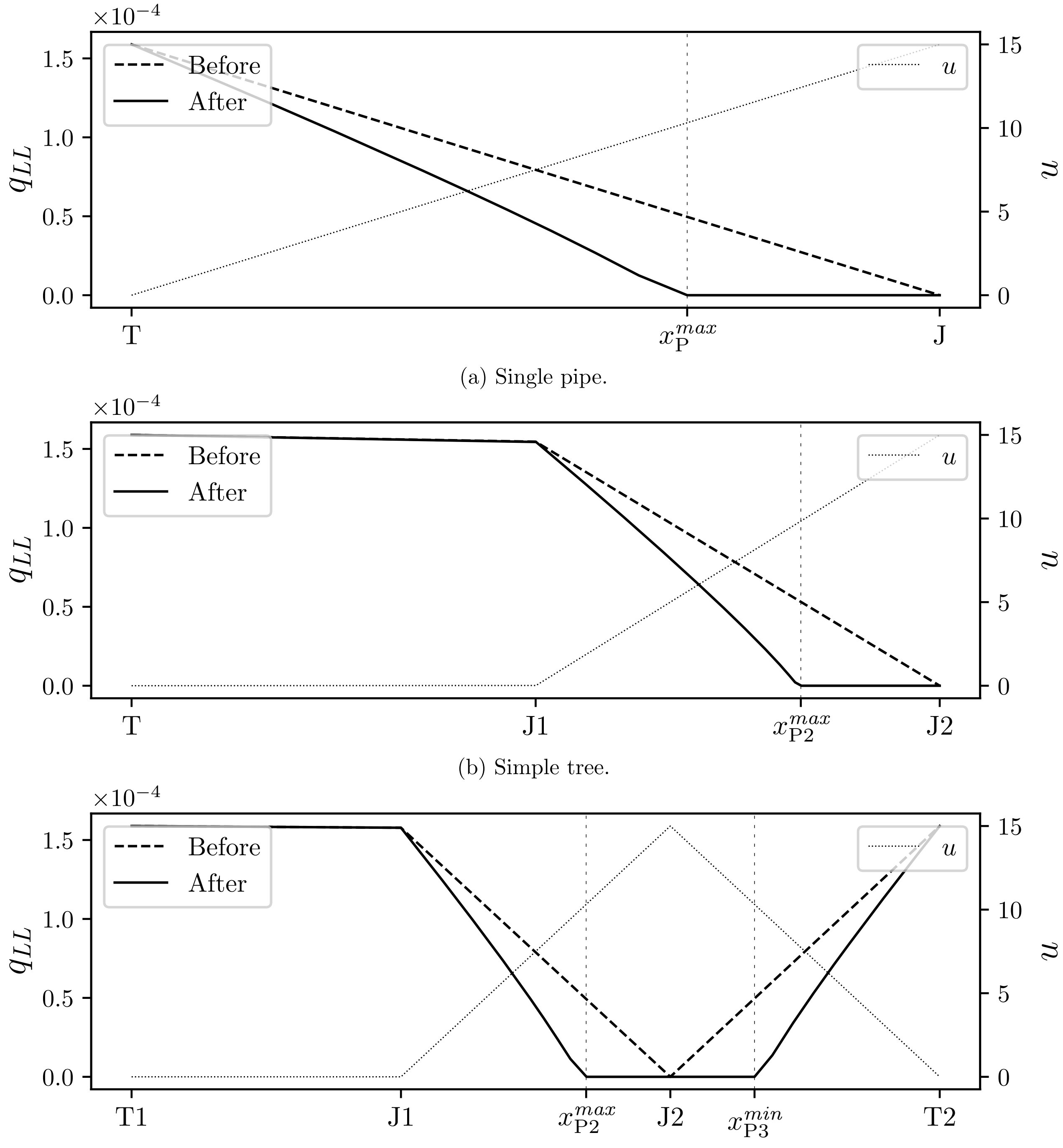
Model	$s_c\!+\!1$	$n_p^{(s_c+1)}$	t_{CPU}
0	5	2070	89.20
1	7	5915	548.17
2	4	2926	120.67
3	4	1893	62.10

The CPU times elapsed are consistent with the numbers of iterations and sub-pipes needed with each leakage model: the higher are the numbers of iterations and sub-pipes, the longer are the simulations (tables 3.2a to 3.2e, column t_{CPU}). For the simple networks, all simulations last less than a second, except with model M1; indeed, with M1 the algorithm needs more than 5 seconds for the path between two tanks and the simple loop. For network C-Town, the simulations last one minute with M3, one minute and a half with M0, two minutes with M2, and more than nine minutes with M1. As expected, the path between two tanks and the simple loop need exactly the same numbers of iterations and sub-pipes for all leakage models, and very close elpased CPU times (tables 3.2c and 3.2d, columns $s_c + 1$, $n_p^{(s_c+1)}$ and t_{CPU}).

We believe that the worse performances of model M1 are due to inconsistency between its computation of lineic leakage outflow rates and flow rates along the pipes. Indeed, M1 considers lineic leakage outflow rates that are invariant along the pipes, but flow rates that vary linearly (see section 2.2.1.5). Thus, in each sub-pipe obtained by discretization, the flow rates and heads are less constrained by positive leakage outflow rates, and their status supplied/unsupplied are more ambiguous to resolve by the algorithm. This limitation makes model M1 less interesting for the simulation of such networks composed of high-lying nodes or subject to strong level of leakages. Conversely, models M2 and M3, which show good performances due to its finer computation of the background leakage outflow rates, could be reused for the simulation of more complex networks.

3.3.3 Gain of accuracy

The profiles of the lineic leakage outflow rates along the simple networks obtained when using the model M2 show strong differences between the results before and after the discretization process (figs. 3.6a to 3.6c). Indeed, the enhanced discretization algorithm permits to find accurately the locations along the pipes where the pressure-heads become 0, and then from where the lineic leakage outflow rates become also 0 and must not contribute to the total leakage outflow rates at the pipe scale.



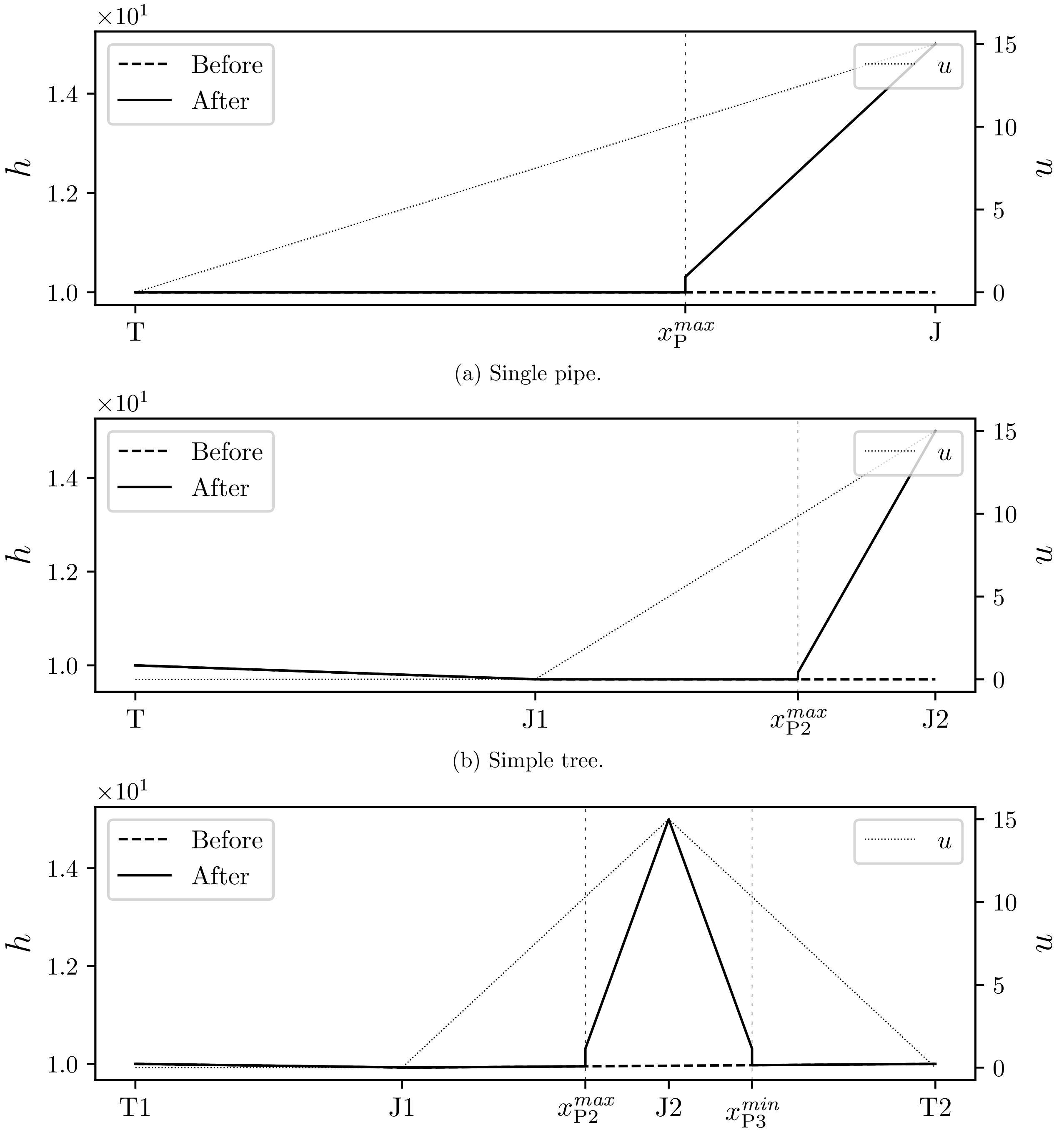
(a) Single pipe.

(b) Simple tree.

(c) Path between two tanks.

Figure 3.6: Profiles of lineic leakage outflow rates (q_{LL}, in $\mathrm{l\,s^{-1}\,m^{-1}}$) along the pipe(s) of the simple networks (fig. 3.2) using the leakage model M2 (section 2.2.1.6), after application of the enhanced discretization algorithm. Lines u show the profiles of elevation (in m). x_{P}^{max}, x_{P2}^{max}, etc., represent the locations where the pressure-heads become 0.

The HGLs in the simple networks before and after the discretization process with model M2 (fig. 3.7) are consistent with the profiles of lineic leakage outflow rates (fig. 3.6), and show the same efficient correction when the pipes become unsupplied. As already observed when comparing the HGLs obtained with the $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ (fig. 3.5), the slight steps at positions x_{P}^{max}, x_{P2}^{max}, etc., are due to the variation of the heads along the sub-pipes that surround these positions at convergence; thinner discretization through the reduction of the convergence criteria tolerance would probably reduce the size of these steps.



(a) Single pipe.

(b) Simple tree.

(c) Path between two tanks.

Figure 3.7: Hydraulic grade lines (in $\mathrm{mH_2O}$) along the pipe(s) of the simple networks (fig. 3.2) using the leakage model M2 (section 2.2.1.6), after application of the enhanced discretization algorithm. Lines u show the profiles of elevation (in m). x_{P}^{max}, x_{P2}^{max}, etc., represent the locations where the pressure-heads become 0.

For each simple network, the global demand satisfactions obtained after discretization using the model M2 is the same or almost the same as the one obtained before discretization (table 3.3, column d_s^{glob}). This is normal because the high-lying node is unsupplied both before and after discretization. Conversely, the total leakage outflow rates (column q_L^{tot}) decrease much: $-30\,\%$ for the single pipe, $-11\,\%$ for the simple tree, and $-15\,\%$ for the path between two tanks and the simple loop. Here also, these results were predictable. Indeed, if we compute, for each simple network, the ratio between the number of pipes connected to an high-lying node and the total number of pipes in the network, we can see that this ratio is 1 for the single pipe, 2/3 for the path between two tanks and the simple loop, and 1/2 for the simple tree. These ratios are also inherently the ones of partly-supplied pipes over total number of pipes. Thus, it is normal that the total leakage outflow rates computed before and after discretization differ more for the single pipe than for the path between two tanks and the simple loop, and more for the path between two tanks and the simple loop than for the simple tree.

Table 3.3: Global demand satisfactions (d_s^{glob}, in %) and total leakage outflow rates (q_L^{tot}, in $\mathrm{l\,s^{-1}}$) before ("*bef*") and after ("*aft*") applying the enhanced discretization algorithm to simple networks (fig. 3.2) and to network C-Town (fig. 3.3), using leakage model M2 (section 2.2.1.6). "Absolute difference" is computed as $\left|d_{s,bef}^{glob} - d_{s,aft}^{glob}\right|$, and "Relative difference" as $(q_{L,aft}^{tot} - q_{L,bef}^{tot})/q_{L,bef}^{tot} \times 100$. "≈" means "equals up to 1‱".

Network	d_s^{glob}			q_L^{tot}		
	Before ($d_{s,bef}^{glob}$)	After ($d_{s,aft}^{glob}$)	Absolute difference	Before ($q_{L,bef}^{tot}$)	After ($q_{L,aft}^{tot}$)	Relative difference
Single pipe	0	0	0	0.12	0.08	-30
Simple tree	≈ 35	≈ 35	$< 10^{-2}$	0.18	0.16	-11
Path between two tanks	≈ 35	≈ 35	$< 10^{-3}$	0.16	0.13	-15
Simple loop	≈ 35	≈ 35	$< 10^{-3}$	0.16	0.13	-15
C-Town	≈ 99	≈ 88	≈ 11	52.03	41.35	-21

For network C-Town, the unsupplied junctions, partly-supplied pipes and unsupplied pipes obtained after discretization (fig. 3.8) are consistent with the junctions initially lifted when preparing the network (fig. 3.3). The absolute difference between the global demand satisfactions (d_s^{glob}) obtained before and after discretization is of 11 %, and the relative difference between the total leakage outflow rate (q_L^{tot}) is $-21\,\%$ (table 3.3, row "C-Town", columns d_s^{glob} and q_L^{tot}). These differences are important. However, they depend much on the locations of the lifted junctions. Indeed, if a junction is located on the only path that permits to connect a large part of a sector to its source(s), then lifting this junction will lead to much bigger differences between the demand satisfactions and the leakage outflow rates before and after discretization than if the cumulated length of the unsupplied pipes and the cumulated users' demands at unsupplied junctions are low. Nevertheless, our algorithm permits to detect the pipes that are critical for a network to not become deficient, to locate the leakiest pipes, and even the leakiest sub-parts of the pipes.

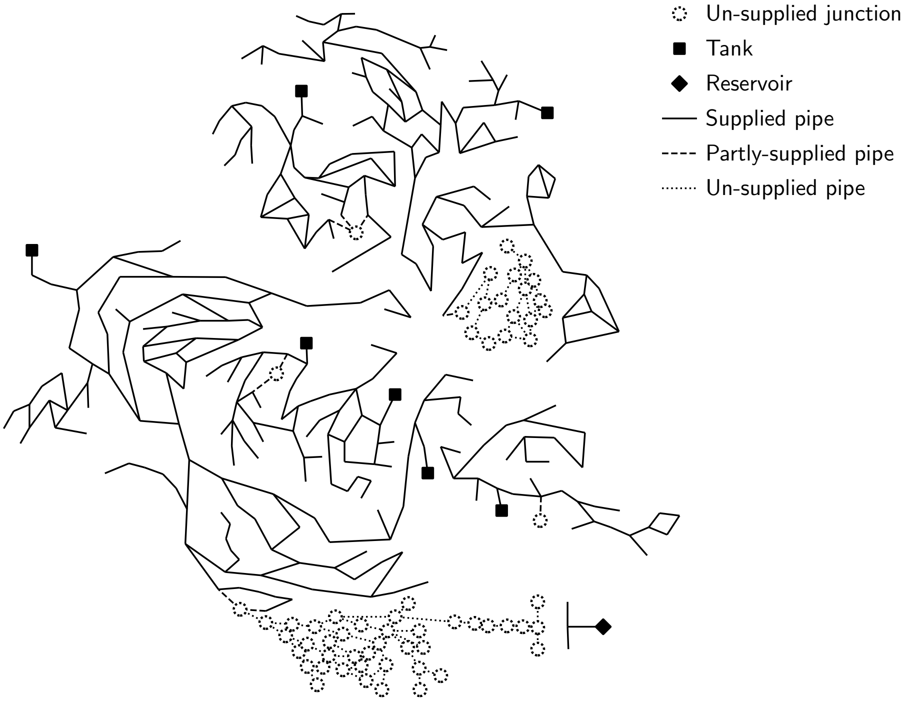


Figure 3.8: Top view (i.e., the elevation is not shown) of the leaky network C-Town with lifted junctions (fig. 3.3), after application of the enhanced discretization algorithm with leakage model M2 (section 2.2.1.6).

3.4 Conclusions

In this third chapter, we proposed a new method to simulate leaky water distribution network (WDNs) that include high-lying nodes. Such nodes can appear from the conception of the networks (e.g., too high elevation at some parts of the network), when a pipe or an equipment (e.g., pump, valves, etc.) is broken or misused, and/or if the level of leakages in pipes becomes progressively or suddenly very strong. The pipes connected to high-lying nodes are partly-supplied; thus, only their supplied part has to contribute to background leakage outflow rates and to the equilibrium of the whole system.

To simulate high-lying nodes and partly-supplied pipes, we first adapted the calculation of the flow rates, friction head-losses and background leakage outflow rates, and of their derivatives with respect to heads at junctions, for each pressure-driven leakage model presented in chapter 2. Then, we reused the discretization algorithm developed in the previous chapter to find the position, along each pipe and if it exists, at which the flow rate becomes 0. As a result, at convergence of the algorithm the partly-supplied pipes of the network are discretized in sub-pipes that are either fully-supplied or fully-unsupplied, which permits to compute more accurate background leakage outflow rates than without discretization. Finally, running a post-convergence simulation with users' demands and lineic leakage outflow rates set to 0 in all the disconnected parts of the network permits to obtain good approximation of the upstream flow rates.

Tests on several small networks and on an adapted version of the larger network C-Town ([57, 120])

permitted to validate all new developments. In particular, we quantified, through different profiles and metrics, the equivalence of the leakage models when they are used for the simulation of networks with high-lying nodes, and the gain of accuracy brought by the discretization algorithm in terms of background leakage outflow rates and hydraulic grade lines (HGLs) along the partly-supplied pipes.

We also added several numerical enhancements to the discretization algorithm of chapter 2. Indeed, in the convergence criterion of the algorithm, the heads at new intermediate junctions are now calculated analytically, while they were computed with a PCHIP interpolator in the previous version of the algorithm. Also, at each iteration of the algorithm, the flow rates and the heads at junctions are initialized using the flow rates and the heads from the previous iteration. These enhancements permit to speed-up the algorithm by up-to 10 %. This gain is particularly valuable for networks with high-lying nodes, since the simulation of these networks requires more iterations.

This new application of the discretization algorithm demonstrates its genericity and re-usability. Our method should permit the location of the leakiest parts of long pipes, or the leakiest pipes of links made from the concatenation of several pipes of a sector; like so, our method provides a continuum between the models of background leakages and the models of local leakage outflow rates like the power equation [49, 91] or the FAVAD (Fixed And Varied Area Discharges) model [102]. A future work could consist in reformulating the equations and adding a numerical constraint to compute straightly the exact flows in pipes upstream of high-lying nodes, and compare the obtained results to the ones we got using recursive discretization and extra post-convergence simulation.

Finally, all our developments are integrated into the Python collaborative framework OOPNET [147, 148]. Thus, the advances that these developments represent for the modeling of WDNs will be more profitable, and easier to reuse and extend by the community. Also, the exhaustive testing and documentation will ask less effort for software developers and modelers to include our models and algorithms as new features in bigger software like EPANET [141] or Porteau [131].

Highlights

- An enhanced model of pressure-dependent outflow rates in deficient networks
- Handles high-lying nodes and partly-supplied pipes
- Simulates both accurate background leakages and user-consumptions
- Based on an efficient algorithm of recursive discretization of pipes
- Deployed through a collaborative and state-of-the-art Python framework

Chapter 4

Inertia phenomena

Abstract

Taking into account inertia phenomena and pressure-dependent outflow rates are primordial to obtain accurate results when simulating water distribution networks (WDNs). Several rigid water column (RWC) simulators already exist, but none of them integrate explicitly both inertia phenomena and pressure-dependent outflows. In this chapter, we propose a new RWC simulator to fill this gap. To do so, we reduce the slow-transient differential algebraic equations (DAEs) of the system to penalized ordinary differential equations (ODEs). A penalty is applied on the inverse function of users' consumptions to maintain the users' consumptions above 0 and below the users' demands at every time step of the simulation. In addition to users' consumptions, our simulator also models background leakages that do not depend on the pressures, and the convective inertia term associated to these leakages. To solve the system of stiff ODEs, we implement a θ-scheme that we compare to a multi-step backward differentiation formulation (BDF) solver from the Python library SciPy. We test our new RWC simulator on two networks, and compare the results with the ones of a dedicated extended-period simulator (EPS) that integrates the same users' consumptions and background leakage models. Our results show significant inertia effect for one of the tested networks. This work is a first step toward the explicit integration of pressure-dependent outflow rates in RWC simulators. Further work is needed to adjust the constraints on consumptions and improve the stability of the θ-scheme. We expect that such comprehensive simulator will permit better analysis of WDNs, and more robust integration of events and processes evolving at the time scale of one minute.

Keywords:
water distribution network (WDN), rigid water column (RWC) simulator, inertia phenomena, pressure-dependent model (PDM) of users' consumptions, background leakage outflows, penalization method

4.1 Introduction

4.1.1 Problem description

According to the first Newton's law of motion [115], any system resists to change when a force applies to it over a given period of time. This resistance is called "inertia".

Inertia exists in water distribution networks (WDNs). For example, when the users' demands vary, valves are opened or closed, or pumps are switched on or off, then the variation of flow velocities is slow down by inertia phenomena [165, p. 578]. These phenomena need to be considered for better analysis of WDNs [83].

Different kinds of 1-D simulators are used to analyze the functioning of WDNs over time. The extended-period (a.k.a., quasi-steady) simulators (EPS) separately solve, at each time step of the simulation, the steady-state equations (i.e., mass balance at junctions and energy conservation in pipes) and the mass balance at tanks [137, 138]. EPS are computationally quick; thus, they permit to run long-term simulations of large WDNs. But EPS simulators neglect inertial effects, skewing the results when the flows change at a time scale of one minute or less.

On the opposite side of EPS simulators, the water hammer (a.k.a., "unsteady-compressible" or "fast-transient") simulators capture not only inertial, but also viscous and compressibility effects. They are well suited for surge analysis to protect the system from excessive transient conditions, which can happen when a power failure occurs, emergency valves are operated, or fire hydrants are used [50, 83, 87]. However, their complexity prevent the simulation of large WDNs over long-term simulations [111].

Between EPS and water hammer simulators, the rigid water column (RWC) (a.k.a., "unsteady-incompressible" or "slow-transient") simulators neglect compressibility and viscous effects, but describe well inertia phenomena related to flow variations at time scale of one minute [112]. These relatively slow variations can be observed when the users' demands fluctuates, or when valves and/or pumps are activated. Among other applications, RWC simulators permit to model the emptying process in a pipeline using pressurized air [25], locate leaks and monitor water quality [34, 157], and to control the pressure [89]. RWC simulators are good compromise between the accuracy of Water Hammer simulators and the computational efficiency of EPSs [111].

Neglecting compressibility and viscous terms in RWC simulators should permit to simplify the transient equations, and to analyze moderately-large WDNs over several days in reasonable computational times. Yet, to reflect accurately inertia phenomena while maintaining the numerical stability of the system, these simplifications should be driven with care, by integrating all time dependent processes at once.

4.1.2 State-of-the-art

Several RWC simulators have been developed in the past decades. [69, 81, 119] solved the slow-transient equations with demand-driven model (DDM) of outflow rates, at a time when pressure-driven model (PDM) did not exist yet. To do so, they implemented a loop-based formulation from the graph theory, reducing the system of differential algebraic equations (DAEs), in which the momentum equation covers all pipes, to a system of first order ordinary differential equations (ODEs), in which the momentum equation covers only the pipes that belong to a minimum spanning tree. Then, [69] linearized the friction head-loss term to obtain a semi-implicit solution, while [81, 119] explicitly integrate the ODEs. [81] also integrated background leakages that do not depend on the pressures, and a term that simulates the convective inertia due to these leakages. However, the system reduction implemented by [69, 81, 119] does not work for pressure-driven outflow rates [126].

Still considering DDM outflow rates, [144] reduced the DAEs to ODEs by variable substitutions, and solved them with explicit scheme. Later, [112] proposed a slow-transient version of the generalized-global gradient algorithm (G-GGA) from [56, 155], adapting the method of [144] and improving its stability with an implicit integration scheme. Finally, [89] integrated the simulator from [112] into the EPANET software [141].

Thus, there still not exists a RWC simulator that integrates slow-transient DAEs with explicit pressure-dependent model (PDM) of outflow rates, such as PDM users' consumptions or PDM leakages; actually, the use of PDM outflow rates makes the DAEs much more complex, which is probably one of the main reasons why such process has not been integrated in the slow transient equations yet. However, as already discussed in previous chapters, modeling pressure-dependent outflow rates is critical for numerous applications (see sections 1.1.1 and 2.1.1).

4.1.3 Hypothesis and objectives

We believe that inertia phenomena could have a significant effect on the state variables in a WDN where the outflow rates depend on the pressures. Thus, our goal is to develop and implement a new

RWC simulator that models explicitly both inertia phenomena and pressure-dependent outflow rates. We expect our RWC simulator to give more accurate results than an EPS simulator that models the same pressure-dependent outflow rates, and to simulate moderately large WDNs in a reasonable time.

To do so, we will make the assumption, as in all RWC simulators, that the compressibility and viscous effects are negligible. Also, to simplify the problem, we will not simulate any hydraulic equipment, and the only outflow rates that will depend on the pressures are users' consumptions. Finally, we will model background leakages as outflow rates that do not depend on the pressures, and take into account the convective inertia that these leakages induce.

First, we will remind the user's consumption function and the slow-transient DAEs of WDNs. Then we will propose a new reduction of the DAEs to penalized ODEs, as well as an implicit scheme to solve these ODEs. Finally, we will test our new RWC simulator on two simple networks, and compare its results with the ones of an EPS simulator that models the same pressure-dependent users' consumptions and same background leakage outflow rates.

4.2 Methods

4.2.1 Consumptions, background leakages and friction head-losses modeling

4.2.1.1 Pressure-dependent users' consumptions

Users' consumptions are modeled by eq. (1.5), as in chapters 1 to 3. At any junction i of a WDN, the Wagner's pressure-outflow rate relation (POR) [162] is used to compute pressure-dependent user's consumption, as:

$$\mathrm{c}_i(p_i) = \begin{cases} 0 & \text{if } p_i \leq p_m \\ d_i\sqrt{\mathrm{z}(p_i)} & \text{if } p_m < p_i < p_s \\ d_i & \text{if } p_i \geq p_s, \end{cases}$$

where d_i is the user's demand (in $\mathrm{l\,s^{-1}}$) at i, p_i is the pressure-head (in $\mathrm{mH_2O}$), p_m and p_s are respectively the minimum and the service pressure-heads (in $\mathrm{mH_2O}$; $p_s > p_m$), and $\mathrm{z}(p_i)$ is the pressure-fraction function defined by eq. (1.4). There exist other PORs to compute PDM user's consumption (e.g., [35, 46, 152]). However, the advantage of the one used here is that its inverse is quadratic, and therefore easy to compute, while being more realistic than the linear POR proposed by [43], or the Heaviside POR implemented by [123]. Indeed, we will see later that we will need to compute the inverse of the POR function to solve the equations of the system.

Denoting u_i the elevation (in m) at the junction i, we also defined in eq. (1.10) the user's consumption as a function of the head h_i (in $\mathrm{mH_2O}$) at i, as:

$$\mathrm{c_i}(h_i) = \begin{cases} 0 & \text{if } h_i \leq u_i + p_m \\ d_i\sqrt{\mathrm{z}(h_i - u_i)} & \text{if } u_i + p_m < h_i < u_i + p_s \\ d_i & \text{if } h_i \geq u_i + p_s. \end{cases}$$

As in chapters 1 to 3, the minimum and service pressure-heads p_m and p_s are chosen respectively equal to 0 and $20\,\mathrm{mH_2O}$.

4.2.1.2 Pressure-independent background leakage outflow rates and flow rates

Conversely to chapters 2 and 3, background leakages are now modeled as pressure-independent outflow rates. This simplification is needed to simulate inertia phenomena without too much computational overhead, as we will see in next sections. Then, the leakage model used in the current chapter is different from the ones defined in chapter 2 (i.e., it is not the same as any of the models $\{\mathrm{M0}, \mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$ from chapter 2).

Denoting $q_{LL,k}$ the pressure-independent lineic outflow rate of background leakage (in $\mathrm{l\,s^{-1}\,m^{-1}}$) in any pipe k of a WDN, the affine function to compute the flow rate (in $\mathrm{l\,s^{-1}}$) at any position x_k (in m) along k is then defined as [81]:

$$\mathrm{q}_k(x_k) = q_{0.5,k} - q_{LL,k}\left(x_k - \frac{\ell_k}{2}\right), \tag{4.1}$$

where $q_{0.5,k} = \mathrm{q}_k(0.5\,\ell_k)$ is the flow rate at the middle of k, and ℓ_k is the length of k (in m).

4.2.1.3 Friction head-losses integrating background leakage outflow rates

As [81], we derive in any pipe k the Hazen-Williams friction head-loss formula [167] to take into account the pressure-independent lineic background leakage outflow rates $q_{LL,k}$. To do so, denoting $\gamma_{\scriptscriptstyle HW} = 1.852$ the Hazen-Williams exponent (unit-less), and reusing the same notation as in section 4.2.1.2, we first define, in each pipe k of a WDN, the unitary friction head-loss (unit-less) at any position x_k along k as:

$$\varphi_k(x_k) = f_k\,\mathrm{q}_k(x_k)\,|\mathrm{q}_k(x_k)|^{\gamma_{\scriptscriptstyle HW}-1}, \tag{4.2}$$

where $\mathrm{q}_k(x_k)$ is defined by eq. (4.1). Thus, $\forall q_{LL,k} \in \mathbb{R}^+$, we have:

$$\varphi_k(x_k) = \begin{cases} f_k\,\left(q_{0.5,k} - q_{LL,k}\,(x_k - {}^{\ell_k}\!/_2)\right) \left|q_{0.5,k} - q_{LL,k}\,(x_k - {}^{\ell_k}\!/_2)\right|^{\gamma_{\scriptscriptstyle HW}-1} & \text{if } q_{LL,k} \neq 0 \\ f_k\,q_{0.5,k}\,|q_{0.5,k}|^{\gamma_{\scriptscriptstyle HW}-1} & \text{otherwise.} \end{cases} \tag{4.3}$$

In eq. (4.3), the expression of $\varphi_k(x_k)$ when $q_{LL,k} = 0$ corresponds to the original Hazen-Williams unitary friction head-loss formula from [167], which does not take into account the background leakage outflows. Finally, for each case, integrating eq. (4.3) over the whole pipe k permits to compute the friction head-loss (in $\mathrm{mH_2O}$), as:

$$\xi_{\mathrm{f},k}(q_{0.5,k}) = \begin{cases} \dfrac{f_k}{(\gamma_{\scriptscriptstyle HW}+1)\,q_{LL,k}} \left(\left|q_{0.5,k} + q_{LL,k}\dfrac{\ell_k}{2}\right|^{\gamma_{\scriptscriptstyle HW}+1} - \left|q_{0.5,k} - q_{LL,k}\dfrac{\ell_k}{2}\right|^{\gamma_{\scriptscriptstyle HW}+1} \right) & \text{if } q_{LL,k} \neq 0 \\ f_k\,\ell_k\,q_{0.5,k}\,|q_{0.5,k}|^{\gamma_{\scriptscriptstyle HW}-1} & \text{otherwise.} \end{cases} \tag{4.4}$$

4.2.2 Slow-transient differential-algebraic equations

4.2.2.1 At the pipe scale

Assuming that the fluid is incompressible, the axisymmetric Navier-Stokes momentum equation can be integrated along the axial direction of a cylindrical pipe as [50, 81]:

$$\frac{1}{10^3}\frac{\partial \mathrm{q}}{\partial t}(t,x) + \frac{\pi \varnothing_p}{\rho}\tau_w + g\,a_p\frac{\partial \mathrm{h}}{\partial x}(t,x) + \frac{1}{(10^3)^2}\frac{\beta_b}{a_p}\frac{\partial\left(\mathrm{q}^2\right)}{\partial x}(t,x) - \frac{1}{10^3}\frac{\mu}{\rho}\frac{\partial^2 \mathrm{q}}{\partial x^2}(t,x) = 0 \tag{4.5}$$

where $\mathrm{q}(t,x)$ is the function to compute the flow rate (in $\mathrm{l\,s^{-1}}$), $\varnothing_p$ is the inside diameter of the pipe (in m), ρ is the fluid density (in $\mathrm{kg\,m^{-3}}$), τ_w is the shear stress at the pipe wall (in $\mathrm{kg\,s^{-2}\,m^{-1}}$), g the standard acceleration due to gravity (in $\mathrm{m\,s^{-2}}$), a_p is the inside cross-sectional area of the pipe (in $\mathrm{m^2}$) defined as:

$$a_p = \pi \frac{{\varnothing_p}^2}{4}, \tag{4.6}$$

$\mathrm{h}(t,x)$ is the function to compute the piezometric head along the pipe (in $\mathrm{mH_2O}$), β_b is the Boussinesq coefficient (unit-less) that reflects the real distribution of speed along the radial direction of the pipe [81], and μ is the dynamic viscosity of the fluid (in $\mathrm{kg\,m^{-1}\,s^{-1}}$). The divisions by 10^3 permit to convert the flow rate computed by $\mathrm{q}(t,x)$ from $\mathrm{l\,s^{-1}}$ to $\mathrm{m^3\,s^{-1}}$.

In our study, the flow can be considered as uniform, and $\beta_b = 1$. Also, $\mathrm{q}(t,x)$ is defined by the affine function (4.1); thus, the last term of eq. (4.5), which corresponds to the diffusion, is 0, and considering affine flow rate along the pipe amounts to neglecting the diffusion phenomena in comparison to inertia (i.e. 1st and 4th terms in eq. (4.5)), the friction (i.e., 2nd term in eq. (4.5)) and the head variation (i.e. 3rd term in eq. (4.5)). Then, after removing the diffusion term, dividing by "$g\,a_p$", replacing β_b by 1, and integrating along the full length ℓ of the pipe, eq. (4.5) becomes:

$$\frac{\iota_p}{10^3}\frac{\mathrm{d}\mathrm{q}_{0.5}}{\mathrm{d}t}(t) + \xi_\mathrm{f}(\mathrm{q}_{0.5}(t)) - \Delta\mathrm{h}(t) - \frac{\mathfrak{l}_c(t)}{10^3}\,\mathrm{q}_{0.5}(t) = 0, \tag{4.7}$$

where ι_p is the constant inertia of the pipe (in $\mathrm{s^2\,m^{-2}}$), defined as:

$$\iota_p = \frac{\ell}{g\,a_p} \neq 0, \tag{4.8}$$

$\mathrm{q}_{0.5}(t) = \mathrm{q}(t, 0.5\,\ell)$ is the function that computes the flow rate (in $\mathrm{l\,s^{-1}}$) at the middle of the pipe, $\xi_\mathrm{f}(\mathrm{q}_{0.5}(t))$ is the function to compute the friction head-loss as in eq. (4.4), $\Delta\mathrm{h}(t) = \mathrm{h}(t,0) - \mathrm{h}(t,\ell)$ is the function to compute the total drop of piezometric head (in $\mathrm{mH_2O}$) along the pipe, and $\mathfrak{l}_c(t)$ is the function to compute the convective inertia coefficient due to background leakage outflow rates as:

$$\mathfrak{l}_c(t) = \frac{2\,\ell}{g\,{a_p}^2}\frac{\mathrm{q}_\mathrm{LL}(t)}{10^3}, \tag{4.9}$$

where $\mathrm{q}_\mathrm{LL}(t)$ is the pressure-independent lineic outflow rate of background leakage (in $\mathrm{l\,s^{-1}\,m^{-1}}$) at t, defined in section 4.2.1.2.

In eq. (4.7), the divisions by 10^3 permit to convert the middle flow rate computed by $\mathrm{q}_{0.5}(t)$ from $\mathrm{l\,s^{-1}}$ to $\mathrm{m^3\,s^{-1}}$. Finally, dividing eq. (4.7) by ι_p and keeping only time derivative on the left side, we obtain:

$$\frac{\mathrm{d}\mathrm{q}_{0.5}}{\mathrm{d}t}(t) = -\frac{10^3}{\iota_p}\Big(\xi_\mathrm{f}(\mathrm{q}_{0.5}(t)) - \Delta\mathrm{h}(t) - \frac{\mathfrak{l}_c(t)}{10^3}\,\mathrm{q}_{0.5}(t)\Big). \tag{4.10}$$

This notation, where the inertial term $-10^3/\iota_p$ is at the right-hand side of the equation, is the one used by the Python solvers[1].

[1] e.g., `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html`

4.2.2.2 At the WDN scale

Considering now all pipes of a WDN, eq. (4.10) becomes:

$$\frac{\mathbf{dq_{0.5}}}{\mathrm{d}t}(t) = -10^3\, \boldsymbol{\mathcal{I}_p}^{-1} \left(\boldsymbol{\xi}_{\mathbf{f}}(\mathbf{q_{0.5}}(t)) - \boldsymbol{A}^T \mathbf{h}(t) - \boldsymbol{A}_{\boldsymbol{t}}^T \mathbf{h_t}(t) - \boldsymbol{A}_{\boldsymbol{r}}^T \boldsymbol{h_r} - \frac{\boldsymbol{\mathcal{I}_c}(t)}{10^3}\, \mathbf{q_{0.5}}(t) \right), \tag{4.11}$$

where $\mathbf{q_{0.5}}$ is a vector function that computes the flow rates (in $\mathrm{l\,s^{-1}}$) at the middle of the pipes at time t, $\boldsymbol{\mathcal{I}_p}$ and $\boldsymbol{\mathcal{I}_c}(t)$ are the diagonal matrices of pipes' and convective inertia, defined, for all pipe $k \in \{1, \dots, n_p\}$, from respectively eq. (4.8) and eq. (4.9), as:

$$[\boldsymbol{\mathcal{I}_p}]_{kk} = \frac{\ell_k}{g\, a_{p,k}} \tag{4.12}$$

and

$$[\boldsymbol{\mathcal{I}_c}(t)]_{kk} = \frac{2\, \ell_k}{g\, {a_{p,k}}^2} \frac{\mathrm{q}_{\mathrm{LL},k}(t)}{10^3}, \tag{4.13}$$

$\boldsymbol{\xi}_{\mathbf{f}}(\mathbf{q_{0.5}}(t))$, is the vector function that computes the leakage dependent Hazen-Williams friction head-losses in pipes (see eq. (4.4)), and $\mathbf{h}(t)$ and $\mathbf{h_t}(t)$ are the vector functions that compute the heads (in $\mathrm{mH_2O}$) at respectively the junctions and the tanks of the WDN.

As in chapter 1, deriving the mass conservation at junctions from the first Kirchhoff's law we obtain:

$$-\boldsymbol{A}\, \mathbf{q_{0.5}}(t) - \mathbf{c}(\mathbf{h}(t)) - \mathbf{q_L}(t) = \boldsymbol{0}, \tag{4.14}$$

where the vector function $\mathbf{c}(\mathbf{h}(t))$ computes the user's consumption at each junction node using the eq. (1.10), and $\mathbf{q_L}(t)$ computes the pressure-independent background leakage outflow rates (in $\mathrm{l\,s^{-1}}$) reported to junctions and is defined for any junction i as:

$$\mathrm{q}_{\mathrm{L},i}(t) = 0.5 \sum_{k \in \mathbb{P}_i^- \cup \mathbb{P}_i^+} \mathrm{q}_{\mathrm{LL},k}(t)\, \ell_k, \tag{4.15}$$

with $\mathbb{P}_i^-$ and $\mathbb{P}_i^+$ the sets of pipes respectively entering and leaving i [81]. Actually, eq. (4.14) corresponds to a simplified version of the mass conservatiin eq. (2.37) used in chapter 2; this simplification is possible because the leakages are not pressure-dependent.

Applying the same first Kirchhoff's law at the tanks, we have:

$$\frac{\mathbf{dh_t}}{\mathrm{d}t}(t) = -\frac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3} \left(\boldsymbol{A_t}\, \mathbf{q_{0.5}}(t) + \mathbf{q_{Lt}}(t) \right), \tag{4.16}$$

where $\boldsymbol{\mathcal{I}_t}$ is the diagonal matrix of constant tanks' inertia, defined for any cylindrical tank i as:

$$[\boldsymbol{\mathcal{I}_t}]_{ii} = a_{t,i} = \pi \frac{{\varnothing_{t,i}}^2}{4} \tag{4.17}$$

with $\varnothing_{t,i}$ the inside diameters of the tank i (in m) and $a_{t,i}$ its cross-sectional area (in $\mathrm{m^2}$). We suppose here that all tanks are cylindrical; otherwise, eq. (4.17) would also depend on the head $h_{t,i}$ at i. In eq. (4.16), $\mathbf{q_{Lt}}(t)$ computes the pressure-independent background leakage outflow rates (in $\mathrm{l\,s^{-1}}$) reported to tanks and is defined, for any tank i, as:

$$\mathrm{q}_{\mathrm{Lt},i}(t) = 0.5 \sum_{k \in \mathbb{P}_{\mathrm{t},i}^- \cup \mathbb{P}_{\mathrm{t},i}^+} \mathrm{q}_{\mathrm{LL},k}(t)\, \ell_k, \tag{4.18}$$

where $\mathbb{P}_{\mathrm{t},i}^{-}$ and $\mathbb{P}_{\mathrm{t},i}^{+}$ are the sets of pipes respectively entering and leaving the tank i, $\mathrm{q}_{\mathrm{LL},k}(t)$ is the pressure-independent lineic outflow rate (in $\mathrm{l\,s^{-1}\,m^{-1}}$) of background leakage from the pipe k and defined in section 4.2.1.2, and ℓ_k is the length (in m) of k.

Finally, from eqs. (4.11), (4.14) and (4.16), we obtain the slow-transient system of differential algebraic equations (DAEs):

$$\begin{cases} \dot{\mathbf{q}}_{\mathbf{0.5}}(t) = -\,10^3\,\boldsymbol{\mathcal{I}_p}^{-1}\left(\boldsymbol{\xi}_{\mathbf{f}}(\mathbf{q}_{\mathbf{0.5}}(t)) - \boldsymbol{A}^T\mathbf{h}(t) - \boldsymbol{A}_{\boldsymbol{t}}^T\mathbf{h}_{\mathbf{t}}(t) - \boldsymbol{A}_{\boldsymbol{r}}^T\boldsymbol{h}_{\boldsymbol{r}} - \dfrac{\boldsymbol{\mathcal{I}_c}(t)}{10^3}\,\mathbf{q}_{\mathbf{0.5}}(t)\right) \\ \qquad\quad \boldsymbol{0} = -\,\boldsymbol{A}\,\mathbf{q}_{\mathbf{0.5}}(t) - \mathbf{c}(\mathbf{h}(t)) - \mathbf{q}_{\mathbf{L}}(t) \\ \quad \dot{\mathbf{h}}_{\mathbf{t}}(t) = -\,\dfrac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3}\left(\boldsymbol{A}_{\boldsymbol{t}}\,\mathbf{q}_{\mathbf{0.5}}(t) + \mathbf{q}_{\mathbf{Lt}}(t)\right), \end{cases} \tag{4.19}$$

where the unknown functions are $\mathbf{q}_{\mathbf{0.5}}(t)$, $\mathbf{h}(t)$ and $\mathbf{h}_{\mathbf{t}}(t)$. As in eq. (4.10), we use in eq. (4.19) the notation of the Python solvers, in which the inertial terms are put at the right-hand side of the equations.

4.2.3 Reduction of DAEs to penalized ordinary differential equations

4.2.3.1 Reduction to ODEs

DAEs, such as eq. (4.19), are difficult to solve numerically, especially when the algebraic equation is non-linear [76, p. 104]. In eq. (4.14), since the function $\mathbf{c}(\mathbf{h}(t))$ is pressure-dependent, we cannot define the loop flow rates and reduce eq. (4.19) to a determined and simpler system of ordinary differential equations (ODEs) as in [81, 126]. Also, differentiating eq. (4.14) to transform the DAEs (4.19) to ODEs that describe the full (i.e., not reduced) system, as in [156], would not make appear the derivative of head with respect to time at zero demand nodes. Thus, to ease the solving of eq. (4.19) while keeping them representative of the initial system, we choose to rather reduce the DAEs (4.19) to a system of penalized ODEs, as:

$$\begin{cases} \dot{\mathbf{q}}_{\mathbf{0.5}}(t) = -\,10^3\,\boldsymbol{\mathcal{I}_p}^{-1}\left(\boldsymbol{\xi}_{\mathbf{f}}(\mathbf{q}_{\mathbf{0.5}}(t)) - \boldsymbol{A}^T\,\widehat{\hat{\mathbf{h}}}(\mathbf{q}_{\mathbf{0.5}}(t)) - \boldsymbol{A}_{\boldsymbol{t}}^T\mathbf{h}_{\mathbf{t}}(t) - \boldsymbol{A}_{\boldsymbol{r}}^T\boldsymbol{h}_{\boldsymbol{r}} - \dfrac{\boldsymbol{\mathcal{I}_c}(t)}{10^3}\,\mathbf{q}_{\mathbf{0.5}}(t)\right) \\ \quad \dot{\mathbf{h}}_{\mathbf{t}}(t) = -\,\dfrac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3}\left(\boldsymbol{A}_{\boldsymbol{t}}\,\mathbf{q}_{\mathbf{0.5}}(t) + \mathbf{q}_{\mathbf{Lt}}(t)\right), \end{cases} \tag{4.20}$$

and to integrate (4.20) rather than (4.19) in our RWC simulator. We can see in eq. (4.20) that the dual vector function $\mathbf{h}(t)$, associated to the mass conservation equatiin eq. (4.14) in eq. (4.19), disappeared. Thus, the unknowns are now $\mathbf{q}_{\mathbf{0.5}}(t)$ and $\mathbf{h}_{\mathbf{t}}(t)$. Once these unknowns are found by solving (4.20), we can deduce $\mathbf{h}(t)$ from the inverse vector function of $\mathbf{c}(\mathbf{h}(t))$, defined, $\forall i \in \{1,\dots,n_j\}$ such that $0 \le \mathrm{c}_i(\mathrm{h}_i(t)) \le \mathrm{d}_i(t)$ and $\mathrm{d}_i(t) \neq 0$, as:

$$\mathrm{h}_i(\mathrm{c}_i(t)) = p_m + u_i + (p_s - p_m)\left(\frac{\mathrm{c}_i(t)}{\mathrm{d}_i(t)}\right)^2, \tag{4.21}$$

where p_m and p_s are respectively the constant minimum pressure-head and service pressure-head at any junction i, u_i is the constant elevation at i (see section 4.2.1.1), and

$$\mathrm{c}_i(t) = \left[-\boldsymbol{A}\,\mathbf{q}_{\mathbf{0.5}}(t) - \mathbf{q}_{\mathbf{L}}(t)\right]_i. \tag{4.22}$$

In eq. (4.20), $\hat{\hat{\mathbf{h}}}(\mathbf{q_{0.5}}(t))$ is the penalized inverse vector function of $\mathbf{c}(\mathbf{h}(t))$. It permits to obtain solutions $\{\mathbf{q_{0.5}}(t), \mathbf{h_t}(t)\}$ of (4.20) that are such that, $\forall i \in \{1, \dots, n_j\}$, eq. (4.22) computes a consumption that belongs as much as possible to the interval $[0, \mathrm{d}_i(t)]$.

The initial conditions of $\mathbf{q_{0.5}}(t)$ and $\mathbf{h_t}(t)$ will be defined a little later, with the description of the numerical scheme used to solve eq. (4.20) (see section 4.2.4.1).

4.2.3.2 Definition of the penalized function

In this sub-section, we will define the penalized vector function $\hat{\hat{\mathbf{h}}}(\mathbf{q_{0.5}}(t))$ for any time t. Thus, for better readability, we will simply denote $c_i = \mathrm{c}_i(t)$, $d_i = \mathrm{d}_i(t)$, and $\mathrm{h}_i(c_i) = \mathrm{h}_i(\mathrm{c}_i(t))$, and we will omit t in the definition of all new variables and functions, without any loss of generality. Then, without any other indication, all variables used and/or defined in this sub-section depend on t.

The not-penalized multivalued inverse function of the users' consumptions at junctions, $\mathbf{h}(\boldsymbol{c})$, is defined, for any junction $i \in \{1, \dots, n_j\}$, by a sub-differential mapping, as [35]:

$$\mathrm{h}_i(c_i) = \begin{cases} \emptyset & \text{if } c_i < 0, \\ (-\infty, p_m + u_i] & \text{if } c_i = 0, \\ p_m + u_i + (p_s - p_m)\left(\dfrac{c_i}{d_i}\right)^2 & \text{if } 0 < c_i < d_i, \\ [p_s + u_i, +\infty) & \text{if } c_i = d_i, \\ \emptyset & \text{if } c_i > d_i. \end{cases} \tag{4.23}$$

Equation (4.23) is undefined when $c_i \in [-\infty, 0[\cup]d_i, +\infty]$. The expression of eq. (4.23) for $0 < c_i < d_i$ corresponds to eq. (4.21).

To define the penalized function $\hat{\hat{\mathrm{h}}}_i(c_i)$, we first write, from the expression of eq. (4.23) when $0 < c_i < d_i$, the monovalued function:

$$\widehat{\mathrm{h}}_i(c_i) = p_m + u_i + (p_s - p_m)\,\frac{c_i|c_i|}{{d_i}^2}. \tag{4.24}$$

(4.24) is definite, continuous, and derivable on $\mathbb{R}$, for any junction i where $d_i \neq 0$. To guarantee that $d_i \neq 0\ \forall i \in \{1, \dots, n_j\}$, we apply a threshold on each d_i such that $d_i \geq 10^{-3}\,\mathrm{l\,s^{-1}}\ \forall i \in \{1, \dots, n_j\}$.

Next, we define a Huber loss [73] penalization function $\xi_{\mathrm{H},i}(c_i)$ to apply in eq. (4.24) when $c_i \in [-\infty, 0[\cup]d_i, +\infty]$, as:

$$\xi_{\mathrm{H},i}(c_i) = \begin{cases} \xi^-_{\mathrm{Q},i}(c^-_i) + \xi^-_{\mathrm{L},i}(c_i) & \text{if } c_i < c^-_i, \\ \xi^-_{\mathrm{Q},i}(c_i) & \text{if } c^-_i \leq c_i < 0, \\ 0 & \text{if } 0 \leq c_i \leq d_i, \\ \xi^+_{\mathrm{Q},i}(c_i) & \text{if } d_i < c_i \leq c^+_i, \\ \xi^+_{\mathrm{Q},i}(c^+_i) + \xi^+_{\mathrm{L},i}(c_i) & \text{if } c_i > c^+_i. \end{cases} \tag{4.25}$$

Huber loss functions are commonly used in robust regression, because they are less sensitive to large outliers in data than the Mean Squared Error (MSE) loss, while being more efficient than the Mean Absolute Error (MAE) loss function for smaller outliers [4].

In eq. (4.25), $\xi^{-}_{\mathrm{Q},i}(c_i)$ is a quadratic penalty defined as :

$$\xi^{-}_{\mathrm{Q},i}(c_i) = -\mu_Q\,\mathrm{c}_i{}^2, \tag{4.26}$$

where the constant factor μ_Q is computed as:

$$\mu_Q = \frac{\delta h}{(\delta c)^2}; \tag{4.27}$$

in eq. (4.27), δc represents a very small and constant variation of a user's consumption (in $\mathrm{l\,s^{-1}}$), and Δh a significant and constant head variation (in $\mathrm{mH_2O}$). $\xi^{+}_{\mathrm{Q},i}(c_i)$ is the symmetric of $\xi^{-}_{\mathrm{Q},i}(c_i)$, and is defined as:

$$\xi^{+}_{\mathrm{Q},i}(c_i) = -\xi^{-}_{\mathrm{Q},i}(c_i - d_i) = \mu_Q\,(c_i - d_i)^2\,. \tag{4.28}$$

Quadratic penalties $\xi^{-}_{\mathrm{Q},i}(c_i)$ and $\xi^{+}_{\mathrm{Q},i}(c_i)$ permit to penalize strongly eq. (4.24) when $c_i^- \leq c_i < 0$ and $d_i < c_i \leq c_i^+$.

$\xi^{+}_{\mathrm{L},i}(c_i)$ and $\xi^{-}_{\mathrm{L},i}(c_i)$ are linear penalties, and c_i^+ and c_i^- are the values of users' consumptions from which to switch from quadratic to linear penalties in the Huber loss penalization function (4.25). The linear penalties are defined as:

$$\xi^{+}_{\mathrm{L},i}(c_i) = \mu_{L,i}\left(c_i - c_i^+\right) \tag{4.29}$$

and

$$\xi^{-}_{\mathrm{L},i}(c_i) = \mu_{L,i}\left(c_i - c_i^-\right). \tag{4.30}$$

$\mu_{L,i}$ is the slope of the linear penalties, defined as:

$$\mu_{L,i} = \mu_H \frac{2\,(p_s - p_m)}{d_i}. \tag{4.31}$$

μ_H is a constant multiplier (unit-less) that permits to control the intervals where the quadratic and the linear penalties have to be respectively applied. Linear penalties $\xi^{+}_{\mathrm{L},i}(c_i)$ and $\xi^{-}_{\mathrm{L},i}(c_i)$ are softer than the quadratic penalties $\xi^{-}_{\mathrm{Q},i}(c_i)$ and $\xi^{+}_{\mathrm{Q},i}(c_i)$. Switching from quadratic to linear penalties prevents the quadratic penalties to become too big when $c_i < c_i^-$ or $c_i > c_i^+$; thus, this switching will improve the convergence of the algorithms that we will use later in this chapter to solve the ODEs (4.20).

In eq. (4.29), c_i^+ is defined as:

$$c_i^+ = \frac{\mu_{L,i} + 2\,\mu_Q\,d_i}{\dfrac{2\,(p_s - p_m)}{d_i{}^2} + 2\,\mu_Q}. \tag{4.32}$$

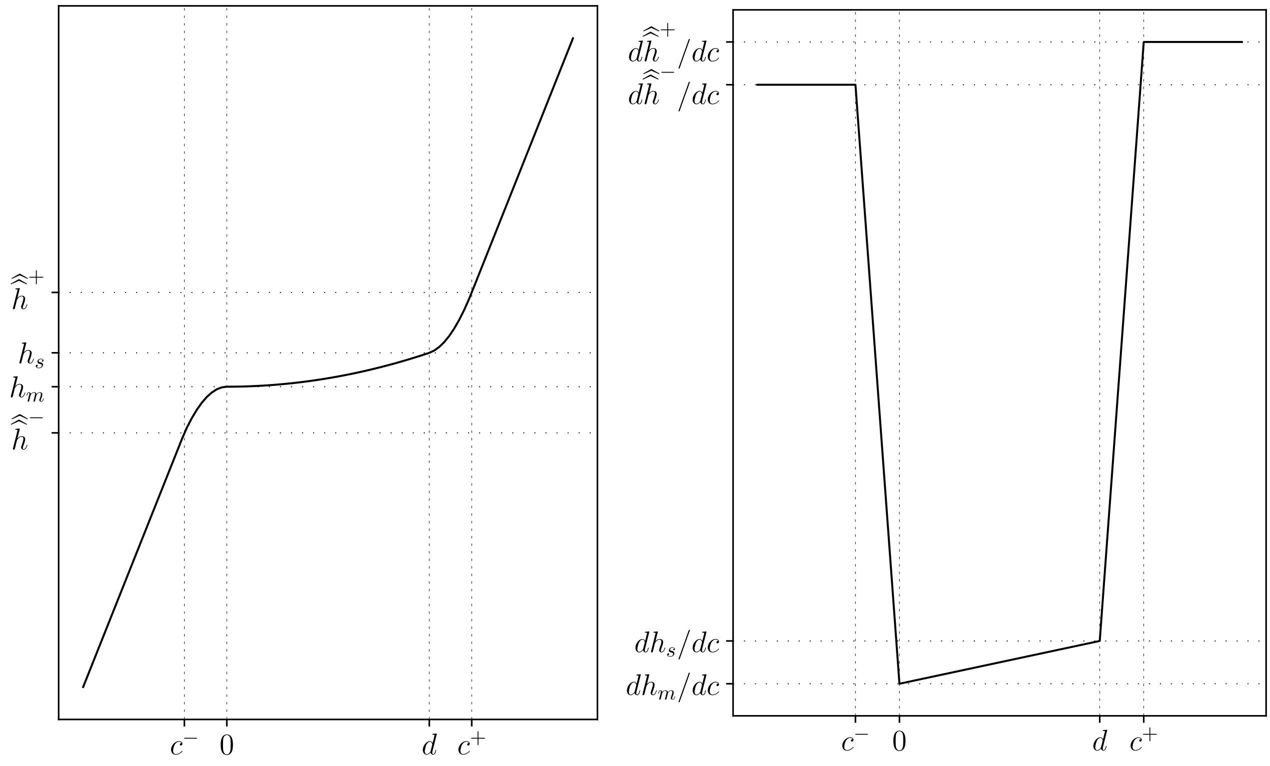
In eq. (4.30), c_i^- is defined as:

$$c_i^- = d_i - c_i^+. \tag{4.33}$$

Finally, from eqs. (4.24) and (4.25), we define the continuous penalized function $\widehat{\widehat{\mathrm{h}}}_i(c_i)$ for any junction i as:

$$\widehat{\widehat{\mathrm{h}}}_i(c_i) = \widehat{\mathrm{h}}_i(c_i) + \xi_{\mathrm{H},i}(c_i). \tag{4.34}$$

Figure 4.1 shows the shapes of eq. (4.34) and its derivative for $\mathrm{c}_i \in \mathbb{R}$ (we remove the subscript i on the figure for better readability, without any loss of generality).



(a) $\widehat{\widehat{\mathrm{h}}}(c)$

(b) $\mathrm{d}\widehat{\widehat{\mathrm{h}}}/\mathrm{dc}\,(c)$

Figure 4.1: Graphical representation of the penalization of users' consumptions. A Huber loss penalization function ([73]) is applied on the inverse function of the consumption, $\mathrm{h}(c)$ (fig. 4.1a), and on its derivative, $\mathrm{d}\widehat{\widehat{\mathrm{h}}}/\mathrm{dc}\,(c)$ (fig. 4.1b), to maintain as much as possible the user's consumption c between 0 and the user's demand d, $\forall c \in \mathbb{R}$ and $d \neq 0$. h_m and h_s: minimum and service-heads, defined as $h_m = u + p_m$ and $h_s = u + p_s$, with u the elevation, and p_m and p_s the minimum and service pressure-heads; c^- and c^+: users' consumptions at which to switch from quadratic to linear penalty; $\widehat{\widehat{h}}^-$ and $\widehat{\widehat{h}}^+$: values of the heads at c^- and c^+.

4.2.3.3 Limit of the reduction method

The reduction of the DAEs (4.19) to the penalized ODEs (4.20) is possible because the inverse of the Wagner's POR [162], $\mathrm{h}_i(t, c_i)$, can be easily extended to the function $\widehat{\mathrm{h}}_i(c_i)$ that is defined, continuous, and derivable on $\mathbb{R}$ for any junction i (see *eq.* (4.24)).

In chapter 2, we described several models of pressure-dependent background leakages. All these models compute, for each leaky pipe, a background leakage outflow rate using the pressure-heads at the nodes located at both extremities of the pipe (sections 2.2.1.3 to 2.2.1.7). Thus, the function that computes the pressure-dependent background leakage outflow rate is not invertible, because there is no relation that links one pressure-head to one background leakage outflow rate. As a result, the DAEs (4.19) cannot be reduced to the ODEs (4.20) if one of the pressure-dependent background leakage models described in chapter 2 is used to simulate background leakage outflow rates. That is the reason why, in section 4.2.1.2, we chose to model pressure-independent background leakage outflow rates.

4.2.4 θ-scheme solver

4.2.4.1 Description of the θ-scheme

To solve the penalized ODEs (4.20) in our RWC simulator, we choose to implement a θ-scheme. We set $\theta = 0.5$, which corresponds to the trapezoidal rule. This method is implicit and of second-order; thus, its global error is $\mathcal{O}(\delta t^2)$ as the hydraulic time step δt (in s) tends to 0 [7]. The trapezoidal rule is the most accurate of the A-stable linear multi-step methods [150]; as a reminder, a linear multi-step method is A-stable when its order is at most 2. Finally, our θ-scheme will be easy to integrate later into the software Porteau [131], in which it has already proved its efficiency for demand-driven modeling (DDM) of users' consumptions.

We denote $\mathbf{y}(t) = (\mathbf{q_{0.5}}, \mathbf{h_t})^T(t)$ the unknown vector function that satisfies the ODEs (4.20). At t_0, we only know the heads at tanks $\boldsymbol{h_t}^{(0)} = \mathbf{h_t}(t_0)$. Thus, to start from good initial values of the middle flow rates in pipes, we first need to solve the balance equation in steady-state from $\boldsymbol{h_t}^{(0)}$ and the demands $\mathbf{d}(t_0)$. To do so, we run a snapshot simulation with the extended-period simulator (EPS) described in appendix D.1; this EPS simulator simulates the same processes (i.e., same consumption and friction head-loss functions and same pressure-independent background leakage outflow rates) as the RWC simulator, but neglecting all inertia phenomena. We then obtain the initial guesses $\boldsymbol{y}^{(0)} = \mathbf{y}(t_0) = \left(\boldsymbol{q_{0.5}}^{(0)}, \boldsymbol{h_t}^{(0)}\right)^T$.

Next, we denote $\mathbf{f}(t, \boldsymbol{y}) = \dot{\mathbf{y}}(t, \boldsymbol{y}) = \left(\dot{\mathbf{q}}_{\mathbf{0.5}}, \dot{\mathbf{h}}_{\mathbf{t}}\right)^T(t)$. At each iteration $n > 0$ of the θ-scheme, we need to solve the non-linear system:

$$\boldsymbol{y}^{(n+1)} = \boldsymbol{y}^{(n)} + \delta t \left((1-\theta)\, \mathbf{f}(t_n, \boldsymbol{y}^{(n)}) + \theta\, \mathbf{f}(t_{n+1}, \boldsymbol{y}^{(n+1)}) \right) \tag{4.35}$$

where $t_n = n\, \delta t$, and $\boldsymbol{y}^{(n)} \approx \mathbf{y}(t_n)$. Equation (4.35) can be rewritten in a linear form, as:

$$\mathbf{g}(\boldsymbol{y}) = \boldsymbol{y} - \delta t \left((1-\theta)\, \mathbf{f}(n\, \delta t, \boldsymbol{y}^{(n)}) + \theta\, \mathbf{f}((n+1)\, \delta t, \boldsymbol{y}) \right) - \boldsymbol{y}^{(n)}. \tag{4.36}$$

Then, we solve eq. (4.36) using the same Newton's method as in section 1.2.4 of chapter 1:

$$\boldsymbol{y}^{(n+1,m+1)} = \boldsymbol{y}^{(n+1,m)} - \left(\mathbf{G}(\boldsymbol{y}^{(n+1,m)}) \right)^{-1} \mathbf{g}(\boldsymbol{y}^{(n+1,m)}), \tag{4.37}$$

where $m = 0, 1, 2, \dots$ is the iteration of the Newton's method, and $\mathbf{G} = \boldsymbol{\partial}\mathbf{g} / \boldsymbol{\partial y}$ is the Jacobian of $\mathbf{g}$, defined as:

$$\mathbf{G}(\boldsymbol{y}) = \boldsymbol{I} - \delta t\, \theta\, \mathbf{F}((n+1)\, \delta t, \boldsymbol{y}). \tag{4.38}$$

In eq. (4.38), $\boldsymbol{I}$ is the identity matrix of size $n_p + n_t$ (n_p: number of pipes; n_t: number of tanks), and $\mathbf{F} = \boldsymbol{\partial}\mathbf{f} / \boldsymbol{\partial y}$ is the Jacobian of $\mathbf{f}(t, \boldsymbol{y}) = \dot{\mathbf{y}}(t, \boldsymbol{y})$, defined as:

$$\mathbf{F} = \begin{pmatrix} [\mathbf{F}]_{\mathbf{11}} & [\mathbf{F}]_{\mathbf{12}} \\ [\mathbf{F}]_{\mathbf{21}} & [\mathbf{F}]_{\mathbf{22}} \end{pmatrix} = \begin{array}{cc} \begin{array}{cc} \boldsymbol{\partial q_{0.5}} & \boldsymbol{\partial h_t} \end{array} & \\ \begin{pmatrix} -10^3\, \boldsymbol{\mathcal{I}_p}^{-1} \left(\dfrac{\boldsymbol{\partial \xi_f}}{\boldsymbol{\partial q_{0.5}}} - \boldsymbol{A}^T \dfrac{\boldsymbol{\partial} \widehat{\widehat{\mathbf{h}}}}{\boldsymbol{\partial q_{0.5}}} - \dfrac{\boldsymbol{\mathcal{I}_c}}{10^3} \right) & 10^3\, \boldsymbol{\mathcal{I}_p}^{-1} \boldsymbol{A}_t^T \\ -\dfrac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3} \boldsymbol{A_t} & \boldsymbol{0} \end{pmatrix} & \begin{array}{c} \dot{\mathbf{q}}_{\mathbf{0.5}} \\ \\ \dot{\mathbf{h}}_{\mathbf{t}} \end{array} \end{array}, \tag{4.39}$$

where:

- $\boldsymbol{q_{0.5}}$ and $\boldsymbol{h_t}$ are the vectors of respectively flow rates at middle of pipes and heads at tanks, and $\dot{\mathbf{q}}_{\mathbf{0.5}}$ and $\dot{\mathbf{h}}_{\mathbf{t}}$ their derivatives with respect to time,
- $\boldsymbol{\xi_f}$ and $\widehat{\widehat{\mathbf{h}}}$, defined by eq. (4.4) and eq. (4.34), are the vector functions to compute the friction head-losses in pipes and the penalized heads at junctions,
- $\boldsymbol{A}$ and $\boldsymbol{A_t}$ are the node-pipe incidence matrices reduced to respectively junctions and tanks,
- $\boldsymbol{\mathcal{I}_p}$, $\boldsymbol{\mathcal{I}_c}$ and $\boldsymbol{\mathcal{I}_t}$, defined by eqs. (4.12), (4.13) and (4.17), are the matrices of respectively pipes', convective and tanks' inertia,
- and the coefficient 10^3 permits to convert the flow rates from $\mathrm{l\,s^{-1}}$ to $\mathrm{m^3\,s^{-1}}$.

The Schur complement of eq. (4.39) on $[\mathbf{F}]_{\mathbf{11}}$ is defined as:

$$\mathbf{S} = [\mathbf{F}]_{\mathbf{22}} - [\mathbf{F}]_{\mathbf{21}} \left([\mathbf{F}]_{\mathbf{11}}\right)^{-1} [\mathbf{F}]_{\mathbf{12}}. \tag{4.40}$$

$\mathbf{S}$ can be made definite after regularization, adapting the preconditioning method from [42]; the Newton's method then becomes a quasi-Newton method. However, $\mathbf{S}$ is not symmetric. Thus, at any iteration $n > 0$ of the θ-scheme, and at each iteration $m = 0, 1, 2, \ldots$ of the (quasi-)Newton's method, we use a LU factorization to compute the descent on $\boldsymbol{h_t}^{(n+1,m)}$.

4.2.4.2 Application of damping to the (quasi-)Newton descents

At any iteration $n > 0$ of the θ-scheme, similarly to section 1.2.5.3, we can apply, at each iteration $m = 0, 1, 2, \ldots$ of the (quasi-)Newton's method, a damping to the descents on $\boldsymbol{y}^{(n+1,m)} = \left(\boldsymbol{q}_{\mathbf{0.5}}^{(n+1,m)}, \boldsymbol{h_t}^{(n+1,m)}\right)^T$, when the Goldstein index is less than 0.1 (see appendix A.1.3). In the current θ-scheme, the Goldstein index is computed from the weighted least square (WLS) of $\boldsymbol{g_\theta}^{(n+1,m)}$. This WLS is calculated as:

$$\psi(\boldsymbol{y}^{(n+1,m)}) = \frac{1}{2}\left\|\left(\boldsymbol{W_\theta}^{(n+1,0)}\right)^{\frac{1}{2}} \mathbf{g}(\boldsymbol{y}^{(n+1,m)})\right\|_2^2 = \frac{1}{2}\left(\boldsymbol{y}^{(n+1,m)}\right)^T \boldsymbol{W_\theta}^{(n+1,0)}\, \boldsymbol{y}^{(n+1,m)}, \tag{4.41}$$

where $\boldsymbol{W_\theta}^{(n+1,0)}$ is the diagonal matrix of positive weights:

$$\boldsymbol{W_\theta}^{(n+1,0)} = \begin{pmatrix} \left(\boldsymbol{M_\theta}^{(n+1,0)}\right)^{-1} & \mathbf{0} \\ \mathbf{0} & \left(\boldsymbol{N_\theta}^{(n+1,0)}\right)^{-1} \end{pmatrix}, \tag{4.42}$$

computed at the Newton iteration $m = 0$ only. We define $\boldsymbol{M_\theta}^{(n+1,0)}$ and $\boldsymbol{N_\theta}^{(n+1,0)}$ to be consistent with eq. (4.36), as:

$$\begin{aligned} \boldsymbol{M_\theta}^{(n+1,0)} = \Bigg(& 10^3\,\theta\,\delta t \times \min_{k\in\{1,\ldots,n_p\}} \iota_{p,k} \times \left(\max_{i\in\{1,\ldots,n_0\}} h_{f,i}^{(n+1,0)} + \max_{k\in\{1,\ldots,n_p\}} \frac{\iota_{c,k}}{10^3}\left|q_{0.5,k}^{(n+1,0)}\right| \right) \\ & + \max_{k\in\{1,\ldots,n_p\}} \left|q_{0.5,k}^{(n+1,0)}\right| \Bigg)^2 \boldsymbol{I_{n_p}} \end{aligned} \tag{4.43}$$

and:

$$\boldsymbol{N}_{\boldsymbol{\theta}}^{(n+1,0)} = \left(\frac{\theta\,\delta t}{10^3} \times \min_{i\in\{1,\ldots,n_t\}} \iota_{t,i} \times \left(\max_{i\in\{1,\ldots,n_t\}} q_{L,i} + \sum_{i\in\{1,\ldots,n_j\}} d_i^{(n+1,0)}\right) + \max_{i\in\{1,\ldots,n_t\}} h_{t,i}^{(n+1,0)}\right)^2 \boldsymbol{I}_{\boldsymbol{n_t}}. \tag{4.44}$$

4.2.5 Simulated networks

During all simulations, we use a pressure-independent background lineic leakage outflow rate $\mathrm{q_{LL}}(t)$ equal to $4 \times 10^{-4}\,\mathrm{l\,s^{-1}\,m^{-1}}$ in all pipes. This value is equivalent to the one used by [55, 57] for an average pressure-head of 20 mH2O. As already explained in sections 4.2.1.2 and 4.2.3.3, the use of a pressure-independent background lineic leakage outflow rate is needed to convert the differential algebraic equations (DAEs) (4.19) to the equivalent system of penalized ordinary differential equations (ODEs) (4.20), and then to simulate inertia phenomena without too much computational overhead, i.e., without the need of complicated DAE solvers.

We simulate two networks. The first one is a single leaky pipe that should permit to check the stability of the models and assess inertia phenomena for a very simple network. The second network to simulate is a district metered area (DMA) of the real leaky network C-Town, including consistent users' demand patterns.

4.2.5.1 Single leaky pipe

We simulate the single leaky pipe of fig. 4.2. This pipe has a length $\ell = 1{,}000\,\mathrm{m}$, a diameter $\varnothing_p = 200\,\mathrm{mm}$, and an Hazen-Williams roughness coefficient $c_{\scriptscriptstyle HW} = 120$ (unit-less). Both junction and tank have zero elevation $u = 0\,\mathrm{m}$. The junction has a peak demand $d_n = 10\,\mathrm{l\,s^{-1}}$. The tank has initial and minimum water-levels respectively $l_{init} = 27.5\,\mathrm{mH_2O}$ and $l_{min} = 0\,\mathrm{mH_2O}$, and a diameter $\varnothing_t = 5\,\mathrm{m}$.

Figure 4.2: Single leaky pipe used to simulate inertia phenomena, connecting a tank ■ to a junction ●.

The hourly pattern of the demand at the junction is described for 24 h in fig. 4.3. There, the markers show the values of the demand multipliers μ_d that are retrieved from the EPANET's input file. For stability matter, the time step of the θ-scheme, used in the RWC simulator and denoted δt, needs to be smaller than 1 hour. Thus, to compute the demand over the time grid of the θ-scheme, we need to interpolate the demand at every time $t_n = n\,\delta t$. To do so, while keeping the variation of the demand smooth and preventing any negative value, we use the Piecewise Cubic Hermite Interpolating Polynomial (PCHIP) from the SciPy library [160]; as mentioned in the documentation: "The [PCHIP] interpolator preserves monotonicity in the interpolation data and does not overshoot if the data is not smooth"[2]. In fig. 4.3, the corresponding interpolated values are represented by the continuous line that passes through all circles.

[2] https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.PchipInterpolator.html

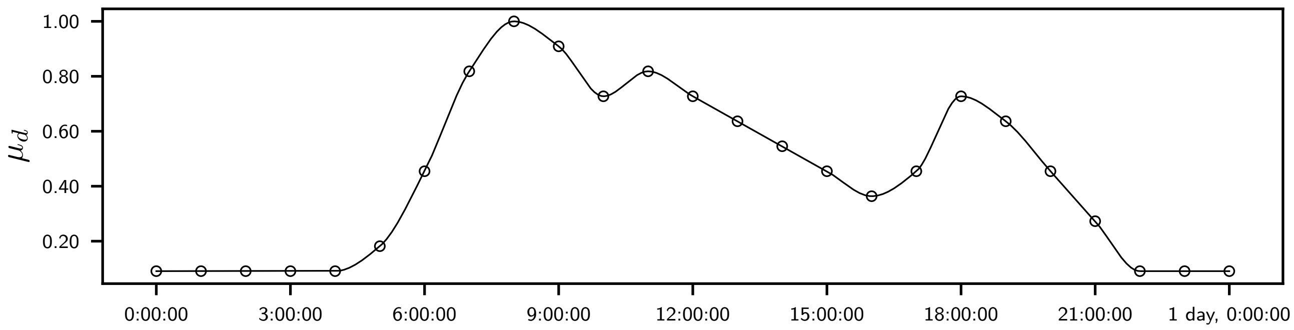


Figure 4.3: Demand pattern at the junction of the single leaky pipe of fig. 4.2. μ_d: demand multiplier to apply on the peak demand.

4.2.6 District metered area of network C-Town

We also simulate the district metered area (DMA) represented in black on the network of fig. 4.4. This network is an adapted version of the network C-Town already used by [57, 120] to test calibration methods and to simulate background leakage outflow rates. It permits to explore our method through a more realistic test case than the previous single leaky pipe. As in chapters 2 and 3, we remove all equipments (i.e., pumps and valves). We then obtain 6 independent sectors, that are easier and quicker to simulate than the whole network C-Town.

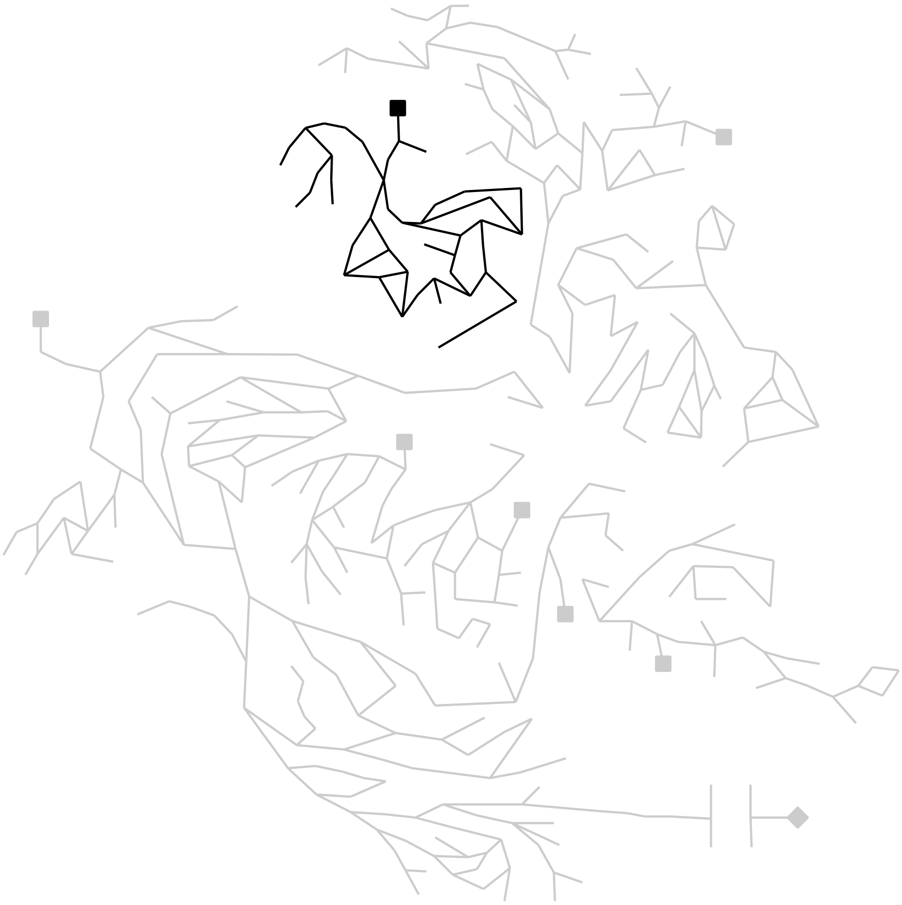


Figure 4.4: In black: district metered area (DMA) of the adapted network C-Town, used to simulate inertia phenomena. In gray: other DMAs (not used). The ■/□ represent the tanks. The ◆ represents a reservoir.

In fig. 4.4, the district metered area (DMA) in black corresponds to the DMA 5 in the EPANET [140]'s input file of the original network C-Town. We choose to simulate this DMA because it is one of the smallest, and it is composed of only one tank. Its small size should permit to compute relevant results quickly; having only one tank make the results easier to interpret and validate, because the tank cannot be filled by another tank.

The hourly pattern to apply on the users' demands in the simulated DMA is described in fig. 4.5. As for the single leaky pipe test case (section 4.2.5.1), the markers show the values of the demand multipliers that are retrieved from the EPANET's input file of the network, and the values of the multipliers between each hourly time step are interpolated using the PCHIP interpolator from the SciPy library [160].

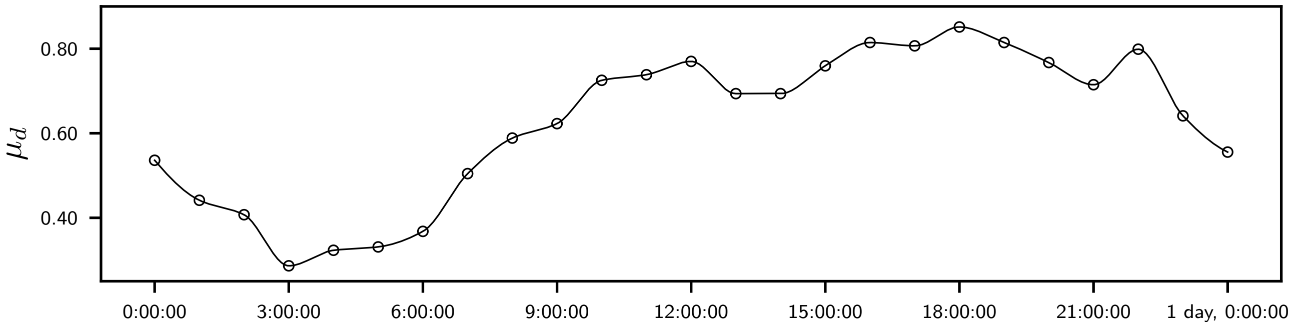


Figure 4.5: Demand pattern in the simulated district metered area (DMA) of the adapted network C-Town (fig. 4.4), used to simulate inertia phenomena. μ_d: demand multipliers to apply on the peak demands.

4.2.7 Tests and metrics

4.2.7.1 Stability of the θ-scheme

To test the stability of the θ-scheme (section 4.2.4) implemented in the RWC simulator, we simulate the single leaky pipe of fig. 4.2 during 24 h, for different values of the time step δt, until the time series of the pressure-head becomes smooth enough. To do so, we define several multipliers $\mu_\theta \in \{1, 0.1, 0.01, 0.001\}$, to be applied on the hourly hydraulic time step $\Delta t = 3{,}600\,\mathrm{s}$. The resulting δt time steps that we use in the θ-scheme are then $\{3600, 360, 36, 3.6\}$ (in seconds).

To check the time series of the pressure-head computed with each μ_θ value, we also solve the ODEs (4.20) with the "BDF" solver[3] from the SciPy library [160], and plot the results obtained with the BDF solver against the ones computed by the θ-scheme. The "BDF" solver from SciPy implements a multi-step variable order implicit method based on a backward-differentiation formula for the derivative approximation; it is adapted for solving stiff ODEs.

Finally, to compare the efficiencies of the θ-scheme and "BDF" solvers, we measure, for each μ_θ, the total CPU time elapsed during each run.

4.2.7.2 Consumption penalties

To assess the capacity of the user's consumption penalties to maintain user's consumption c between 0 and the demand d, we compare the time series of the demand satisfaction d_s (in %) computed by the rigid water column (RWC) simulator in the single leaky pipe of fig. 4.2 during 24 h, with and without applying user's consumption penalties, and using either the θ-scheme or the "BDF" solver. For all runs, we set the time step multiplier of the θ-scheme, μ_θ, to 10^{-3} (i.e., $\delta t = 3.6\,\mathrm{s}$).

The demand satisfaction is computed as in chapter 1:

$$d_s = \frac{c}{d} \times 100. \tag{4.45}$$

[3] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.BDF.html`

We also compute the user's consumption cumulated over the simulation time grid (in m^3), as:

$$c^{cmul} = \frac{1}{10^3} \int_{t_0}^{t_{end}} \mathrm{c}^{\mathrm{cont}}(t)\,\mathrm{d}t. \tag{4.46}$$

where $\mathrm{c}^{\mathrm{cont}}(t)$ is the continuous function obtained by fitting a cubic spline passing through the user's consumption computed every $\Delta t = 1\,\mathrm{h}$; $t_0 = 0\,\mathrm{s}$ and $t_{end} = 24\,\mathrm{h} = 86{,}400\,\mathrm{s}$ are respectively the start and the end times of the simulation. The fit and the definite integral of the spline are accomplished by the 1-D interpolating spline function from the SciPy library[4].

To compare the results obtained with and without penalties, we compute the absolute differences as:

$$\Delta = c_{with}^{cmul} - c_{wout}^{cmul}, \tag{4.47}$$

where c_{with}^{cmul} is the cumulated user's consumption obtained with penalties, and c_{wout}^{cmul} is the one obtained without penalty. We also compute the relative difference (in %) as:

$$\delta = 100\,\frac{\Delta}{c_{wout}^{cmul}}. \tag{4.48}$$

Finally, we measure and compare the total CPU times elapsed, obtained with and without penalties, in the same way as for the cumulated users' consumptions.

4.2.7.3 Inertia phenomena

To quantify the effect of inertia in WDNs, we simulate the single leaky pipe of fig. 4.2 and a DMA of the adapted network C-Town (fig. 4.4) during 24 h, with the RWC simulator and the extended period simulator (EPS). For the θ-scheme of the RWC simulator, we set the time step multiplier μ_θ to 10^{-3} (i.e., $\delta t = 3.6\,\mathrm{s}$).

For the single leaky pipe of fig. 4.2, we compute and plot the time-series of the demand satisfaction, user's consumption, pressure-head at junction, and head at tank; the demand satisfaction is computed as in eq. (4.45). We also compute and compare the cumulated user's consumption, the global demand satisfaction, and the total elapsed CPU time. The cumulated user's consumption is computed as in eq. (4.46), and the global demand satisfaction (in %) as:

$$d_s^{gbl} = 100\,\frac{\displaystyle\int_{t_0}^{t_{end}} \mathrm{c}^{\mathrm{cont}}(t)\,\mathrm{d}t}{\displaystyle\int_{t_0}^{t_{end}} \mathrm{d}^{\mathrm{cont}}(t)\,\mathrm{d}t}. \tag{4.49}$$

where $\mathrm{c}^{\mathrm{cont}}(t)$ and $\mathrm{d}^{\mathrm{cont}}(t)$ are continuous functions obtained by fitting a cubic spline passing through the user's consumption or demand at every $\Delta t = 1\,\mathrm{h}$. We then compute absolute and relative differences between cumulated users' consumptions, global demand satisfactions and total elapsed CPU times, using eqs. (4.47) and (4.48).

[4] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.InterpolatedUnivariateSpline.html`

For the simulated district metered area (DMA) of the adapted network C-Town, we plot the time-series of cumulated user's consumption, global demand satisfaction, minimum pressure-head at junctions, and head at tank. The cumulated user's consumption (in m^3) is computed as:

$$c^{cmul} = \frac{1}{10^3} \int_{t_0}^{t_{end}} \sum_{i \in \{1,\ldots,n_j\}} \mathrm{c}_i^{\mathrm{cont}}(t)\, \mathrm{d}t, \tag{4.50}$$

and the global demand satisfaction as:

$$d_s^{gbl} = 100 \, \frac{c^{cmul}}{\displaystyle\int_{t_0}^{t_{end}} \sum_{i \in \{1,\ldots,n_j\}} \mathrm{d}_i^{\mathrm{cont}}(t)\, \mathrm{d}t}. \tag{4.51}$$

As for the single leaky pipe, we then compute the absolute and relative differences between the cumulated users' consumptions, global demand satisfactions and total elapsed CPU times obtained with either the EPS or the RWC simulator (see eqs. (4.47) and (4.48)).

4.2.8 Implementation

We implement the RWC simulator described in sections 4.2.1 to 4.2.4 and the EPS simulator of appendix D.1 in Python. For the EPS simulator, we extend the code of the steady-state simulator developed in chapter 1 (see section 1.2.6), adding pressure-independent background leakage outflow rates and the update of heads at tanks at each hydraulic time step.

In the RWC simulator, we use the UMFPACK solver [32] to compute the LU decomposition of the Schur complement of the Jacobian matrix (see section 4.2.4.1).

In eq. (4.27), we choose $\delta c = 10^{-4}\,\mathrm{l\,s^{-1}}$ and $\Delta h = 15\,\mathrm{mH_2O}$. Also, we set $\mu_H = 100$ in eq. (4.31). These values of δc, Δh and $\mu_H = 100$ were found, after several numerical tests, to give the best compromise between accuracy and efficiency.

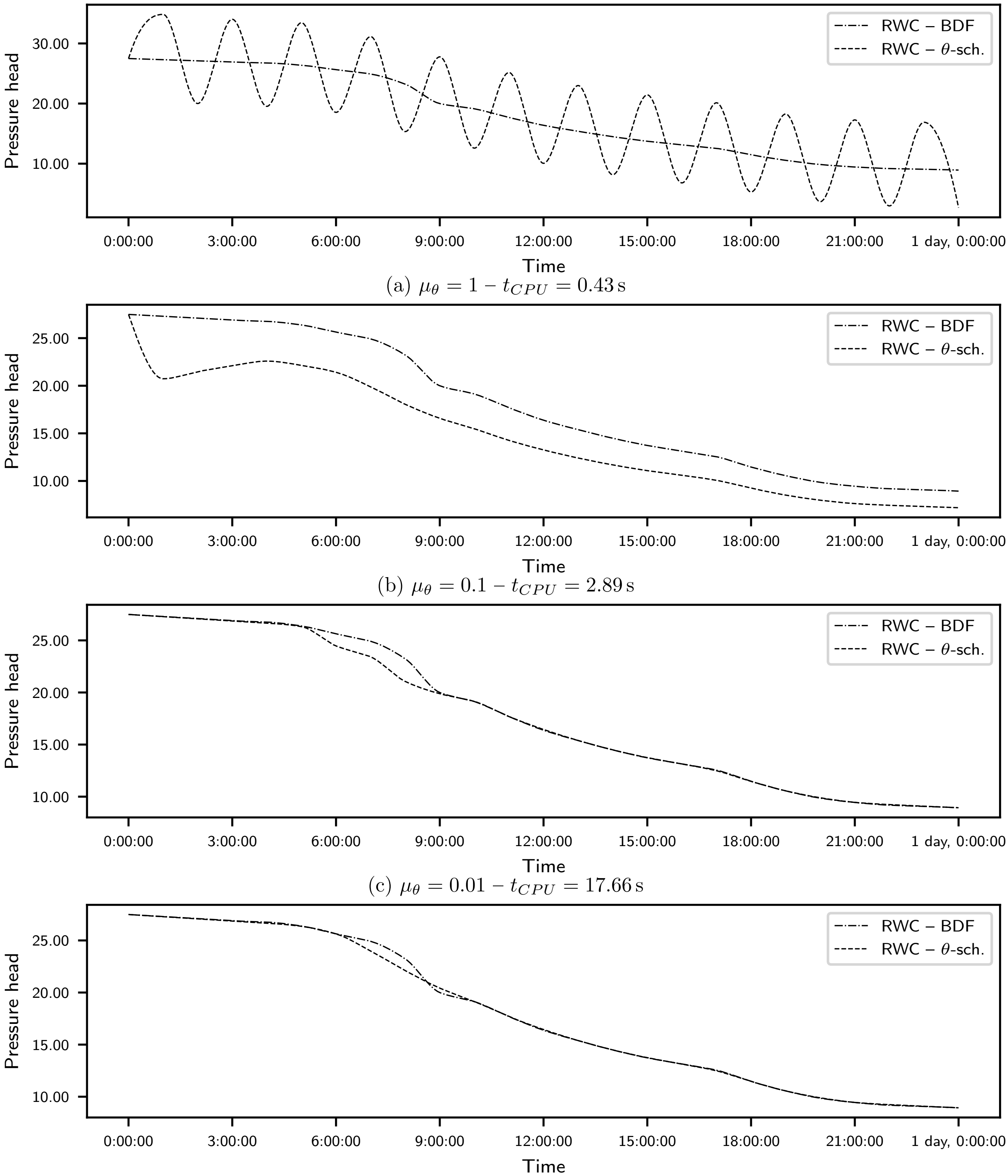
As in previous chapters, all developments are integrated into the Python framework OOPNET [147, 148].

4.3 Results and discussion

This section presents the results obtained when running the tests explained in section 4.2.7.

4.3.1 Stability of the θ-scheme

When simulating the single leaky pipe of fig. 4.2, the θ-scheme starts to stabilize from $\mu_\theta = 0.1$, and computes a time-series of pressure-head that is close to the one calculated by the "BDF" solver from $\mu_\theta = 0.01$ (figs. 4.6a to 4.6c). For $\mu_\theta = 0.001$, the curve obtained from the θ-scheme is even smoother than the one from the "BDF" solver.



(a) $\mu_\theta = 1 - t_{CPU} = 0.43\,\text{s}$

(b) $\mu_\theta = 0.1 - t_{CPU} = 2.89\,\text{s}$

(c) $\mu_\theta = 0.01 - t_{CPU} = 17.66\,\text{s}$

(d) $\mu_\theta = 0.001 - t_{CPU} = 136.63\,\text{s}$

Figure 4.6: Stability of the θ-scheme (section 4.2.4) according to the time discretization multiplier μ_θ when simulating the single leaky pipe of fig. 4.2 with the rigid water column (RWC) simulator. The line "BDF" corresponds to the reference pressure-head (in mH_2O) obtained from the "BDF" solver of the SciPy library [160] (`https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.BDF.html`). With the "BDF" solver, the simulation lasts $t_{CPU} = 0.78\,\text{s}$.

However, the θ-scheme is slower than the "BDF" solver from the SciPy library [160]. Indeed, while the RWC simulator needs only 0.78 s with the "BDF" solver, it needs up to 136.63 s with the θ-scheme. This difference is probably due to the regular time step of integration in the θ-scheme, while the "BDF" solver uses an adaptive time step.

The smoother shape of the time-series obtained with the θ-scheme for $\mu_\theta = 0.001$ is probably due to the tolerance parameter passed to the "BDF" solver. Indeed, in our implementation, the θ-scheme reaches convergence when the relative difference between two iterates becomes less than 10^{-6}, while the "BDF" solver from SciPy stops as soon as this relative difference is less than 10^{-3}. However, it is possible to reduce the relative tolerance of the "BDF" solver; we would then probably obtain exactly the same results with the θ-scheme and "BDF" solvers.

4.3.2 Consumption penalties

Adding penalties to user's consumption when simulating the single leaky pipe of fig. 4.2 permits to maintain the demand satisfaction between 0 and 100 % over the whole simulation time grid (fig. 4.7). Without those penalties, we can see that the demand satisfaction exceeds 100 % during the first 9 hours of the simulation, going up to ≃ 115 %. Both the θ-scheme and "BDF" solvers produce the same outputs.

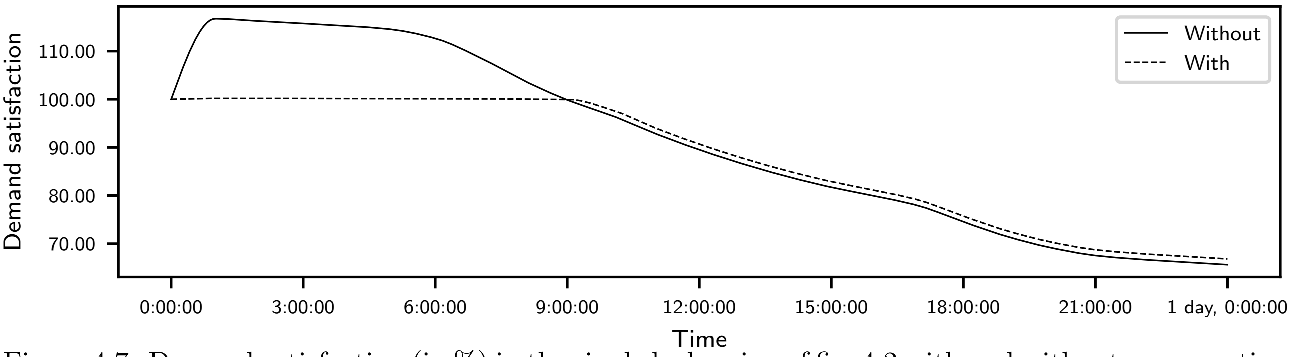


Figure 4.7: Demand satisfaction (in %) in the single leaky pipe of fig. 4.2 with and without consumption penalties.

We also see in fig. 4.7 that the demand satisfaction computed without penalties after time "9:00:00" remains slightly lower than the one obtained with penalties. This is probably a consequence of previous excessive values. Indeed, greater demand satisfaction means also bigger user's consumption, which leads to larger reduction of the volume of water into the tank and of the head at tank. Thus, in this test case, not applying penalties skews the values of the demand satisfaction even when the penalties are not needed anymore.

The differences between the cumulated users' consumptions computed with and without penalties are significant (table 4.1). In fact, using either the θ-scheme or the "BDF" solver, the absolute difference is $\Delta \simeq 6\,\mathrm{m}^3$, which represents a relative difference $\delta \simeq 1.7\,\%$.

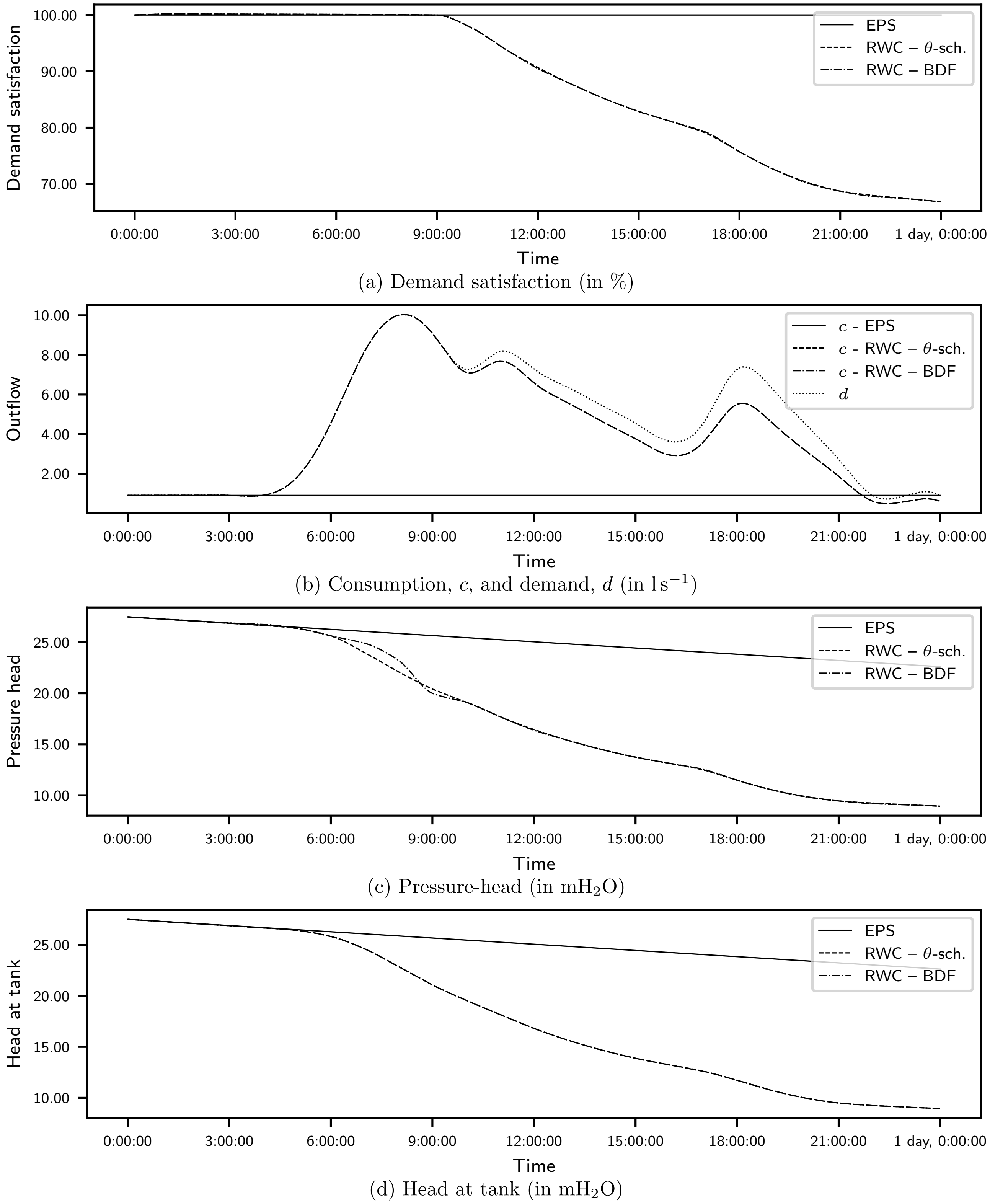
Table 4.1: Cumulated consumptions (c^{cmul}, in m^3) and total elapsed CPU times (t^{tot}_{CPU}, in s) obtained when simulating the single leaky pipe of fig. 4.2 with and without penalties on user's consumption.

Simulator	c^{cmul}				t^{tot}_{CPU}			
	Without	With	Δ	δ	Without	With	Δ	δ
θ-scheme	353.38	347.33	-6.05	-1.71	74.52	134.45	59.93	80.42
BDF	353.51	347.28	-6.23	-1.76	0.20	0.63	0.43	217.75

With this simple test case, we demonstrate the importance of penalties to obtain consistent values of demand satisfaction and cumulated user's consumption. However, we also see that adding these penalties increases the computational times needed by the θ-scheme and "BDF" solvers, by around respectively 80 % and 218 % comparing to the computational time needed when there is no penalty. Actually, adding penalties makes the ODEs (4.20) stiffer and more difficult to solve. But they still remain less complex than the DAEs (4.19).

4.3.3 Inertia phenomena in the single leaky pipe

The results obtained from the RWC simulator show significant differences with the ones computed by the EPS simulator when simulating the single leaky pipe of fig. 4.8. In particular, the time series of demand satisfaction (fig. 4.8a), pressure-head (fig. 4.8c) and head at tank (fig. 4.8d) confirm that taking into account the inertia has a visible effect on the simulated outputs. Also, the time series of the user's consumption shows good consistency with the pattern of the demand (fig. 4.8a), using either the θ-scheme or the "BDF" solver. This confirms the stability of our θ-scheme for this simple test case.



(a) Demand satisfaction (in %)

(b) Consumption, c, and demand, d (in $\mathrm{l\,s^{-1}}$)

(c) Pressure-head (in $\mathrm{mH_2O}$)

(d) Head at tank (in $\mathrm{mH_2O}$)

Figure 4.8: Comparison of the outputs obtained from the extended period (EPS) and the rigid water column (RWC) simulators, when simulating the single leaky pipe of fig. 4.2.

The differences observed graphically in fig. 4.8 are confirmed numerically when calculating the cumulated user's consumption and the global demand satisfaction over the whole time grid (table 4.2). Indeed, whatever the solver we use in the RWC simulator (i.e., θ-scheme or "BDF"), we observe for the cumulated user's consumption an absolute difference $\Delta \simeq 4.6\,\mathrm{m}^3$, which represents a relative difference $\delta \simeq 1.3\,\%$. In terms of global demand satisfaction, the differences are around 1.2 %.

Table 4.2: Cumulated users' consumptions (c^{cmul}, in m^3), global demand satisfactions (d_s^{gbl}, in %) and total CPU times (t_{CPU}^{tot}, in s) obtained when simulating the single leaky pipe of fig. 4.2, with the extended period (EPS) and the rigid water column (RWC) simulator. Δ and δ: absolute and relative (in %) differences between RWC simulator and EPS. "NA" means "not applicable".

Simulator	c^{cmul}			d_s^{gbl}		t_{CPU}^{tot}		
	Value	Δ	δ	Value	Δ	Value	Δ	δ
EPS	351.89	NA	NA	90.24	NA	0.06	NA	NA
RWC – θ-scheme	347.33	-4.56	-1.29	89.08	-1.16	136.71	136.65	213802.63
RWC – BDF	347.22	-4.67	-1.33	89.06	-1.18	0.68	0.62	967.37

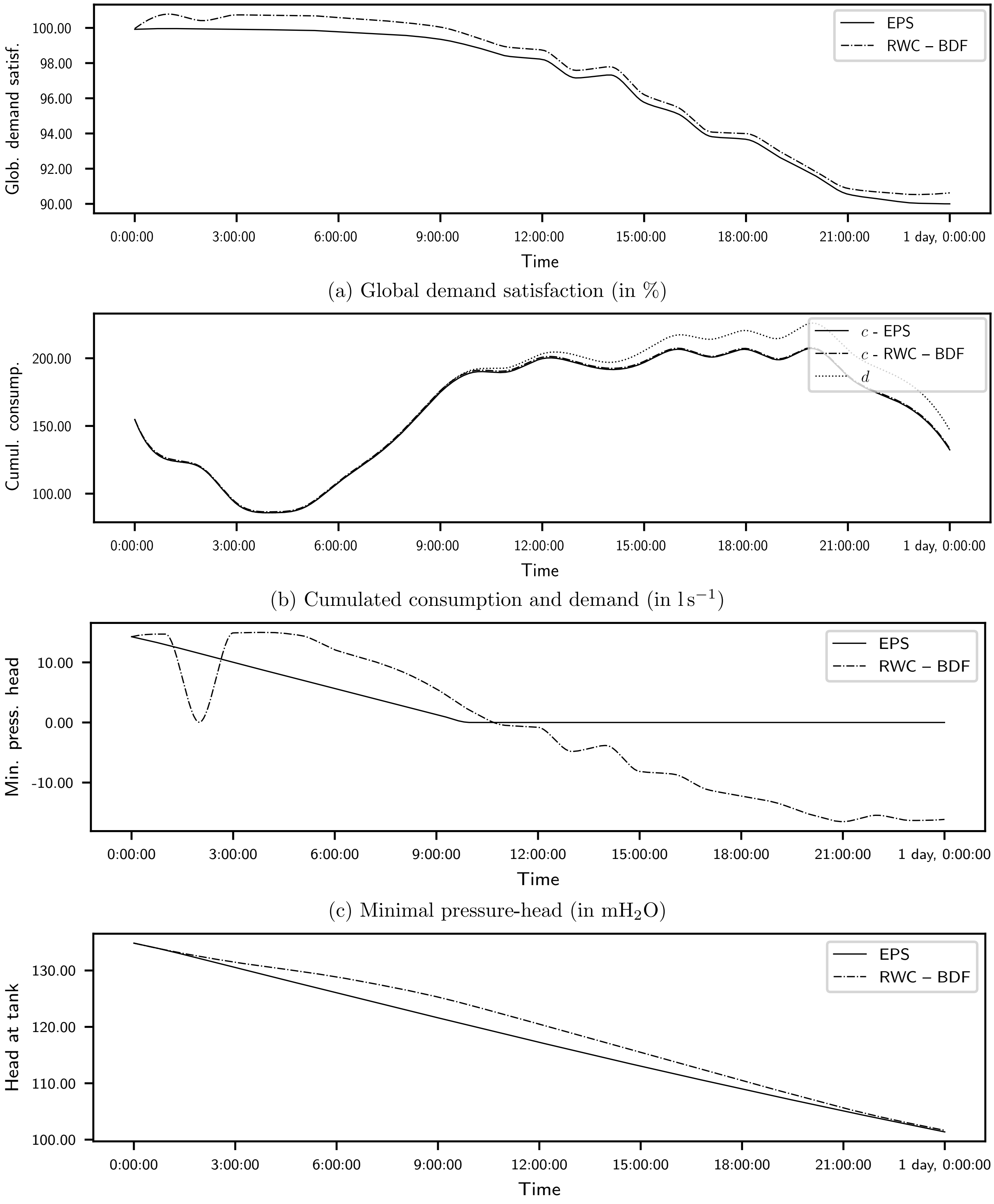
For the θ-scheme, we used in this test case a θ time step multiplier $\mu_\theta = 0.001$, which leads to a time-step $\delta t = 3.6\,\mathrm{s}$. Thus, our results confirm the observations of [112], who found significant inertia phenomena from time steps of a minute.

However, the RWC simulator is much more demanding in terms of computational time, especially when using the θ-scheme. The total CPU time elapsed is multiplied by 10 with the "BDF" solver, and by more than 2000 with the θ-scheme. For this last one, the use of an adaptive time step should reduce considerably the computational time.

4.3.4 Inertia phenomena in the simulated DMA of the adapted network C-Town

The θ-scheme does not converge when simulating the district metered area (DMA) of the network C-Town (fig. 4.4) for any demand multiplier $\mu_d \in \{1, 0.1, 0.01, 0.001\}$. This is probably due to the too strong stiffness of the penalized ODEs (4.20). One way to reduce this stiffness is to diminish the penalty factor μ_Q, defined by eq. (4.27), either decreasing the δh, or increasing the δc parameter (see section 4.2.3). A way to gain in stability, while preserving a high enough μ_Q, is to decrease the μ_d. However, the use of $\mu_d = 10^{-4}$ still not permits the quasi-Newton's method to converge at some simulation times, and setting an even smaller μ_d becomes prohibiting in terms of computational time (i.e., several days) on the machine we use for testing (i.e., an Intel Core i9 with 32 GB of memory). Thus, all RWC simulator results presented in this section come from the SciPy "BDF" solver, its adaptive time step making it much more suitable for such applications. Nevertheless, the θ-scheme in its current state could still be useful for quick tests on smaller systems, since it allows a better control on the solving process and easier introspection.

The RWC simulator converges when it uses the "BDF" solver (fig. 4.9). However, during the first 9 h of the simulation, the global demand satisfaction exceeds slightly 100 %, even after increasing the penalty factor μ_Q to 15×10^{10} (fig. 4.9a). Smaller values of μ_Q leads to more excessive global demand satisfaction.



(a) Global demand satisfaction (in %)

(b) Cumulated consumption and demand (in $\mathrm{l\,s^{-1}}$)

(c) Minimal pressure-head (in $\mathrm{mH_2O}$)

(d) Head at tank (in $\mathrm{mH_2O}$)

Figure 4.9: Comparison of the outputs obtained from the extended period (EPS) and the rigid water column (RWC) simulator, when simulating a DMA of the adapted network C-Town (fig. 4.4).

The over users' consumptions are barely visible at the whole network scale (fig. 4.9b). Also, the time-series of the extrema of the demand satisfactions shows that the penalties maintain quite well the users' consumptions between 0 and the demands (fig. 4.10). Yet, the slight excesses are already enough to hide the differences due to inertia. Thus, solving the penalized ODEs (4.20) does not permit here to estimate the importance of the inertia phenomena. The values of cumulated consumptions c^{cmul} and global demand satisfactions d_s^{gbl} (table 4.3) are presented for the sake of completeness, but are not really exploitable.

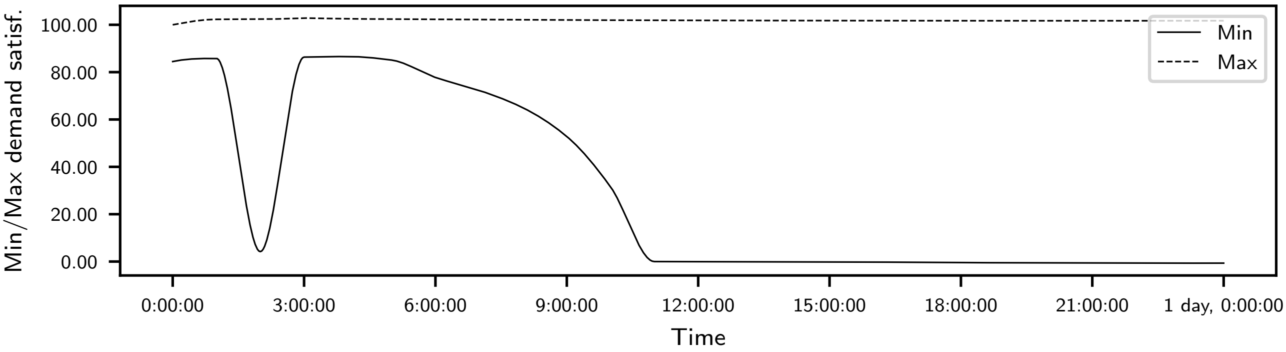


Figure 4.10: Extrema of the demand satisfactions (in %) at all junctions of the simulated DMA (fig. 4.4), obtained from the RWC simulator.

Looking more deeply into the critical variables of the network during the simulation, we observe that the minimum pressure-head at the junctions becomes negative in the second half of the simulation (fig. 4.9c). This is probably just a local behavior, otherwise the cumulated consumptions computed by the RWC simulator would not be so close to the ones computed by the EPS simulator (fig. 4.9b). Still, the presence of these negative pressure-head(s) indicate some instabilities that prevent us from quantifying the differences between the EPS and the RWC simulators.

The CPU times elapsed for the simulation of the EPS and the RWC simulators show that the RWC simulator is now more than 500 times slower than the EPS simulator (table 4.3). This huge difference is probably due to the tremendous penalty factor used to constraint consumptions ($\mu_Q = 15 \times 10^{10}$). Thus, these penalty method, at least in its current state, is maybe not the most suitable to assess inertia phenomena in WDNs. Nevertheless, using progressive values for the penalty factor μ_Q could improve the convergence rate. E.g., at each integration time of the θ-scheme, we could first set $\mu_Q = 15 \times 10^3$ and try to solve the system eq. (4.36) with the Newton's method; if the Newton's method converges to a solution such that the user's consumption c_i verifies $0 \lesssim c_i \lesssim d_i \ \forall i \in \{1, \dots, n_j\}$, then we go to next integration time of the θ-scheme; otherwise, we multiplied μ_Q by 10^n, $n = 1, 2, \dots$, and retry to solve the system eq. (4.36) with the Newton's method, and so forth until the Newton's method converges to a solution such that the user's consumption c_i verifies $0 \lesssim c_i \lesssim d_i \ \forall i \in \{1, \dots, n_j\}$.

Table 4.3: Cumulated users' consumptions (c^{cmul}, in m^3), global demand satisfactions (d_s^{gbl}, in %) and total CPU times (t_{CPU}^{tot}, in s) when simulating the DMA of the adapted network C-Town (fig. 4.4). Δ and δ: absolute and relative (in %) differences between RWC and EPS simulators. "NA" means "not applicable".

Simulator	c^{cmul}			d_s^{gbl}		t_{CPU}^{tot}		
	Value	Δ	δ	Value	Δ	Value	Δ	δ
EPS	14,091.32	NA	NA	95.88	NA	0.17	NA	NA
RWC – BDF	14,164.20	72.88	0.52	96.36	0.48	94.89	94.72	54931.29

4.4 Conclusions

The integration of the inertia phenomena permits better analysis of WDNs, and more robust simulation of events and processes such as users' demands variation, valves opening and closure, pumps switching on/off, etc. Simulating inertia phenomena related to flow and head variations at the time scale of one minute has many applications such as leaks location, water quality monitoring and pressure control [34, 89, 157].

In this chapter, to take into account inertia phenomena, we proposed a new slow-transient formulation of the differential algebraic equations (DAEs) that govern the evolution of the flow rates and of the heads in water distribution networks (WDNs), assuming that the water flowing into the pipes is incompressible. The equations describe pressure-dependent users' consumptions, pressure-independent background leakage outflow rates, Hazen-Williams friction head-losses, and the convective inertia due to background leakages. Pressure-dependent users' consumptions are modeled by the Wagner's pressure-outflow relationship (POR) [162]. Background leakage outflow rates were chosen independent of the pressures to allow the reduction of the DAEs to a simpler system of ordinary differential equations (ODEs). In each pipe, the friction head-loss function takes into account the axial flow variation due to background leakages. From the slow-transient equations, we implemented, with the Python language, a new rigid water column (RWC) simulator that reads EPANET's input files, interpolates the demands, solves the ODEs, and computes derived outputs such as cumulated consumptions, demand satisfactions, and time series of the most important variables.

To solve the reduced ODEs while maintaining users' consumptions between 0 and the users' demands, we defined and applied a Huber loss [73] that penalizes the users' consumptions below 0 and above the users' demands. Then, we implemented a θ-scheme, reusing and adapting the (quasi-)Newton's method and the numerical enhancements already proposed and validated in chapter 1. We chose to implement a θ-scheme to ease the future integration of our developments into the software Porteau [131].

The implemented penalty method requires non-zero demands at junctions. To simulate networks in which some junctions have zero-demand, we therefore set a very small artificial demand to the junctions were the actual demand is zero. These demands increase the stiffness of the system. Another approach would consist in replacing the current penalties by an active-set method, to identify the active constraints and express them as equality constraints, thereby transforming the inequality-constrained initial problem into a stabler equality-constrained subproblem [35].

To quantify inertia phenomena in WDNs, we simulated a single leaky pipe and a district metered area (DMA) of a real network for 24 h with our new RWC simulator, and with an extended period simulator (EPS) that models the same processes but neglects inertia phenomena. We observed significant differences between the RWC and EPS simulators for the single leaky pipe. In the DMA, there still remains a slight over user's consumption when simulated with the RWC simulator. This makes the results difficult to compare with the ones obtained with the EPS simulator. However, the use of incremental penalty factors would probably permit to obtain more accurate results (see, e.g., [118, p. 490]), and/or implementing a multi-step implicit method in place of the θ-scheme would lead to greater stability. Finally, it would be worth testing the solving of the initial slow-transient DAEs with a dedicated solver, like the one from the SUNDIALS (SUite of Nonlinear and DIfferential/ALgebraic equation Solvers) library[5] [47, 67].

Current work represents a first step towards the integration of both pressure-dependent consumptions and inertia in the simulation of WDNs. The formulation of the slow-transient equations as a system of ODEs allows the use of very robust solvers, like the "BDF" solver from the SciPy library [160], based on a multi-step variable order implicit method with backward-differentiation formula for the derivative approximation. With the method of penalties implemented, it is possible to not only model pressure-dependent users' consumptions, but also any other process that can be described by an invertible function of the pressure-heads at nodes. Thus, a future work could consist in simulating local leakage outflow rates that are pressure-dependent, using for example the Fixed And Varied Area Discharges (FAVAD) model [102]. Finally, to observe more inertia phenomena, it would be interesting to simulate valves and pumps (see, e.g., [112, 119]).

RWC simulators are a good compromise between the accuracy of inertia modeling and computational efficiency [111]. The significant results observed here encourage us to carry out further investigations. Also, to ease future collaborative work, all the developments of this chapter are integrated into the Python framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147, 148].

Highlights

- A new pressure-dependent rigid water column simulator
- Explicit integration of inertia phenomena and pressure-dependent consumptions
- Reduction of the system to penalized ordinary differential equations
- First results are encouraging

[5] `https://computing.llnl.gov/projects/sundials`

Chapter 5

Calibration of leakage parameters

Abstract

The leakage parameters in Water Distribution Networks (WDNs), i.e., the type of the leaks and the level of degradation of the pipes, are very difficult to measure physically. Thus, several authors already proposed calibration methods to estimate numerically a part of these parameters. In current chapter, we describe two implicit methods of calibration, to adjust respectively all network-wise and pipe-wise leakage parameters. The calibrated network-wise parameters are computed from the linearization of the Germanopoulos's power function [49], and using average measured data. The pipe-wise parameters are the ones used in the leakage models presented in chapter 2; they can be computed by mean least-squares minimization of the residuals between the simulation outputs and the measured data. The first calibration method (i.e., linearization of the power function to obtain network-wise parameters) is tested on the semi-real experimental network of the International Office for Water (OiEau) and on a sub-sector of the Vienne-Briance-Gorre (VBG) water supply trade union. To do so, we used measures collected during the Oriented Renewal of Pipes (ROC) French research project. The results show parameter values that are consistent with the ones of the literature, and could be used as initial guesses for the calibration of more complex models. The second calibration method is presented analytically for the same networks and discussed; it should permit to identify accurately the leakiest pipes of WDNs and the amount of leakages, provided that enough relevant data are measured.

Keywords:
water distribution network (WDN), pressure-dependent model (PDM), leakage outflow, implicit calibration method, minimization problem

5.1 Introduction

5.1.1 Problem description

Model calibration consists in adjusting uncertain parameters of a model from field data, so that the calibrated model better simulates the real functioning of the system [84, 120].

A model of water distribution network (WDN) is said "calibrated" when it is able to predict the behavior of a WDN over a wide range of operating conditions and water use [163]. Uncertain parameters in WDNs can be pipe roughnesses and diameters, users' demands, valve status and coefficients, pump speed factors, leakage parameters (i.e., type of leakages and level of degradation of pipes), etc. [120, 165]. Field data are usually pressures and flow rates, measured at only few locations of the WDNs [143].

Well calibrated WDN models can produce meaningful results and simulate extreme scenarios (e.g., peak demands, fire flows, burst, etc.) [94, 164]. Then, these models can be used for many different applications: design, control, supervision and operating [116, 142], long term rehabilitation and investments planning [9, 13], optimal leakage management [11, 85, 121], reducing pump costs [117], better use of isolation valves [38], etc. In particular, model calibration is used since several decades to detect leaks and estimate the level of leakage (e.g., [101, 161, 171]).

As already explained in chapter 2, leakages in WDNs can cause significant water losses; thus, leaks have to be detected and repaired as soon as possible, and the components of the networks must be replaced when they become too old [91, 136].

We proposed and implemented in chapter 2 several new pressure-dependent models (PDMs) to simulate leakages accurately. We also described a method for the calibration of these models considering a single leaky pipe. Thus, the question that we would like to answer now is: how to calibrate, for each of the leakage models presented in chapter 2, the leakage parameters associated to each pipe of a WDN?

5.1.2 State of the art

Different categories of methods exist for the calibration of WDN models. In "trial-and-error" methods, the modeler iteratively (1) chooses estimates of the parameters, (2) runs a simulation with these estimates, and (3) compares the results to observed data, repeating steps (1) to (3) until the simulation results and the observations agree (e.g., [163]); however, the convergence rate of these methods is slow [13]. Another category of methods are the "explicit" ones, which extend the initial system of mass-balance and energy equations by adding one new equation per measured data, and solve the extended system numerically; with these methods, the number of unknown parameters to calibrate is limited by the number of available measurements, though [165]. Finally, the third type of methods are the "implicit" methods, which express the calibration process as an optimization problem that tends to minimize the residuals between the simulated and the observed flow rates and pressures [143]; the implicit methods permit to calibrate a larger number of parameters than the number of available measures.

Because of the cost of the sensors and the difficulty of accessing physically many parts of the WDNs, only a few measurements are generally conducted in WDNs; then, the number of parameters to calibrate is very often higher than the number of available measures. Only implicit methods can deal with such situations; for this reason, these methods have already been used for a wide range of applications (e.g., [26, 37, 38, 146, 151]).

The calibration of the leakage parameters with implicit methods permit to identify the leakiest parts of WDNs, and to quantify the level of leakages. In particular, some authors (e.g., [11, 105, 159]) calibrated the pressure-dependent leakage parameters associated to the Fixed and Varied Area Discharges (FAVAD) model [102]. Other authors calibrated the parameters associated to the power function proposed by [49], and which we used as a basis for all new models developed in chapter 2. For example, [117] calibrated the parameter that indicates the level of degradation of the pipes, but not the one that corresponds to the type of the leakages; [101, 171] calibrated both parameters, but either per group of pipes in a real WDN, or for a very small experimental network.

5.1.3 Hypothesis and objectives

We believe that the calibration of a global type of leakage and a global level of degradation for all pipes of a WDN could give a first interesting approximation of the type and the level of leakage in the pipes of the WDN. Thus, our first objective is to fit the leakage parameters of the power function from [49], supposing that the parameters are the same for all pipes.

Also, it is possible to calibrate, for each new model developed in chapter 2, the type of leakage and the level of degradation of each pipe of a WDN, using an implicit calibration method. Thus, our second objective will consist in describing mathematically and discussing a suitable method for such calibration.

To reach these two goals, we will first describe the experimental data that we used for the calibration of the leakage parameters. Next, we will remind the leakage model of [49] and the ones developed in chapter 2, and we will propose an implicit method to calibrate all of them. Finally, we will present the results obtained for the model of [49], and discuss the a priori advantages and limits of the implicit method when applied to the models of chapter 2.

5.2 Methods

5.2.1 Experimental data

5.2.1.1 The ROC project

The experimental data used for the calibration of the models were collected during the Oriented Renewal of Pipes (ROC) research project; this project was leaded by D.Gilbert during 2017-2022. The goal of the ROC project was to carry out multidisciplinary (i.e., engineering sciences, applied mathematics, and human and social sciences) research to develop tools and methods, focused on the understanding and reduction of background leakage outflows and on water quality. The ROC project was funded by the Loire-Bretagne[1] and Adour-Garonne[2]' water agencies, and by the Regional Health Agency (ARS) of the Aquitaine region[3].

To measure the experimental data, [52] designed a mobile measuring device, composed of a towed van equipped with a full hydraulic circuit, which can be connected to a fire hydrant (fig. 5.1). This device permits to inject water into a WDN under regulated pressures, and to record the pressures and the flow rates at the way out of the device. The device will be called just “the Trailer” in all current chapter. The reader can refer to [52] for deeper explanations.

[1] https://agence.eau-loire-bretagne.fr
[2] https://eau-grandsudouest.fr
[3] https://www.nouvelle-aquitaine.ars.sante.fr

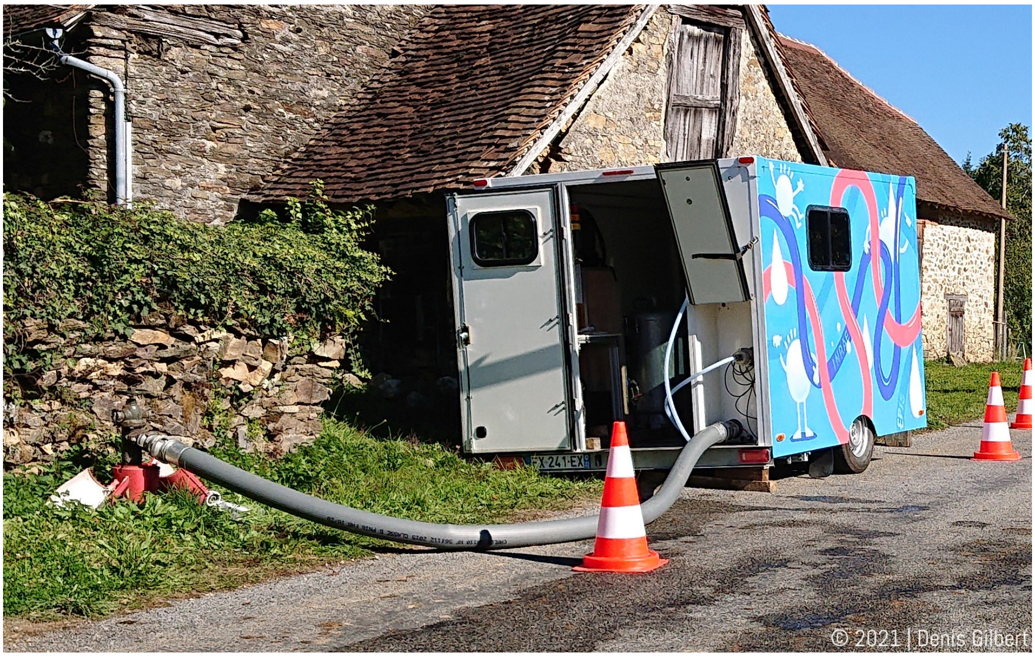


Figure 5.1: Mobile measuring device connected to a fire hydrant, designed by [52]; this device is called "the Trailer" in all current chapter.

5.2.1.2 Semi-real network of the OiEau

The first network that we will calibrate is the experimental network of the International Office for Water[4] (OiEau), located at Limoges (France). This network is not connected to the WDN of the Limoges urban area. Thus, it allows experimentation with more controlled conditions than in a real WDN. In particular, there cannot be unexpected users' consumptions while measuring the flow rates. The network has a total length (i.e., sum of the lengths of all pipes) $\ell^{tot}_{\text{OiE}} = 864\,\text{m}$. It was used in the ROC project to test the Trailer.

To produce leaks, a few holes of different diameters are present at known locations of the network (fig. 5.2). For some holes, the diameters are known accurately; for some other, there are unknown or known with much uncertainty. Most of the holed pipes are doubled in parallel by not holed ones. Gate valves permit to bypass or not some of the leaks, and to isolate some parts of the network. In fig. 5.2, only the valves that will be later used in this chapter to bypass some leaks and isolate some parts of the network are represented.

[4] `https://www.oieau.org/en`

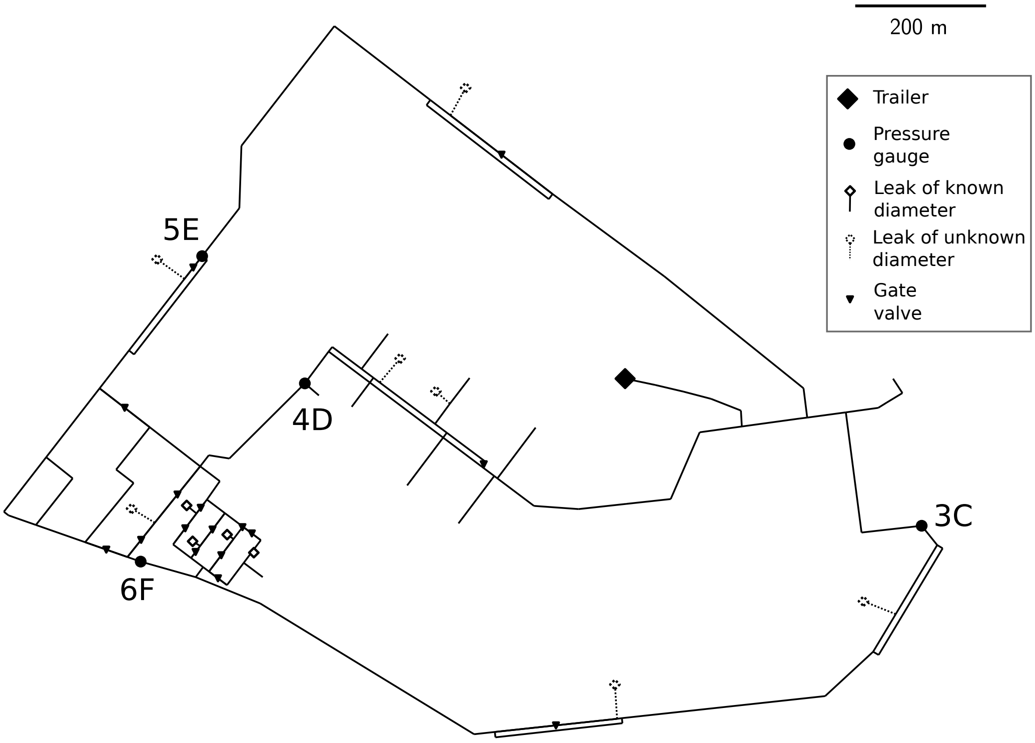


Figure 5.2: Diagram of the experimental network of the International Office for Water (OiEau).

Some of the pipes are made of cast iron, some of PVC (polyvinyl chloride), and some other of PEHD (polyethylene high-density), but these properties are not used in this chapter. The pressures (in bar) can be measured at every second from 4 pressure gauges $\{3C, 4D, 5E, 6F\}$, distributed across the network (fig. 5.2). After conversion to mH2O, we denote $\{p_{3C}^{(t)}, p_{4D}^{(t)}, p_{5E}^{(t)}, p_{6F}^{(t)}\}$ the corresponding pressure-heads measured at any acquisition time t. As a reminder, a pressure p_{bar} in bar can be converted to a pressure-head $p_{\text{mH}_2\text{O}}$ in mH$_2$O as:

$$p_{\text{mH}_2\text{O}} = \frac{10^5\, p_{\text{bar}}}{\rho\, g}, \tag{5.1}$$

where $\rho \approx 1{,}000\,\text{kg}\,\text{m}^{-3}$ is the water density, and $g \approx 9.81\,\text{m}\,\text{s}^{-2}$ is the standard acceleration due to gravity.

Several measuring campaigns were driven on June 16-17th, 2022. For each of them, [52] first opened or closed some of the gate valves, and chose one or several pressure(s) $p_{\text{T}}^{(t)}$ to be imposed by the Trailer at its way out. Then, for each imposed pressure,

1. the Trailer injected water into the WDN, and filled it until the steady-state is reached and $p_{\text{T}}^{(t)}$ became constant;
2. the pressure gauges measured the pressures $\{p_{3C}^{(t)}, p_{4D}^{(t)}, p_{5E}^{(t)}, p_{6F}^{(t)}\}$ at every 1 s, during around 1 h;
3. the acquisition system of the Trailer recorded all these pressures, as well as the flow rates $q_{\text{T}}^{(t)}$ of the water injected by the Trailer;
4. the Trailer continued to inject water to maintain the network under the same pressure.

In this chapter, we choose to use the data measured by [52] on June 17th, 2022, from 14:10 to 15:00. Figure 5.3 shows the pipes that are supplied during these measures; for better clarity, the components disconnected due to closed valves are hidden. The sum of the lengths of all connected pipes is $\ell^{sub}_{\text{OiE}} = 721\,\text{m}$.

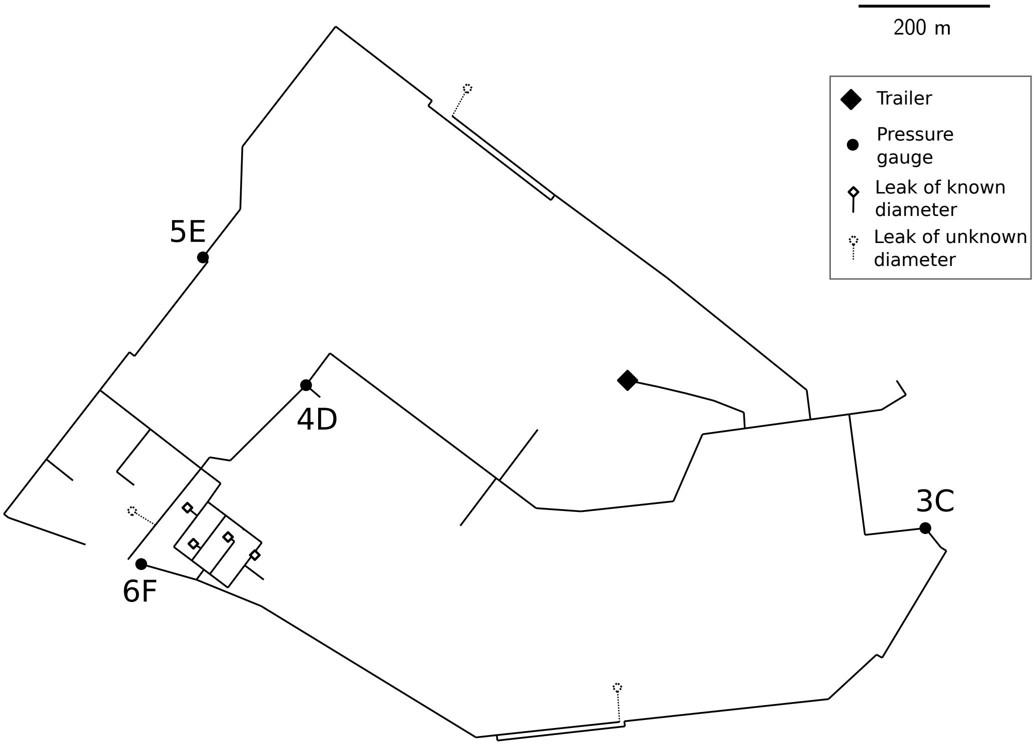
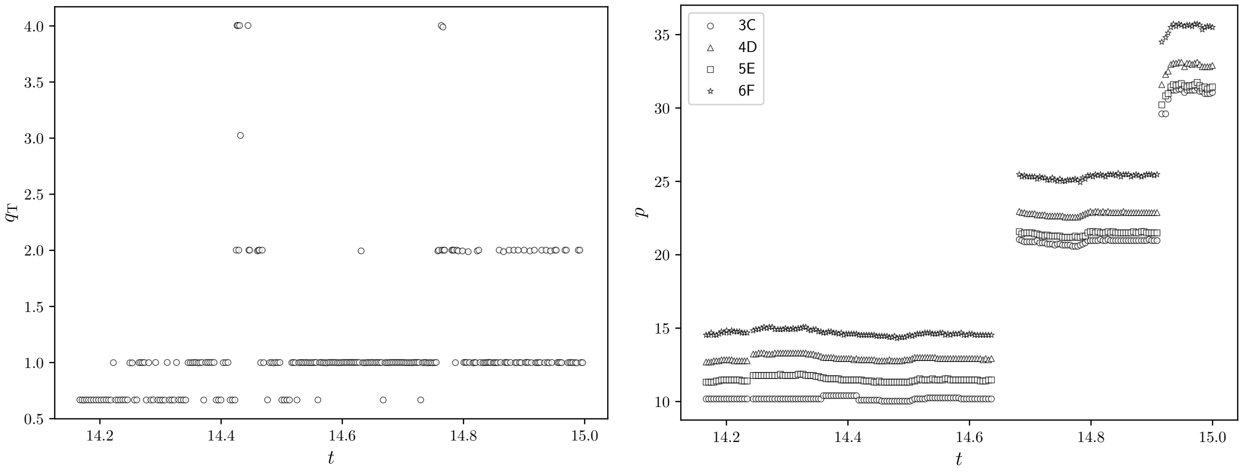


Figure 5.3: Diagram of the experimental network of the International Office for Water (OiEau) after opening or closing some of the gate valves. For better readability, the components disconnected due to closed valves are not displayed on this diagram.

Figure 5.4a shows the time series of the flow rates at the way out of the Trailer during the measuring campaign of June 17th, 2022, from 14:10 to 15:00. Figure 5.4b presents the time series of the pressure-heads measured by the pressure gauges $\{3\text{C}, 4\text{D}, 5\text{E}, 6\text{F}\}$ for the same period of time. The pressure gauge used to measure the pressures has an accuracy of 1 ‰ for pressures between 0.5 and 10 bar [44, p. 1], which corresponds to an error of $\pm 0.1\,\text{mH}_2\text{O}$. The static flow meter used to measure the flow rates has an accuracy of 1 % of the measured flow rates, for flow rates between $16\,\text{l}\,\text{h}^{-1}$ and $80\,\text{m}^3\,\text{h}^{-1}$ [78, p. 3].



(a) Flow rates (in $\mathrm{l\,s^{-1}}$).

(b) Pressure-heads (in $\mathrm{mH_2O}$).

Figure 5.4: Time series of the flow rates at the way out of the Trailer and of the pressure-heads at the pressure gauges $\{3C, 4D, 5E, 6F\}$, measured during the campaign conducted by [52] on June 17th, 2022, from 14:10 to 15:00, in the International Office for Water (OiEau) experimental network. Times are in decimal hours.

We choose the measures made on June 17th, 2022, from 14:10 to 15:00 to calibrate the leakage parameters for several reasons. First, as we can see in fig. 5.3, the open and closed valves leaded to a network with well distributed leaks (i.e., leaks are not all located at the same place). Also, all but one leaks have a pressure gauge at both upstream and downstream; this is very important for measuring the decreases of pressures due to the leaks [165]. Moreover, most of the pressure gauges are far from the leaks; this is also important because the data for model calibration have to be collected as far as possible from the boundary condition (i.e, which are the leaks in this experimentation) [164]. Finally, we can observe in fig. 5.4b that there are 3 distinct levels of pressures imposed by the Trailer during this campaign; like so, it should be possible to see the variation of the lineic leakage outflow rate according to the pressure.

5.2.1.3 Sub-sector from real water distribution network

Next, we calibrate a sub-sector of the Vienne-Briance-Gorre (VBG) water supply trade union[5]; this trade union produces and supplies drinking water to 59 communes of the Haute-Vienne French department. The sub-sector we calibrate was used in the ROC project to assess background leakage outflows in a real networks. The pipes are made of PVC, but this property is not used in the calibration process. The total length of all pipes (i.e., sum of the lengths of all pipes) is $\ell_{\mathrm{VBG}} = 1{,}784\,\mathrm{m}$. The fig. 5.5 shows a top-view of the sub-sector and the location of the Trailer.

[5] `https://www.synd-vbg-eaux.com/`

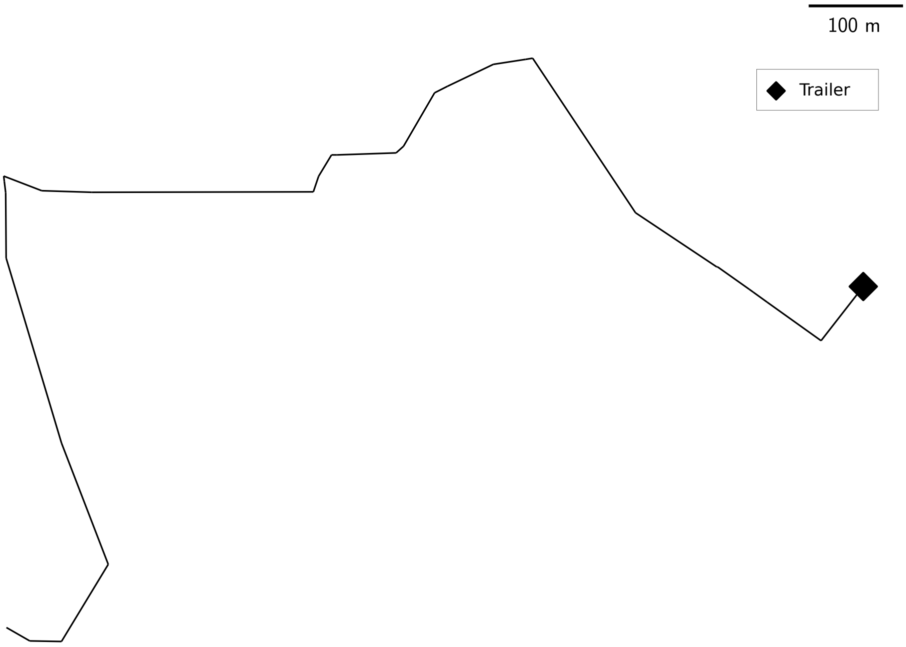


Figure 5.5: Sub-sector of the Vienne-Briance-Gorre (VBG) water supply trade union connected to the Trailer.

During off-peak hours of November, 2021, [52] injected water into the network during 3 periods of time $\{\Delta t_1, \Delta t_2, \Delta t_3\}$, until reaching 3 constant levels of pressures $\{p_{\mathrm{T},1}, p_{\mathrm{T},2}, p_{\mathrm{T},3}\}$ at the way out of the Trailer; for all periods, once the steady-state was reached, the flow rates $\{q_{\mathrm{T},1}, q_{\mathrm{T},2}, q_{\mathrm{T},3}\}$ were measured at the Trailer (table 5.1). As for the experimentation in the semi-real network of the OiEau (see section 5.2.1.2), the error on the measured pressures is of $\pm 0.1\,\mathrm{mH_2O}$, and the one on the measured flow rates is of $1\,\%$ of the measured flow rates.

Time	p_{T}		q_{T}	
	bar	$\mathrm{mH_2O}$	$\mathrm{l\,h^{-1}}$	$\mathrm{l\,s^{-1}}$
Δt_1	2.13	21.71	104.34	2.90×10^{-2}
Δt_2	3.18	32.42	124.13	3.45×10^{-2}
Δt_3	4.26	43.42	180.00	5.00×10^{-2}

Table 5.1: Pressures p_{T} and flow rates q_{T} measured by [52] at the way out of the Trailer, in the network of fig. 5.5. p_{T} and q_{T} are measured respectively in bar and in $\mathrm{l\,h^{-1}}$. p_{T} in $\mathrm{mH_2O}$ are computed using formula (5.1), from p_{T} in bar, $\rho = 1{,}000\,\mathrm{kg\,m^{-3}}$ and $g = 9.81\,\mathrm{m\,s^{-2}}$.

5.2.2 Leakage models to calibrate

5.2.2.1 Network-wise lineic leakage outflow rate

In chapter 2, we used the model:

$$\mathrm{q_{LL}}(\widetilde{p}) = \beta_L \left([\widetilde{p}]^+\right)^{\alpha_L}$$

from [49], to compute the lineic leakage outflow rate q_{LL} (in $\mathrm{l\,s^{-1}\,m^{-1}}$) in any pipe of a WDN, from the approximated average pressure-head $\widetilde{p}$ (in $\mathrm{mH_2O}$) into the pipe. α_L and β_L are the leakage parameters: α_L corresponds to the type of leakage (unit-less) in the pipe, and β_L is the level of degradation of the pipe (in $\mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$). $[...]^+$ represents the positive-part function (see eq. (2.3)). Finally, the approximated average pressure-head $\widetilde{p}$ is computed as:

$$\widetilde{p} = \frac{p_0 + p_\ell}{2},$$

where p_0 and p_ℓ are the pressure-heads at respectively the start and the end of the pipe (see section 2.2.1.1). This model of lineic leakage outflow rate was used as a basis for all background leakage models presented in chapter 2.

In current chapter, we simplify the model from [49], to now suppose that the leakage types and the degradation levels (i.e., parameters α_L and β_L) are the same for all pipes of a WDN. Then, the simplified model computes the average network-wise lineic leakage outflow rate as:

$$\overline{\mathrm{q_{LL}}}(\overline{p}) = \beta_L \left([\overline{p}]^+\right)^{\alpha_L}, \tag{5.2}$$

where $\overline{p}$ is the mean pressure-head in the WDN, and α_L and β_L represent respectively the unique leakage type and degradation level for all pipes in the WDN.

5.2.2.2 Pipe-wise lineic leakage outflow rate

In chapter 2, we described several models $\{\mathrm{M0, M1, M2, M3}\}$ in which the lineic leakage outflow rate $\mathrm{q}_{\mathrm{LL},k}(x)$ along each pipe k is computed from the pressure-heads $p_{0,k}$ and $p_{\ell,k}$ at the extremities of k (see eqs. (2.7), (2.21), (2.26) and (2.30)). Moreover, in models $\{\mathrm{M1, M2, M3}\}$, the flow rate $\mathrm{q}_k(x)$ along k is computed from the flow rate $q_{0.5,k}$ at the middle of k and from the pressures-heads $p_{0,k}$ and $p_{\ell,k}$ at the extremities of k (see eqs. (2.22), (2.27) and (2.31)). In model M0, $\mathrm{q}_k(x) = q_{0.5,k}\ \forall x \in [0, \ell_k]$, with ℓ_k the length of k (see eq. (2.8)).

In any pipe k and $\forall x \in \{0, \ell\}$, the pressure-head $p_{x,k}$ is such that:

$$p_{x,k} = \begin{cases} h_i - u_i & \text{if the node } i \text{ that is located at } x \text{ is a junction,} \\ h_{0,i} & \text{if the node } i \text{ that is located at } x \text{ is a source,} \end{cases} \tag{5.3}$$

where h_i or $h_{0,i}$ is the head at i, and u_i is the elevation at i [165, p. 29]. Thus, for any of $\{\mathrm{M0, M1, M2, M3}\}$, the steady-state equations to compute the vector of flow rates $\boldsymbol{q_{0.5}}$ at the middle of all pipes and the vector of heads $\boldsymbol{h}$ at all junctions at equilibrium was defined in section 2.2.2.1 as:

$$\boldsymbol{\rho}(\boldsymbol{q_{0.5}}, \boldsymbol{h}) = \begin{pmatrix} \boldsymbol{\xi}_{\mathbf{f}}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \boldsymbol{A}^T\boldsymbol{h} - \boldsymbol{A}_0^T\boldsymbol{h_0} \\ \boldsymbol{A}^-\, \mathbf{q}_\ell(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \boldsymbol{A}^+\, \mathbf{q}_0(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_0}) - \mathbf{c}(\boldsymbol{h}) \end{pmatrix} = \boldsymbol{0},$$

where:

- $\boldsymbol{h_O}$ is the vector of heads at source nodes, supposed known and fixed,
- $\boldsymbol{\xi}_{\mathbf{f}}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O})$, $\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O})$ and $\mathbf{q}_{\boldsymbol{\ell}}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O})$ are the vector functions to compute the friction head-losses, the flow rates at pipe starts and the flow rates at pipe ends (for model M0, $\mathbf{q_0}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O}) = \mathbf{q}_{\boldsymbol{\ell}}(\boldsymbol{q_{0.5}}, \boldsymbol{h}, \boldsymbol{h_O}) = \boldsymbol{q_{0.5}}$),
- $\boldsymbol{A}$ and $\boldsymbol{A_O}$ are the incidence matrices reduced to respectively the junction and the source nodes,
- $\boldsymbol{A}^+$ and $\boldsymbol{A}^-$ are respectively the positive and the negative parts of $\boldsymbol{A}$, such that $\boldsymbol{A}^+ - \boldsymbol{A}^- = \boldsymbol{A}$,
- and $\mathbf{c}(\boldsymbol{h})$ is the vector function to compute the users' consumptions at all junctions using the Wagner's [162] POR (pressure outflow relation) defined by eq. (1.10).

5.2.3 Calibration of leakage parameters

5.2.3.1 Network-wise leakage parameters

For the experimental network of the OiEau represented in fig. 5.3, from the measured data plotted in fig. 5.4, we first compute, at each time $t \in [14{:}10, 15{:}00]$, the algebraic mean of the pressure-head (in mH$_2$O) in the whole network as:

$$\overline{p^{(t)}} = \frac{1}{4} \sum_{i \in \{3\text{C},4\text{D},5\text{E},6\text{F}\}} p_i^{(t)}, \tag{5.4}$$

where $p_i^{(t)}$ is the pressure-head (in mH$_2$O) measured at the pressure gauge i at time t. Since there is no other outflow rate than the ones due to leakages, the flow rate $q_{\mathrm{T}}^{(t)}$ (in $\mathrm{l\,s^{-1}}$) measured at every time t at the way out of the Trailer corresponds to the total leakage outflow rate in the network. Thus, we can compute the average lineic leakage outflow rate (in $\mathrm{l\,s^{-1}\,m^{-1}}$) in the whole network at every t as:

$$\overline{q_{LL}^{(t)}} = \frac{q_{\mathrm{T}}^{(t)}}{\ell_{\mathrm{OiE}}^{sub}}, \tag{5.5}$$

where $\ell_{\mathrm{OiE}}^{sub} = 721\,\mathrm{m}$ is the sum of the lengths of all connected pipes. We suppose, $\forall t \in [14{:}10, 15{:}00]$, that $\overline{p^{(t)}}$ and $\overline{q_{LL}^{(t)}}$ verify the model of [49]:

$$\overline{q_{LL}^{(t)}} = \beta_L \left(\overline{p^{(t)}}\right)^{\alpha_L}, \tag{5.6}$$

where α_L and β_L represent respectively the type of the leaks and the level of degradation of all pipes in the network. Thus, the calibrated parameters $\{\alpha_L, \beta_L\}$ are solution of the minimization problem:

$$\begin{gathered}\{\alpha_L, \beta_L\} : \min \left(0.5 \sum_{t \in \{14:10,\dots,15:00\}} \left[\overline{q_{LL}^{(t)}} - \beta_L \left(\overline{p^{(t)}}\right)^{\alpha_L}\right]^2\right) \\ \text{such that:} \{\alpha_L, \beta_L\} \in \mathbb{R}_*^+ \times \mathbb{R}_*^+,\end{gathered} \tag{5.7}$$

and where $\{14{:}10, \dots, 15{:}00\} \subset [14{:}10, 15{:}00]$. To ease the solving of (5.7), we linearize eq. (5.6) as:

$$\log\left(\overline{q_{LL}^{(t)}}\right) = \log(\beta_L) + \alpha_L \log\left(\overline{p^{(t)}}\right), \tag{5.8}$$

where "log" is the natural logarithm function. Then, from all pairs $(\overline{p^{(t)}}, \overline{q_{LL}^{(t)}})$ of the data set

$$\left\{ \left(\overline{p^{(t)}}, \overline{q_{LL}^{(t)}} \right) \mid t \in [14{:}10, 15{:}00] \right\}, \tag{5.9}$$

we can easily find $\{\alpha_L, \log(\beta_L)\}$ by linear regression of eq. (5.8). Finally, we can compute β_L as:

$$\beta_L = e^{\log(\beta_L)}. \tag{5.10}$$

We can take into account the uncertainties on the measured pressures and flow rates by weighting the residuals of problem (5.7), using the accuracy of the measuring devices (i.e., pressure gauges and flow meters).

We can apply the same linear regression method for the sub-sector of the VBG water supply trade union of fig. 5.5, supposing that the users' consumptions remain zero during the entire measuring campaign. Indeed, if the users' consumptions are zero, then the flow rates $q_{\mathrm{T},i}$, $i \in \{1, 2, 3\}$, measured at the way out of the Trailer, correspond to the total leakage outflow rates in the whole network. Thus, from the measured pressure-heads $p_{\mathrm{T},i}$ (in $\mathrm{mH_2O}$) and flow rates $q_{\mathrm{T},i}$ (in $\mathrm{l\,s^{-1}}$) presented in table 5.1, we can compute the parameters α_L and β_L of the network by linear regression of:

$$\log(\overline{q_{LL,i}}) = \log(\beta_L) + \alpha_L \log(p_{\mathrm{T},i}), \tag{5.11}$$

where $\overline{q_{LL,i}}$ is computed as:

$$\overline{q_{LL,i}} = \frac{q_{\mathrm{T},i}}{\ell_{\mathrm{VBG}}}, \tag{5.12}$$

$\ell_{\mathrm{VBG}} = 1{,}784\,\mathrm{m}$ being the sum of the lengths of all pipes in the network.

5.2.3.2 Pipe-wise leakage parameters

The models $\{\mathrm{M0}, \mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$ described in chapter 2 associate one pair of leakage parameters $\{\alpha_{L,k}, \beta_{L,k}\}$ to each pipe k of a WDN. Thus, to calibrate the leakage parameters of each pipe of the experimental network from the OiEau (see fig. 5.3), we can minimize, $\forall t \in [14{:}10, 15{:}00]$, the mean least-squares of the residuals:

- between the flow rates measured $q_{\mathrm{T}}^{meas^{(t)}}$ and simulated $q_{\mathrm{T}}^{simu^{(t)}}$ at the way out of the Trailer,
- and between the pressure-heads measured $\{p_{\mathrm{3C}}^{meas^{(t)}}, p_{\mathrm{4D}}^{meas^{(t)}}, p_{\mathrm{5E}}^{meas^{(t)}}, p_{\mathrm{6F}}^{meas^{(t)}}\}$ and simulated $\{p_{\mathrm{3C}}^{simu^{(t)}}, p_{\mathrm{4D}}^{simu^{(t)}}, p_{\mathrm{5E}}^{simu^{(t)}}, p_{\mathrm{6F}}^{simu^{(t)}}\}$ at the pressure-gauges $\{\mathrm{3C}, \mathrm{4D}, \mathrm{5E}, \mathrm{6F}\}$.

At any acquisition time t, we have:

$$q_{\mathrm{T}}^{simu^{(t)}} = \mathrm{q}_{k_{\mathrm{T}}}(x, q_{0.5}, p_{0,k}, p_{\ell,k}), \tag{5.13}$$

where k_{T} represents the index of the pipe connected to the Trailer; the position x is such that:

$$x = \begin{cases} 0 & \text{if the Trailer is located at the start of } k_{\mathrm{T}}, \\ \ell_{k_{\mathrm{T}}} & \text{if the Trailer is located at the end of } k_{\mathrm{T}}; \end{cases} \tag{5.14}$$

$\mathrm{q}_{k_{\mathrm{T}}}$ is the function to compute the flow rate with any of the models $\{\mathrm{M0}, \mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$ (see eqs. (2.8), (2.22), (2.27) and (2.31)).

In the network from the OiEau, the pressure-gauges $\{3\text{C}, 4\text{D}, 5\text{E}, 6\text{F}\}$ are located at junction nodes; then, each simulated pressure-head $p_i^{simu(t)}$, $i \in \{3\text{C}, 4\text{D}, 5\text{E}, 6\text{F}\}$, is equal to the simulated pressure-head $p_j^{simu(t)}$ at a junction node $j \in \{1..n_j\}$ (n_j: number of junction nodes in the network), such that $h_j^{simu(t)} = p_j^{simu(t)} + u_j$ is an element of the vector of the simulated heads at junctions $\boldsymbol{h^{simu(t)}}$ (u_j: elevation at j). Thus, the leakage parameters $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\}$ of the n_p pipes of the network are solution of the minimization problem:

$$\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\} : \min \left(0.5 \sum_{t \in \{14:10,\dots,15:00\}} \left[\rho(\boldsymbol{q_{0.5}^{simu(t)}}, \boldsymbol{h^{simu(t)}}; \boldsymbol{\alpha_L}, \boldsymbol{\beta_L}; q_{\text{T}}^{meas(t)}, p_{3\text{C}}^{meas(t)}, \dots, p_{6\text{F}}^{meas(t)}) \right]^2 \right)$$
$$\text{such that:} \begin{cases} \{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\} \in \left(\mathbb{R}_*^+\right)^{n_p} \times \left(\mathbb{R}_*^+\right)^{n_p}, \\ q_{\text{T}}^{simu(t)} \approx q_{\text{T}}^{meas(t)}, \\ h_i^{simu(t)} \approx p_i^{meas(t)} + u_i, \ i \in \{3\text{C}, 4\text{D}, 5\text{E}, 6\text{F}\}, \\ \boldsymbol{c}^{(t)} \approx \boldsymbol{0}, \end{cases} \tag{5.15}$$

where $\boldsymbol{c}^{(t)}$ is the vector of users' consumptions at t, and $\{14:10, \dots, 15:00\} \subset [14:10, 15:00]$. To have good initial guesses of $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\}$, we can use the linear regression method described at section 5.2.3.1.

For the sub-sector of the VBG water supply trade union represented in fig. 5.5, we can use the same minimization method. In this case, we look for the $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\}$ such that:

$$\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\} : \min \left(0.5 \sum_{\Delta t_i \in \{\Delta t_1, \Delta t_2, \Delta t_3\}} \left[\rho(\boldsymbol{q_{0.5}^{simu(\Delta t_i)}}, \boldsymbol{h^{simu(\Delta t_i)}}; \boldsymbol{\alpha_L}, \boldsymbol{\beta_L}; q_{\text{T}}^{meas(\Delta t_i)}, p_{\text{T}}^{meas(\Delta t_i)}) \right]^2 \right)$$
$$\text{such that:} \begin{cases} \{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\} \in \left(\mathbb{R}_*^+\right)^{n_p} \times \left(\mathbb{R}_*^+\right)^{n_p}, \\ q_{\text{T}}^{simu(\Delta t_i)} \approx q_{\text{T}}^{meas(\Delta t_i)}, \\ h_{\text{T}}^{simu(\Delta t_i)} \approx p_{\text{T}}^{meas(\Delta t_i)}, \\ \boldsymbol{c}^{(\Delta t_i)} \approx \boldsymbol{0}, \end{cases} \tag{5.16}$$

where $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\}$ are the leakage parameters of the n_p pipes of the sub-sector, and $\boldsymbol{c}^{(\Delta t_i)}$ is the vector of users' consumptions (in $\mathrm{l\,s^{-1}}$) during the time period Δt_i. As for the experimental network of the OiEau, we can also use here the linear regression method described at section 5.2.3.1 to have good initial guesses of $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\}$ and ease the solving of the minimization problem.

5.2.4 Tests and metrics

The tests consist, for the networks of figs. 5.3 and 5.5, in calibrating the network-wise leakage parameters $\{\alpha_L, \beta_L\}$ of the simple model of lineic leakage outflow rate described at section 5.2.2.1 (i.e., the model that considers the same pair $\{\alpha_L, \beta_L\}$ for all pipes). To do so, we use the linear regression method explained at section 5.2.3.1. Then, we compare the obtained calibrated leakage parameters to the ones of the literature. Finally, we will discuss the a priori advantages and limits of the implicit calibration method described at section 5.2.3.2 for the calibration of pipe-wise leakage parameters, according to the sets of measured data available at the writing time of the chapter.

5.2.5 Implementation

We implement the calibration method of section 5.2.3.1 in Python. To compute the linear regression of eq. (5.8), we use the linear least-squares regression routine `scipy.stats.linregress`[6] from the SciPy library.

Our code is integrated into the framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147, 148]. Like so, we can easily read the EPANET's [140] input files that describe the networks of figs. 5.3 and 5.5, find which pipes remain connected to the Trailer after switching on or off the gate valves in the network of fig. 5.2, and plot clear diagrams of each network (see figs. 5.2, 5.3 and 5.5).

The measured data of the whole campaigns were provided by [52] as tab-separated values (TSV) files; we use the Pandas library [103] to read and process them. Finally, to plot the time series of the flow rates and pressure-heads (see fig. 5.4), and to present the results of the calibration graphically, we use the Matplotlib Python library [74].

5.3 Results and discussion

This section presents the results obtained from the calibration of the leakage parameters with the linear regression method described at section 5.2.3.1. We also discuss here the a priori advantages and limits of the least-squares minimization method presented in section 5.2.3.2.

5.3.1 Calibration by linear regression

We present and discuss here the results obtained from the calibration of the leakage parameters with the linear regression method described at section 5.2.3.1.

5.3.1.1 Semi-real network of the OiEau

The linear regression of the eq. (5.8) from the pressures and flow rates measured in the semi-real network of the OiEau (see fig. 5.3) is presented in fig. 5.6. Then, the leakage type of all pipes in the network corresponds to the slope of the straight line; it is equal to $\alpha_L = 0.32$. The level of degradation of all pipes is $\beta_L = 6.79 \times 10^{-4}\,\mathrm{l\,s^{-1}\,m^{-\alpha_L - 1}}$; it is computed by eq. (5.10), with $\log(\beta_L)$ equal to the value of $\log(\overline{q_{LL}})$ when $\log(\overline{p}) = 0$ (i.e., $\log(\beta_L)$ is equal to the y-intercept).

[6] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.linregress.html`

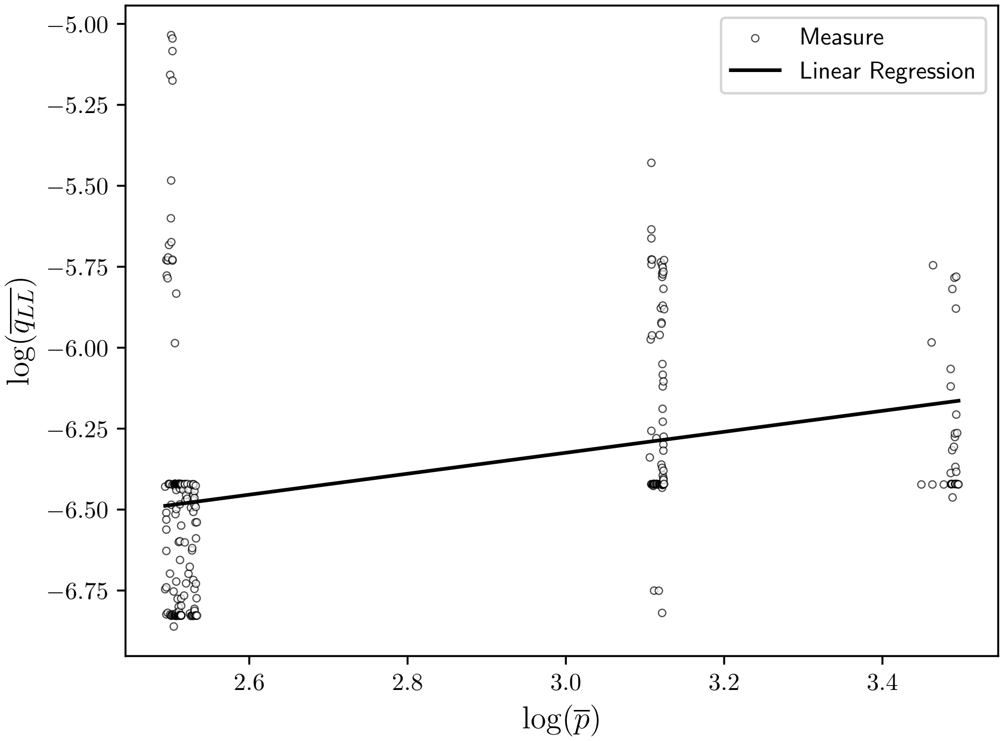


Figure 5.6: Natural logarithm of the average lineic leakage outflow rate ($\log(\overline{q_{LL}})$) in function of the natural logarithm of the average pressure-head ($\log(\overline{p})$). Each $\{\overline{q_{LL}}, \overline{p}\}$ is derived from the measures of the campaign conducted by [52] on June 17th, 2022, from 14:10 to 15:00, in the International Office for Water (OiEau) experimental network.

Since $\alpha_L \leq 0.5$, we can deduce, according to the literature ([9, 91, 101, 102]), that the values of the calibrated leakage parameters correspond to local leaks. This is consistent with the real characteristics of the network. Indeed, as explained in section 5.2.1.2, the actual leaks are holes of different diameters distributed at few positions across the network.

The coefficient of determination of the linear regression is $R^2 = 0.10$; it is very low. This is due to the instability of the flow rate measured at the way out of the Trailer. The use of a flow control valve (FCV) could probably improve this stability.

This method permits to obtain values of the leakage parameters that agree with the usual ones found in the literature. However, it supposes that the type of the leaks and the level of degradation of the pipes are the same everywhere in the network. Leakage parameters depend intrinsically on other pipe characteristics, like their materials (see e.g., [11]). In the network of the OiEau, the material is not the same for all pipes. Thus, one of the limits of this method when applied to this network is that it neglects pipe materials, leading to uncertainties on the calibrated values of the leakage parameters.

Another limit is the use of an average pressure-head, computed from the pressures measured at the pressure gauges. This is needed because we only know, at each acquisition time, the flow rate at the way out of the Trailer. But doing so, we also neglect the difference of head-loss across the network, which leads to erroneous values of leakage outflow rates at every leaks where the pressure-head is far from the average pressure-head. Using flow rates measured at several locations in the network could reduce these errors. Also, as for the pressures, the flow rates should be measured far from the leaks if we want to measure background leakage outflow rates [164, 165].

5.3.1.2 Sub-sector of the VBG water supply trade union

The linear regression of the eq. (5.11) from the pressures and flow rates of table 5.1 is presented in fig. 5.7. As for the semi-real network of the OiEau, the leakage type of all pipes in the sub-sector of the VBG water supply trade union corresponds to the slope of the straight line; it is now equal to $\alpha_L = 0.77$. For this network, the level of degradation associated to all pipes is $\beta_L = 1.48 \times 10^{-6}\,\mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$.

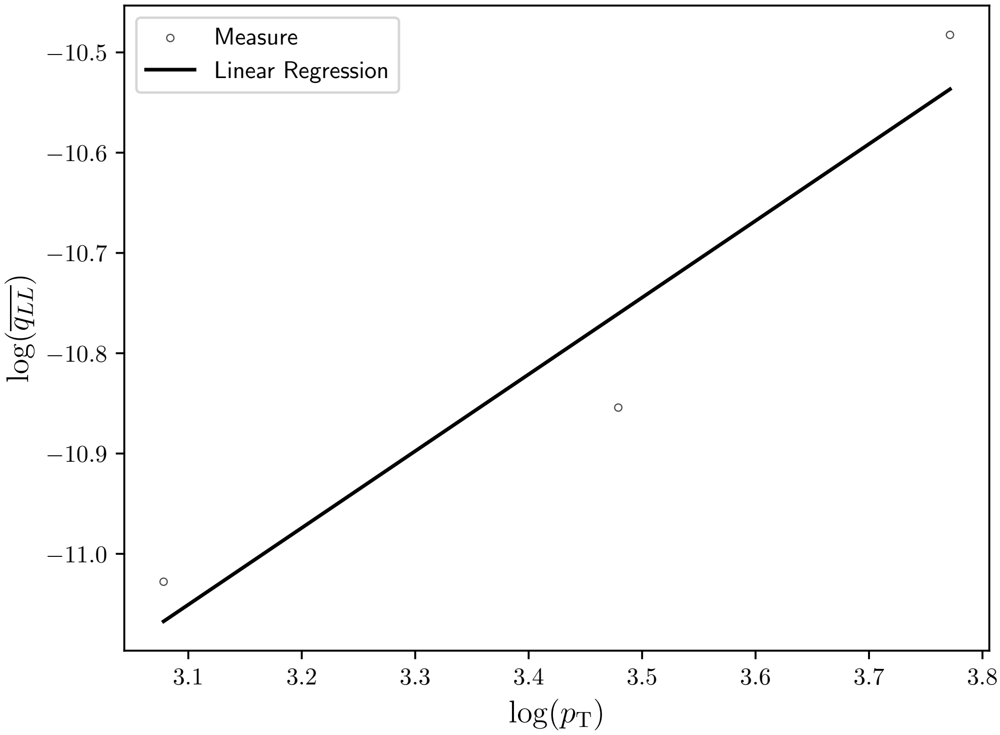


Figure 5.7: Natural logarithm of measured average lineic leakage outflow rate ($\log(\overline{q_{LL}})$) in function of natural logarithm of measured pressure-head at the Trailer ($\log(p_\mathrm{T})$).

Since $\alpha_L > 0.5$, we can deduce, according to the literature ([9, 91, 101, 102]), that the values of the calibrated leakage parameters correspond to background leakage outflows. This is realistic, because the network studied here is maintained by the VBG water supply trade union and does not have any detected leak. Thus, supposing that the users' consumptions are zero, the outflows come from background leakages only.

The coefficient of determination of the linear regression is $R^2 = 0.91$, which shows that the network-wise lineic leakage outflow rate model described at section 5.2.2.1 replicates very well the measures from the VBG's sub-sector. Conversely, we saw in section 5.3.1.1 that the same model gives a much lower R^2 (i.e., $R^2 = 0.10$) when calibrating the local leakage parameters in the semi-real network of the OiEau (see fig. 5.3). Thus, we can deduce that the model (5.2), which is a simplified version of the one from [49], represents more accurately background than local leakages.

Finally, as for the network of the OiEau, this method does not permit to find the leakage parameters associated to each pipe. Moreover, users' consumptions cannot be distinguished from leakage outflows. Thus, even if the measures were conducted during off-peak hours, the leakage outflow rates are potentially overestimated due to unexpected users' consumptions.

5.3.2 Calibration by least-squares minimization

We just discuss here the a priori advantages and limits of the least-squares minimization method presented in section 5.2.3.2 to calibrate the leakage parameters associated to each pipe of the WDNs.

5.3.2.1 Advantages

The obvious advantage of this method is the potential calibration of the leakage type $\alpha_{L,k}$ and the degradation level $\beta_{L,k}$ of each pipe k of a WDN. Indeed, for any pipe k, if $\alpha_{L,k} \leq 0.5$, then it means that there is probably one or several local leak(s) along k, which water authorities should repair as soon as possible. If $\alpha_{L,k} > 0.5$, then k is probably subject to background leakage outflows; in this case, the calibrated $\beta_{L,k}$ could be used by decision makers to prioritize or not the replacement of k, according to the service pressure of the sector where k is located, and taking also into account the levels of degradation of the other pipes in the sector.

In the network of fig. 5.3, there are no other outflows than the ones due to the local leaks at some known locations of the network; likewise, we supposed for the network of fig. 5.5, since the measures were made for off-peak hours, that users' consumptions were zero. For these reasons, we impose for both networks, when calibrating the pipe-wise leakage parameters, the constraint $\boldsymbol{c}^{(t)} \approx \boldsymbol{0}$ at each acquisition time t (see section 5.2.3.2). However, for the real sub-sector of the VBG water supply trade union (see fig. 5.5), unexpected and non negligible users' consumptions could occur even during off-peak hours. In this case, and conversely to the linear regression method described at section 5.2.3.1, the least-squares minimization method should permit to avoid overestimation of leakage outflows. Indeed, if we impose a constant pressure at the way out of the Trailer for a long enough time, and if we filter the measures to keep only the ones that lead to the lowest outflow rates along the pipes, then there is a high probability that the kept measures correspond to the actual leakage outflows.

Finally, the least-squares minimization method is flexible, as other implicit calibration methods. It gives the most suitable leakage parameters according to the available data. But, conversely to some other methods that just calibrate one of the 2 leakage parameters $\alpha_{L,k}$ and $\beta_{L,k}$ (e.g., [9, 117] calibrate only $\beta_{L,k}$), our method tends to calibrate all leakage parameters at once.

5.3.2.2 Limits

When solving the minimization problems (5.15) and (5.16), we actually calibrate not only all leakage parameters $\{\boldsymbol{\alpha_L}, \boldsymbol{\beta_L}\} \in (\mathbb{R}_*^+)^{n_p} \times (\mathbb{R}_*^+)^{n_p}$ (where n_p is the number of pipes in each network), but also the flow rates at the middle of all pipes $\{\boldsymbol{q_{0.5}}^{(t_1)}, \boldsymbol{q_{0.5}}^{(t_2)}, \ldots, \boldsymbol{q_{0.5}}^{\left(t_{n_{acq}}\right)}\} \in \mathbb{R}^{n_p \times n_{acq}}$ and the heads at all junctions $\{\boldsymbol{h'}^{(t_1)}, \boldsymbol{h'}^{(t_2)}, \ldots, \boldsymbol{h'}^{\left(t_{n_{acq}}\right)}\} \in \mathbb{R}^{n'_j \times n_{acq}}$ over the whole measuring campaign

(n_{acq}: number of acquisition times; n'_j: number of junction nodes at which the pressure is not measured by a pressure gauge). Then, despite the few number of junctions where the pressures are measured, the problems (5.15) and (5.16) should have a solution as soon as the number of acquisition times is large enough. However, the relevance of the calibrated values depends much on the locations where the pressures are measured (see e.g., [107, 164]). Ideally, the pressures and the flow rates at both extremities of each pipe should be measured at every acquisition time. But such exhaustive measuring is very difficult to conduct in real WDNs. Thus, it is essential to check the accuracy of any solution of (5.15) and (5.16).

To check the accuracy of the calibrated leakage parameters, it is possible to compute the ranks of the Jacobians associated to the minimization problems (5.15) and (5.16). If this rank is maximal, then it means that there is enough measured data. Otherwise, there is not enough measured data, or the data is not measured at appropriate locations.

In case the rank of one of (5.15) and (5.16) is not maximal, one solution consists in increasing the number of locations where the pressures and the flow rates are measured, and/or to optimize these locations [20, 31, 113]. Another solution is to aggregate some pipes to one longer pipe, and to describe the leakages along this pipe by only one α_L and one β_L; the choice of the pipes that can be aggregated while preserving the accuracy of the calibrated leakage parameters is also an optimization problem [9, 101, 105].

5.4 Conclusions

In this chapter, we described two different implicit methods to calibrate several models of leakage outflows in water distribution networks (WDNs), from few measured data. The first calibration method is based on the linearization of the lineic leakage outflow power equation from [49]. The second calibration method consists in minimizing the mean of the least-squares residuals between the measures and the values simulated by the pressure-dependent leakage models described in chapter 2.

The calibration methods were applied on the semi-real network of the International Office for Water (OiEau) located at Limoges (France), and on a sub-sector of the Vienne-Briance-Gorre (VBG) water supply trade union (Haute-Vienne French department). The measures on both networks, needed for the calibration, were designed and conducted by [52], into the scope of the Oriented Renewal of Pipes (ROC) French research project, using a dedicated mobile measuring device. In this chapter, we presented succinctly the two networks and the measuring device; deeper explanations are available in the annual progress reports of the ROC project (see [52]).

For the first calibration method, after linearization of the lineic leakage outflow power equation from [49], the global leakage parameters associated to each network (i.e., one type of leakage and one level of degradation of the pipes) were determined by least-squares regression of the averaged measured data. We then obtained values of the parameters that are consistent with the literature. However, this method did not permit to find out which pipes are the most critical across the networks, and it neglects the specificity of each pipe (e.g., pipe material) in the calibration process. Nevertheless, it permits to obtain first approximations of the leakage parameters, which could then be reused as initial values for more complex and accurate calibration methods.

We then described mathematically and discussed the a priori advantages and limits of the second calibration method, regarding the available data and the models to calibrate. Actually, the second method should permit to calibrate the leakage type and the level of degradation of each pipe of the networks, provided that enough consistent data are measured. More than their number, the location of the measures is essential to obtain relevant calibrated values of the parameters, and experiments should be conducted considering this need [165]. Future work will consist in the implementation and numerical testing of the method. It will then permit to identify accurately the leakiest pipes of WDNs, and the amount of leakage in each of them.

Finally, all our Python code is integrated into the framework OOPNET (Object-Oriented Python framework for water distribution NETworks analysis) [147, 148], which enhance the reusability of our developments. Once the second calibration method will be implemented, we could then study the propagation of the uncertainties, made on the experimental data when they are measured, to the calibrated leakage parameters, therefore determining the confidence interval for the calibrated parameters. Also, we would like to estimate from how many local leakages, simulated by emitter nodes with the Porteau [131] or EPANET [141] software, it is possible to get calibrated values of leakage parameters that correspond to background leakage outflows; like so, we would know how to use Porteau and Epanet to simulate background leakages accurately, and provide a continuum between the calibration of local and background leakages in WDNs.

Highlights

- Calibration of leakage parameters for two water distribution networks (WDNs)
- Mean least-squares minimization of residuals between simulated and measured data
- Network-wise parameters by linearization of a power function
- Pipe-wise parameters for the pressure-dependent models (PDMs) of chapter 2
- First numerical results are consistent with the literature and spark interest

General conclusion

Main researches and findings

The main goal of this thesis was to develop a more accurate model of background leakage outflows. The 5 sub-objectives were: (1) implement a pressure-dependent model (PDM) of users' consumptions in a quick steady-state Python simulator, (2) develop new models of background leakage outflows that take into account the gradient of pressure along the pipes, (3) extend our models to simulate high-lying nodes, (4) extend our models to consider inertia phenomena, and (5) solve the inverse problem of leakage parameter calibration. Each sub-objective was treated in a dedicated chapter of the thesis, and their level of achievement are reported in the next paragraphs.

In chapter 1, we fully reached our first sub-objective: "implement a pressure-dependent model of users' consumptions in a quick steady-state Python simulator". We tested our simulator on 8 large WDNs (i.e., up-to 19647 pipes and 17986 nodes), and for different levels of users' demands. When the users' demands were fully satisfied, our Python simulator was almost as quick as EPANET. Our simulator is integrated into the framework OOPNET. We finally obtained a well functioning, efficient, flexible and mastered tool to simulate pressure-dependent users' consumptions in WDNs, easy to reuse by the whole community of WDN modelers, and to extend for studying new hydraulic processes.

In chapter 2, we attained our second sub-objective: "develop new models of background leakage outflows that take into account the gradient of pressure along the pipes". We developed a new reference model that discretizes recursively the pipes into sub-pipes until the hydraulic grade line (HGL) along each pipe converges. We also gradually refined a state-of-the-art model, to explicitly integrate flow rates that vary along the pipes because of background leakages, leading to three other new models. All the models were integrated into the Python simulator developed in chapter 1. Numerical tests on a single leaky pipe and on the real network C-Town demonstrated the superiority of our new models when compared with the state-of-the-art one. The choice between our new models depends on the exhaustiveness of the measured data needed to calibrate them, and/or on the level of accuracy to reach.

We globally met our third sub-objective in chapter 3: "extend our models to simulate high-lying nodes". We improved the models of chapter 2, to identify accurately the parts of the pipes that are supplied and the ones that are not. Numerical tests on several small dedicated networks and on an adapted real network showed better estimations of the leakage outflows and HGLs at the pipes' scale. We had some difficulties to find the exact flow rates in parts of a network that are upstream to an high-lying node located in a branch. However, running a post-convergence simulation with users' demands and lineic leakage outflow rates set to zero in all the disconnected parts of the network permitted to obtain good approximation of the upstream flow rates.

We partially reached our fourth-objective in chapter 4. As a reminder, this objective consisted in extending the models developed in the previous chapters so that they can take into account inertia phenomena. To do so, we developed a new rigid water column (RWC) simulator, that neglects water compressibility but considers slow-transient inertia phenomena, i.e., phenomena that occur when the flow rates and the heads vary significantly at the time scale of one minute. We managed to integrate the pressure-dependent users' consumptions model from the chapter 1, but not the pressure-dependent background leakage models from the chapters 2 and 3. Numerical tests on a single leaky pipe confirmed significant effects of inertia phenomena. However, on a district metered area (DMA) of a real network, the penalty method did not permit to constrain enough the consumptions so that they always remain below the demands, and the θ-scheme implemented showed some instabilities.

Finally, in chapter 5, we wanted to solve the inverse problem of calibrating the background leakage models developed in chapter 2. We initiated this research work, describing analytically a well-suited calibration method, and providing a good initial approximation of the parameters at the network-scale. The experimental data used for the calibration were collected during the Oriented Renewal of Pipes (ROC) project, from one semi-real and one real networks.

Main objective achievement and contributions summary

Even if all sub-objectives have not been completely reached, we think that the main objective of the thesis, “develop a more accurate model of background leakage outflows”, is achieved. We brought in this thesis new knowledge and understanding of the pressure-dependent leakage outflows and inertia phenomena in WDNs. In particular, we demonstrated the interest in considering the gradient of pressure along the pipes for better prediction of background leakage outflows, HGLs, and partly-supplied pipes due to high-lying nodes, and the need to model inertia phenomena for more robust simulation of events and processes at the time scale of one minute (e.g., users’ demands variation, valves opening and closure, pumps switching on/off). New Python EPS and RWC simulators that encompass all this researches have been implemented and integrated into the collaborative Python framework OOPNET, for better reusability and easier extension by the community. Finally, this thesis contributes to deciding of the best strategies for optimal functioning and rehabilitation of the WDNs, and to reducing water losses. The related publications are listed at the end of the general introduction (page 37).

Future work

The approaches and methods proposed in this thesis could be improved and/or extended on many aspects. Below are some ideas.

First, the algorithm we implemented to solve the PDM equations of equilibrium in steady-state should be compared mathematically and numerically to the ones that were developed by other authors during the time of this thesis (e.g., [30, 35]). In case the more recent algorithms are better for some applications, it would be interesting to integrate them in our EPS simulator.

Also, in case of high-lying nodes, our current algorithm has difficulty to find the exact flow rates in the upstream parts of a network. We could improve this computation by applying an additional mathematical constraint.

Then, to improve the stability of the θ-scheme used in the RWC simulator, we could apply incremental rather than fixed penalty factors, and use an adaptive time step. It would also be worth testing specific solvers of algebraic differential equations (DAEs). With the current penalty method, we could also try to simulate pressure-dependent local leakage outflows.

Finally, extra work is needed to calibrate accurately the leakage parameters of the background leakage models that we developed. The implicit calibration method described analytically should be implemented and tested; if needed, additional measures could be conducted to obtain more relevant calibrated parameters, or the pipes could be aggregated to reduce the number of parameters to calibrate. Then, we could also study the propagation of the uncertainties to find the confidence interval of the calibrated parameters.

All the future work described above is wide, and concerns not only hydraulic modeling, but also numerical analysis and optimization, model calibration, steady-state and slow-transient simulators, and software development and integration. Thus, a particular attention should be given to the prioritization of the next work, keeping in mind that, to produce useful and reusable research tools: “working software is the primary measure of progress” [8].

Bibliography

[1] J. Almandoz, E. Cabrera, F. Arregui, E. Cabrera, and R. Cobacho. Leakage Assessment through Water Distribution Network Simulation. *J. Water Resour. Plann. Manage.*, 131(6):458–466, Nov. 2005. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)0733-9496(2005)131:6(458). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%282005%29131%3A6%28458%29`.

[2] E. Alperovits and U. Shamir. Design of optimal water distribution systems. *Water Resources Research*, 13(6):885–900, 1977. ISSN 1944-7973. doi:10.1029/WR013i006p00885. URL `https://onlinelibrary.wiley.com/doi/abs/10.1029/WR013i006p00885`. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/WR013i006p00885.

[3] S. Alvisi and M. Franchini. Near-optimal rehabilitation scheduling of water distribution systems based on a multi-objective genetic algorithm. *Civil Engineering and Environmental Systems*, 23(3):143–160, Sept. 2006. ISSN 1028-6608. doi:10.1080/10286600600789300. URL `https://doi.org/10.1080/10286600600789300`. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10286600600789300.

[4] R. Andersen. *Modern methods for robust regression.* Number 152 in Quantitative Applications in the Social Sciences. Sage, 2008. URL `https://us.sagepub.com/en-us/nam/book/modern-methods-robust-regression`.

[5] S. Ateş. Hydraulic modelling of closed pipes in loop equations of water distribution networks. *Applied Mathematical Modelling*, 40(2):966–983, Jan. 2016. ISSN 0307-904X. doi:10.1016/j.apm.2015.06.017. URL `https://www.sciencedirect.com/science/article/pii/S0307904X15003960`.

[6] D. Ayala-Cabrera, O. Piller, M. Herrera, D. Gilbert, and J. Deuerlein. Absorptive Resilience Phase Assessment Based on Criticality Performance Indicators for Water Distribution Networks. *Journal of Water Resources Planning and Management*, 145(9):04019037, Sept. 2019. ISSN 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0001097. URL `https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0001097`. Publisher: American Society of Civil Engineers.

[7] G. J. Barclay, D. F. Griffiths, and D. J. Higham. Theta Method Dynamics. *LMS J. Comput. Math.*, 3:27–43, 2000. ISSN 1461-1570. doi:10.1112/S146115700000019X. URL `https://www.cambridge.org/core/product/identifier/S146115700000019X/type/journal_article`.

[8] K. L. Beck, M. A. Beedle, A. v. Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. J. Mellor,

K. Schwaber, J. Sutherland, and D. A. Thomas. Manifesto for Agile Software Development - Principles behind the Agile Manifesto, 2001. URL https://agilemanifesto.org/principles.html.

[9] L. Berardi, D. Laucelli, R. Ugarelli, and O. Giustolisi. Hydraulic System Modelling: Background Leakage Model Calibration in Oppegård Municipality. In *Procedia Engineering*, volume 119 of *Computing and Control for the Water Industry (CCWI2015) Sharing the best practice in water management*, pages 633–642, Jan. 2015. doi:10.1016/j.proeng.2015.08.916. URL http://www.sciencedirect.com/science/article/pii/S1877705815025862.

[10] L. Berardi, D. Laucelli, R. Ugarelli, and O. Giustolisi. Leakage Management: Planning Remote Real Time Controlled Pressure Reduction in Oppegård Municipality. *Procedia Engineering*, 119:72–81, Jan. 2015. ISSN 1877-7058. doi:10.1016/j.proeng.2015.08.855. URL http://www.sciencedirect.com/science/article/pii/S1877705815025254.

[11] L. Berardi, A. Simone, D. Laucelli, and O. Giustolisi. Feasibility of Mass Balance Approach to Water Distribution Network Model Calibration. In *Procedia Engineering*, volume 186 of *XVIII International Conference on Water Distribution Systems, WDSA2016*, pages 551–558, Jan. 2017. doi:10.1016/j.proeng.2017.03.269. URL http://www.sciencedirect.com/science/article/pii/S1877705817314224.

[12] D. Bernoulli. *Hydrodynamica: sive de viribus et motibus fluidorum commentarii.* Sumptibus Johannis Reinholdi Dulseckeri, 1738.

[13] P. R. Bhave. Calibrating Water Distribution Network Models. *Journal of Environmental Engineering*, 114(1):120–136, Feb. 1988. ISSN 0733-9372. doi:10.1061/(ASCE)0733-9372(1988)114:1(120). URL https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9372%281988%29114%3A1%28120%29. Publisher: American Society of Civil Engineers.

[14] G. J. Bonthuys, M. van Dijk, and G. Cavazzini. Leveraging water infrastructure asset management for energy recovery and leakage reduction. *Sustainable Cities and Society*, 46:101434, Apr. 2019. ISSN 2210-6707. doi:10.1016/j.scs.2019.101434. URL https://www.sciencedirect.com/science/article/pii/S221067071832081X.

[15] P. Boulos and T. Altman. A graph-theoretic approach to explicit nonlinear pipe network optimization. *Applied Mathematical Modelling*, 15(9):459–466, Sept. 1991. ISSN 0307-904X. doi:10.1016/0307-904X(91)90035-N. URL https://www.sciencedirect.com/science/article/pii/0307904X9190035N.

[16] M. Braun, T. Bernard, H. Ung, O. Piller, and D. Gilbert. Model based Investigation of Transport Phenomena in Water Distribution Networks for Contamination Scenarios. *Procedia Engineering*, 70:191–200, Jan. 2014. ISSN 1877-7058. doi:10.1016/j.proeng.2014.02.022. URL https://www.sciencedirect.com/science/article/pii/S1877705814000241.

[17] M. Braun, O. Piller, J. Deuerlein, and I. Mortazavi. Limitations of demand- and pressure-driven modeling for large deficient networks. *Drinking Water Engineering and Science*, 10(2): 93–98, Oct. 2017. ISSN 1996-9457. doi:https://doi.org/10.5194/dwes-10-93-2017. URL https://www.drink-water-eng-sci.net/10/93/2017/dwes-10-93-2017.html.

[18] W. Chen, D. De Kee, and P. N. Kaloni. *Advanced mathematics for engineering and science.* World Scientific, River Edge, NJ, 2003. ISBN 978-981-238-292-4 978-981-238-291-7. OCLC: ocm52571540.

[19] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: CHOLMOD, Supernodal Sparse Cholesky Factorization and Update/Downdate. *ACM Trans. Math. Softw.*, 35(3):1–14, Oct. 2008. ISSN 0098-3500, 1557-7295. doi:10.1145/1391989.1391995. URL `https://dl.acm.org/doi/10.1145/1391989.1391995`.

[20] L. Cheng, D. Kun, T. Jia-peng, and D. Wei-xin. Optimal Placement of Pressure Sensors in Water Distribution System Based on Clustering Analysis of Pressure Sensitive Matrix. *Procedia Engineering*, 186:405–411, Jan. 2017. ISSN 1877-7058. doi:10.1016/j.proeng.2017.03.242. URL `https://www.sciencedirect.com/science/article/pii/S1877705817313954`.

[21] R. Clark, W. Grayman, S. Buchberger, Y. Lee, and D. Hartman. Drinking water distribution systems: an overview. *Water supply systems security. McGraw-Hill, New York*, pages 4–1, 2004.

[22] D. Cohen, M. Lindvall, and P. Costa. An Introduction to Agile Methods. In *Advances in Computers*, volume 62, pages 1–66. Elsevier, 2004. ISBN 978-0-12-012162-5. doi:10.1016/S0065-2458(03)62001-2. URL `https://linkinghub.elsevier.com/retrieve/pii/S0065245803620012`.

[23] M. Collins, L. Cooper, R. Helgason, J. Kennington, and L. LeBlanc. Solving the Pipe Network Analysis Problem Using Optimization Techniques. *Management Science*, 24(7):747–760, Mar. 1978. ISSN 0025-1909, 1526-5501. doi:10.1287/mnsc.24.7.747. URL `http://pubsonline.informs.org/doi/abs/10.1287/mnsc.24.7.747`.

[24] A. F. Colombo and B. W. Karney. Energy and Costs of Leaky Pipes: Toward Comprehensive Picture. *J. Water Resour. Plann. Manage.*, 128(6):441–450, Nov. 2002. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)0733-9496(2002)128:6(441). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%282002%29128%3A6%28441%29`.

[25] Coronado-Hernández Oscar E., Fuertes-Miquel Vicente S., Iglesias-Rey Pedro L., and Martínez-Solano Francisco J. Rigid Water Column Model for Simulating the Emptying Process in a Pipeline Using Pressurized Air. *Journal of Hydraulic Engineering*, 144(4):06018004, Apr. 2018. doi:10.1061/(ASCE)HY.1943-7900.0001446. URL `http://ascelibrary.org/doi/full/10.1061/%28ASCE%29HY.1943-7900.0001446`.

[26] F. Costanzo, A. F. Morosini, P. Veltri, and D. Savić. Model Calibration as a Tool for Leakage Identification in WDS: A Real Case Study. In *Procedia Engineering*, volume 89 of *16th Water Distribution System Analysis Conference, WDSA2014*, pages 672–678, Jan. 2014. doi:10.1016/j.proeng.2014.11.493. URL `https://www.sciencedirect.com/science/article/pii/S1877705814026083`.

[27] B. Coulbeck. Dynamic simulation of water distribution systems. *Mathematics and Computers in Simulation*, 22(3):222–230, Sept. 1980. ISSN 0378-4754. doi:10.1016/0378-4754(80)90049-X. URL `https://www.sciencedirect.com/science/article/pii/037847548090049X`.

[28] N. R. Council. *Drinking Water Distribution Systems: Assessing and Reducing Risks*. National Academies Press, Washington, D.C., Dec. 2006. ISBN 978-0-309-10306-0. doi:10.17226/11728. URL http://www.nap.edu/catalog/11728.

[29] E. Creaco and T. Walski. Economic Analysis of Pressure Control for Leakage and Pipe Burst Reduction. *Journal of Water Resources Planning and Management*, 143(12):04017074, Dec. 2017. doi:10.1061/(ASCE)WR.1943-5452.0000846. URL https://ascelibrary.org/doi/full/10.1061/(ASCE)WR.1943-5452.0000846.

[30] E. Creaco, A. Di Nardo, M. Iervolino, and G. Santonastaso. High-Order Global Algorithm for the Pressure-Driven Modeling of Water Distribution Networks. *J. Water Resour. Plann. Manage.*, 148(3):04021109, Mar. 2022. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0001524. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0001524.

[31] M. A. Cugueró-Escofet, V. Puig, and J. Quevedo. Optimal pressure sensor placement and assessment for leak location using a relaxed isolation index: Application to the Barcelona water network. *Control Engineering Practice*, 63:1–12, June 2017. ISSN 0967-0661. doi:10.1016/j.conengprac.2017.03.003. URL https://www.sciencedirect.com/science/article/pii/S096706611730059X.

[32] T. A. Davis. *Direct Methods for Sparse Linear Systems*. Fundamentals of Algorithms. Society for Industrial and Applied Mathematics, Jan. 2006. ISBN 978-0-89871-613-9. doi:10.1137/1.9780898718881. URL https://epubs.siam.org/doi/book/10.1137/1.9780898718881.

[33] M. De Marchis, C. M. Fontanazza, G. Freni, G. L. Loggia, E. Napoli, and V. Notaro. Analysis of the impact of intermittent distribution by modelling the network-filling process. *Journal of Hydroinformatics*, 13(3):358–373, Dec. 2010. ISSN 1464-7141. doi:10.2166/hydro.2010.026. URL https://doi.org/10.2166/hydro.2010.026.

[34] J. A. Delgado-Aguiñaga, F. I. Becerra-López, L. Torres, G. Besançon, V. Puig, and M. R. J. Magaña. Dynamic model for a water distribution network: application to leak diagnosis and quality monitoring. *IFAC-PapersOnLine*, 53(2):16679–16684, Jan. 2020. ISSN 2405-8963. doi:10.1016/j.ifacol.2020.12.1091. URL https://www.sciencedirect.com/science/article/pii/S2405896320314671.

[35] J. W. Deuerlein, O. Piller, S. Elhay, and A. R. Simpson. Content-Based Active-Set Method for the Pressure-Dependent Model of Water Distribution Systems. *J. Water Resour. Plann. Manage.*, 145(1):04018082, 2019. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0001003. URL http://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0001003.

[36] K. Diao, Z. Wang, G. Burger, C.-H. Chen, W. Rauch, and Y. Zhou. Speedup of water distribution simulation by domain decomposition. *Environmental Modelling & Software*, 52:253–263, Feb. 2014. ISSN 13648152. doi:10.1016/j.envsoft.2013.09.025. URL https://linkinghub.elsevier.com/retrieve/pii/S1364815213002247.

[37] N. C. Do, A. R. Simpson, J. W. Deuerlein, and O. Piller. Calibration of Water Demand Multipliers in Water Distribution Systems Using Genetic Algorithms. *Journal of Water Resources Planning and Management*, 142(11):04016044, Nov. 2016. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0000691. URL http://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000691.

[38] N. C. Do, A. Simpson, J. Deuerlein, and O. Piller. A Hybrid Method for Calibration of Unknown Partially/Fully Closed Valves in Water Distribution Systems. In *WDSA / CCWI Joint Conference Proceedings*, volume 1, 2018. URL https://ojs.library.queensu.ca/index.php/wdsa-ccw/article/view/12222.

[39] H.-F. Duan. Transient Wave Scattering and Its Influence on Transient Analysis and Leak Detection in Urban Water Supply Systems: Theoretical Analysis and Numerical Validation. *Water*, 9(10):789, Oct. 2017. ISSN 2073-4441. doi:10.3390/w9100789. URL https://www.mdpi.com/2073-4441/9/10/789. Number: 10 Publisher: Multidisciplinary Digital Publishing Institute.

[40] G. Dubois. *Modeling and Simulation : Challenges and Best Practices for Industry.* CRC Press, Mar. 2018. ISBN 978-1-351-24112-0. doi:10.1201/9781351241137. URL https://www.taylorfrancis.com/books/9781351241120.

[41] H. A. El-Ghandour, S. M. Elabd, and E. Elbeltagi. Assessment of optimal water distribution systems design under steady-state and transient conditions due to pipe roughness uncertainty. *Ain Shams Engineering Journal*, 12(1):465–473, Mar. 2021. ISSN 2090-4479. doi:10.1016/j.asej.2020.09.014. URL https://www.sciencedirect.com/science/article/pii/S2090447920302094.

[42] S. Elhay and A. R. Simpson. Dealing with Zero Flows in Solving the Nonlinear Equations for Water Distribution Systems. *J. Hydraul. Eng.*, 137(10):1216–1224, Oct. 2011. ISSN 0733-9429, 1943-7900. doi:10.1061/(ASCE)HY.1943-7900.0000411. URL http://ascelibrary.org/doi/10.1061/%28ASCE%29HY.1943-7900.0000411.

[43] S. Elhay, O. Piller, J. Deuerlein, and A. R. Simpson. A Robust, Rapidly Convergent Method That Solves the Water Distribution Equations for Pressure-Dependent Models. *J. Water Resour. Plann. Manage.*, 142(2):04015047, Feb. 2016. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0000578. URL http://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000578.

[44] Endress+Hauser. Deltabar PMD55B, 2023. URL https://portal.endress.com/dla/5001127/3770/000/04/TI01510PEN_0622-00.pdf.

[45] M. Ferrante, E. Todini, C. Massari, B. Brunone, and S. Meniconi. Equivalent hydraulic resistance to simulate pipes subject to diffuse outflows. *Journal of Hydroinformatics*, 14(1):65–74, Jan. 2012. ISSN 1464-7141. doi:10.2166/hydro.2011.043. URL /jh/article/14/1/65/3142/Equivalent-hydraulic-resistance-to-simulate-pipes.

[46] O. Fujiwara and T. Ganesharajah. Reliability assessment of water supply systems with storage and distribution networks. *Water Resour. Res.*, 29(8):2917–2924, Aug. 1993. ISSN 00431397. doi:10.1029/93WR00857. URL http://doi.wiley.com/10.1029/93WR00857.

[47] D. J. Gardner, D. R. Reynolds, C. S. Woodward, and C. J. Balos. Enabling new flexibility in the SUNDIALS suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, 2022. doi:10.1145/3539801. Publisher: ACM.

[48] A. George and J. W. H. Liu. *Computer solution of large sparse positive definite systems.* Prentice-Hall series in computational mathematics. Prentice-Hall, Englewood Cliffs, N.J, 1981. ISBN 978-0-13-165274-3.

[49] G. Germanopoulos. A technical note on the inclusion of pressure dependent demand and leakage terms in water supply network models. *Civil Engineering Systems*, 2(3):171–179, Sept. 1985. ISSN 0263-0257. doi:10.1080/02630258508970401. URL `http://www.tandfonline.com/doi/abs/10.1080/02630258508970401`.

[50] M. S. Ghidaoui. On the fundamental equations of water hammer. *Urban Water Journal*, 1(2): 71–83, June 2004. ISSN 1573-062X. doi:10.1080/15730620412331290001. URL `https://doi.org/10.1080/15730620412331290001`.

[51] D. Gilbert, E. Abraham, I. Montalvo, and O. Piller. Iterative Multistage Method for a Large Water Network Sectorization into DMAs under Multiple Design Objectives. *Journal of Water Resources Planning and Management*, 143(11):04017067, Nov. 2017. ISSN 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0000835. URL `https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000835`. Publisher: American Society of Civil Engineers.

[52] D. Gilbert, T. Hémion, and A. Husson. Assessment of losses in water supply network – ROC project. Progress reports 2019-2022, INRAE, 2022.

[53] O. Giustolisi and L. Berardi. Prioritizing Pipe Replacement: From Multiobjective Genetic Algorithms to Operational Decision Support. *Journal of Water Resources Planning and Management*, 135(6):484–492, Nov. 2009. ISSN 0733-9496. doi:10.1061/(ASCE)0733-9496(2009)135:6(484). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%282009%29135%3A6%28484%29`. Publisher: American Society of Civil Engineers.

[54] O. Giustolisi and D. Laucelli. Water Distribution Network Pressure-Driven Analysis Using the Enhanced Global Gradient Algorithm (EGGA). *J. Water Resour. Plann. Manage.*, 137(6):498–510, Nov. 2011. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0000140. URL `http://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000140`.

[55] O. Giustolisi, D. Savic, and Z. Kapelan. Pressure-Driven Demand and Leakage Simulation for Water Distribution Networks. *Journal of Hydraulic Engineering*, 134(5):626–635, 2008. doi:10.1061/(ASCE)0733-9429(2008)134:5(626). URL `https://ascelibrary.org/doi/abs/10.1061/(ASCE)0733-9429(2008)134:5(626)`.

[56] O. Giustolisi, L. Berardi, and D. Laucelli. Generalizing WDN simulation models to variable tank levels. *Journal of Hydroinformatics*, 14(3):562–573, July 2012. ISSN 1464-7141. doi:10.2166/hydro.2011.224. URL `/jh/article/14/3/562/3165/Generalizing-WDN-simulation-models-to-variable`.

[57] O. Giustolisi, L. Berardi, D. Laucelli, D. Savic, T. Walski, and B. Brunone. Battle of Background Leakage Assessment for Water Networks (BBLAWN) at WDSA Conference 2014. *Procedia*

Engineering, 89:4–12, Jan. 2014. ISSN 1877-7058. doi:10.1016/j.proeng.2014.11.153. URL `http://www.sciencedirect.com/science/article/pii/S1877705814022681`.

[58] A. A. Goldstein. *Constructive real analysis.* Courier Corporation, 2013.

[59] R. Guidotti, H. Chmielewski, V. Unnikrishnan, P. Gardoni, T. McAllister, and J. van de Lindt. Modeling the resilience of critical infrastructure: the role of network dependencies. *Sustainable and Resilient Infrastructure*, 1(3-4):153–168, Nov. 2016. ISSN 2378-9689. doi:10.1080/23789689.2016.1254999. URL `https://doi.org/10.1080/23789689.2016.1254999`. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/23789689.2016.1254999.

[60] I. Gupta, J. Bassin, A. Gupta, and P. Khanna. Optimization of water distribution system. *Environmental Software*, 8(2):101–113, 1993. ISSN 02669838. doi:10.1016/0266-9838(93)90020-I. URL `https://linkinghub.elsevier.com/retrieve/pii/026698389390020I`.

[61] R. Gupta and P. R. Bhave. Comparison of Methods for Predicting Deficient-Network Performance. *Journal of Water Resources Planning and Management*, 122(3):214–217, May 1996. ISSN 0733-9496. doi:10.1061/(ASCE)0733-9496(1996)122:3(214). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%281996%29122%3A3%28214%29`. Publisher: American Society of Civil Engineers.

[62] L. Gyergyek and S. Presern. Simulation and optimal control of large water distribution systems. *Mathematics and Computers in Simulation*, 24(5):385–392, Oct. 1982. ISSN 0378-4754. doi:10.1016/0378-4754(82)90027-1. URL `https://www.sciencedirect.com/science/article/pii/0378475482900271`.

[63] A. Hagberg, P. Swart, and D. S Chult. Exploring network structure, dynamics, and function using networkx. Technical Report LA-UR-08-05495; LA-UR-08-5495, Los Alamos National Lab. (LANL), Los Alamos, NM (United States), Jan. 2008. URL `https://www.osti.gov/biblio/960616`.

[64] E. Hairer, G. Wanner, and E. Hairer. *Stiff and differential-algebraic problems.* Number E. Hairer; G. Wanner ; 2 in Solving ordinary differential equations. Springer, Berlin, 2., rev. ed., corrected 2. printing edition, 2002. ISBN 978-3-642-05221-7 978-3-540-60452-5 978-3-642-05220-0. OCLC: 249300232.

[65] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. ISSN 1476-4687. doi:10.1038/s41586-020-2649-2. URL `https://www.nature.com/articles/s41586-020-2649-2`. Number: 7825 Publisher: Nature Publishing Group.

[66] N. J. Higham. *Accuracy and stability of numerical algorithms.* Society for Industrial and Applied Mathematics, Philadelphia, 2nd ed edition, 2002. ISBN 978-0-89871-521-7.

[67] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, 31(3):363–396, 2005. doi:10.1145/1089014.1089020. Publisher: ACM.

[68] Y. Hold-Geoffroy, O. Gagnon, and M. Parizeau. Once you SCOOP, no need to fork. In *Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment*, XSEDE '14, pages 1–8, New York, NY, USA, July 2014. Association for Computing Machinery. ISBN 978-1-4503-2893-7. doi:10.1145/2616498.2616565. URL `https://doi.org/10.1145/2616498.2616565`.

[69] M. B. Holloway. *Dynamic pipe network computer model.* PhD thesis, Pullman, Washington State Univ., Ph.D.Thesis, 1986.

[70] H. P. Hong, E. N. Allouche, and M. Trivedi. Optimal Scheduling of Replacement and Rehabilitation of Water Distribution Systems. *Journal of Infrastructure Systems*, 12(3):184–191, Sept. 2006. ISSN 1076-0342. doi:10.1061/(ASCE)1076-0342(2006)12:3(184). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%291076-0342%282006%2912%3A3%28184%29`. Publisher: American Society of Civil Engineers.

[71] S. Hoyer and J. Hamman. xarray: N-D labeled Arrays and Datasets in Python. *Journal of Open Research Software*, 5(1):10, Apr. 2017. ISSN 2049-9647. doi:10.5334/jors.148. URL `http://openresearchsoftware.metajnl.com/article/10.5334/jors.148/`. Number: 1 Publisher: Ubiquity Press.

[72] Y. Huang, F. Zheng, H.-F. Duan, and Q. Zhang. Multi-Objective Optimal Design of Water Distribution Networks Accounting for Transient Impacts. *Water Resour Manage*, 34(4):1517–1534, Mar. 2020. ISSN 1573-1650. doi:10.1007/s11269-020-02517-4. URL `https://doi.org/10.1007/s11269-020-02517-4`.

[73] P. J. Huber. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, Mar. 1964. ISSN 0003-4851, 2168-8990. doi:10.1214/aoms/1177703732. URL `https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-35/issue-1/Robust-Estimation-of-a-Location-Parameter/10.1214/aoms/1177703732.full`. Publisher: Institute of Mathematical Statistics.

[74] J. D. Hunter. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, 9(3):90–95, May 2007. ISSN 1558-366X. doi:10.1109/MCSE.2007.55. Conference Name: Computing in Science & Engineering.

[75] IEC. International Electrotechnical Vocabulary, 2023. URL `https://www.electropedia.org/`.

[76] A. Ilchmann and T. Reis, editors. *Surveys in Differential-Algebraic Equations II.* Differential-Algebraic Equations Forum. Springer International Publishing, Cham, 2015. ISBN 978-3-319-11049-3 978-3-319-11050-9. doi:10.1007/978-3-319-11050-9. URL `https://link.springer.com/10.1007/978-3-319-11050-9`.

[77] P. Ingeduld, A. Pradhan, Z. Svitak, and A. Terrai. Modelling Intermittent Water Supply Systems with EPANET. In *Water Distribution Systems Analysis Symposium 2006*, pages 1–8, Cincinnati,

Ohio, United States, Mar. 2008. American Society of Civil Engineers. ISBN 978-0-7844-0941-1. doi:10.1061/40941(247)37. URL http://ascelibrary.org/doi/abs/10.1061/40941%28247%2937.

[78] Itron. Static Water Meter INTELIS, 2021. URL https://www.itron.com/-/media/feature/products/documents/brochure/intelis-water-static-meter-en-web.pdf.

[79] M. Ivetic. Forensic transient analyses of two pipeline failures. *Urban Water Journal*, 1(2): 85–95, June 2004. ISSN 1573-062X, 1744-9006. doi:10.1080/15730620412331289986. URL http://www.tandfonline.com/doi/abs/10.1080/15730620412331289986.

[80] M. Jarraud and I. Bokova. *International Glossary of Hydrology.* World Meteorological Organization (WMO) & United Nations Educational, Scientific and Cultural Organization (UNESCO), 2012. ISBN 978-92-63-03385-8 & 978-92-3-001154-3.

[81] E. Jaumouillé, O. Piller, and J. E. Van Zyl. A hydraulic model for water distribution systems incorporating both inertia and leakage. In *Water Management Challenges in Global Change (CCWI2007 and SUWM2007 Conference), Leicester, GBR*, pages 3–5, 2007.

[82] R. D. Jenks and R. S. Sutor. *AXIOM — The Scientific Computation System.* Springer, 2013.

[83] B. S. Jung, B. W. Karney, P. F. Boulos, and D. J. Wood. The need for comprehensive transient analysis of distribution systems. *Journal AWWA*, 99(1):112–123, 2007. ISSN 1551-8833. doi:10.1002/j.1551-8833.2007.tb07851.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1551-8833.2007.tb07851.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/j.1551-8833.2007.tb07851.x.

[84] Z. Kapelan. *Calibration of water distribution system hydraulic models.* Ph.D., University of Exeter, 2002. URL https://ethos.bl.uk/OrderDetails.do;jsessionid=8A493F1042EC733E453563232FC4D14B?uin=uk.bl.ethos.370011.

[85] Z. Kapelan, D. Savic, and G. Walters. Incorporation of prior information on parameters in inverse transient analysis for leak detection and roughness calibration. *Urban Water Journal*, 1 (2):129–143, June 2004. ISSN 1573-062X, 1744-9006. doi:10.1080/15730620412331290029. URL http://www.tandfonline.com/doi/abs/10.1080/15730620412331290029.

[86] I. E. Karadirek, S. Kara, G. Yilmaz, A. Muhammetoglu, and H. Muhammetoglu. Implementation of Hydraulic Modelling for Water-Loss Reduction Through Pressure Management. *Water Resour Manage*, 26(9):2555–2568, July 2012. ISSN 1573-1650. doi:10.1007/s11269-012-0032-2. URL https://doi.org/10.1007/s11269-012-0032-2.

[87] B. W. Karney and D. McInnis. Efficient Calculation of Transient Flow in Simple Pipe Networks. *Journal of Hydraulic Engineering*, 118(7):1014–1030, July 1992. ISSN 0733-9429, 1943-7900. doi:10.1061/(ASCE)0733-9429(1992)118:7(1014). URL http://ascelibrary.org/doi/10.1061/%28ASCE%290733-9429%281992%29118%3A7%281014%29.

[88] K. A. Klisel, R. Murray, and T. Haxton. AN OVERVIEW OF THE WATER NETWORK TOOL FOR RESILIENCE (WNTR). In *1st International WDSA / CCWI2018 Joint Conference*, Kingston, Ontario, Canada, 2018.

[89] M. M. Koşucu and M. C. Demirel. Smart pressure management extension for EPANET: source code enhancement with a dynamic pressure reducing valve model. *Journal of Hydroinformatics*, 24(3):642–658, Mar. 2022. ISSN 1464-7141. doi:10.2166/hydro.2022.172. URL `https://doi.org/10.2166/hydro.2022.172`.

[90] M. Kuznetsov, B. Kaldybaeva, S. Tsibulskiy, and S. Boldyryev. Optimisation of heat distribution in Tomsk City by inter-cluster connections model implemented in P-graph environment. In *2021 6th International Conference on Smart and Sustainable Technologies (SpliTech)*, pages 01–06, Sept. 2021. doi:10.23919/SpliTech52315.2021.9566473.

[91] A. Lambert. What do we know about pressure-leakage relationships in distribution systems? In *IWA Conference 'System Approach to Leakage Control and Water Distribution Systems Management'*, page 8, 2001.

[92] S. Lao, S. Portela, J. Dequesne, and O. Debuf. National report of SISPEA data. Technical Report Edition of Jan. 2022 from data of 2020, French national observatory of water and sanitation services, 2022. URL `https://www.services.eaufrance.fr/docs/synthese/rapports/Rapport_Sispea_2020_VF.pdf`.

[93] C. Larman and V. Basili. Iterative and incremental developments. a brief history. *Computer*, 36(6):47–56, June 2003. ISSN 1558-0814. doi:10.1109/MC.2003.1204375. Conference Name: Computer.

[94] D. Laucelli, L. Berardi, O. Giustolisi, L. S. Vamvakeridou-Lyroudia, Z. Kapelan, D. Savic, and G. Barbaroand. Calibration of Water Distribution System Using Topological Analysis. In *Water Distribution Systems Analysis 2010*, pages 1664–1681, Tucson, Arizona, United States, Dec. 2011. American Society of Civil Engineers. ISBN 978-0-7844-1203-9. doi:10.1061/41203(425)147. URL `http://ascelibrary.org/doi/10.1061/41203%28425%29147`.

[95] D. Laucelli, L. Berardi, and O. Giustolisi. WDNetXL: Efficient Research Transfer For Management, Planning And Design Of Water Distribution Networks. In *11th International Conference on Hydroinformatics*, page 9, 2014.

[96] D. Laucelli, A. Simone, L. Berardi, and O. Giustolisi. Optimal Design of District Metering Areas for the Reduction of Leakages. *Journal of Water Resources Planning and Management*, 143(6):04017017, June 2017. doi:10.1061/(ASCE)WR.1943-5452.0000768. URL `https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000768`.

[97] D. B. Laucelli, L. Berardi, O. Giustolisi, and C. Andrei. Reducing background leakages and energy consumption in a real WDN by optimal DMA design. In *2017 International Conference on ENERGY and ENVIRONMENT (CIEM)*, pages 241–245, Oct. 2017. doi:10.1109/CIEM.2017.8120768.

[98] B. Lehner, P. Döll, J. Alcamo, T. Henrichs, and F. Kaspar. Estimating the Impact of Global Change on Flood and Drought Risks in Europe: A Continental, Integrated Analysis. *Climatic Change*, 75(3):273–299, Apr. 2006. ISSN 1573-1480. doi:10.1007/s10584-006-6338-4. URL `https://doi.org/10.1007/s10584-006-6338-4`.

[99] M. Lutz. *Learning Python*. O'Reilly, Beijing, 5 edition, 2013. ISBN 978-1-4493-5573-9. URL `https://www.safaribooksonline.com/library/view/learning-python-5th/9781449355722/`.

[100] A. Mailhot, A. Poulin, and J.-P. Villeneuve. Optimal replacement of water pipes. *Water Resources Research*, 39(5), 2003. ISSN 1944-7973. doi:10.1029/2002WR001904. URL `https://onlinelibrary.wiley.com/doi/abs/10.1029/2002WR001904`. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1029/2002WR001904.

[101] M. Maskit and A. Ostfeld. Leakage Calibration of Water Distribution Networks. In *Procedia Engineering*, volume 89 of *16th Water Distribution System Analysis Conference, WDSA2014*, pages 664–671, Jan. 2014. doi:10.1016/j.proeng.2014.11.492. URL `https://www.sciencedirect.com/science/article/pii/S1877705814026071`.

[102] J. May. Pressure dependent leakage. *World water and environmental engineering*, 17(8):10, 1994.

[103] W. McKinney. pandas: a foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, pages 1–9, 2011.

[104] M. M. Mekonnen and A. Y. Hoekstra. Four billion people facing severe water scarcity. *Science Advances*, 2(2):e1500323, Feb. 2016. doi:10.1126/sciadv.1500323. URL `https://www.science.org/doi/10.1126/sciadv.1500323`. Publisher: American Association for the Advancement of Science.

[105] R. Moasheri, M. J. Ghazizadeh, and M. Tashayoei. Leakage detection in water networks by a calibration method. *Flow Measurement and Instrumentation*, 80:101995, Aug. 2021. ISSN 0955-5986. doi:10.1016/j.flowmeasinst.2021.101995. URL `https://www.sciencedirect.com/science/article/pii/S0955598621001035`.

[106] H. Monsef, M. Naghashzadegan, R. Farmani, and A. Jamali. Pressure management in water distribution systems in order to reduce energy consumption and background leakage. *Journal of Water Supply: Research and Technology-Aqua*, 67(4):397–403, June 2018. ISSN 0003-7214. doi:10.2166/aqua.2018.002. URL `/aqua/article/67/4/397/38998/Pressure-management-in-water-distribution-systems`.

[107] A. F. Morosini, P. Veltri, F. Costanzo, and D. Savić. Identification of Leakages by Calibration of WDS Models. In *Procedia Engineering*, volume 70 of *12th International Conference on Computing and Control for the Water Industry, CCWI2013*, pages 660–667, Jan. 2014. doi:10.1016/j.proeng.2014.02.072. URL `https://www.sciencedirect.com/science/article/pii/S1877705814000745`.

[108] A. Nafi and Y. Kleiner. Scheduling Renewal of Water Pipes While Considering Adjacency of Infrastructure Works and Economies of Scale. *Journal of Water Resources Planning and Management*, 136(5):519–530, Sept. 2010. ISSN 0733-9496. doi:10.1061/(ASCE)WR.1943-5452.0000062. URL `https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000062`. Publisher: American Society of Civil Engineers.

[109] J. A. Nathanson. Water supply system - Encyclopedia Britannica, Apr. 2023. URL `https://www.britannica.com/technology/water-supply-system`.

[110] U. Nations. World Population Dashboard, 2023. URL https://www.unfpa.org/data/world-population-dashboard.

[111] J. D. Nault. *Comprehensive Simulation of One-Dimensional Unsteady Pipe Network Hydraulics: Improved Formulations and Adaptive Hybrid Modeling*. Thesis, University of Toronto, Nov. 2017. URL https://tspace.library.utoronto.ca/handle/1807/80861.

[112] J. D. Nault and B. W. Karney. Improved Rigid Water Column Formulation for Simulating Slow Transients and Controlled Operations. *J. Hydraul. Eng.*, 142(9):04016025, Sept. 2016. ISSN 0733-9429, 1943-7900. doi:10.1061/(ASCE)HY.1943-7900.0001145. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29HY.1943-7900.0001145.

[113] F. Nejjari, R. Sarrate, and J. Blesa. Optimal Pressure Sensor Placement in Water Distribution Networks Minimizing Leak Location Uncertainty. *Procedia Engineering*, 119:953–962, Jan. 2015. ISSN 1877-7058. doi:10.1016/j.proeng.2015.08.979. URL https://www.sciencedirect.com/science/article/pii/S1877705815026491.

[114] D. Nerantzis and I. Stoianov. Optimization-Based Selection of Hydrants and Valves Control in Water Distribution Networks for Fire Incidents Management. *IEEE Systems Journal*, 17(1): 134–145, Mar. 2023. ISSN 1937-9234. doi:10.1109/JSYST.2022.3159764. Conference Name: IEEE Systems Journal.

[115] I. Newton, N. W. L. o. S. I. N. Chittenden, D. Adee, A. Motte, and T. P. E. A. m. b. C.-B. Hill. *Newton's Principia : the mathematical principles of natural philosophy*. New-York : Published by Daniel Adee, 1846. URL http://archive.org/details/newtonspmathema00newtrich.

[116] M. Nicolini and L. Falcomer. Genetic Algorithm for Calibration and Leakage Identification in Water Distribution System. In *2020 3rd IEEE International Conference on Knowledge Innovation and Invention (ICKII)*, pages 273–276, Aug. 2020. doi:10.1109/ICKII50300.2020.9318899.

[117] M. Nicolini, C. Giacomello, and K. Deb. Calibration and Optimal Leakage Management for a Real Water Distribution Network. *J. Water Resour. Plann. Manage.*, 137(1):134–142, Jan. 2011. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0000087. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000087.

[118] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer series in operations research. Springer, New York, 1999. ISBN 978-0-387-98793-4.

[119] K. Onizuka. System Dynamics Approach to Pipe Network Analysis. *Journal of Hydraulic Engineering*, 112(8):728–749, Aug. 1986. ISSN 0733-9429, 1943-7900. doi:10.1061/(ASCE)0733-9429(1986)112:8(728). URL http://ascelibrary.org/doi/10.1061/%28ASCE%290733-9429%281986%29112%3A8%28728%29.

[120] A. Ostfeld, E. Salomons, L. Ormsbee, J. G. Uber, C. M. Bros, P. Kalungi, R. Burd, B. Zazula-Coetzee, T. Belrain, D. Kang, K. Lansey, H. Shen, E. McBean, Z. Yi Wu, T. Walski, S. Alvisi, M. Franchini, J. P. Johnson, S. R. Ghimire, B. D. Barkdoll, T. Koppel, A. Vassiljev, H. Kim Joong, G. Chung, G. Yoo Do, K. Diao, Y. Zhou, J. Li, Z. Liu, K. Chang, J. Gao, S. Qu, Y. Yuan, T. D. Prasad, D. Laucelli, L. S. Vamvakeridou Lyroudia, Z. Kapelan,

D. Savic, L. Berardi, G. Barbaro, O. Giustolisi, M. Asadzadeh, B. A. Tolson, and R. McKillop. Battle of the Water Calibration Networks. *Journal of Water Resources Planning and Management*, 138(5):523–532, Sept. 2012. doi:10.1061/(ASCE)WR.1943-5452.0000191. URL `https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000191`.

[121] R. Perez, G. Sanz, V. Puig, J. Quevedo, M. A. Cuguero Escofet, F. Nejjari, J. Meseguer, G. Cembrano, J. M. Mirats Tur, and R. Sarrate. Leak Localization in Water Networks: A Model-Based Methodology Using Pressure Sensors Applied to a Real Network in Barcelona [Applications of Control]. *IEEE Control Systems Magazine*, 34(4):24–36, Aug. 2014. ISSN 1941-000X. doi:10.1109/MCS.2014.2320336. Conference Name: IEEE Control Systems Magazine.

[122] O. Piller. *Modeling the behavior of a network - Hydraulic analysis and a sampling procedure for estimating the parameters.* PhD Thesis in Applied Mathematics, University of Bordeaux, France, 1995. URL `http://www.theses.fr/1995BOR10511`.

[123] O. Piller. A unified framework for pressure driven network analysis. In *Water Management Challenges in Global Change (CCWI2007 and SUWM2007 Conference), Leicester, GBR, 3-5 September 2007, Sep 2007, Leicester, United Kingdom. pp.25-30*, page 7, 2007.

[124] O. Piller. *Water distribution system modeling and optimization.* French habilitation, Ecole doctorale Sciences des métiers de l'ingénieur, ED 432, Paris, France, 2019. URL `http://rgdoi.net/10.13140/RG.2.2.23972.53122`.

[125] O. Piller and B. Brémond. A stochastic model for peak period analysis of pipe networks. *ASCE Environmental & Water Resources Systems Analysis (EWRSA)*, 2002.

[126] O. Piller and M. Propato. Slow Transient Pressure Driven Modeling in Water Distribution Networks. In *Water Distribution Systems Analysis Symposium 2006*, pages 1–13, Cincinnati, Ohio, United States, 2008. American Society of Civil Engineers. ISBN 978-0-7844-0941-1. doi:10.1061/40941(247)38. URL `http://ascelibrary.org/doi/abs/10.1061/40941%28247%2938`.

[127] O. Piller and J. E. van Zyl. Modeling Control Valves in Water Distribution Systems Using a Continuous State Formulation. *Journal of Hydraulic Engineering*, 140(11):04014052, Nov. 2014. doi:10.1061/(ASCE)HY.1943-7900.0000920. URL `https://ascelibrary.org/doi/abs/10.1061/(ASCE)HY.1943-7900.0000920`.

[128] O. Piller and J. E. V. Zyl. Pressure-driven analysis of network sections supplied via high-lying nodes. *Integrating Water Systems: Proceedings of the Tenth International Conference on Computing and Control in the Water Industry*, page 6, 2010.

[129] O. Piller, B. Brémond, and P. Eisenbeis. Water supply network modelling including head-dependent demand and elevated nodes. In *IWA specialised conference: system approach to leakage control and water distribution systems management*, volume 1/1, page 72, May 2001. URL `https://hal.inrae.fr/hal-02579643`.

[130] O. Piller, B. Bremond, and M. Poulton. Least Action Principles Appropriate to Pressure Driven Models of Pipe Networks. In *World Water &amp; Environmental Resources Congress 2003*, pages 1–15, Philadelphia, Pennsylvania, United States, June 2003. American Society of Civil

Engineers. ISBN 978-0-7844-0685-4. doi:10.1061/40685(2003)113. URL http://ascelibrary.org/doi/abs/10.1061/40685%282003%29113.

[131] O. Piller, D. Gilbert, K. Haddane, and S. Sabatié. Porteau: An Object-Oriented programming hydraulic toolkit for water distribution system analysis. In *Eleventh International Conference on Computing and Control for the Water Industry (CCWI 2011)*, volume 1/3, pages p. 27 – p. 32, Exeter, United Kingdom, Sept. 2011. Centre for Water Systems, University of Exeter. URL https://hal.archives-ouvertes.fr/hal-00653763.

[132] O. Piller, F. Sedehizade, T. Bernard, M. Braun, N. Cheifetz, J. Deuerlein, A. Korth, E. Lapébie, I. Trick, J. Weber, and C. Werey. ResiWater: A Franco-German Project for Augmented Resilience of Water Distribution Systems following Severe Abnormal Events. In *14th CCWI international conference, Computing and Control in Water Industry*, page 7 p., Amsterdam, Netherlands, Nov. 2016. URL https://hal.archives-ouvertes.fr/hal-01555788.

[133] O. Piller, S. Elhay, J. Deuerlein, and A. Simpson. Why are Line Search Methods Needed for Hydraulic DDM and PDM Solvers? In *Computing and Control for the Water Industry (CCWI) 2017*, page 327189 Bytes. The University of Sheffield, 2017. doi:10.15131/SHEF.DATA.5364229.V1. URL https://figshare.shef.ac.uk/articles/journal_contribution/CCWI2017_F41_Why_are_Line_Search_Methods_Needed_for_Hydraulic_DDM_and_PDM_Solvers_/5364229/1. Artwork Size: 327189 Bytes.

[134] O. Piller, F. Sedehizade, T. Bernard, M. Braun, N. Cheifetz, J. Deuerlein, M. Wagner, E. Lapébie, I. Trick, J.-M. Weber, and C. Werey. Augmented Resilience of Water Distribution Systems following Severe Abnormal Events. In *CCWI2017*, page 373528 Bytes, 2017. doi:10.15131/SHEF.DATA.5363509.V1. URL https://figshare.shef.ac.uk/articles/journal_contribution/CCWI2017_F96_Augmented_Resilience_of_Water_Distribution_Systems_following_Severe_Abnormal_Events_/5363509/1. Artwork Size: 373528 Bytes Publisher: The University of Sheffield.

[135] O. Piller, S. Elhay, J. W. Deuerlein, and A. R. Simpson. A Content-Based Active-Set Method for Pressure-Dependent Models of Water Distribution Systems with Flow Controls. *Journal of Water Resources Planning and Management*, 146(4):04020009, Apr. 2020. ISSN 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0001160. URL https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29WR.1943-5452.0001160. Publisher: American Society of Civil Engineers.

[136] R. Puust, Z. Kapelan, D. A. Savic, and T. Koppel. A review of methods for leakage management in pipe networks. *Urban Water Journal*, 7(1):25–45, Feb. 2010. ISSN 1573-062X. doi:10.1080/15730621003610878. URL https://doi.org/10.1080/15730621003610878.

[137] H. S. Rao and D. W. Bree. Extended Period Simulation of Water Systems—Part A. *Journal of the Hydraulics Division*, 103(2):97–108, Feb. 1977. doi:10.1061/JYCEAJ.0004711. URL https://ascelibrary.org/doi/10.1061/JYCEAJ.0004711. Publisher: American Society of Civil Engineers.

[138] H. S. Rao, L. C. Markel, and D. W. Bree. Extended Period Simulation of Water Systems—Part B. *Journal of the Hydraulics Division*, 103(3):281–294, Mar. 1977. doi:10.1061/JYCEAJ.0004716. URL https://ascelibrary.org/doi/10.1061/JYCEAJ.0004716. Publisher: American Society of Civil Engineers.

[139] H. Ritchie and M. Roser. Water Use and Stress. *Our World in Data*, 2017. URL https://ourworldindata.org/water-use-stress.

[140] L. A. Rossman. EPANET 2 USERS MANUAL. *U.S. Environmental Protection Agency, Washington, D.C., EPA/600/R-00/057*, 2000.

[141] L. A. Rossman, H. Woo, M. Tryby, F. Shang, R. Janke, and T. Haxton. EPANET 2.2 User Manual. *U.S. Environmental Protection Agency, Washington, DC, EPA/600/R-20/133*, 2020.

[142] Sanz Gerard, Pérez Ramon, Kapelan Zoran, and Savic Dragan. Leak Detection and Localization through Demand Components Calibration. *Journal of Water Resources Planning and Management*, 142(2):04015057, Feb. 2016. doi:10.1061/(ASCE)WR.1943-5452.0000592. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000592.

[143] D. A. Savic, Z. S. Kapelan, and P. M. Jonkergouw. Quo vadis water distribution model calibration? *Urban Water Journal*, 6(1):3–22, Mar. 2009. ISSN 1573-062X, 1744-9006. doi:10.1080/15730620802613380. URL http://www.tandfonline.com/doi/abs/10.1080/15730620802613380.

[144] M. Shimada. Graph-Theoretical Model for Slow Transient Analysis of Pipe Networks. *Journal of Hydraulic Engineering*, 115(9):1165–1183, Sept. 1989. ISSN 0733-9429, 1943-7900. doi:10.1061/(ASCE)0733-9429(1989)115:9(1165). URL http://ascelibrary.org/doi/10.1061/%28ASCE%290733-9429%281989%29115%3A9%281165%29.

[145] A. Shirzad, M. Tabesh, R. Farmani, and M. Mohammadi. Pressure-Discharge Relations with Application to Head-Driven Simulation of Water Distribution Networks. *J. Water Resour. Plann. Manage.*, 139(6):660–670, Nov. 2013. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0000305. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0000305.

[146] S. Sophocleous, D. A. Savić, Z. Kapelan, and O. Giustolisi. A Two-stage Calibration for Detection of Leakage Hotspots in a Real Water Distribution Network. In *Procedia Engineering*, volume 186 of *XVIII International Conference on Water Distribution Systems, WDSA2016*, pages 168–176, Jan. 2017. doi:10.1016/j.proeng.2017.03.223. URL http://www.sciencedirect.com/science/article/pii/S1877705817313723.

[147] D. Steffelbauer and D. Fuchs-Hanusch. OOPNET: An object-oriented EPANET in Python. *Procedia Engineering*, 119:710–718, 2015. ISSN 18777058. doi:10.1016/j.proeng.2015.08.924. URL https://linkinghub.elsevier.com/retrieve/pii/S1877705815025941.

[148] D. Steffelbauer, O. Piller, C. Chambon, and E. Abraham. Towards a novel multi-purpose simulation software of water distribution systems in Python. In *14th International Conference on Hydroinformatics, Water INFLUENCE Water INFormatic soLUtions and opEN problems in the cycle from Clouds to ocEan*, page 4, Bucharest, Romania, July 2022.

[149] D. B. Steffelbauer, J. Deuerlein, D. Gilbert, E. Abraham, and O. Piller. Pressure-Leak Duality for Leak Detection and Localization in Water Distribution Systems. *J. Water Resour. Plann. Manage.*, 148(3):04021106, Mar. 2022. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)WR.1943-5452.0001515. URL https://ascelibrary.org/doi/10.1061/%28ASCE%29WR.1943-5452.0001515.

[150] E. Süli and D. F. Mayers. *An introduction to numerical analysis.* Cambridge University Press, Cambridge ; New York, 2003. ISBN 978-0-521-81026-5 978-0-521-00794-8. OCLC: ocm50525488.

[151] M. Tabesh, M. Jamasb, and R. Moeini. Calibration of water distribution hydraulic models: A comparison between pressure dependent and demand driven analyses. *Urban Water Journal*, 8 (2):93–102, Apr. 2011. ISSN 1573-062X, 1744-9006. doi:10.1080/1573062X.2010.548525. URL `http://www.tandfonline.com/doi/abs/10.1080/1573062X.2010.548525`.

[152] T. Tanyimboh and A. Templeman. A New Nodal Outflow Function for Water Distribution Networks. In *The Fourth International Conference on Engineering Computational Technology*, page 64, Lisbon, Portugal, 2004. doi:10.4203/ccp.80.64. URL `http://www.ctresources.info/ccp/paper.html?id=3699`.

[153] P. Tchebichef. Sur l'intégration des différentielles irrationnelles. *Journal de mathématiques pures et appliquées*, 18:87–111, 1853.

[154] W. Tinney and W. Meyer. Solution of large sparse systems by ordered triangular factorization | IEEE Journals & Magazine | IEEE Xplore. *IEEE TACON*, 18(4):333 – 346, 1973. ISSN 0018-9286, 1558-2523. doi:10.1109/TAC.1973.1100352. URL `https://ieeexplore.ieee.org/abstract/document/1100352`.

[155] E. Todini. Extending the global gradient algorithm to unsteady flow extended period simulations of water distribution systems. *Journal of Hydroinformatics*, 13(2):167–180, Mar. 2011. ISSN 1464-7141. doi:10.2166/hydro.2010.164. URL `/jh/article/13/2/167/31084/Extending-the-global-gradient-algorithm-to`.

[156] N. Tonnet. *Hydraulic model driven by slow transient phenomena.* Project Graduation, MATMECA, 2003.

[157] L. Torres, C. Verde, and J. Rojas. Minimal-order observers for locating leaks in a pipeline with a branch **Funded by CONACyT: Atención a Problemas Nacionales, Convocatoria 2017; and CONACyT-Secretaría de energía, Proyecto 280170: Cero incidentes en la red de ductos de México. *IFAC-PapersOnLine*, 52(23):67–72, Jan. 2019. ISSN 2405-8963. doi:10.1016/j.ifacol.2019.11.011. URL `https://www.sciencedirect.com/science/article/pii/S2405896319309462`.

[158] J. E. van Zyl, J. Borthwick, and A. Hardy. OOTEN: An object-oriented programmers toolkit for Epanet. In *Conference: CCWI 2003 Advances in water supply management*, Imperial College, London, United Kingdom, 2003.

[159] van Zyl J. E. and Cassa A. M. Modeling Elastically Deforming Leaks in Water Distribution Pipes. *Journal of Hydraulic Engineering*, 140(2):182–189, Feb. 2014. doi:10.1061/(ASCE)HY.1943-7900.0000813. URL `https://ascelibrary.org/doi/full/10.1061/%28ASCE%29HY.1943-7900.0000813`.

[160] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, I. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero,

C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, and P. van Mulbregt. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, Mar. 2020. ISSN 1548-7105. doi:10.1038/s41592-019-0686-2. URL `https://www.nature.com/articles/s41592-019-0686-2`. Number: 3 Publisher: Nature Publishing Group.

[161] J. P. Vítkovský, A. R. Simpson, and M. F. Lambert. Leak Detection and Calibration Using Transients and Genetic Algorithms. *Journal of Water Resources Planning and Management*, 126(4):262–265, July 2000. ISSN 0733-9496. doi:10.1061/(ASCE)0733-9496(2000)126:4(262). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%282000%29126%3A4%28262%29`. Publisher: American Society of Civil Engineers.

[162] J. M. Wagner, U. Shamir, and D. H. Marks. Water Distribution Reliability: Simulation Methods. *Journal of Water Resources Planning and Management*, 114(3):276–294, May 1988. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)0733-9496(1988)114:3(276). URL `http://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%281988%29114%3A3%28276%29`.

[163] T. M. Walski. Technique for Calibrating Network Models. *J. Water Resour. Plann. Manage.*, 109(4):360–372, Oct. 1983. ISSN 0733-9496, 1943-5452. doi:10.1061/(ASCE)0733-9496(1983)109:4(360). URL `https://ascelibrary.org/doi/10.1061/%28ASCE%290733-9496%281983%29109%3A4%28360%29`.

[164] T. M. Walski. Model calibration data: the good, the bad, and the useless. *Journal - American Water Works Association*, 92(1):94–99, Jan. 2000. ISSN 0003150X. doi:10.1002/j.1551-8833.2000.tb08791.x. URL `https://onlinelibrary.wiley.com/doi/10.1002/j.1551-8833.2000.tb08791.x`.

[165] T. M. Walski, D. V. Chase, D. Savic, W. Grayman, S. Beckwith, E. Koelle, and I. Haestad Methods. *Advanced water distribution modeling and management.* Bentley Institute Press, Exton, PA, 2007. ISBN 978-1-934493-01-4. OCLC: 154213231.

[166] E. Weisstein. *CRC Concise encyclopedia of mathematics.* Chapman & Hall, Boca Raton, 1999.

[167] G. S. Williams, A. Hazen, and others. *Hydraulic tables: the elements of gagings and the friction of water flowing in pipes, aqueducts, sewers, etc.* J. Wiley & sons, inc.: London, Chapman & Hall, limited,, 1933.

[168] L. Xing and L. Sela. Transient simulations in water distribution networks: TSNet python package. *Advances in Engineering Software*, 149:102884, Nov. 2020. ISSN 09659978. doi:10.1016/j.advengsoft.2020.102884. URL `https://linkinghub.elsevier.com/retrieve/pii/S0965997819309342`.

[169] L. Xing, T. Raviv, and L. Sela. Sensor placement for robust burst identification in water systems: Balancing modeling accuracy, parsimony, and uncertainties. *Advanced Engineering Informatics*, 51:101484, Jan. 2022. ISSN 1474-0346. doi:10.1016/j.aei.2021.101484. URL `https://www.sciencedirect.com/science/article/pii/S1474034621002330`.

[170] S. Yoo and M. Harman. Regression testing minimization, selection and prioritization: a survey. *Software Testing, Verification and Reliability*, 22(2):67–120, 2012. ISSN 1099-1689. doi:10.1002/stvr.430. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/stvr.430`. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/stvr.430.

[171] D. Zaman, A. K. Gupta, V. Uddameri, M. K. Tiwari, and D. Sen. Exploring the key facets of leakage dynamics in water distribution networks: Experimental verification, hydraulic modeling, and sensitivity analysis. *Journal of Cleaner Production*, 362:132236, Aug. 2022. ISSN 0959-6526. doi:10.1016/j.jclepro.2022.132236. URL https://www.sciencedirect.com/science/article/pii/S0959652622018418.

Appendix A

Supplementary material of chapter 1: Pressure-dependent users' consumptions

A.1 Numerical enhancements to deal with sources of instabilities

A.1.1 Regularization of friction head-loss for flow rate close to zero

At each iteration of the Newton's method described at section 1.2.4, and for any pipe k in $\{1, \ldots, n_p\}$, the element $\left[\boldsymbol{J_{11}}\right]_{kk}$ of the Jacobian matrix is equal to the derivative of the friction head-loss $d\xi_{f,k}/dq_k$, which is computed as:

$$\frac{\mathrm{d}\xi_{\mathrm{f},k}}{\mathrm{d}q_k}(q_k) = \gamma_{_{HW}}\, f_k\, \ell_k\, |q_k|^{\gamma_{_{HW}}-1}, \tag{A.1}$$

where $\gamma_{_{HW}} = 1.852$ and f_k are the Hazen-Williams exponent and friction coefficient of k [167], l_k is the length of k, and q_k is the flow rate in k. Conversely to the friction coefficient of the Darcy–Weisbach equation [165, p. 32-33], the f_k used here does not depend on the q.

However, eq. (A.1) gives 0 for $q_k = 0$. Thus, since we need to compute the inverse of $\left[\boldsymbol{J_{11}}\right]_{kk}$ at each iteration (see eqs. (1.20), (1.21) and (1.23)), the use of eq. (A.1) to compute $d\xi_{f,k}/dq_k$ could lead to a division by zero error.

To prevent such error, we choose, as [122], to regularize the friction head-loss

$$\xi_{\mathrm{f},k}(q_k) = f_k\, \ell_k\, q_k\, |q_k|^{\gamma_{_{HW}}-1}. \tag{A.2}$$

and its derivative (eq. (A.1)) for q_k close to 0, by respectively a cubic and a quadratic polynomials.

To do so, we first choose an $\varepsilon_{\xi_f} \in \mathbb{R}_*^+$ close to $0\,\mathrm{l\,s^{-1}}$ (e.g., $\varepsilon_{\xi_f} = 10^{-3}\,\mathrm{l\,s^{-1}}$) that defines the interval $[-\varepsilon_{\xi_f}, \varepsilon_{\xi_f}]$ into which the regularizations have to be done. Next, we look for the polynomial $\mathrm{P}_{\xi_\mathrm{f}}(q) = a_{\xi_f,3}\, q^3 + a_{\xi_f,1}\, q$ and its derivative $\mathrm{dP}_{\xi_\mathrm{f}}/\mathrm{d}q\,(q_k) = 3\, a_{\xi_f,3}\, q^2 + a_{\xi_f,1}$, which coefficients $\{a_{\xi_f,1}, a_{\xi_f,3}\}$ are solutions of the system:

$$\begin{cases} a_{\xi_f,3}\, {\varepsilon_{\xi_f}}^3 & + & a_{\xi_f,1}\, \varepsilon_{\xi_f} & = & \varepsilon_{\xi_f} \left|\varepsilon_{\xi_f}\right|^{\gamma_{_{HW}}-1} & \text{(A.3a)} \\ 3\, a_{\xi_f,3}\, {\varepsilon_{\xi_f}}^2 & + & a_{\xi_f,1} & = & \gamma_{_{HW}} \left|\varepsilon_{\xi_f}\right|^{\gamma_{_{HW}}-1} & \text{(A.3b)} \end{cases}$$

where eq. (A.3a) insures the continuity between $\xi_{\mathrm{f}}(q)$ and $\mathrm{P}_{\xi_{\mathrm{f}}}(q)$ at $q \in \{-\varepsilon_{\xi_f}, \varepsilon_{\xi_f}\}$, and eq. (A.3b) the continuity between their derivative at the same positions. Equation (A.3) needs to be solve only once before any simulation run, and the two coefficients $\{a_{\xi_f,1}, a_{\xi_f,3}\}$ apply to any pipe k, whatever the values of f_k, ℓ_k and q_k.

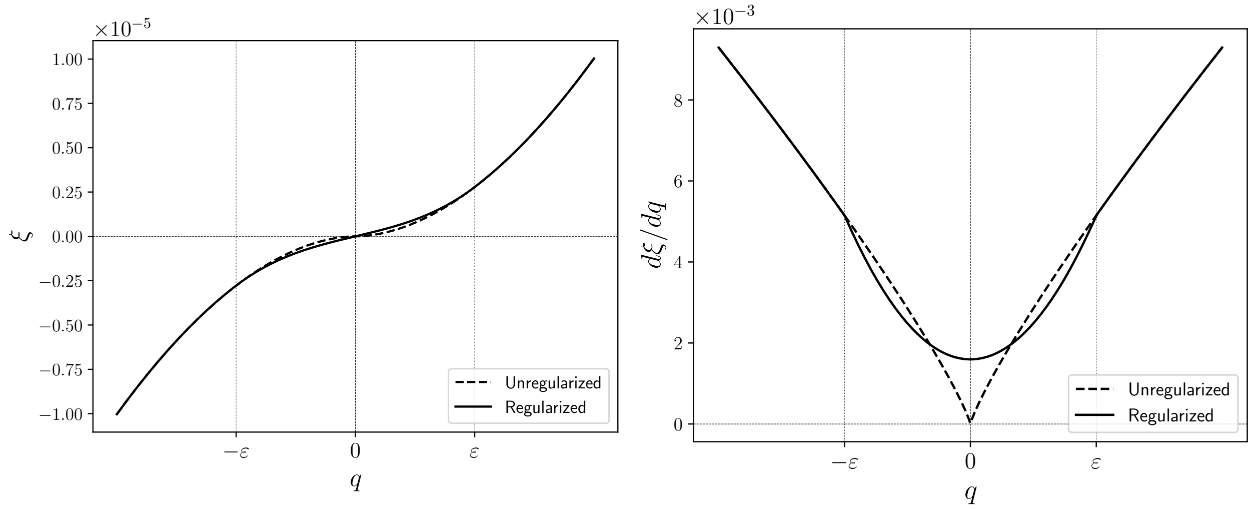
Once we found $\left\{a_{\xi_f,1}, a_{\xi_f,3}\right\}$, we can then compute $\xi_{f,k}$, $\forall k \in \{1, \dots, n_p\}$, as:

$$\xi_{\mathrm{f},k}(q_k) = \begin{cases} f_k\, \ell_k\, \mathrm{P}_{\xi_{\mathrm{f}}}(q_k) & \text{if } |q_k| \leq \varepsilon_{\xi_f} \\ \text{eq. (A.2)} & \text{otherwise,} \end{cases} \tag{A.4}$$

and $d\xi_{f,k}/dq_k$ as:

$$\frac{\mathrm{d}\xi_{\mathrm{f},k}}{\mathrm{d}q_k}(q_k) = \begin{cases} f_k\, \ell_k\, \dfrac{\mathrm{dP}_{\xi_{\mathrm{f}}}}{\mathrm{d}q}(q_k) & \text{if } |q_k| \leq \varepsilon_{\xi_f} \\ \text{eq. (A.1)} & \text{otherwise.} \end{cases} \tag{A.5}$$

Figure A.1 illustrates these regularizations for a pipe of length $\ell = 1\,\mathrm{m}$ and friction coefficient $f = 1\ \mathrm{s}^{1.852}\,\mathrm{l}^{-1.852}$, without any loss of generality. Indeed, eqs. (A.1) and (A.2) and their regularizations depend linearly on ℓ and f. Thus, fig. A.1 can be transposed to any other ℓ an f, by just multiplying the values plotted on the vertical axis by ℓ and f.



(a) $\xi_{\mathrm{f}}(q)$

(b) $\mathrm{d}\xi_{\mathrm{f}}/\mathrm{d}q\ (q)$

Figure A.1: Cubic regularization of the Hazen-Williams friction head-loss function (in $\mathrm{mH_2O}$, fig. A.1a) [167], and quadratic regularization of its derivative (in $\mathrm{mH_2O\,l^{-1}\,s}$, fig. A.1b), for flow rates q close to $0\,\mathrm{l\,s^{-1}}$, as initially proposed by [122]. Here, $\varepsilon = 10^{-3}\,\mathrm{l\,s^{-1}}$, which is also the value of ε used in the source code.

A.1.2 Regularizations of user's consumption for pressure-head close to the minimum or to the service pressure-head

At each iteration of the Newton's method described at section 1.2.4, and $\forall i \in \{1, \dots, n_j\}$, the element $[\boldsymbol{J_{22}}]_{ii}$ of the Jacobian matrix is equal to $-dc_i/dh_i$, where dc_i/dh_i is the derivative of the user's consumption with respect to head. dc_i/dh_i can be computed as:

$$\frac{\mathrm{dc}_i}{\mathrm{d}h_i}\,(h_i) = \Big(\frac{\mathrm{dc}_i}{\mathrm{d}p_i}\frac{\mathrm{dp}_i}{\mathrm{d}h_i}\Big)(h_i)\,, \tag{A.6}$$

where

$$\frac{\mathrm{dp}_i}{\mathrm{d}h_i}\,(h_i) = \begin{cases} 1 & \text{if } h_i \in \mathbb{R}_*^+ \\ 0 & \text{otherwise} \end{cases} \tag{A.7}$$

and

$$\frac{\mathrm{dc}_i}{\mathrm{d}p_i}\,(p_i) = \begin{cases} d_i \dfrac{1}{p_s - p_m}\dfrac{1}{2\sqrt{\mathrm{z}(p_i)}} = \dfrac{1}{2\,(p_s - p_m)}\dfrac{\mathrm{c}_i(p_i)}{\mathrm{z}(p_i)} & \text{if } p_m < p_i < p_s \\ 0 & \text{otherwise,} \end{cases} \tag{A.8}$$

with $\mathrm{z}(p_i)$ defined as:

$$\mathrm{z}(p_i) = \frac{p_i - p_m}{p_s - p_m}. \tag{A.9}$$

But eq. (A.8) is undefined at $p_i = p_m$ and discontinuous at $p_i = p_s$. Thus, using eq. (A.8) to compute dc_i/dp_i could lead to convergence failure when p_i is close to p_m or p_s. To make eq. (A.8) definite and continuous $\forall p_i \in \mathbb{R}$, we choose, as [130], to regularize the function $\mathrm{c_i}(p_i)$, defined as

$$\mathrm{c_i}(p_i) = \begin{cases} 0 & \text{if } p_i \le p_m \\ d_i\sqrt{\mathrm{z}(p_i)} & \text{if } p_m < p_i < p_s \\ d_i & \text{if } p_i \ge p_s, \end{cases} \tag{A.10}$$

and its derivative eq. (A.8) by respectively a cubic and a quadratic polynomial, when p_i is close to p_m and p_s.

To do so, denoting $z = \mathrm{z}(p) =$ eq. (A.9), we first define the demand satisfaction function $\mathrm{d_s}(z) = (\mathrm{c}/d) \circ \mathrm{z}(p)$ as:

$$\mathrm{d_s}(z) = \begin{cases} 0 & \text{if } z \le 0 \\ \sqrt{z} & \text{if } 0 < z < 1 \\ 1 & \text{if } z \ge 1 \end{cases} \tag{A.11}$$

and its derivative $\mathrm{dd_s}/\mathrm{d}z\ (z) = \mathrm{d(c/}d)/\mathrm{dz} \cdot \mathrm{dz/d}p\ (p)$ as:

$$\frac{\mathrm{dd_s}}{\mathrm{d}z}\,(z) = \begin{cases} \dfrac{1}{p_s - p_m}\dfrac{1}{2\sqrt{z}} = \dfrac{1}{2\,(p_s - p_m)}\dfrac{\mathrm{d_s}(z)}{z} & \text{if } 0 < z < 1 \\ 0 & \text{otherwise.} \end{cases} \tag{A.12}$$

Next, we choose an $\varepsilon_c \in \mathbb{R}_*^+$ close to 0 (e.g., $\varepsilon_c = 10^{-3}$) that defines the intervals $[-\varepsilon_c, \varepsilon_c]$ and $[1-\varepsilon_c, 1+\varepsilon_c]$ into which the regularizations have to be done. Then, we look for the polynomial $\mathrm{P_c^m}(z) = a_{c,3}^m\, z^3 + a_{c,2}^m\, z^2 + a_{c,1}^m\, z + a_{c,0}^m$ and its derivative $\mathrm{dP_c^m}/\mathrm{d}z\ (z) = 3\, a_{c,3}^m\, z^2 + 2\, a_{c,2}^m\, z + a_{c,1}^m$, which coefficients $\{a_{c,0}^m, \ldots, a_{c,3}^m\}$ are solutions of the system:

$$\begin{cases} -\,a_{c,3}^m\,{\varepsilon_c}^3 + a_{c,2}^m\,{\varepsilon_c}^2 - a_{c,1}^m\,\varepsilon_c + a_{c,0}^m = 0 & \text{(A.13a)} \\ a_{c,3}^m\,{\varepsilon_c}^3 + a_{c,2}^m\,{\varepsilon_c}^2 + a_{c,1}^m\,\varepsilon_c + a_{c,0}^m = \sqrt{\varepsilon_c} & \text{(A.13b)} \\ 3\,a_{c,3}^m\,{\varepsilon_c}^2 - 2\,a_{c,2}^m\,\varepsilon_c + a_{c,1}^m = 0 & \text{(A.13c)} \\ 3\,a_{c,3}^m\,{\varepsilon_c}^2 + 2\,a_{c,2}^m\,\varepsilon_c + a_{c,1}^m = \dfrac{1}{2\sqrt{\varepsilon_c}}, & \text{(A.13d)} \end{cases}$$

where eqs. (A.13a) and (A.13b) insure the continuity between $\mathrm{d_s}(z)$ and $\mathrm{P_c^m}(z)$ at $z \in \{-\varepsilon_c, \varepsilon_c\}$, and eqs. (A.13c) and (A.13d) the continuity between their derivative at the same positions. Likewise, we look for the polynomial $\mathrm{P_c^s}(z) = a^s_{c,3}\, z^3 + a^s_{c,2}\, z^2 + a^s_{c,1}\, z + a^s_{c,0}$ and its derivative $\mathrm{dP_c^s}/\mathrm{d}z\ (z) = 3\, a^s_{c,3}\, z^2 + 2\, a^s_{c,2}\, z + a^s_{c,1}$, which coefficients $\{a^s_{c,0}, \dots, a^s_{c,3}\}$ are solutions of the system:

$$\begin{cases}
a^s_{c,3}\,(1-\varepsilon_c)^3 + a^s_{c,2}\,(1-\varepsilon_c)^2 + a^s_{c,1}\,(1-\varepsilon_c) + a^s_{c,0} = \sqrt{1-\varepsilon_c} & \text{(A.14a)}\\
a^s_{c,3}\,(1+\varepsilon_c)^3 + a^s_{c,2}\,(1+\varepsilon_c)^2 + a^s_{c,1}\,(1+\varepsilon_c) + a^s_{c,0} = 1 & \text{(A.14b)}\\
3\,a^s_{c,3}\,(1-\varepsilon_c)^2 + 2\,a^s_{c,2}\,(1-\varepsilon_c) + a^s_{c,1} = \dfrac{1}{2\sqrt{1-\varepsilon_c}} & \text{(A.14c)}\\
3\,a^s_{c,3}\,(1+\varepsilon_c)^2 + 2\,a^s_{c,2}\,(1+\varepsilon_c) + a^s_{c,1} = 0, & \text{(A.14d)}
\end{cases}$$

where eqs. (A.14a) and (A.14b) insure the continuity between $\mathrm{d_s}(z)$ and $\mathrm{P_c^s}(z)$ at $z \in \{1-\varepsilon_c, 1+\varepsilon_c\}$, and eqs. (A.14c) and (A.14d) the continuity between their derivative at the same positions. Equations (A.13) and (A.14) need to be solve only once before any simulation run, and the coefficients $\{a^m_{c,0}, \dots, a^m_{c,3}\}$ and $\{a^s_{c,0}, \dots, a^s_{c,3}\}$ apply to any junction i, whatever the values of d_i, p_m, p_s and p_i.

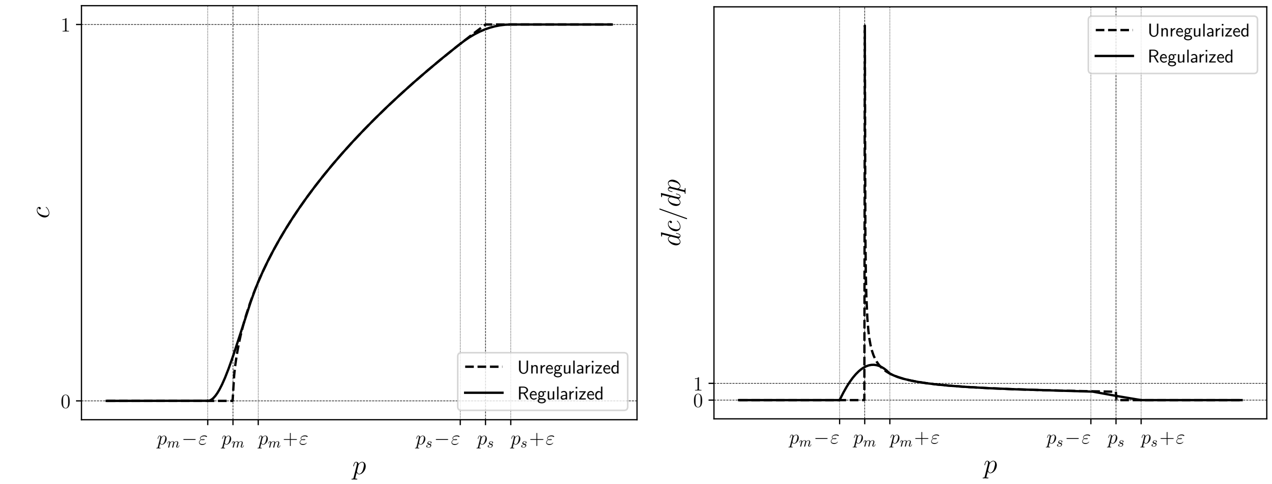
Once we found $\left\{a^m_{c,0}, \dots, a^m_{c,3}\right\}$ and $\left\{a^s_{c,0}, \dots, a^s_{c,3}\right\}$, we can then compute c_i $\forall i \in \{1, \dots, n_j\}$ as:

$$\mathrm{c}_i(p_i) = \begin{cases}
d_i\, \mathrm{P_c^m} \circ \mathrm{z}(p_i) & \text{if} \quad -\varepsilon_c \le \mathrm{z}(p_i) \le \varepsilon_c\\
d_i\, \mathrm{P_c^s} \circ \mathrm{z}(p_i) & \text{if} \quad 1-\varepsilon_c \le \mathrm{z}(p_i) \le 1+\varepsilon_c\\
\text{eq. (A.10)} & \text{otherwise,}
\end{cases} \tag{A.15}$$

and dc_i/dp_i as:

$$\frac{\mathrm{dc}_i}{\mathrm{d}p_i}(p_i) = \begin{cases}
d_i\, \dfrac{1}{p_s - p_m}\, \dfrac{\mathrm{dP_c^m}}{\mathrm{d}z} \circ \mathrm{z}(p_i) & \text{if} \quad -\varepsilon_c \le \mathrm{z}(p_i) \le \varepsilon_c\\
d_i\, \dfrac{1}{p_s - p_m}\, \dfrac{\mathrm{dP_c^s}}{\mathrm{d}z} \circ \mathrm{z}(p_i) & \text{if} \quad 1-\varepsilon_c \le \mathrm{z}(p_i) \le 1+\varepsilon_c\\
\text{eq. (A.8)} & \text{otherwise.}
\end{cases} \tag{A.16}$$

Figure A.2 illustrates these regularizations for a junction node where user's demand $d = 1\,\mathrm{l\,s^{-1}}$, and any minimum and service pressure-head p_m and p_s such that $p_m \ge 0\,\mathrm{mH_2O}$ and $p_s > p_m$, without any loss of generality. Indeed, eq. (A.10), eq. (A.8) and their regularizations depend linearly on d. Thus, fig. A.2 can be transposed to any other d, by just multiplying the values plotted on the vertical axis by d.



(a) c(p)

(b) dc/dp (p)

Figure A.2: Cubic regularizations of the Wagner's consumption function (in $\mathrm{l\,s^{-1}}$, fig. A.2a) [162], and quadratic regularizations of its derivative (in $\mathrm{l\,s^{-1}\,mH_2O^{-1}}$, fig. A.2b), for local pressure-heads p close to p_m and p_s, as initially proposed by [130]. Here, ε is chosen equal to $0.1\,\mathrm{mH_2O}$ for better readability, but in the source code $\varepsilon = 10^{-3}\,\mathrm{mH_2O}$.

A.1.3 Damping of descent directions

The bottleneck of the Newton's method described at section 1.2.4 is the solving, at each iteration, of the system (1.22). Numerous iterations can be needed if the initial guesses of the flow rates and heads at junctions are far from the solutions at equilibrium. In this case, the simulation of large WDNs can become unfeasible in reasonable computational time. Also, the convergence of the method is not guaranteed if the Jacobian of the non-linear system of equations (1.11) does not have a super-linear growth [133]. Thus, to reduce the number of iterations needed by the Newton's method and to guarantee its convergence for any network configuration, [43] proposed a damping method based on a line search approach. To summary, this strategy first looks for the descent directions along which the residuals will be reduced, and then computes a step size that determines how far the iterates can move along that direction. We decide to reuse this damping method in our simulator. Hereafter is a short description.

[43] denoted the diagonal matrix of positive weights $\boldsymbol{W} \in \mathbb{R}^{(n_p+n_j)\times(n_p+n_j)}$, and defined the weighted least squares (WLS) objective scalar function:

$$\psi(\boldsymbol{q},\boldsymbol{h}) = \frac{1}{2}\left\|\boldsymbol{W}^{\frac{1}{2}}\boldsymbol{\rho}(\boldsymbol{q},\boldsymbol{h})\right\|_2^2 = \frac{1}{2}\boldsymbol{\rho}^T\boldsymbol{W}\boldsymbol{\rho}. \tag{A.17}$$

The optimization problem to solve is:

$$\min_{\boldsymbol{q},\boldsymbol{h}} \psi(\boldsymbol{q},\boldsymbol{h}), \tag{A.18}$$

and the damped scheme to solve eq. (A.18) consists, at each iteration m of the Newton's method, in computing:

$$\begin{pmatrix} \boldsymbol{q}^{(m+1)} \\ \boldsymbol{h}^{(m+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{q}^{(m)} \\ \boldsymbol{h}^{(m)} \end{pmatrix} + \sigma^{(m)} \begin{pmatrix} \boldsymbol{\delta_q}^{(m)} \\ \boldsymbol{\delta_h}^{(m)} \end{pmatrix}, \tag{A.19}$$

where $\sigma^{(m)}$ is a well suited step-size.
Remark: another objective function, which involves the inverse of the Hazen-Williams friction head-loss function, is proposed by [43]. But, for genericity matter, and also because we will need it in the next chapter, we choose the objective function defined by eq. (A.17), which can be used even when the inverse function of the friction head-loss does not exist.

To select the correct step-size $\sigma^{(m)}$, [43] used the Goldstein criteria scalar function [58]:

$$\mathrm{c_G}\Big(\psi^{(m)}, \sigma^{(m)}\Big) = \frac{\psi^{(m)} - \psi(\hat{\boldsymbol{q}}^{(m+1)}, \hat{\boldsymbol{h}}^{(m+1)})}{2\,\sigma^{(m)}\psi^{(m)}}, \tag{A.20}$$

where

$$\begin{pmatrix} \hat{\boldsymbol{q}}^{(m+1)} \\ \hat{\boldsymbol{h}}^{(m+1)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{q}^{(m)} \\ \boldsymbol{h}^{(m)} \end{pmatrix} + \sigma^{(m)} \begin{pmatrix} \boldsymbol{\delta_q}^{(m)} \\ \boldsymbol{\delta_h}^{(m)} \end{pmatrix}, \tag{A.21}$$

in an algorithm that divides $\sigma^{(m)}$ by 2 until $\mathrm{c_G}(\psi^{(m)}, \sigma^{(m)}) \geq 0.1$.

Finally, [43] computed the weights $\boldsymbol{W}$ from the maximal values of the demands at junctions $\boldsymbol{d}$ and of the heads at source nodes $\boldsymbol{h_0}$. To do so, they first defined the diagonal matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ to apply respectively on energy (eq. (1.12)) and mass (eq. (1.13)) residuals as:

$$\boldsymbol{M} = \left(\max_{i \in \{1,\dots,n_0\}} h_{f,i}\right)^2 \boldsymbol{I_{n_p}} \quad \text{and} \quad \boldsymbol{N} = \left(\max_{i \in \{1,\dots,n_j\}} d_i\right)^2 \boldsymbol{I_{n_j}}, \tag{A.22}$$

where $\boldsymbol{I_{n_p}}$ and $\boldsymbol{I_{n_j}}$ are respectively the $n_p \times n_p$ and the $n_j \times n_j$ identity matrices. Then, they compute the weights $\boldsymbol{W}$ as:

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{M}^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{N}^{-1} \end{pmatrix}. \tag{A.23}$$

A.1.4 Preconditioning of the Jacobian matrix

At each iteration of the Newton's method, the magnitude of the elements in the Jacobian matrix can vary strongly from one part of the WDN to another, according to the demands, the tank levels, the use of pumps and/or valves, etc. In this case, using the Jacobian matrix $\boldsymbol{J}$ computed as eq. (1.15) can cause instabilities in the Newton's method, and increase significantly the number of iterations needed to converge.

Like [122], [42] proposed a method to deal with zero flow rates in pipes. To do so, they limited, $\forall k \in \{1, \dots, n_p\}$, each singular value λ_k of the sub-block $\boldsymbol{J_{11}}$ of $\boldsymbol{J}$ by a minimum value λ_{min} defined as:

$$\lambda_{min} = \frac{\lambda_{max}}{\kappa_{max}}, \tag{A.24}$$

where λ_{max} is such that

$$\lambda_{max} = \max_k\Big([\boldsymbol{J_{11}}]_{kk}\Big), \tag{A.25}$$

and κ_{max} corresponds to a maximum threshold condition number (e.g., $\kappa_{max} = 10^{10}$). Like so, [42] did not only avoid zero values on the diagonal of $\boldsymbol{J_{11}}$, but also reduced the difference of magnitude order between all elements of $\boldsymbol{J_{11}}$. However, their preconditioning method only applies to the case where consumption is independent of the pressure-head.

Thus, we choose to extend the method proposed by [42], to make it now deal with pressure-dependent consumption. To do so, we recompute λ_{max}, at each iteration of the Newton's method, taking into account the elements on the diagonals of both matrices $\boldsymbol{J_{11}}$ and $\boldsymbol{J_{22}}$, such that:

$$\begin{gathered}\lambda_{max} = \max\Big(\max_k\Big([\boldsymbol{J_{11}}]_{kk}\Big), \max_i\Big(-[\boldsymbol{J_{22}}]_{ii}\Big)\Big),\\ \{k,i\} \in \{1,\ldots,n_p\} \times \{1,\ldots,n_j\}\end{gathered} \tag{A.26}$$

and we recompute $\lambda_{min} = \lambda_{max}/\kappa_{max}$. Finally, we limit:

- each element on the diagonal of $\boldsymbol{J_{11}}$ as:

$$[\boldsymbol{J_{11}}]_{kk} = \max\left([\boldsymbol{J_{11}}]_{kk}, \lambda_{min}\right), \tag{A.27}$$

- and each element on the diagonal of $\boldsymbol{J_{22}}$ as:

$$[\boldsymbol{J_{22}}]_{ii} = \min\left([\boldsymbol{J_{22}}]_{ii}, -\lambda_{min}\right). \tag{A.28}$$

Remark: when using this preconditioning method, the Newton's method described at section 1.2.4 becomes then a quasi-Newton method.

A.2 Discussion on convergence criteria

In iterative algorithms, convergence criteria are generally based on absolute and/or relative differences between the iterates of successive iterations.

Convergence criteria based only on absolute differences are not accurate when the iterates have different magnitude orders, which can occur in WDNs. Conversely, convergence criteria that consider relative differences only can lead to instabilities when the iterates tend toward 0. Thus, the convergence criterion (1.27) that we chose in section 1.2.4.4 is relative when the infinity norm of the new iterates is not too small, and becomes absolute otherwise; we remind it here:

$$\begin{cases}\dfrac{\|\boldsymbol{y}^{(m+1)} - \boldsymbol{y}^{(m)}\|_\infty}{\|\boldsymbol{y}^{(m+1)}\|_\infty} \leq 10^{-6} & \text{if } \|\boldsymbol{y}^{(m+1)}\|_\infty \geq 10^{-6}\\ \|\boldsymbol{y}^{(m+1)} - \boldsymbol{y}^{(m)}\|_\infty \leq 10^{-6} & \text{otherwise,}\end{cases}$$

where $\boldsymbol{y} \in \{\boldsymbol{q}, \boldsymbol{h}\}$.

In the first row of the criterion (1.27), we compare the relative differences between the infinity norms of the iterates. We could also have compared the infinity norms of the relative differences, as:

$$\left\|\frac{\boldsymbol{y}^{(m+1)} - \boldsymbol{y}^{(m)}}{\boldsymbol{y}^{(m+1)}}\right\|_\infty \leq 10^{-6}. \tag{A.29}$$

Criterion (A.29) is commonly used in Runge-Kutta methods [64, p. 120-121]. However, we can see, for $\boldsymbol{y} = \boldsymbol{q}$ and $\forall k \in \{1, \ldots, n_p\}$, that:

$$\frac{\left\|\boldsymbol{q}^{(m+1)} - \boldsymbol{q}^{(m)}\right\|_\infty}{\left\|\boldsymbol{q}^{(m+1)}\right\|_\infty} = \frac{\max\limits_k \left|q_k^{(m+1)} - q_k^{(m)}\right|}{\max\limits_k \left|q_k^{(m+1)}\right|} \leq \max_k \frac{\left|q_k^{(m+1)} - q_k^{(m)}\right|}{\left|q_k^{(m+1)}\right|} = \left\|\frac{\boldsymbol{q}^{(m+1)} - \boldsymbol{q}^{(m)}}{\boldsymbol{q}^{(m+1)}}\right\|_\infty .$$

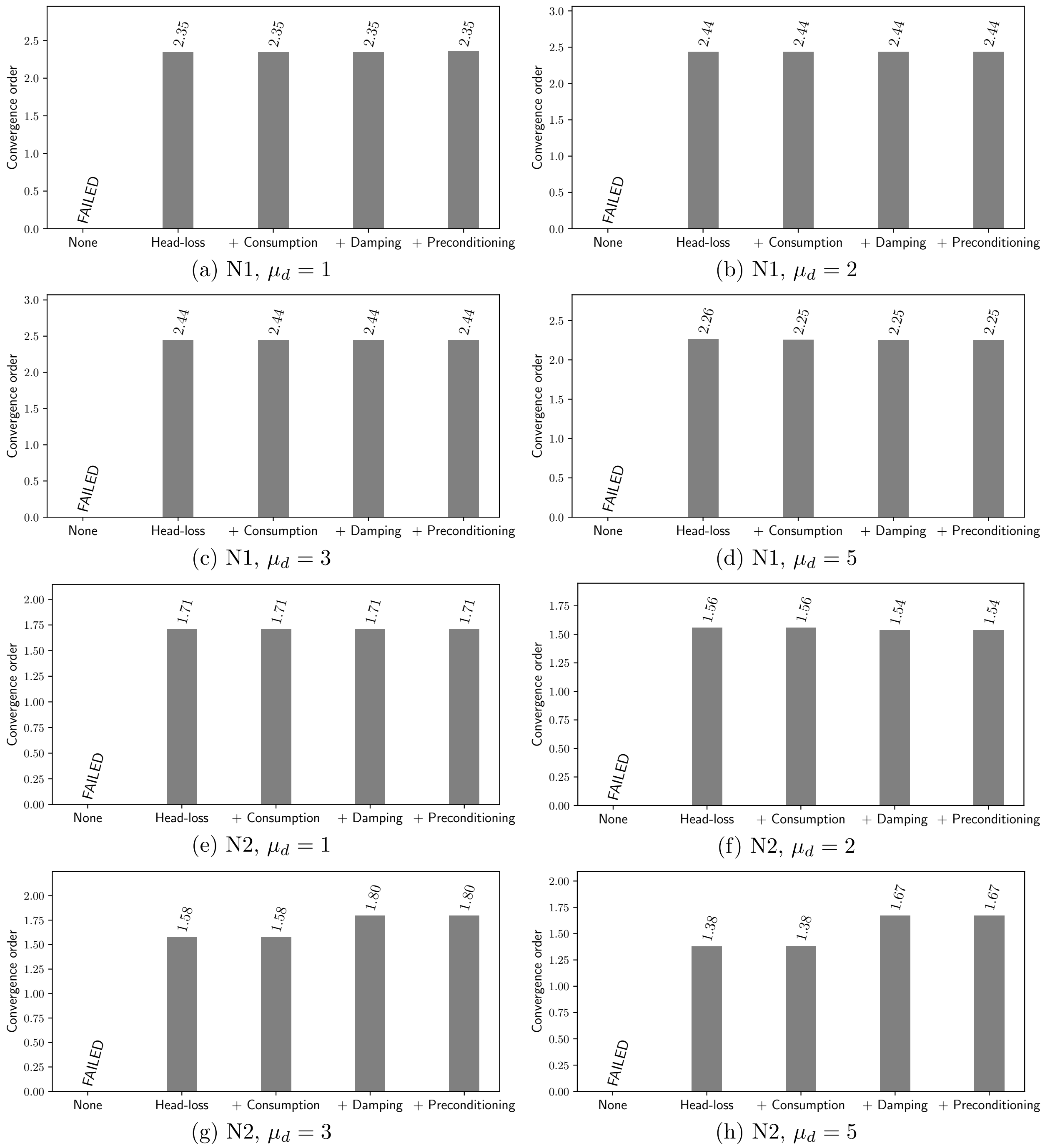
Likewise, for $\boldsymbol{y} = \boldsymbol{h}$ and $\forall i \in \{1, \ldots, n_j\}$, we have:

$$\frac{\left\|\boldsymbol{h}^{(m+1)} - \boldsymbol{h}^{(m)}\right\|_\infty}{\left\|\boldsymbol{h}^{(m+1)}\right\|_\infty} = \frac{\max\limits_i \left|h_i^{(m+1)} - h_i^{(m)}\right|}{\max\limits_i \left|h_i^{(m+1)}\right|} \leq \max_i \frac{\left|h_i^{(m+1)} - h_i^{(m)}\right|}{\left|h_i^{(m+1)}\right|} = \left\|\frac{\boldsymbol{h}^{(m+1)} - \boldsymbol{h}^{(m)}}{\boldsymbol{h}^{(m+1)}}\right\|_\infty .$$

Thus, the criterion (A.29) can need more iterations than (1.27) to be satisfied. This property was also observed by [66, chap. 7]. Since we are not solving stiff differential equations in chapter 1, we decided to choose the quicker criterion (1.27).

A.3 Orders of convergence per network and demand multiplier

Figures A.3 to A.6 present the not-averaged orders of convergence when simulating each network for each demand multiplier, first with no numerical enhancement, and then adding successively each of the numerical enhancements described in appendix A.1.



(a) N1, $\mu_d = 1$

(b) N1, $\mu_d = 2$

(c) N1, $\mu_d = 3$

(d) N1, $\mu_d = 5$

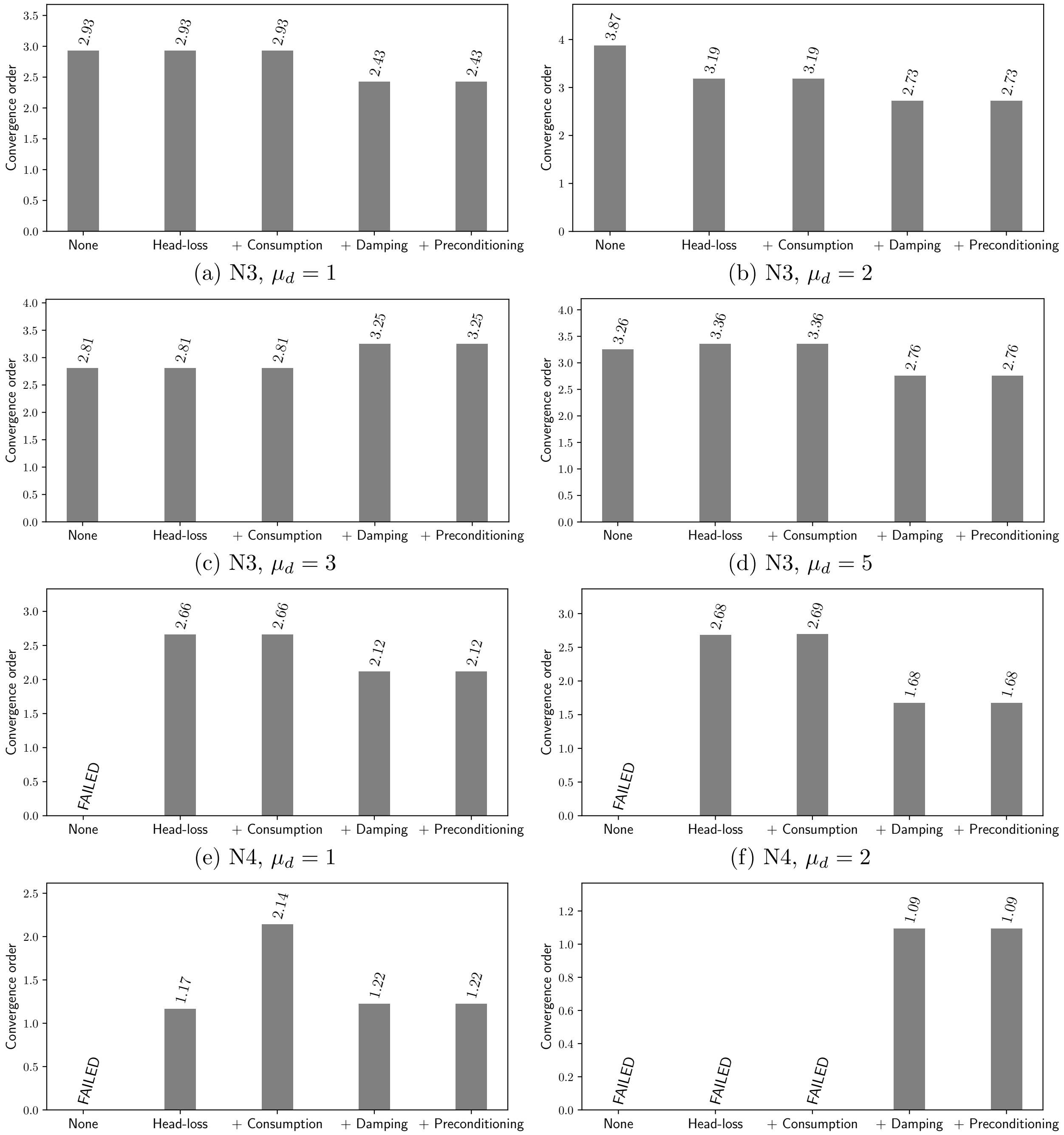
(e) N2, $\mu_d = 1$

(f) N2, $\mu_d = 2$

(g) N2, $\mu_d = 3$

(h) N2, $\mu_d = 5$

Figure A.3: Orders of convergence of the Newton's method described at section 1.2.4 when simulating the networks N1 and N2 for each demand multiplier $\mu_d \in \{1, 2, 3, 5\}$, first with no numerical enhancement (bar "None"), then with friction head-loss regularization only (bar "Head-loss"), then with friction head-loss and consumption regularization (bar "+ Consumption"), then with friction head-loss regularization, consumption regularization and damping correction (bar "+ Damping"), and finally with all enhancements including preconditioning of derivatives (bar "+ Preconditioning"). "FAILED" means that the simulation did not converge.



(a) N3, $\mu_d = 1$

(b) N3, $\mu_d = 2$

(c) N3, $\mu_d = 3$

(d) N3, $\mu_d = 5$

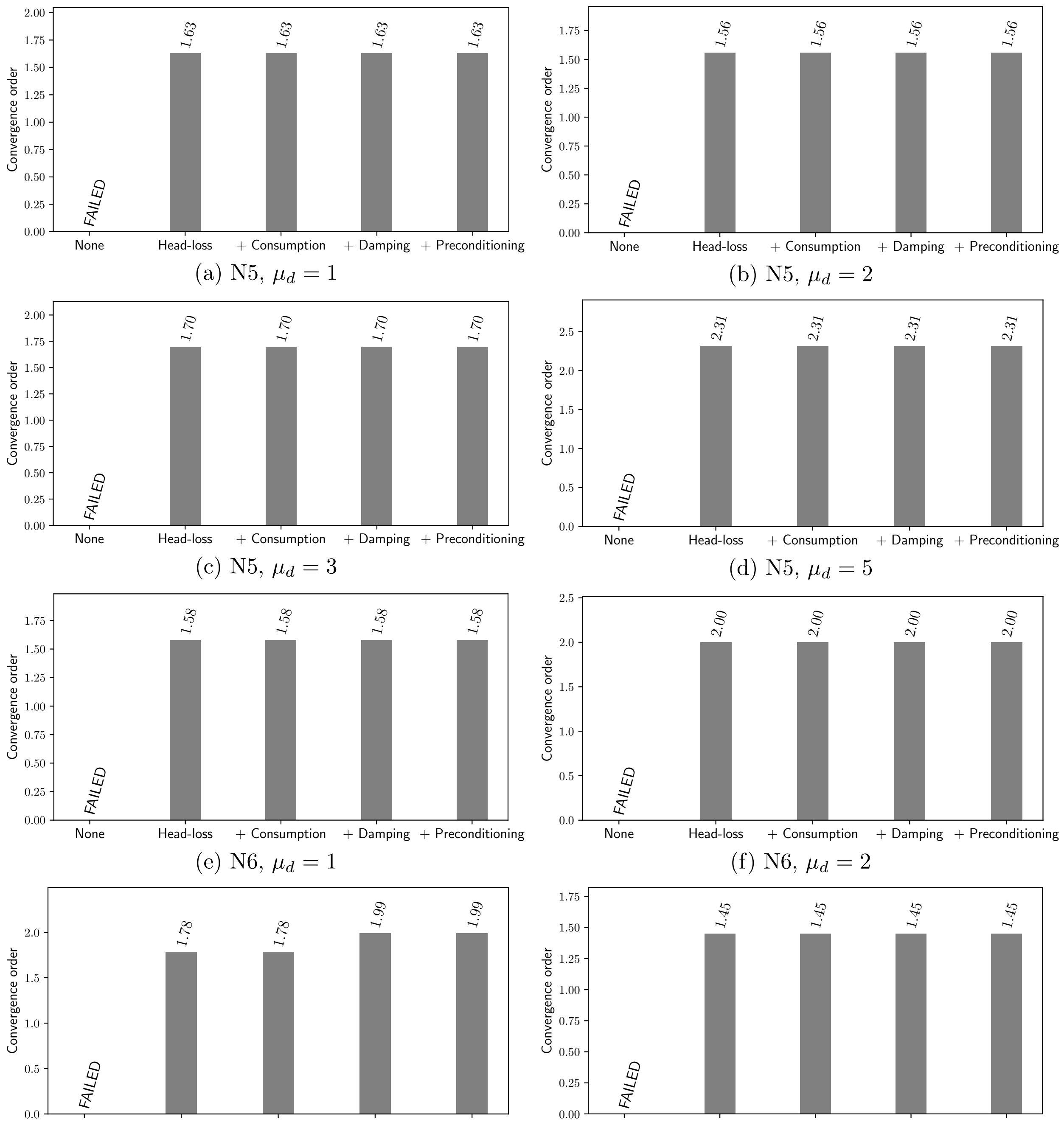
(e) N4, $\mu_d = 1$

(f) N4, $\mu_d = 2$

(g) N4, $\mu_d = 3$

(h) N4, $\mu_d = 5$

Figure A.4: Orders of convergence of the Newton's method described at section 1.2.4 when simulating the networks N3 and N4 for each demand multiplier $\mu_d \in \{1, 2, 3, 5\}$, first with no numerical enhancement (bar "None"), then with friction head-loss regularization only (bar "Head-loss"), then with friction head-loss and consumption regularization (bar "+ Consumption"), then with friction head-loss regularization, consumption regularization and damping correction (bar "+ Damping"), and finally with all enhancements including preconditioning of derivatives (bar "+ Preconditioning"). "FAILED" means that the simulation did not converge.



(a) N5, $\mu_d = 1$

(b) N5, $\mu_d = 2$

(c) N5, $\mu_d = 3$

(d) N5, $\mu_d = 5$

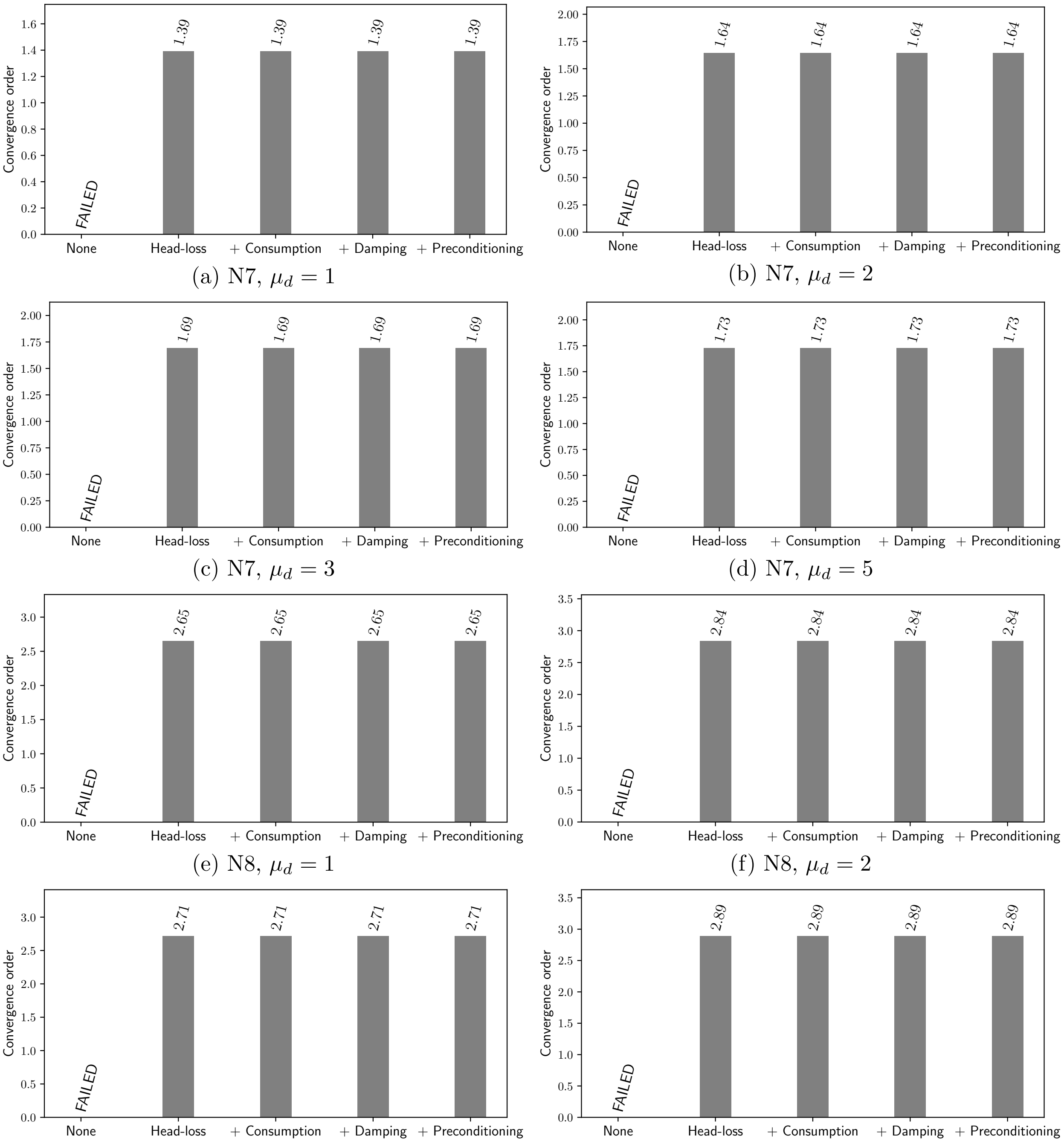
(e) N6, $\mu_d = 1$

(f) N6, $\mu_d = 2$

(g) N6, $\mu_d = 3$

(h) N6, $\mu_d = 5$

Figure A.5: Orders of convergence of the Newton's method described at section 1.2.4 when simulating the networks N5 and N6 for each demand multiplier $\mu_d \in \{1, 2, 3, 5\}$, first with no numerical enhancement (bar "None"), then with friction head-loss regularization only (bar "Head-loss"), then with friction head-loss and consumption regularization (bar "+ Consumption"), then with friction head-loss regularization, consumption regularization and damping correction (bar "+ Damping"), and finally with all enhancements including preconditioning of derivatives (bar "+ Preconditioning"). "FAILED" means that the simulation did not converge.



(a) N7, $\mu_d = 1$

(b) N7, $\mu_d = 2$

(c) N7, $\mu_d = 3$

(d) N7, $\mu_d = 5$

(e) N8, $\mu_d = 1$

(f) N8, $\mu_d = 2$

(g) N8, $\mu_d = 3$

(h) N8, $\mu_d = 5$

Figure A.6: Orders of convergence of the Newton's method described at section 1.2.4 when simulating the networks N7 and N8 for each demand multiplier $\mu_d \in \{1, 2, 3, 5\}$, first with no numerical enhancement (bar "None"), then with friction head-loss regularization only (bar "Head-loss"), then with friction head-loss and consumption regularization (bar "+ Consumption"), then with friction head-loss regularization, consumption regularization and damping correction (bar "+ Damping"), and finally with all enhancements including preconditioning of derivatives (bar "+ Preconditioning"). "FAILED" means that the simulation did not converge.

Appendix B

Supplementary material of chapter 2: Pressure-dependent background leakages

B.1 Approximation of continuous theoretical functions $\{\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}, \mathrm{q}^{\mathrm{theo}}\}$ by piecewise constant functions $\{\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}, \mathrm{q}^{\mathrm{M0}}\}$

In section 2.2.1.4, we proposed a new model based on the recursive discretization of a pipe into sub-pipes until the difference between the hydraulic grade lines (HGLs) of two consecutive discretization levels becomes small enough.

Since the functions $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}$ and q^{M0} are piecewise constant functions (i.e., constant per sub-pipe; see eqs. (2.7) and (2.8)), they can lead to a good numerical approximation of the continuous theoretical functions $\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}$ and $\mathrm{q}^{\mathrm{theo}}$ (eqs. (2.4) and (2.5)), providing that the pipe is discretized in enough sub-pipes. Below is the proof for $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}$ and $\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}$. The same reasoning can be used for q^{M0} and $\mathrm{q}^{\mathrm{theo}}$.

Proof.

$\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}$ is uniformly continuous because it is continuous on the compact interval $[0, \ell]$. Then, $\forall \varepsilon > 0$ and $\forall\,(x_1, x_2) \in [0, \ell] \times [0, \ell]$, $\exists \delta > 0$ such that $\left|\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x_1) - \mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x_2)\right| < \varepsilon$, provided that $|x_1 - x_2| < \delta$.

We discretize the pipe into n sub-pipes of equal length, with n large enough so that ${}^{\ell}\!/_{n} < \delta$. Then, $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}$ being constant per sub-pipe, we have $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}(x) = \mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}({}^{[nx]}\!/_{n})$.

$\forall x \in [{}^{k\,\ell}\!/_{n}, {}^{(k+1)\,\ell}\!/_{n})$, $k = 0, 1, \ldots, n$, we have ${}^{[nx]}\!/_{n} = {}^{k\,\ell}\!/_{n}$. Then, we also have:

$$\left|\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x) - \mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}(x)\right| = \left|\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x) - \mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}({}^{[nx]}\!/_{n})\right| = \left|\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x) - \mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}({}^{k\,\ell}\!/_{n})\right| < \varepsilon \tag{B.1}$$

because $|x - {}^{k\,\ell}\!/_{n}| < \delta$. Thus,

$$\sup_{x \in [0, \ell]} \left|\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}(x) - \mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}(x)\right| < \varepsilon$$

and $\mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}$ approximates the continuous function $\mathrm{q}_{\mathrm{LL}}^{\mathrm{theo}}$, with ε the error of the approximation.

□

B.2 Integration of the unitary friction head-loss for the models M2 and M3

In sections 2.2.1.6 and 2.2.1.7, we integrate numerically the unitary friction head-loss functions $\{\varphi^{\mathrm{M2}}, \varphi^{\mathrm{M3}}\}$, defined by eqs. (2.29) and (2.34) as:

$$\varphi^{\mathrm{M2}}(y) = f\mathrm{q}^{\mathrm{M2}}(y)|\mathrm{q}^{\mathrm{M2}}(y)|^{\gamma_{HW}-1} \qquad \text{and} \qquad \varphi^{\mathrm{M3}}(y) = f\mathrm{q}^{\mathrm{M3}}(y)|\mathrm{q}^{\mathrm{M3}}(y)|^{\gamma_{HW}-1}.$$

To do so, we use a Newton-Cotes formula of degree 2, which leads to the friction head-loss functions $\{\xi_{\mathrm{f}}^{\mathrm{M2}}, \xi_{\mathrm{f}}^{\mathrm{M3}}\}$, defined by eqs. (2.28) and (2.33) as:

$$\xi_{\mathrm{f}}^{\mathrm{M2}}(x) = \frac{x}{6}\Big(\varphi^{\mathrm{M2}}(0) + 4\,\varphi^{\mathrm{M2}}(x/2) + \varphi^{\mathrm{M2}}(x)\Big) \quad \text{and} \quad \xi_{\mathrm{f}}^{\mathrm{M3}}(x) = \frac{x}{6}\Big(\varphi^{\mathrm{M3}}(0) + 4\,\varphi^{\mathrm{M3}}(x/2) + \varphi^{\mathrm{M3}}(x)\Big).$$

Indeed, (2.29) and (2.34) do not have any elementary antiderivative that can be expressed in terms of elementary functions. Below is the proof for (2.29); the same reasoning can be used for (2.34).

Proof.

In model M2, the unitary friction head-loss is defined by eq. (2.29) as:

$$\varphi^{\mathrm{M2}}(y) = f\mathrm{q}^{\mathrm{M2}}(y)|\mathrm{q}^{\mathrm{M2}}(y)|^{\gamma_{HW}-1},$$

where the flow rate $\mathrm{q}^{\mathrm{M2}}(y)$ is computed by eq. (2.27) as:

$$\mathrm{q}^{\mathrm{M2}}(x) = q_{0.5} - \widehat{\mathrm{q}_{\mathrm{LL}}^{\mathrm{M2}}}(x)\left(x - \frac{\ell}{2}\right).$$

We can rewrite eq. (2.27) as a polynomial of degree 2 such that:

$$\mathrm{q}^{\mathrm{M2}}(x) = \frac{q_{LL0} - q_{LL\ell}}{2\,\ell}\,x^2 - q_{LL0}\,x + \frac{(3\,q_{LL0} + q_{LL\ell})\,\ell}{8} + q_{0.5}. \tag{B.2}$$

Thus, we can also rewrite eq. (2.29) as:

$$\widetilde{\varphi^{\mathrm{M2}}}(x) = \Big(1 + x^2\Big)^{\gamma_{HW}}, \tag{B.3}$$

if and only if:

$$\exists\,\{q_{LL0}, q_{LL\ell}, \ell, q_{0.5}\} \in \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}_+^* \times \mathbb{R} \tag{B.4}$$

such that:

$$\frac{q_{LL0} - q_{LL\ell}}{2\,\ell} \neq 0 \tag{B.5}$$

and

$$-q_{LL0}x + \frac{(3q_{LL0} + q_{LL\ell})\,\ell}{8} + q_{0.5} > 0. \tag{B.6}$$

Indeed, if we choose $q_{LL\ell} = 0$, then the inequality (B.5) is true $\forall q_{LL0} \neq 0$, and the inequality (B.6) is equivalent to:

$$0 \leq x < \frac{3\,\ell}{8} + \frac{q_{0.5}}{q_{LL0}} \leq \ell. \tag{B.7}$$

Also, if we choose any $q_{0.5} > 0$, then the inequality (B.7) is true $\forall q_{LL0}$ such that:

$$q_{LL0} \geq \frac{8\, q_{0.5}}{5\, \ell}. \tag{B.8}$$

Thus, there exists, $\forall x \in [0, \ell]$, at least one set $\{q_{LL0}, q_{LL\ell}, \ell, q_{0.5}\} \in \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}_+^* \times \mathbb{R}$ such that eq. (2.29) can be rewritten as eq. (B.3).

We remind now the Tchebichef's theorem on the integration of binomial differentials [153]:

Tchebichef's theorem on the integration of binomial differentials.

Let f *be a function such that:*

$$\begin{aligned} \mathrm{f}\colon \mathbb{R} &\to \mathbb{R} \\ x &\mapsto x^m \,(\alpha + \beta x^n)^p \,, \end{aligned} \tag{B.9}$$

where α and β are real numbers, m, n and p are rational numbers, and $\alpha\beta n \neq 0$; functions like f *are called "differential binomials".* f *has an elementary antiderivative, that is, an antiderivative that can be expressed in terms of elementary functions, if and only if at least one of p, $(m+1)/n$ or $p+(m+1)/n$ is an integer.*

We can prove that eq. (B.3) can be rewritten in the form of eq. (B.9) by construction. Indeed, if we choose $\alpha = \beta = 1$, $m = 0$, $n = 2$ and $p = \gamma_{HW}$, then $\alpha\beta n = 2 \neq 0$, and eq. (B.3) can be rewritten in the form of eq. (B.9). Moreover, we have $p = \gamma_{HW} = 1.852 \notin \mathbb{Z}$, $(m+1)/n = 0.5 \notin \mathbb{Z}$ and $p + (m+1)/n = 1.852 + 0.5 = 2.352 \notin \mathbb{Z}$. Thus, eq. (B.3) does not have any elementary antiderivative, and, by deduction, the unitary friction head-loss eq. (2.29) also does not have an elementary antiderivative $\forall \{q_{LL0}, q_{LL\ell}, \ell, q_{0.5}\} \in \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}_+^* \times \mathbb{R}$ (i.e., there exists at least one set $\{q_{LL0}, q_{LL\ell}, \ell, q_{0.5}\} \in \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}_+^* \times \mathbb{R}$ such that the unitary friction head-loss eq. (2.29) does not have an elementary antiderivative).

□

Remark: we also check the nonexistence of elementary antiderivative of eq. (B.3) using the symbolic algebra system AXIOM [82].

B.3 Numerical enhancements to deal with sources of instabilities

B.3.1 Regularization of lineic leakage outflow rate and of its derivative

B.3.1.1 Cubic regularization

At each iteration of the Newton's method (section 2.2.2.2) and for each pipe $k \in \{1, \dots, n_p\}$, we need to compute the lineic leakage outflow rate for one or several local pressure-head(s) $p_k \in \mathbb{R}$, as [49]:

$$\mathrm{q}_{\mathrm{LL},k}(p_k) = \beta_{L,k} \left([p_k]^+\right)^{\alpha_{L,k}} , \tag{B.10}$$

where $[p_k]^+$ is the positive part of p_k, and $\alpha_{L,k} \in \,]0.5, 2.5]$ and $\beta_{L,k} \in \left[10^{-7}, 10^{-1}\right] \mathrm{l\,s^{-1}\,m^{-\alpha_{L,k}-1}}$ are the leakage parameters associated to pipe k. Also, we need to compute the derivative of eq. (B.10) with respect to p_k, as:

$$\frac{\mathrm{dq}_{\mathrm{LL},k}}{\mathrm{d}p_k}(p_k) = \begin{cases} \alpha_{L,k}\,\beta_{L,k}\,p_k^{\alpha_{L,k}-1} & \text{if } p_k > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{B.11}$$

Equation (B.11) is discontinuous at $p_k = 0\ \forall \alpha_{L,k} \in]0.5, 1[$; this discontinuity could lead to convergence failure. To make eq. (B.11) continuous $\forall p_k \in \mathbb{R}$ and $\forall \alpha_{L,k} \in]0.5, 2.5[$, we propose a new cubic regularization of eq. (B.10) and a new quadratic regularization of eq. (B.11) for p_k close to 0.

To do so, we first choose an $\varepsilon_{q_{LL}} \in \mathbb{R}_*^+$ close to $0\,\mathrm{mH_2O}$ (e.g., $\varepsilon_{q_{LL}} = 10^{-3}\,\mathrm{mH_2O}$) that defines the interval $[-\varepsilon_{q_{LL}}, \varepsilon_{q_{LL}}]$ into which the regularization has to be done. Next, we look for the polynomial $\mathrm{P_{q_{LL},k}}(p_k) = a_{q_{LL},3,k}\,{p_k}^3 + a_{q_{LL},2,k}\,{p_k}^2 + a_{q_{LL},1,k}\,p_k + a_{q_{LL},0,k}$ and its derivative $\mathrm{dP_{q_{LL},k}}/\mathrm{d}p_k\ (p_k) = 3\,a_{q_{LL},3,k}\,{p_k}^2 + 2\,a_{q_{LL},2,k}\,p_k + a_{q_{LL},1,k}$, which coefficients $\{a_{q_{LL},0,k}, \dots, a_{q_{LL},3,k}\}$ are solutions of the system:

$$\begin{cases} -\,a_{q_{LL},3,k}\,{\varepsilon_{q_{LL}}}^3 + a_{q_{LL},2,k}\,{\varepsilon_{q_{LL}}}^2 - a_{q_{LL},1,k}\,\varepsilon_{q_{LL}} + a_{q_{LL},0,k} = 0 & \text{(B.12a)} \\ a_{q_{LL},3,k}\,{\varepsilon_{q_{LL}}}^3 + a_{q_{LL},2,k}\,{\varepsilon_{q_{LL}}}^2 + a_{q_{LL},1,k}\,\varepsilon_{q_{LL}} + a_{q_{LL},0,k} = \beta_{L,k}\,(\varepsilon_{q_{LL}})^{\alpha_{L,k}} & \text{(B.12b)} \\ 3a_{q_{LL},3,k}\,{\varepsilon_{q_{LL}}}^2 - 2\,a_{q_{LL},2,k}\,\varepsilon_{q_{LL}} + a_{q_{LL},1,k} = 0 & \text{(B.12c)} \\ 3a_{q_{LL},3,k}\,{\varepsilon_{q_{LL}}}^2 + 2\,a_{q_{LL},2,k}\,\varepsilon_{q_{LL}} + a_{q_{LL},1,k} = \alpha_{L,k}\,\beta_{L,k}\,{\varepsilon_{q_{LL}}}^{\alpha_{L,k}-1}, & \text{(B.12d)} \end{cases}$$

where eqs. (B.12a) and (B.12b) insure the continuity between $\mathrm{q}_{\mathrm{LL},k}(p_k)$ and $\mathrm{P_{q_{LL},k}}(p_k)$ at $p_k \in \{-\varepsilon_{q_{LL}}, \varepsilon_{q_{LL}}\}$, and eqs. (B.12c) and (B.12d) the continuity between their derivative at the same positions. Equation (B.12) need to be solve only once before any simulation run, and for each distinct couple $(\alpha_{L,k}, \beta_{L,k})$, $k \in \{1, \dots, n_p\}$.

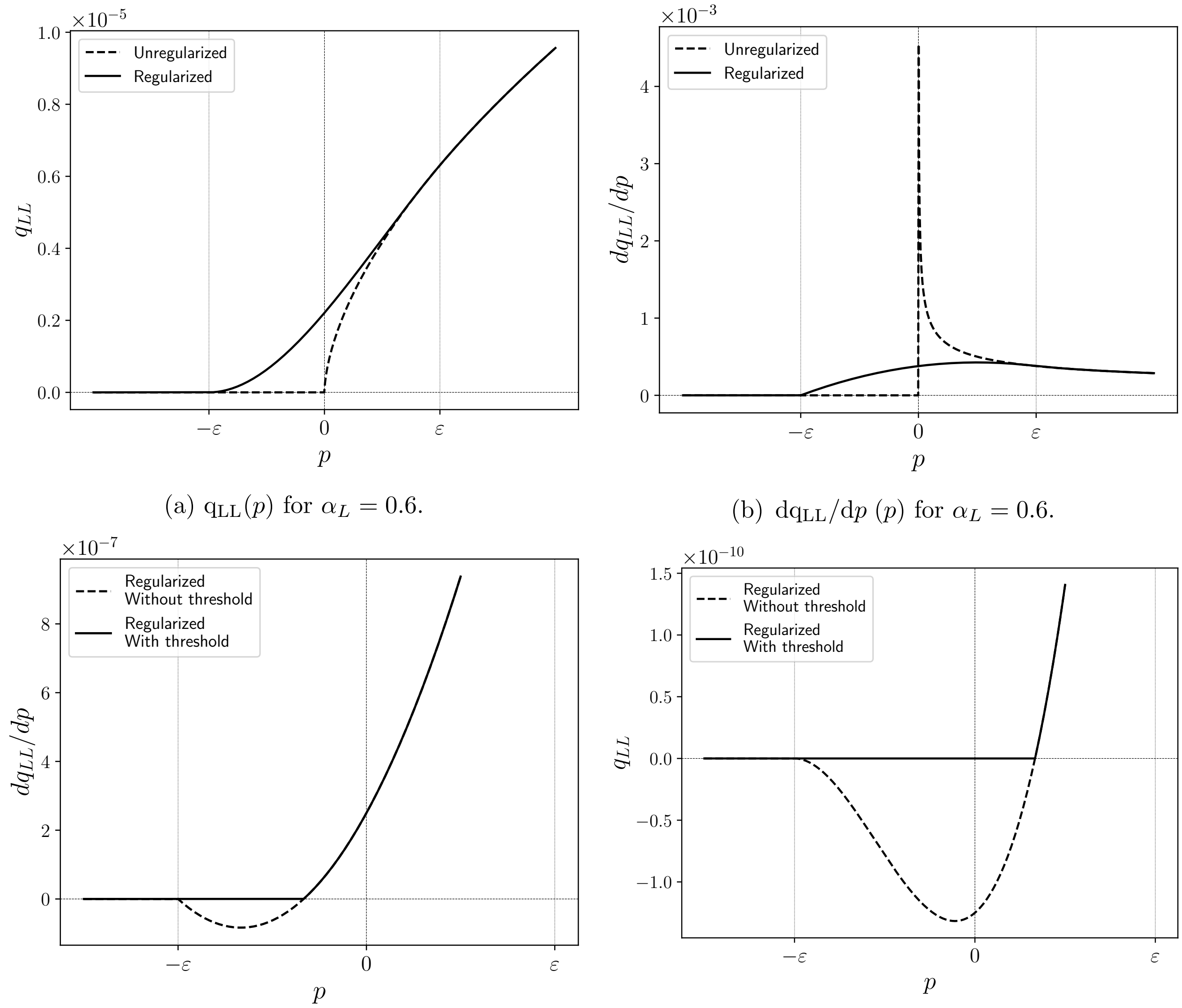
Once we found $\{a_{q_{LL},0,k}, \dots, a_{q_{LL},3,k}\}$, we can then compute, at each iteration of the Newton's method and for any local pressure-head p_k along k, the lineic leakage outflow rate $q_{LL,k}$ in k as:

$$\mathrm{q}_{\mathrm{LL},k}(p_k) = \begin{cases} \mathrm{P_{q_{LL},k}}(p_k) & \text{if } -\varepsilon_{q_{LL}} \le p_k \le \varepsilon_{q_{LL}} \\ \text{eq. (B.10)} & \text{otherwise,} \end{cases} \tag{B.13}$$

and its derivative $dq_{LL,k}/dp_k$ as:

$$\frac{\mathrm{dq}_{\mathrm{LL},k}}{\mathrm{d}p_k}(p_k) = \begin{cases} \dfrac{\mathrm{dP_{q_{LL},k}}}{\mathrm{d}p_k}(p_k) & \text{if } -\varepsilon_{q_{LL}} \le p_k \le \varepsilon_{q_{LL}} \\ \text{eq. (B.11)} & \text{otherwise.} \end{cases} \tag{B.14}$$

Figures B.1a and B.1b illustrate eqs. (B.13) and (B.14) when $\alpha_L = 0.6$ and $\beta_L = 10^{-4}\ \mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$ (we omit the subscripts for better readability), without any loss of generality. Indeed, eqs. (B.10) to (B.14) depend linearly on β_L. Thus, figs. B.1a and B.1b can be transposed to any other β_L, by just multiplying the values plotted on the vertical axis by $\beta_L/10^{-4}$.



(a) $\mathrm{q_{LL}}(p)$ for $\alpha_L = 0.6$.

(b) $\mathrm{dq_{LL}/d}p\ (p)$ for $\alpha_L = 0.6$.

(c) $\mathrm{dq_{LL}/d}p\ (p)$ for $\alpha_L = 2$.

(d) $\mathrm{q_{LL}}(p)$ for $\alpha_L = 2.5$.

Figure B.1: New cubic regularization of the lineic leakage outflow rate function $\mathrm{q_{LL}}(p)$ [49] (in $\mathrm{l\,s^{-1}\,m^{-1}}$, figs. B.1a and B.1d) and quadratic regularization of its derivative $\mathrm{dq_{LL}/d}p\ (p)$ (in $\mathrm{l\,s^{-1}\,m^{-1}\,mH_2O^{-1}}$, figs. B.1b and B.1c), in a pipe with leakage type $\alpha_L \in \{0.6, 2, 2.5\}$ (unit-less), degradation level $\beta_L = 10^{-4}\,\mathrm{l\,s^{-1}\,m^{-\alpha_L-1}}$, for local pressure-heads p close to $0\,\mathrm{mH_2O}$. Figures B.1c and B.1d represent the regularization with and without threshold of negative values. Here, ε is chosen equal to $10^{-2}\,\mathrm{mH_2O}$ for better readability, but in the source code $\varepsilon = 10^{-3}\,\mathrm{mH_2O}$.

B.3.1.2 Threshold of the negative values induced by the cubic regularization

In any pipe k, the cubic regularization proposed at B.3.1.1 can lead to negative derivative of the lineic leakage outflow rate when $\alpha_{L,k} \geq 1$. In other words, when $\alpha_{L,k} \geq 1$,

$$\exists \{\varepsilon_1, \varepsilon_2\} \in \mathbb{R} \times \mathbb{R}, \varepsilon_1 \neq \varepsilon_2 \mid \frac{\mathrm{dP_{q_{LL},k}}}{\mathrm{d}p_k}(p_k) < 0 \ \forall p_k \in]\varepsilon_1, \varepsilon_2[\,. \tag{B.15}$$

Proof.

To simplify the notation, we denote $p = p_k$, $\mathrm{dP}/\mathrm{d}p = \mathrm{dP}_{\mathrm{q_{LL},k}}/\mathrm{d}p_k$, $\varepsilon = \varepsilon_{q_{LL}}$, and $a_i = a_{q_{LL},i,k}$ $\forall i \in \{1, \ldots, 3\}$. Then, $\forall p \in \mathbb{R}$ such that $|p| \leq \varepsilon$, and $\forall \{a_3, a_2, a_1\}$ solutions of the system (B.12), we can rewrite eq. (B.12c) as:

$$\frac{\mathrm{dP}}{\mathrm{d}p}(p) = 3\, a_3\, p^2 - 2\, a_2\, p + a_1. \tag{B.16}$$

The discriminant of $\mathrm{dP}/\mathrm{d}p$ is:

$$\Delta = 4\left({a_2}^2 - 3\, a_1 a_3\right). \tag{B.17}$$

Combining two by two the rows of (B.12), we obtain, $\forall \{\alpha_L, \beta_L\} \in \mathbb{R}_+ \times \mathbb{R}_+$ and $\forall \varepsilon \in \mathbb{R}_+^*$:

$$a_0 = \frac{1}{4}\, \beta_L\, \varepsilon^{\alpha_L}\, (2 - \alpha_L)\,, \tag{B.18a}$$

$$a_1 = \frac{1}{4}\, \beta_L\, \varepsilon^{\alpha_L - 1}\, (3 - \alpha_L)\,, \tag{B.18b}$$

$$a_2 = \frac{1}{4}\, \alpha_L\, \beta_L\, \varepsilon^{\alpha_L - 2}, \tag{B.18c}$$

$$a_3 = \frac{1}{4}\beta_L\, \varepsilon^{\alpha_L - 3}\, (\alpha_L - 1)\,. \tag{B.18d}$$

Injecting eqs. (B.18b) to (B.18d) into eq. (B.17), we get:

$$\Delta = \left[\beta_L\, \varepsilon^{\alpha_L - 2}\left(\alpha_L - \frac{3}{2}\right)\right]^2. \tag{B.19}$$

$\Delta > 0$. Thus, $\mathrm{dP}/\mathrm{d}p$ has 2 distinct real roots:

$$\varepsilon_1 = \frac{1}{3\, a_3}\left(a_2 - \frac{\sqrt{\Delta}}{2}\right), \tag{B.20a}$$

$$\varepsilon_2 = \frac{1}{3\, a_3}\left(a_2 + \frac{\sqrt{\Delta}}{2}\right). \tag{B.20b}$$

Next, injecting eqs. (B.18c), (B.18d) and (B.19) into eqs. (B.20a) and (B.20b), we obtain:

$$\varepsilon_1 = \frac{\varepsilon\,(3 - \alpha_L)}{3\,(\alpha_L - 1)}, \tag{B.21a}$$

$$\varepsilon_2 = \varepsilon. \tag{B.21b}$$

$\varepsilon_1 > 0$ $\forall \alpha_L \in]1, 3[$. Also, $\{\varepsilon_1, \varepsilon_2\}$ are such that:

$$\begin{cases} \varepsilon_1 < \varepsilon_2 & \text{if } \alpha_L > \dfrac{3}{2} \\ \varepsilon_1 = \varepsilon_2 & \text{if } \alpha_L = \dfrac{3}{2} \\ \varepsilon_1 > \varepsilon_2 & \text{otherwise.} \end{cases} \tag{B.22}$$

Thus, the $\mathrm{dP}/\mathrm{d}p$ has the same sign as $-3\, a_3$ between ε_1 and ε_2. $a_3 > 0$ $\forall \varepsilon \in \mathbb{R}_+^*$ because $\alpha_L > 1$. Therefore, $\mathrm{dP}/\mathrm{d}p$ is negative between ε_1 and ε_2.

□

Also, in any pipe k, the cubic regularization proposed at B.3.1.1 can lead to negative lineic leakage outflow rate when $\alpha_{L,k} \geq 2$. In other words, when $\alpha_{L,k} \geq 2$,

$$\exists \{\varepsilon_1, \varepsilon_2\} \in \mathbb{R}_- \times \mathbb{R}_+, \varepsilon_1 \neq \varepsilon_2 \mid \mathrm{P}_{\mathrm{q_{LL}},\mathrm{k}}(p_k) < 0 \ \forall p_k \in \left]\varepsilon_1, \varepsilon_2\right[. \tag{B.23}$$

Proof.

To simplify the notation, we denote $p = p_k$, $\mathrm{P} = \mathrm{P}_{\mathrm{q_{LL}},\mathrm{k}}$, $\varepsilon = \varepsilon_{q_{LL}}$, and $a_i = a_{q_{LL},i,k}\ \forall i \in \{0, \dots, 3\}$. Then, $\forall \varepsilon \in \mathbb{R}_+^*$ and $\forall \{a_i \mid i \in \{0, \dots, 3\}\}$ solutions of (B.12), we can rewrite eq. (B.12a) as:

$$\mathrm{P}(\varepsilon) = -a_3\, \varepsilon^3 + a_2\, \varepsilon^2 - a_1\, \varepsilon + a_0 = 0. \tag{B.24}$$

Replacing ε by $-\varepsilon$ in eq. (B.24), we have also:

$$\mathrm{P}(-\varepsilon) = a_3\, \varepsilon^3 + a_2\, \varepsilon^2 + a_1\, \varepsilon + a_0 = 0. \tag{B.25}$$

Thus, $\varepsilon_1 = -\varepsilon < 0$ is a root of P.

From (B.18), we have also, $\forall \alpha_L \in \left]0.5, 2.5\right]$:

$$\begin{cases} \mathrm{sgn}(a_3) & = \mathrm{sgn}(\alpha_L - 1) \\ \mathrm{sgn}(a_2) & = 1 \\ \mathrm{sgn}(a_1) & = \mathrm{sgn}(3 - \alpha_L) \\ \mathrm{sgn}(a_0) & = \mathrm{sgn}(2 - \alpha_L), \end{cases} \tag{B.26}$$

where sgn is the sign function, defined, $\forall x \in \mathbb{R}$, as:

$$\mathrm{sgn}(x) = \begin{cases} -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases} \tag{B.27}$$

In particular, if $\alpha_L \in \left]2, 2.5\right]$, we have $\mathrm{sgn}(a_3) = \mathrm{sgn}(a_2) = \mathrm{sgn}(a_1) = 1$ and $\mathrm{sgn}(a_0) = -1$. Thus, according to the Descartes's rule of signs, P has also 1 positive root that we denote ε_2.

Finally, $\mathrm{sgn}(-a_3) = -\,\mathrm{sgn}(a_3) = -1\ \forall \alpha_L \in \left]2, 2.5\right]$. Thus, P is negative between ε_1 et ε_2.

□

Negative lineic leakage outflow rate or negative derivative of lineic leakage outflow rate is not physically realistic. Thus, if the cubic regularization proposed at B.3.1.1 leads to negative lineic leakage outflow rate or to negative derivative of the lineic leakage outflow rate, we choose to replace these negative values by 0. Equations (B.13) and (B.14) then become respectively:

$$\mathrm{q}_{\mathrm{LL},k}(p_k) = \begin{cases} 0 & \text{if } -\varepsilon_{q_{LL}} \leq p_k \leq \varepsilon_{q_{LL}} \text{ and } \mathrm{P}_{\mathrm{q_{LL}},\mathrm{k}}(p_k) \leq 0 \\ \text{eq. (B.13)} & \text{otherwise,} \end{cases} \tag{B.28}$$

and:

$$\frac{\mathrm{d}\mathrm{q}_{\mathrm{LL},k}}{\mathrm{d}p_k}(p_k) = \begin{cases} 0 & \text{if } -\varepsilon_{q_{LL}} \leq p_k \leq \varepsilon_{q_{LL}} \text{ and } \dfrac{\mathrm{dP}_{\mathrm{q_{LL}},\mathrm{k}}}{\mathrm{d}p_k}(p_k) \leq 0 \\ \text{eq. (B.14)} & \text{otherwise.} \end{cases} \tag{B.29}$$

Figures B.1c and B.1d illustrate eqs. (B.28) and (B.29) for $\alpha_L \in \{2, 2.5\}$ and $\beta_L = 10^{-4}\ \mathrm{l\,s^{-1}\,m^{-\alpha_L - 1}}$.

B.3.2 Preconditioning of the Jacobian matrix to prevent division by zero error

At each iteration of the Newton's method, we need to compute the inverse of the sub-block $\mathbf{J_{11}}$ of the Jacobian matrix function, defined at section 2.2.2.2 as:

$$\mathbf{J_{11}} = \frac{\boldsymbol{\partial \xi_f}}{\boldsymbol{\partial q_{0.5}}},$$

where $\boldsymbol{\xi_f}$ is the friction head-loss function, computed, according to the leakage model used, by one of eqs. (2.11), (2.25), (2.28) and (2.33). For all models, we can rewrite the friction head-loss function along the whole pipe k in the generic form:

$$\xi_{\mathrm{f},k}(\ell_k) = f_k \int_0^{\ell_k} \mathrm{q}_k(x)|\mathrm{q}_k(x)|^{\gamma_{HW}-1}\,\mathrm{d}x\,,$$

and its derivative as:

$$\frac{\mathrm{d}\xi_{\mathrm{f},k}}{\mathrm{d}q_{0.5,k}}(\ell_k) = \gamma_{HW}\, f_k \int_0^{\ell_k} \frac{\mathrm{d}\mathrm{q}_k}{\mathrm{d}q_{0.5,k}}(x)\cdot|\mathrm{q}_k(x)|^{\gamma_{HW}-1}\,\mathrm{d}x\,. \tag{B.30}$$

Equation (2.40) is 0 if $\mathrm{q}_k(x) = 0\ \forall x \in [0, \ell_k]$. In this case, we will have a division by zero error. To avoid this error, we choose in chapter 2,

- for model M0: to reuse, as in chapter 1, the cubic regularization of the friction head-loss function initially proposed by [122] for flow rate close to 0; this regularization is fully described in appendix A.1.1,
- for model $\{\mathrm{M1}, \mathrm{M2}, \mathrm{M3}\}$: to reuse the preconditioning method initially proposed by [42].

To reuse the preconditioning method initially proposed by [42], we limit each singular value λ_k, $k \in \{1, \dots, n_p\}$, of the sub-block $\boldsymbol{J_{11}}$ of $\boldsymbol{J}$ by a minimum value λ_{min} defined as:

$$\lambda_{min} = \frac{\lambda_{max}}{\kappa_{max}}, \tag{B.31}$$

where λ_{max} is such that

$$\lambda_{max} = \max_k\Big([\boldsymbol{J_{11}}]_{kk}\Big), \tag{B.32}$$

and κ_{max} corresponds to a maximum threshold condition number (e.g., $\kappa_{max} = 10^{10}$). Like so, we prevent zero values on the diagonal of $\boldsymbol{J_{11}}$ and reduce the difference of magnitude order between all elements of $\boldsymbol{J_{11}}$. When an element of $\boldsymbol{J_{11}}$ is actually modified by this preconditioning method, the Newton's method used in chapter 2 (see section 2.2.2.2) then becomes a quasi-Newton method.

B.3.3 Damping of descent directions

As in chapter 1, to avoid numerous iterations due to initial guesses of flow rates at middle of pipes and heads at junctions far from the solutions at equilibrium, and to guarantee the convergence of the Newton's method even if the Jacobian of the non-linear system of equations (2.35) does not have a super-linear growth [133], we chose to extend the damping algorithm initially proposed by [43] and described in appendix A.1.3. This extension permits the algorithm to deal with pressure-dependent background leakages.

To do so, replacing $\boldsymbol{q}$ by $\boldsymbol{q_{0.5}}$ in eq. (A.17), we obtain the weighted least squares (WLS) objective scalar function:

$$\theta(\boldsymbol{q_{0.5}}, \boldsymbol{h}) = \frac{1}{2}\left\| \boldsymbol{W}^{\frac{1}{2}} \rho(\boldsymbol{q_{0.5}}, \boldsymbol{h}) \right\|_2^2 = \frac{1}{2}\boldsymbol{\rho}^T \boldsymbol{W} \boldsymbol{\rho}, \tag{B.33}$$

(see appendix A.1.3 for the definition of other variables). Next, the different steps of the algorithm are the same as in the original version from [43], but the computation of the weights $\boldsymbol{W}$ is different. Indeed, to extend the algorithm to pressure-dependent background leakages, we need to take into account the leakage outflow rates $\boldsymbol{q_L}$ (in $\mathrm{l\,s^{-1}}$) reported to junctions too, which is defined at any junction $j \in \{1, \ldots, n_j\}$ as:

$$q_{L,j} = \sum_{k \in \mathbb{P}_j^+} \int_0^{\ell_k/2} \mathrm{q_{LL}}_k(x_k)\, \mathrm{d}x_k + \sum_{k \in \mathbb{P}_j^-} \int_{\ell_k/2}^{\ell_k} \mathrm{q_{LL}}_k(x_k)\, \mathrm{d}x_k\,, \tag{B.34}$$

where $\mathbb{P}_j^-$ and $\mathbb{P}_j^+$ are the sets of pipes respectively entering and leaving the node j, ℓ_k is the length of pipe $k \in \mathbb{P}_j^- \cup \mathbb{P}_j^+$, and $\mathrm{q_{LL}}_k(x_k)$ is the lineic leakage outflow rate at the position x_k of k. Then, we compute the weights $\boldsymbol{W}$ as:

$$\boldsymbol{W} = \begin{pmatrix} \boldsymbol{M}^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{N}^{-1} \end{pmatrix}, \tag{B.35}$$

with

$$\boldsymbol{M} = \left(\max_{i \in \{1,\ldots,n_0\}} \boldsymbol{h_{f,i}} \right)^2 \boldsymbol{I_{n_p}} \qquad \text{and} \qquad \boldsymbol{N} = \left(\max_{j \in \{1,\ldots,n_j\}} (\boldsymbol{d}_j + \boldsymbol{q_{L,j}}) \right)^2 \boldsymbol{I_{n_j}}, \tag{B.36}$$

where $\boldsymbol{I_{n_p}}$ and $\boldsymbol{I_{n_j}}$ are the $n_p \times n_p$ and $n_j \times n_j$ identity matrices.

B.4 Calibration of background leakage parameters

In chapter 2, we propose several background leakage models, denoted $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$. Model M0 corresponds to the state-of-the-art one, as initially proposed by [55]. Models $\{\mathrm{M1}, \ldots, \mathrm{M3}\}$ are new gradually refined versions of M0.

In this section, we propose a method to calibrate the leakage parameters $\{\alpha_L, \beta_L\}$ for each model, considering a single leaky pipe of length ℓ, and supposing that the needed measured data is available. This method is used in section 2.3.1.4 of chapter 2, to explore the potential of models $\{\mathrm{M0}, \ldots, \mathrm{M3}\}$ to adjust model Ref. A method to calibrate the leakage parameters at the scale of a whole water distribution network is introduced in chapter 5.

B.4.1 State-of-the-art background leakage model (M0)

To calibrate the leakage parameter β_L associated to model M0, denoted hereafter β_L^{M0}, we suppose that the pressure-heads at pipe extremities and the flow rate at pipe end have been measured; we denote them respectively p_0^{meas}, p_ℓ^{meas} and q_ℓ^{meas}. Then, denoting u_0 and u_ℓ the elevations (in m) at the start and end of the pipe, and knowing that, at any $x \in [0, \ell]$, the pressure-head p_x, the elevation u_x and the head h_x are linked by the relation:

$$h_x = p_x + u_x, \tag{B.37}$$

we have in particular:

$$h_{\mathit{0}}^{meas} = p_{\mathit{0}}^{meas} + u_{\mathit{0}} \qquad \text{and} \qquad h_{\ell}^{meas} = p_{\ell}^{meas} + u_{\ell}. \tag{B.38}$$

Next, we suppose that the sources of head-loss other than the friction one are negligible. Thus, we can calculate the friction head-loss over the full pipe as:

$$\xi_{f}^{meas} = h_{\ell}^{meas} - h_{\mathit{0}}^{meas}, \tag{B.39}$$

Also, we can obtain the flow at the middle of the pipe, $q_{\mathit{0.5}}^{\mathrm{M0}}$, by solving the equation of conservation of energy:

$$\xi_{f}^{meas} - f\,\ell\, q_{\mathit{0.5}}^{\mathrm{M0}}\, |q_{\mathit{0.5}}^{\mathrm{M0}}|^{\gamma_{HW}-1} = 0 \tag{B.40}$$

with, for example, the nonlinear least-squares solver `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html` from the SciPy library [160]. Next, denoting $q_{LL}^{\mathrm{M0}} = \mathrm{q}_{\mathrm{LL}}^{\mathrm{M0}}(x)$, the conservation of the mass at $x = \ell$ leads to:

$$q_{\mathit{0.5}}^{\mathrm{M0}} - \frac{\ell}{2} q_{LL}^{\mathrm{M0}} - q_{\ell}^{meas} = 0, \tag{B.41}$$

and permits to calculate q_{LL}^{M0} as:

$$q_{LL}^{\mathrm{M0}} = \frac{2\left(q_{\mathit{0.5}}^{\mathrm{M0}} - q_{\ell}^{meas}\right)}{\ell}. \tag{B.42}$$

Finally, defining, from $p_{\mathit{0}}^{meas} = h_{\mathit{0}}^{meas} - u_{\mathit{0}}$ and $p_{\ell}^{meas} = h_{\ell}^{meas} - u_{\ell}$, the average measured pressure-head in the pipe as

$$\widetilde{p^{\,meas}} = \left(p_{\mathit{0}}^{meas} + p_{\ell}^{meas}\right)/2, \tag{B.43}$$

and supposing that $\widetilde{p^{\,meas}} > 0$, we obtain, $\forall \alpha_L \in \,]0.5, 2.5]$:

$$\beta_{L}^{\mathrm{M0}} = \frac{q_{LL}^{\mathrm{M0}}}{\left(\widetilde{p^{\,meas}}\right)^{\alpha_L}}. \tag{B.44}$$

Notes:

- We choose here to use q_{ℓ}^{meas} to calibrate β_L; but using q_{0}^{meas} in place of q_{ℓ}^{meas} would give the same calibrated β_L^{M0}, because $\mathrm{q}^{\mathrm{M0}}(x)$ is symmetrical about $x = \ell/2$.
- It is not possible to calibrate the parameter α_L because q_{LL}^{M0} is invariant along the pipe.

B.4.2 Lineic leakage outflow rate invariant along the pipe but affine flow rate (M1)

To calibrate the leakage parameter β_L, we suppose, as for model M0, that the pressure-heads at pipe extremities and the flow rate at pipe end have been measured, and we denote them respectively p_{0}^{meas}, p_{ℓ}^{meas} and q_{ℓ}^{meas}. Thus, we also have $\xi_{f}^{meas} = p_{\ell}^{meas} + u_{\ell} - p_{\mathit{0}}^{meas} - u_{\mathit{0}}$, and we can compute q_{LL}^{M1} by solving the equation of conservation of energy along the full pipe:

$$\xi_{f}^{meas} - \frac{f}{(\gamma_{HW}+1)\; q_{LL}^{\mathrm{M1}}}\left(|\mathrm{q}_{0}^{\mathrm{M1}}(q_{LL}^{\mathrm{M1}})|^{\gamma_{HW}+1} - |q_{\ell}^{meas}|^{\gamma_{HW}+1}\right) = 0, \tag{B.45}$$

where

$$\mathrm{q}_0^{\mathrm{M1}}(q_{LL}^{\mathrm{M1}}) = q_\ell^{meas} + \ell\, q_{LL}^{\mathrm{M1}}. \tag{B.46}$$

Equation (B.45) can be solved with, for example, the nonlinear least-squares solver `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html` from the SciPy library [160]. Finally, as for model M0, we obtain, $\forall \alpha_L \in\,]0.5, 2.5]$:

$$\beta_L^{\mathrm{M1}} = \frac{q_{LL}^{\mathrm{M1}}}{\left(\widetilde{p^{\,meas}}\right)^{\alpha_L}}, \tag{B.47}$$

where $\widetilde{p^{\,meas}}$ is defined by eq. (B.43).

Remark: As for model M0, it is not possible to calibrate the parameter α_L, because q_{LL}^{M1} is invariant along the pipe.

B.4.3 Affine lineic leakage outflow rate (M2)

For model M2, we suppose that the pressure-heads and the flow rates at both extremities of the pipe have been measured, and we denote them respectively $\{p_0^{meas}, p_\ell^{meas}\}$ and $\{q_0^{meas}, q_\ell^{meas}\}$. Then, to calibrate the parameters α_L and β_L, we first reformulate eqs. (2.3) and (2.27) to (2.29) as functions of $\{\alpha_L, \beta_L\}$, such that:

$$\mathrm{q}_{\mathrm{LL}0}(\alpha_L,\beta_L) = \beta_L\left([p_0^{meas}]^+\right)^{\alpha_L} \quad \text{and} \quad \mathrm{q}_{\mathrm{LL}\ell}(\alpha_L,\beta_L) = \beta_L\left([p_\ell^{meas}]^+\right)^{\alpha_L}, \tag{B.48}$$

$$\mathrm{q}_{0.5}^{\mathrm{M2}}(\alpha_L,\beta_L) = q_\ell^{meas} + \frac{\ell}{8}\Big(\mathrm{q}_{\mathrm{LL}0}(\alpha_L,\beta_L) + 3\,\mathrm{q}_{\mathrm{LL}\ell}(\alpha_L,\beta_L)\Big), \tag{B.49}$$

$$\mathrm{q}_0^{\mathrm{M2}}(\alpha_L,\beta_L) = \mathrm{q}_{0.5}^{\mathrm{M2}}(\alpha_L,\beta_L) + \frac{\ell}{8}\Big(3\,\mathrm{q}_{\mathrm{LL}0}(\alpha_L,\beta_L) + \mathrm{q}_{\mathrm{LL}\ell}(\alpha_L,\beta_L)\Big), \tag{B.50}$$

$$\varphi_{\ell/2}^{\mathrm{M2}}(\alpha_L,\beta_L) = f\mathrm{q}_{0.5}^{\mathrm{M2}}(\alpha_L,\beta_L)|\mathrm{q}_{0.5}^{\mathrm{M2}}(\alpha_L,\beta_L)|^{\gamma_{HW}-1}, \tag{B.51}$$

and

$$\xi_{\mathrm{f}}^{\mathrm{M2}}(\alpha_L,\beta_L) = \frac{\ell}{6}\left(\varphi_0^{meas} + \varphi_{\ell/2}^{\mathrm{M2}}(\alpha_L,\beta_L) + \varphi_\ell^{meas}\right), \tag{B.52}$$

where $\varphi_0^{meas} = f q_0^{meas}|q_0^{meas}|^{\gamma_{HW}-1}$ and $\varphi_\ell^{meas} = f q_\ell^{meas}|q_\ell^{meas}|^{\gamma_{HW}-1}$. Then, we obtain the calibrated parameters $\left\{\alpha_L^{\mathrm{M2}}, \beta_L^{\mathrm{M2}}\right\}$ by solving the system:

$$\begin{cases} \xi_f^{meas} - \xi^{\mathrm{M2}}(\alpha_L^{\mathrm{M2}}, \beta_L^{\mathrm{M2}}) = 0 & \text{(B.53a)} \\ q_0^{meas} - \mathrm{q}_0^{\mathrm{M2}}(\alpha_L^{\mathrm{M2}}, \beta_L^{\mathrm{M2}}) = 0 & \text{(B.53b)} \end{cases}$$

with, for example, the nonlinear least-squares solver `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html` from the SciPy library [160].

B.4.4 Pseudo-quadratic lineic leakage outflow rate (M3)

For model M3, we suppose, as for model M2, that the pressure-heads and the flow rates at both extremities of the pipe have been measured, and we denote them p_0^{meas}, p_ℓ^{meas}, q_0^{meas} and q_ℓ^{meas}. Then, to calibrate parameters α_L and β_L, we reuse the functions $\mathrm{q}_{\mathrm{LL}0}(\alpha_L, \beta_L)$ and $\mathrm{q}_{\mathrm{LL}\ell}(\alpha_L, \beta_L)$ defined by eq. (B.48), and we reformulate eqs. (2.3), (2.31), (2.33) and (2.34) as functions of $\{\alpha_L, \beta_L\}$, such that:

$$\widetilde{\mathrm{q}_{\mathrm{LL}}}(\alpha_L, \beta_L) = \mathrm{q}_{\mathrm{LL}}\Big(\widetilde{p^{meas}}\Big) = \beta_L \left(\left[\frac{p_0^{meas} + p_\ell^{meas}}{2} \right]^+ \right)^{\alpha_L}, \tag{B.54}$$

$$\mathrm{q}_{0.5}^{\mathrm{M3}}(\alpha_L, \beta_L) = q_\ell^{meas} + \frac{\ell}{24}\Big(-\mathrm{q}_{\mathrm{LL}0}(\alpha_L, \beta_L) + 8\,\widetilde{\mathrm{q}_{\mathrm{LL}}}(\alpha_L, \beta_L) + 5\,\mathrm{q}_{\mathrm{LL}\ell}(\alpha_L, \beta_L) \Big), \tag{B.55}$$

$$\mathrm{q}_{0}^{\mathrm{M3}}(\alpha_L, \beta_L) = \mathrm{q}_{0.5}^{\mathrm{M3}}(\alpha_L, \beta_L) + \frac{\ell}{24}\Big(5\,\mathrm{q}_{\mathrm{LL}0}(\alpha_L, \beta_L) + 8\,\widetilde{\mathrm{q}_{\mathrm{LL}}}(\alpha_L, \beta_L) - \mathrm{q}_{\mathrm{LL}\ell}(\alpha_L, \beta_L) \Big), \tag{B.56}$$

$$\varphi_{\ell/2}^{\mathrm{M3}}(\alpha_L, \beta_L) = f \mathrm{q}_{0.5}^{\mathrm{M3}}(\alpha_L, \beta_L) \left| \mathrm{q}_{0.5}^{\mathrm{M3}}(\alpha_L, \beta_L) \right|^{\gamma_{HW}-1}, \tag{B.57}$$

and

$$\xi_{\mathrm{f}}^{\mathrm{M3}}(\alpha_L, \beta_L) = \frac{\ell}{6} \left(\varphi_0^{meas} + \varphi_{\ell/2}(\alpha_L, \beta_L) + \varphi_\ell^{meas} \right). \tag{B.58}$$

Finally, we compute the calibrated parameters $\left\{ \alpha_L^{\mathrm{M3}}, \beta_L^{\mathrm{M3}} \right\}$ by solving the system:

$$\begin{cases} \xi_f^{meas} - \xi_{\mathrm{f}}^{\mathrm{M3}}(\alpha_L^{\mathrm{M3}}, \beta_L^{\mathrm{M3}}) = 0 & \text{(B.59a)} \\ q_0^{meas} - \mathrm{q}_0^{\mathrm{M3}}(\alpha_L^{\mathrm{M3}}, \beta_L^{\mathrm{M3}}) = 0 & \text{(B.59b)} \end{cases}$$

with, for example, the nonlinear least-squares solver `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.least_squares.html` from the SciPy library [160].

<u>Remark</u>: writing $\widetilde{\mathrm{q}_{\mathrm{LL}}}(\alpha_L, \beta_L) = \mathrm{q}_{\mathrm{LL}}\Big(\widetilde{p^{meas}}\Big)$ in eq. (B.54), we then suppose that $p_{\ell/2}^{meas} = \widetilde{p^{meas}}$. Making this assumption is the only way to calibrate the leakage parameters for model M3 if $p_{\ell/2}^{meas}$ has not been measured.

Appendix C

Supplementary material of chapter 3: High-lying nodes and partly-supplied pipes

C.1 Analytic calculation of the real root(s) of quadratic and cubic polynomials

C.1.1 Quadratic polynomial

Denote

$$\mathrm{P}(x) = a_2\, x^2 + a_1\, x + a_0 \tag{C.1}$$

a quadratic polynomial with $a_1 < 0$. The real roots of eq. (C.1), if they exist, can be calculated analytically using the quadratic and Viète's formulas.

To do so, first calculate the discriminant of eq. (C.1), as:

$$\Delta = {a_1}^2 - 4\, a_2\, a_0. \tag{C.2}$$

Then, compute the real roots of eq. (C.1) according to the sign of Δ:

- if $\Delta > 0$, then the two real roots are:

$$r_1^{\mathrm{M2}} = \frac{q}{a_2} \qquad \text{and} \qquad r_2^{\mathrm{M2}} = \begin{cases} \dfrac{a_0}{q} & \text{if } q \neq 0, \\ r_1^{\mathrm{M2}} & \text{if } q = 0, \end{cases} \tag{C.3}$$

  where

$$q = \frac{1}{2}\left(-a_1 + \sqrt{\Delta}\right). \tag{C.4}$$

  Equations (C.3) and (C.4) are an enhanced version of the usual quadratic formula, and lead to more accurate numerical results [166, p. 1479].

- If $\Delta = 0$, then there is only one real root:

$$r^{\mathrm{M2}} = -\frac{a_1}{2\, a_2}. \tag{C.5}$$

- If $\Delta < 0$, then eq. (C.1) has no real root.

C.1.2 Cubic polynomial

Denote

$$\mathrm{P}(x) = a_3\, x^3 + a_2\, x^2 + a_1\, x + a_0, \tag{C.6}$$

a cubic polynomial. The real roots of eq. (C.1), if they exist, can be calculated analytically using the Cardano's formula [166, p. 364-365].

To do so, first define $\{\widetilde{a_i} \mid i \in \{0, \ldots, 3\}\}$ as:

$$\widetilde{a_i} = \frac{a_i}{a_3}. \tag{C.7}$$

Then, the real roots of eq. (C.6) are also the real roots of:

$$\widetilde{\mathrm{P}}(x) = x^3 + \widetilde{a_2}\, x^2 + \widetilde{a_1}\, x + \widetilde{a_0}. \tag{C.8}$$

Define p and q as:

$$p = \frac{9\,\widetilde{a_1}\,\widetilde{a_2} - 27\,\widetilde{a_0} - 2\,\widetilde{a_2}^3}{27} \qquad \text{and} \qquad q = \frac{3\,\widetilde{a_1} - \widetilde{a_2}^2}{3}. \tag{C.9}$$

and y as:

$$y = \sqrt{\frac{3}{4\,|p|}}. \tag{C.10}$$

Equation (C.8) can be rewritten as a function of y:

$$\widetilde{\mathrm{P}}(y) = 4\, y^3 + 3\ \mathrm{sgn}(p)\, y = \frac{1}{2}\, q \left(\frac{3}{|p|}\right)^{3/2}, \tag{C.11}$$

where sgn(...) is the sign function.

Define C as:

$$C = \frac{1}{2}\, q \left(\frac{3}{|p|}\right)^{3/2}. \tag{C.12}$$

The real root(s) of eq. (C.11), if they exist, can then be calculated according to the sign of p and q (eq. (C.9)) and the value of C:

- if $p > 0$, then eq. (C.11) has only one real root:

$$\widetilde{r^{\mathrm{M3}}} = \sinh\!\left(\frac{\operatorname{arcsinh}(C)}{3}\right); \tag{C.13}$$

- if $p < 0$, then:
  - if $C \leq -1$, then eq. (C.11) has only one real root:

$$\widetilde{r^{\mathrm{M3}}} = -\cosh\!\left(\frac{\operatorname{arccosh}(|C|)}{3}\right); \tag{C.14}$$

  - otherwise, if $C \geq 1$, then eq. (C.11) has only one real root:

$$\widetilde{r^{\mathrm{M3}}} = \cosh\!\left(\frac{\operatorname{arccosh}(C)}{3}\right); \tag{C.15}$$

- otherwise (i.e., $-1 < C < 1$), then, writing

$$z = \frac{\arccos(C)}{3}, \tag{C.16}$$

eq. (C.11) has three real roots:

$$\widetilde{r_1^{\mathrm{M3}}} = \cos\left(z - \frac{2\,\pi}{3}\right), \quad \widetilde{r_2^{\mathrm{M3}}} = \cos(z) \quad \text{and} \quad \widetilde{r_3^{\mathrm{M3}}} = \cos\left(z + \frac{2\,\pi}{3}\right); \tag{C.17}$$

- if $p = 0$, then:
  - if $q \geq 0$, then eq. (C.11) has only one real root:

$$\widetilde{r^{\mathrm{M3}}} = q^{1/3}; \tag{C.18}$$

  - if $q < 0$, then eq. (C.11) has no real root.

Finally, from the real roots of eq. (C.11), the real roots of eq. (C.6) can be deduced as follows:

- if eq. (C.11) has no real root, then eq. (C.6) does not have neither;
- if eq. (C.11) has only one real root, then eq. (C.6) has only one real root as well:

$$r^{\mathrm{M3}} = 2\sqrt{\frac{|p|}{3}}\widetilde{r^{\mathrm{M3}}} - \frac{\widetilde{a_2}}{3}; \tag{C.19}$$

- if eq. (C.11) has three real roots $\left\{\widetilde{r_i^{\mathrm{M3}}} \mid i \in \{1,2,3\}\right\}$, then eq. (C.6) has also three real roots:

$$r_i^{\mathrm{M3}} = 2\sqrt{\frac{|p|}{3}}\widetilde{r_i^{\mathrm{M3}}} - \frac{\widetilde{a_2}}{3}. \tag{C.20}$$

Appendix D

Supplementary material of chapter 4: Inertia phenomena

D.1 Extended-period simulator (EPS)

Knowing the heads at tanks $\boldsymbol{h}_{\boldsymbol{t}}^{(0)}$ at start time t_0, and the users' demands $\boldsymbol{d}^{(n)}$ at every time t_n, $n = 0, 1, 2, \ldots$, the extended-period simulator successively computes, at each time t_n, from users' demands $\boldsymbol{d}^{(n)} = \mathbf{d}(t_n)$ and heads at tanks $\boldsymbol{h}_{\boldsymbol{t}}^{(n)} \approxeq \mathbf{h}_{\mathbf{t}}(t_n)$:

1. the flow rates at the middles of the pipes $\boldsymbol{q}_{\boldsymbol{0.5}}^{(n)}$ and the heads at the junctions $\boldsymbol{h}^{(n)}$ that satisfy the equation of equilibrium:

$$\begin{pmatrix} \xi_{\mathbf{f}}(\boldsymbol{q}_{\boldsymbol{0.5}}^{(n)}) - \boldsymbol{A}^T\boldsymbol{h}^{(n)} - \boldsymbol{A}_{\boldsymbol{t}}^T\boldsymbol{h}_{\boldsymbol{t}}^{(n)} - \boldsymbol{A}_{\boldsymbol{r}}^T\boldsymbol{h}_{\boldsymbol{r}} \\ -\boldsymbol{A}\,\boldsymbol{q}_{\boldsymbol{0.5}}^{(n)} - \mathbf{c}(\boldsymbol{h}^{(n)}) - \boldsymbol{q}_{\boldsymbol{L}} \end{pmatrix} = \boldsymbol{0}, \tag{D.1}$$

2. and the new approximated heads at tanks $\boldsymbol{h}_{\boldsymbol{t}}^{(n+1)}$ at t_{n+1} as:

$$\boldsymbol{h}_{\boldsymbol{t}}^{(n+1)} \approx \mathbf{h}_{\mathbf{t}}(t_{n+1}) = \boldsymbol{h}_{\boldsymbol{t}}^{(n)} - \frac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3} \int_{t_n}^{t_{n+1}} \left(\boldsymbol{A}_{\boldsymbol{t}}\, \mathbf{q}_{\mathbf{0.5}}(t) + \boldsymbol{q}_{\boldsymbol{Lt}}\right) \mathrm{d}t. \tag{D.2}$$

The solving of eq. (D.1) can be done using the Newton's method of section 1.2.4, replacing:

- $\boldsymbol{q}$, $\boldsymbol{h}$, $\boldsymbol{h}_{\boldsymbol{t}}$ and $\boldsymbol{d}$ by respectively $\boldsymbol{q}_{\boldsymbol{0.5}}^{(n)}$, $\boldsymbol{h}^{(n)}$, $\boldsymbol{h}_{\boldsymbol{t}}^{(n)}$ and $\boldsymbol{d}^{(n)}$,
- $\xi_{\mathbf{f}}(\boldsymbol{q})$ by $\xi_{\mathbf{f}}(\boldsymbol{q}_{\boldsymbol{0.5}}^{(n)})$, where, for any pipe k, $\xi_{\mathrm{f},k}(q_{0.5,k}^{(n)})$ is defined by eq. (4.4),
- and $\mathbf{c}(\boldsymbol{h})$ by $\mathbf{c}(\boldsymbol{h}^{(n)}) + \boldsymbol{q}_{\boldsymbol{L}}$.

To compute $\boldsymbol{h}_{\boldsymbol{t}}^{(n+1)}$, $n = 0, 1, 2, \ldots$, we use the Heun's method for $n = 0$, and the trapezoidal rule for $n = 1, 2, 3, \ldots$ [18, p. 629 and 643]. The Heun's method for $n = 0$ is composed of two steps:

1. the "predictor step", which consists in computing a first-order approximation of $\mathbf{h}_{\mathbf{t}}(t_1)$ with the Euler method, as:

$$\widetilde{\mathbf{h}_{\mathbf{t}}(t_1)} = \boldsymbol{h}_{\boldsymbol{t}}^{(0)} - \Delta t\, \frac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3} \left(\boldsymbol{A}_{\boldsymbol{t}}\, \boldsymbol{q}_{\boldsymbol{0.5}}^{(0)} + \boldsymbol{q}_{\boldsymbol{Lt}}\right) = \mathbf{h}_{\mathbf{t}}(t_1) + O(\Delta t^2), \tag{D.3}$$

where $\Delta t = t_1 - t_0$;

2. the "corrector step", which consists in computing:
   - $\widetilde{\boldsymbol{q}_{\boldsymbol{0.5}}^{(1)}}$ solution of eq. (D.1) after replacing $\boldsymbol{h}^{(n)}$ by $\widetilde{\mathbf{h_t}(t_1)}$,
   - and $\boldsymbol{h_t}^{(1)}$ as a second-order approximation of $\mathbf{h_t}(t_1)$ with the trapezoidal rule:

$$\boldsymbol{h_t}^{(1)} = \boldsymbol{h_t}^{(0)} - \frac{\Delta t}{2} \frac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3} \left(\boldsymbol{A_t} \left(\boldsymbol{q}_{\boldsymbol{0.5}}^{(0)} + \widetilde{\boldsymbol{q}_{\boldsymbol{0.5}}^{(1)}} \right) + 2\, \boldsymbol{q_{Lt}} \right) = \mathbf{h_t}(t_1) + O(\Delta t^3). \tag{D.4}$$

For $n = 1, 2, 3, \ldots$, the trapezoidal rule computes a second order approximation of $\mathbf{h_t}(t_n)$, as:

$$\boldsymbol{h_t}^{(n+1)} = \boldsymbol{h_t}^{(n)} - \frac{\Delta t}{2} \frac{\boldsymbol{\mathcal{I}_t}^{-1}}{10^3} \left(\boldsymbol{A_t} \left(\boldsymbol{q}_{\boldsymbol{0.5}}^{(n-1)} + \boldsymbol{q}_{\boldsymbol{0.5}}^{(n)} \right) + 2\, \boldsymbol{q_{Lt}} \right) = \mathbf{h_t}(t_{n+1}) + O(\Delta t^3). \tag{D.5}$$

INRAE

Camille CHAMBON

Modeling of background leakages and inertia phenomena in water distribution networks

HESAM UNIVERSITÉ

Résumé : L'objectif de cette thèse est de modéliser des réseaux de distribution d'eau potable sujets à des fuites diffuses et à des phénomènes d'inertie. Les pressions dans les réseaux doivent être suffisantes pour que tous les consommateurs aient de l'eau avec une bonne qualité de service. Cependant, pour limiter les fuites diffuses, ces pressions ne doivent pas être excessives. Un élément clé pour résoudre ce problème d'optimisation est de modéliser avec précision la dépendance des fuites diffuses à la pression. Nous proposons donc dans cette thèse plusieurs nouveaux modèles de fuites diffuses qui prennent en compte le gradient de pression le long des conduites. Nous montrons, à travers plusieurs expérimentations numériques sur des réseaux théoriques et réels, la supériorité de nos modèles par rapport à ceux de l'état de l'art. Notre approche permet également d'identifier les points hauts isolés en cas de pression insuffisante, et les parties les plus fuyardes des tronçons. Après validation de nos modèles en régime permanent, nous explorons la faisabilité de les intégrer dans un nouveau simulateur transitoire-lent qui décrit les phénomènes d'inertie. Ces phénomènes apparaissent par exemple lorsque les demandes des utilisateurs ou les hauteurs des réservoirs varient rapidement, des pompes sont démarrées, ou quand des vannes s'ouvrent ou se ferment en moins d'une minute. Nous observons des différences significatives entre les résultats de notre modèle transitoire-lent et ceux d'un simulateur pseudo-transitoire qui néglige les phénomènes d'inertie. Nous mettons aussi en évidence un accroissement important de la raideur du système à résoudre lorsque des fuites diffuses dépendant de la pression sont modélisées. Enfin, nous introduisons le calage des paramètres de fuite à partir des données expérimentales collectées lors du projet de Renouvellement Orienté des Conduites (ROC). Tous nos développements sont parties intégrantes d'un cadriciel collaboratif dédié à la modélisation des réseaux d'eau.

Mots clés : réseau de distribution d'eau potable, fuite diffuse, modélisation dépendant de la pression, phénomène d'inertie, analyse hydraulique

Abstract: The purpose of this thesis is to model water distribution networks (WDNs) subject to background leakage outflows and inertia phenomena. Pressures in WDNs must be high enough for all consumers to have water with a good quality of service, but low enough to limit background leakages. A key element to solve this optimization problem is to model accurately the dependence of background leakages to pressures. For this purpose, we propose in this thesis several new background leakage models that take into account the gradient of pressure along the pipes. We show, through numerical experimentation on both theoretical and real networks, the superiority of our models when compared to the state-of-the-art ones. Also, our approach allows the simulation of high-lying nodes in case of insufficient pressures, and the identification of the leakiest parts of the pipes. Once our models are validated in steady-state, we explore the feasibility of integrating them into a new rigid water column (RWC) simulator that takes into account inertia phenomena. These phenomena appear, e.g., when users' demands or heads at tanks vary quickly, pumps are started, or valves are opening or closing in less than a minute. We observe significant differences between the results of our RWC and the ones of an extended-period simulator (EPS) that neglects inertia phenomena. We also highlight the increase of stiffness due to the integration of pressure-dependent outflows in the slow-transient equations. Finally, we initiate the calibration of the leakage parameters from the experimental data collected during the Oriented Renewal of Pipes (ROC) project. All our developments are integrated into a collaborative framework dedicated to WDNs modeling.

Keywords: water distribution network (WDN), background leakage, pressure-dependent model (PDM), inertia phenomena, hydraulic analysis