Ana Margarida Barradinhas da Silva

Licenciatura em Ciências de Engenharia do Ambiente

Heat Recovery from Wastewater: Numerical Modelling of Sewer Systems

Dissertação para obtenção do Grau de Mestre em
Engenharia do Ambiente, perfil Engenharia Sanitária

Orientadora:
Professora Doutora Leonor Monteiro do Amaral
Professora Auxiliar, FCT/UNL

Jurí:

Presidente e Arguente:	Prof. Doutor Pedro Manuel Hora Santos Coelho
Vogal:	Prof. Doutor António Pedro de Coimbra Macedo Mano
Vogal:	Prof. Doutora Leonor Miranda Monteiro do Amaral

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Outubro 2012

Ana Margarida Barradinhas da Silva

Licenciatura em Ciências de Engenharia do Ambiente

Heat Recovery from Wastewater: Numerical Modelling of Sewer Systems

Dissertação para obtenção do Grau de Mestre em
Engenharia do Ambiente, perfil Engenharia Sanitária

Orientadora:
Professora Doutora Leonor Monteiro do Amaral
Professora Auxiliar, FCT/UNL

Jurí:

Presidente e Arguente:	Prof. Doutor Pedro Manuel Hora Santos Coelho
Vogal:	Prof. Doutor António Pedro de Coimbra Macedo Mano
Vogal:	Prof. Doutora Leonor Miranda Monteiro do Amaral

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Outubro 2012

Heat Recovery from Wastewater: Numerical Modelling of Sewer Systems

Acknowledgments/ Agradecimentos

I would like to thank to Jorge Maxil (TU Delft), Dr Jan Hofman (KWR), Dr Mirjam Blokker (KWR) and Dr Luuk Rietveld (TU Delft). I am very grateful to Dr Bas Wols (KWR) to support me.

To all my friends in the Netherlands thank you. Nevertheless some of them should be mentioned due to their true friendship and support:

Maria Lascas and Bruno Gracio, to be my second family in the Netherlands and to treat me like their younger sister with the right to have a special room in their home.

To Diana Brandão and Maria Lousada Ferreira for integrate me in TU Delft, Delft life and for their true friendship.

To Shirin Malekpour and Eveline van Amstel for their friendship and encouragement every day.

Aos professores da FCT/UNL que contribuíram para a minha formação.

À Professora Doutora Leonor Amaral pela sua amizade, apoio, motivação, orientação e por tornar este projecto possível.

A todos os meus amigos que durante estes cinco anos de Faculdade me apoiaram e que contribuíram directa ou indirectamente para a realização deste trabalho. Algumas destas pessoas devem ser mencionadas:

À Vanessa Tavares, por toda a amizade demonstrada e por ter sido sempre uma presença constante.

À Diana Garcia, grande amiga e companheira de casa que muito me aturou.

À Susana Inocêncio, companheira de Guerra, que muito me ajudou e me fez não desesperar.

À Marta Alves, Ana Rodrigues, Sandra Gonçalves, Mariana Marques e Luís Ramos por todo o apoio e amizade. E claro, ao João Martins e João Wang pelos conselhos ao longo do curso e amizade.

À Marta Brito, por todo o carinho e interesse que sempre demonstrou ao longo destes anos.

À Helena Graça, Catarina Cravo, Marta Ramalho, Mariana Soares, Inês Crujo, Joana Correia, Marlene Soares, Miguel Fitas, Miguel Jeremias, Hugo Medeiros, Miguel Curado, João Oliveira, Luís Cabecinha e João Santos pela amizade e grande carinho.

À minha gata Nabiça pela companhia neste últimos anos, nunca me deixando estudar sozinha.

Ao grupo de dança contemporânea *Modernus*, à casa Largo do Andaluz, à República Manjerigata e à Casinha da Vanessa por me acolherem.

Por fim, o meu maior agradecimento é para aqueles que mais amo. Aos meus pais, irmão e namorado. Aos meus pais por me apoiarem sempre e fazerem do impossível possível. Ao meu namorado pela dedicação e paciência neste anos de Faculdade e nomeadamente no período da Tese. Dedico assim este trabalho aos meus pais, André e à memória dos meus avós.

“Plan the work and work the plan”.

Resumo

Este trabalho enquadra-se num projecto desenvolvido numa colaboração entre a Universidade Técnica de Delft (TU Delft) com a empresa Waternet e KWR. Pretende-se estudar a viabilidade do aproveitamento de energia térmica associada as águas residuais e assim reduzir as emissões de dióxido de carbono (CO_2) associadas ao sector energético. O presente trabalho parte de um modelo computacional previamente desenvolvido que simula recuperação de calor proveniente de águas residuais para casos de caudal e temperatura constante.

Um dos objectivos pretende simular uma descarga de águas residuais. Para tal e por forma a torná-los variáveis, foi adicionada uma função Gaussiana às condições de fronteira da temperatura da água e caudal. Como segundo objectivo, pretende-se determinar a importância dos termos presentes na equação de balanço de calor na água e na equação de balanço de calor no ar. Para isso, introduziram-se coeficientes binários por cada termo destas equações e processaram-se os sistemas resultantes das combinações possíveis.

Simulou-se com sucesso uma descarga principal para a situação variável, as previsões numéricas para a temperatura da água e caudal foram apresentadas. Quanto à avaliação dos termos nas duas equações em questão, conclui-se que os termos fluxo de calor tubo – água ($\dot{q}_{Rw}$) e fluxo de calor tubo – ar ($\dot{q}_{Rl}$) apresentam ser fulcrais no balaço de calor na água e ar, respectivamente. Com o intuito de reduzir oscilações indesejadas o *dt* foi reduzido, verificando-se uma diminuição das oscilações, bem como uma variação na altura de água na tubagem que não tinha sido verificada até agora.

Este trabalho culmina assim com uma secção de desenvolvimentos futuros por forma a melhorar os resultados obtidos. Apesar do estado actual destas rotinas não permitir uma avaliação precisa das trocas de calor em tubagens, os resultados obtidos são bastante promissores. Provando-se que a modelação numérica de recuperação de calor contribuirá para o desenvolvimento do projecto principal.

Palavras-chave: Água residual, recuperação de calor, modelação numérica, método Lax-Wendroff, equações de balanço de massa, equações de blanço de calor.

Abstract

This thesis was carried as a collaboration of Delft University of Technology (TU Delft) and the companies Waternet and KWR. The main project aims to study the possibility of thermal energy recovery from wastewater, reducing the carbon dioxide (CO_2) emissions linked to the energy sector. The present work is based on a previous computational model that was developed to simulate heat recovery from wastewater for constant flow rate and temperature of water.

The first goal is to simulate a wastewater discharge. In order to achieve this, a Gaussian function was added to the boundary conditions for water flow rate and water temperature. As a second goal, this work aims to assess the significance of the terms present in the water heat balance and air heat balance equations. Binary coefficients were added in each term of both equations and then all the combinations were computed.

The unsteady situation successfully simulated a main discharge and numerical predictions for water temperature and flow rate are presented. The deviations associated with the modified cases for the two equations suggest that the *heat flux pipe to water* ($\dot{q}_{Rw}$) and *heat flux pipe to air* ($\dot{q}_{Rl}$) terms are crucial for water and air heat balance predictions, respectively. In order to smooth extra oscillations, the time step (*dt*) was reduced and a smaller relative size of oscillations was obtained.

This work concludes with a section of future developments in order to improve the results obtained. Despite of the fact that the current state of these routines does not allow us to accurately assess heat exchanges in pipes, promising results were obtained, proving that numerical modelling of heat recovery will contribute greatly to the development of the main project.

Keywords: Wastewater, heat recovery, numerical modelling, Lax-Wendroff method, mass balance equations, heat balance equations.

Contents

List of figures

List of tables

List of symbols

F	Hydrostatic pressure
F_f	Friction force
F_g	Gravity force
ψ	Ambient air pressure
ϕ	Relative humidity ϕ
k_{VK}	Von Karman constant ($\approx 0{,}4$)
g	Gravitational constant
p_{s0}	Saturation pressure
p_L	Partial pressure of steam vapour
$p_{sat}(T_{pl})$	Saturated vapour pressure at the surface of the at temperature T_{pl}
ζ, ξ	Control the condensation
hfg	Evaporation enthalpy
ρ	Density
ρ_w	Density water
ρ_l	Density air
ρ_p	Density of pipe
c_p	Specific heat capacity
λ_l	Thermal conductivity air
λ_s	Thermal conductivity soil
τ	Wall sheer stress
Ω, P, L_w	Wetted perimeter
z	Bottom elevation
S_0 ,i_b	Bottom slope
$c_{p,w}$	Specific heat capacity water
$c_{p,l}$	Specific heat capacity air
$c_{p,p}$	Specific heat capacity for the pipe
$c_{p,s}$	Specific heat capacity for the soil
ν_w	Kinematic viscosity water
ν_l	Kinematic viscosity air
α_w	Thermal diffusivity water
α_l	Thermal diffusivity air
α_s	Thermal diffusivity soil
α_p	Thermal diffusivity pipe
α_{wl}	Heat transfer water-air interface
α_{rw}	Heat transfer flowing water
α_{rl}	Heat transfer flowing air
α_{kP}	Heat transfer condensation pipe
α_{vP}	Heat transfer condensation water-pipe
K_P	Thermal transmittance pipe
λ, k	Thermal conductivity
λ_w	Thermal conductivity water
Pr	Prandtl number water/air
A	Cross sectional area
A_w	Water cross sectional area
r	Cylindrical coordinate radial direction, ray
θ	Cylindrical coordinate polar angle, angle

S_f, i_f	Friction slope
c_f	Friction constant (of pipe wall)
ρ_p	Density pipe
c_p	Specific heat capacity pipe
δ_s	Influence distance of soil
λ_p	Thermal conductivity pipe
Dx	Spatial grid distance
T	Temperature
∇T	Gradient of temperature
Tair	Air temperature
Tsoil	Soil temperature
T_w	Temperature of water
T_l	Temperature of air
T_{pl}	Pipe temperature for air part
T_{pw}	Pipe temperature for water part
T_{sw}	Soil temperature for water part
T_{sl}	Soil temperature for air part
T_{s0}	Temperature of bottom (steady state)
X	Water vapour fraction/ Fraction of water vapour by dry air
X_{sat}	Saturated loading with steam vapour
j_{vP}	Evaporation or condensation
j_{kP}	Condensation in the pipe (condensation layer)
j_{kL}	Condensation oversaturation or condensation in the air volume due to (under)cooling in air
$\dot{q}'''$	Energy generated or consumed per unit of time and volume
$\dot{q}_{Rw}$, $\dot{q}_{pw}$	Heat flux pipe to water

D	Pipe diameter
D_p	Thickness of pipe
L	Pipe length
Δx	Length of the control volume
A_l	Air cross sectional area
L_t	Length
t, t_n	Time (numerical time)
t_0	Start time
t_L	Duration of the function
A'	Amplitude
x	Position
y	Water depth
Q_l	Air flow rate
$u_{w,c}$	Water velocity at the interface water and air
U_w	Velocity of water
v	Maximum velocity of the water plus propagation of waves
l_t	Transverse mass flow
$\sum r$	Sum of sources or sinks
∇	Del differential operator
$Q_{w,bc}$	Gaussian function added to a constant flow of water
$T_{w,bc}$	Gaussian function added to the temperature in water
$\dot{q}_{kl}$	Heat flux due oversaturation
$\dot{q}_{vP}$	Evaporation or condensation
$\dot{q}_w$	Biochemical activity produced in the wastewater
f_{Rw}	Binary coefficient for heat flux pipe to water

$\dot{q}_{Rl}, \dot{q}_{pl}$	Heat flux pipe to air
$\dot{q}_{sw}$	Heat flux soil to water
$\dot{q}_{sl}$	Heat flux soil to air
$\dot{q}_{wl}$	Heat flux water to air

f_{wl}	Binary coefficient for the heat water to air
f_{Rl}	Binary coefficient for the heat flux pipe to the air
f_{kl}	Binary coefficient for heat flux due oversaturation
f_{vP}	Binary coefficient for evaporation or condensation.

Abbreviators

BC	Boundary conditions
CFL	Courant-Friedrich-Levy
CO_2	Carbon dioxide
GC	General case
GHG	Greenhouse gases
MC	Modified cases
PDE	Partial differential equation
SVE	Saint Venant Equations

Chapter 1

Introduction

1.1 General context

Changes over the weather and climate are an emergent concern due to growing evidence of global warming. Such manifestations have been observed around the world, for instance changes in rainfall that causes floods or droughts and severe heat waves are more frequent. The glaciers are melting and as consequence the sea levels are increasing. These climate oscillations became more common in the past decades (US EPA, 2012).

In the last century, human activities contributed for the rising atmospheric concentration of greenhouse gases (GHG), specifically carbon dioxide (CO_2). The main source of these gases is the burning of fossil fuels to produce energy, though some agricultural practices, such as deforestation, and industrial processes also contribute (US EPA, 2012).

In the Netherlands the same happens, the principal source for GHG emissions is the energy sector. In 2008 the contribution of this sector to the GHG emissions was 83% (PBL, 2010). In order to prevent worst impacts of climate change, the GHG emissions are required to be reduced to at least 80% by 2050 (IPCC, 2007).

The municipality of Amsterdam aims to accomplish by 2025 a 40% reduction in CO_2 emissions when compared to 1999. In agreement with IPCC the target is based on the reduction of 75% of CO_2 emissions by 2040 compared to the same year. In order to perform these goals Amsterdam implemented the Amsterdam Climate Program. In order to achieve these targets, the climate policy is focused in energy savings, specifically in an efficient use of fossil fuels and the production of sustainable energy frameworks (Amsterdam Climate Office, 2009).

Waternet is the company that is responsible for water tap supply, wastewater collection and treatment, and also responsible for the quality of surface water and the water level in

Amsterdam and surrounds (Waternet, 2012). Waternet has a program called *Energy from Water* to meet the target to reduce CO_2 emissions through green energy and producing renewable energy from water (Nauffal, 2011).

Wastewater contains a significant amount of energy, this energy could be recovered to produce heat and warm water through a heat pump and a heat exchanger installed in sewers (Dürrenmatt & Wanner, 2008). This results in heat recovery from wastewater a energy reduction in the Urban water cycle (Maxil & Rietveld, 2011).

This technique of heat recovery form wastewater is presently being used in a major scale in countries like Switzerland and Germany. In Switzerland there is 30 facilities in operation. A heat exchanger of 200 m in length is installed in the sewer system of Zurich and produces heat and warm water for 800 apartments (Dürrenmatt & Wanner, 2008). Over 500 wastewater heat pumps are in operation worldwide with thermal ratings from 10kW to 20MW (Schmid 2009).

Studies made in Switzerland and Germany show that 3% of all buildings could be supplied with heat on the basic of wastewater. Due to ideal source temperatures available, wastewater heat pumps reach high performance figures. On top of that, this installations to recovery heat from wastewater have an outstanding environmental performance (Schmid, 2009).

A study carried out by the Swiss Federal Office of Energy shows that the amount available of energy in wastewater is dependent of the use of water in buildings. The amount of water consumed is increasing in countries with strong economic development and increasing standard of living and falling in industrial nations, as a result of efforts being made concerning the efficient use of water. When planning wastewater energy plants, the long-term development of wastewater quantities must therefore be carefully analysed. On top of that, the local availability of wastewater as a source of energy is limited. Places where wastewater is available, both continuously and in large quantities, are economically interesting, such as hospitals, industry, housing estates, etc., since these are the main drains of local settlements and sewage treatment plants. The amount of energy available in wastewater is high despite these restrictions (Schmid, 2009).

According to the same study, the economic viability of the use of heat from wastewater depends of the prices of traditional sources of energy (oil), system size (heating power requirements) and heat-density. The cost of energy production in wastewater heating installations including amortization could be 0.07$ to 0.22$ per kWh for a considering the oil prices to be 90$/100L (Schmid, 2009).

A computer model of wastewater heat recovery could be very usefully to study the behaviour of this complex system. The heat content in sewage is often known by measuring the flow and temperature *in situ*. Employing a computational model in this situation could minimize complicated measurement campaigns.

1.2 State of art on heat modelling in sewers

A literature review was conducted for modelling approaches in the sewer temperature.

Bischofberger and Seyfied (1984) developed a model to estimate the longitudinal profile of the temperature in a sewer pipe. They documented that water (wastewater) temperature in the sewer is mostly affected by the heat exchange from water and the air duct, water evaporation and heat transfer through the pipe walls, assuming air duct temperature and relative humidity constant values (Dürrenmatt & Gujer, 2006).

Starting with a wastewater temperature model from previous authors, Wanner et al. (2004) developed a mathematical model for wastewater temperature in a sewage system. Adding air temperature and relative air humidity as variables, the model can predict the cooling-down of wastewaters in a sewage system and register the main parameters responsible for it (Wanner et al., 2004; Dürrenmatt & Gujer, 2006).

Other mathematical models were developed, such as Krarti & Kreider (1996), Kurpaska & Slipek (1996) and Hollmuller (2003). The first model can predict air temperature along an air tunnel for any hour of the day. Kurpaska and Slipek (1996) developed a model to predict the temperature and water content at a given time within garden subsoil. Using the concept of moisture diffusion and mass transfer coefficients in forced and natural convection similar to heat and heat transfer coefficients. Hollumuller (2003) developed a model for air/soil heat exchange for constant airflow with harmonic temperature signal as input. These three models aim to calculate variations in air temperature, however evaporation and condensation were not considered.

Edwini-Bonsu and Steffler (2004, 2006a and 2006b) focused on the modeling of atmospheric pressurization in sanitary sewer conduit by employing computational fluid dynamics methods to study odorous-compound emissions, design of ventilation systems and sewer fabric corrosion (Edwini-Bonsu & Steffler, 2004; Edwini-Bonsu & Steffler, 2006a; Edwini-Bonsu & Steffler, 2006b).

Dürrenmatt and Gujer (2006) developed a model based in Wanner et. al. (2004) Besides the temperature of wastewater and air, and relative humidity in the sewers from the previous model, a condensation layer was added allowing for the prediction of condensation of water vapour on the channel wall. Another developed concern about the air flow. In past models air flow was assumed as a constant but here it is calculated by physical models. Dürrenmatt and Gujer (2006) presented in the last model the inclusion of manholes and lateral inflows in the sewage for the simulation of the temperature profile.

Two years later Dürrenmatt and Wanner (2008) developed a program called TEMPEST (*Eawag*, Dübendorf, Switzerland) to calculate the dynamics and longitudinal spatial profiles of the wastewater temperature in sewage systems. The model is established with the heat balance in sewers from Dürrenmatt and Gujer (2006) in Dürrenmatt & Wanner (2008).

In a similar context, recent approaches were made in order to directly regenerate the hot water supply through greywater heat recovery instead of blackwater. These types of systems were used to supply hot water for showers (Meggers & Leibundgut, 2011; Wong et al., 2010; Liu et al., 2010).

1.3 Thesis objective

The main topic of this project is modelling temperature exchange between sewer pipes, wasterwater and air.

The work developed in this thesis uses a MATLAB® code developed by Bas Wols model at KWR Water Research Institute (Nieuwegein, The Netherlands), based on the model from Dürrenmatt and Wanner (2008), and this work adapts some of its characteristics. Two goals could be highlighted. The first is the implementation of a case of unsteady conditions for the water temperature and water flow rate in order to simulate a discharge, since the model was previously developed for steady flow conditions. As a second goal, this work aims to assess the significance of the terms present in the equations that describe the physics of the model: the water and air heat balance equations.

This thesis also includes the derivation of the model equations from the general equations (Saint Venant equations and heat conduction equations).

1.4 Thesis organization

This thesis is divided in six chapters:

Chapter 1

This chapter explains the general context and objectives of this work. Closes with a short summary of the document outline.

Chapter 2

Contains the theoretical background. This chapter starts by introducing the Saint Venant equations and heat conduction equation. Subsequently, the model equations and the derivation process from the general equations to the model equations are presented. Finally, the chapter closes with an introduction to the numerical solver Lax Wendroff.

Chapter 3

Describes all the methodology followed in this thesis in order to reach the main objectives. The first section describes the algorithm. To implement the unsteady conditions a Gaussian function was added to the boundary conditions of water temperature and water flow rate. To determine

the contribution of each term of the water heat balance equation and air heat balance equation it was introduced binary coefficient for each heat flux for both equations and all combination of terms were computed.

Chapter 4

The fourth chapter encloses the results obtained in this work. This chapter is divided in results for steady situation and results for unsteady situation. In both section there are the results from the contribution of each term of the water heat balance equation and air heat balance equation, *i.e.* deviation measure between general case and each modified case. Lastly, an extra situation was added in order to improve the results for the unsteady situation.

Chapter 5

The fifth chapter contemplates a discussion for results obtained.

Chapter 6

The last chapter includes the most important conclusions and suggestions for future developments associated to the employed model and optimization techniques.

Chapter 2

Background

In fluid mechanics and hydraulics, the basic principles are the equations of continuity or conservation of mass, equation of momentum and conservation of energy (Fox et al., 2008). The present work is focused in the continuity equations and momentum equations known as Saint Venant Equations. In addition to these two equations a heat balance is required to implement the heat exchange over the pipes.

The heat process inside the pipe is described as the principal of a balance. However, for the heat transfer in the pipe and soil the heat exchanger is based in the heat conduction equation.

The chapter begins with a presentation of the general equations, in Section 2.1 the model equations are available in Section 2.2, the next section 2.3 is about the boundary and initial conditions. Finally the last Section 2.4 is centred in a numerical solver called Lax-Wendroff.

2.1 General equations

An introduction to Saint Venant equations and heat conduction equation are presented.

2.1.1 Saint Venant equations

The continuity and momentum equations are developed for one-dimensional unsteady open channel flows using the Saint Venant Equations (SVE). It considers open channels in which liquid flows with a free surface.

The SVE has some assumptions and these are valid for any channel cross-sectional shape. These are: (1) the flow is one dimensional, the velocity is uniform in a cross-section and the transverse free-surface profile is horizontal; (2) the streamline curvature is very small and the vertical fluid acceleration are negligible, as a result, the pressure distributions are hydrostatic; (3) the flow resistance and turbulent losses are the same as for a steady uniform equilibrium

flow for the same depth and velocity, regardless of trends of the depth; (4) the bottom slope is small enough to satisfy the following approximation: $\cos\theta \approx 1$, $\sin\theta \approx \tan\theta \approx \theta$; (5) the water density is constant; (6) the SVE were developed for fixed boundary channels: sediment motion is neglected (Chanson, 2004).

Following, continuity equation and momentum equations are presented.

Saint Venant continuity equation

The mass conservation principle or continuity equation states that the mass of the system remains constant. In general, the rate of increase of mass in the control volume is due to the net inflow of mass that means the mass within a closed system remains constant with time.

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{2.1}$$

where A (m^2) corresponds to the cross sectional area, Q (m^3s^{-1}) to the flow rate, ρ ($kg.m^{-3}$) to the density and Δx (m) to the length of the control volume. t and x correspond to time and position (Chanson, 2004).

Saint Venant momentum equation

The momentum principle applies for a given control volume. The rate of change in momentum flux equals the sum of the forces acting on the control volume. The forces acting in the fluid control volume are the friction force (F_f) due to shear stress along the bottom; the gravity force (F_g) that relates to the weight of the fluid; and the hydrostatic pressure (F) on the left and right hand side of control volume (Chanson, 2004). Figure 2.1 illustrates the open-channel flow with these three forces.

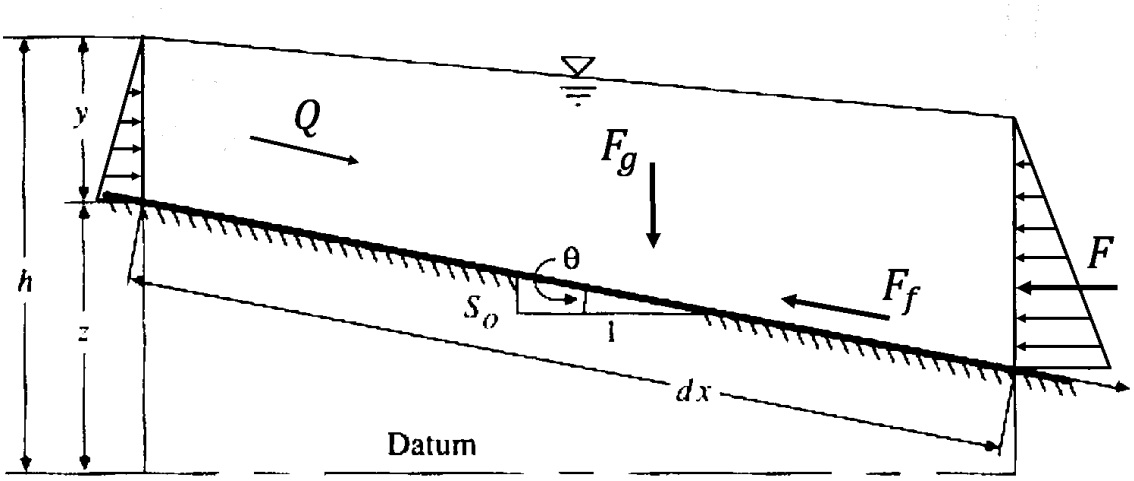

Figure 2.1: Open-Channel flow nonuniform flow. (Adapted from Maidment & Merwade, 2005; Fox et al., 2008; Chanson, 2004).

The Saint Venant Momentum Equation yields:

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{Q^2}{A}\right) + gA\frac{\partial (y+z)}{\partial x} + \frac{\tau}{\rho}\Omega = 0 \tag{2.2}$$

with Q as the flow rate (m^3s^{-1}), A the cross sectional area (m^2), g the gravitational force (m^2s^{-1}), y the water depth (m), z bottom elevation (m), τ wall sheer stress (-), ρ density ($kg.m^{-3}$), and Ω wetted perimeter (m). The first term corresponds to the advection acceleration (inertia), the second to the convective acceleration, the third term to the gravity force, and the last one to the friction force (Pothof, 2012).

2.1.2 Heat conduction equation

The heat conduction equation is valid for isotropic, homogenous or heterogeneous static solids and for static incompressible fluids. The next equation is independent of the coordinate system. In general is a non linear equation since the thermal conductivity is a function of temperature. If the thermal conductivity could be assumed as independent of temperature, the equation becomes a linear relation (Bergman et al., 2011). The heat conduction equations yields:

$$\nabla \cdot (k\,\nabla T) + \dot{q}''' = \rho c_p \frac{\partial T}{\partial t} \tag{2.3}$$

here ∇ is the Del differential operator, k ($Wm^{-1}K^{-1}$) is the thermal conductivity, ∇T (K) is the gradient of temperature, $\dot{q}'''$ (Wm^{-3}) is the energy generated or consumed per unit of time and volume, ρ ($kg.m^{-3}$) is the density, c_p ($Jkg^{-1}K^{-1}$) specific heat capacity, T (K) is the temperature.

The same equation in cylindrical coordinates becomes:

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{1}{r}\frac{\partial}{\partial r}\left(k\,r\frac{\partial T}{\partial r}\right) + \frac{1}{r^2}\frac{\partial}{\partial \theta}\left(k\frac{\partial T}{\partial \theta}\right) + \dot{q}''' = \rho c_p \frac{\partial T}{\partial t} \tag{2.4}$$

where r (m) is the cylindrical coordinate radial direction, θ is the cylindrical coordinate polar angle.

2.2 Model Equations

The model equations came from the Saint Venant Equations and could be classify in three main categories: **mass balance** (described in Subsection 2.2.1), **heat balance** (in Subsection 2.2.2), and finally the third **momentum balance** (in Subsection 2.2.3). To simplify the visualization of the exchange process (heat and mass), **Figure 2.2** is presented:

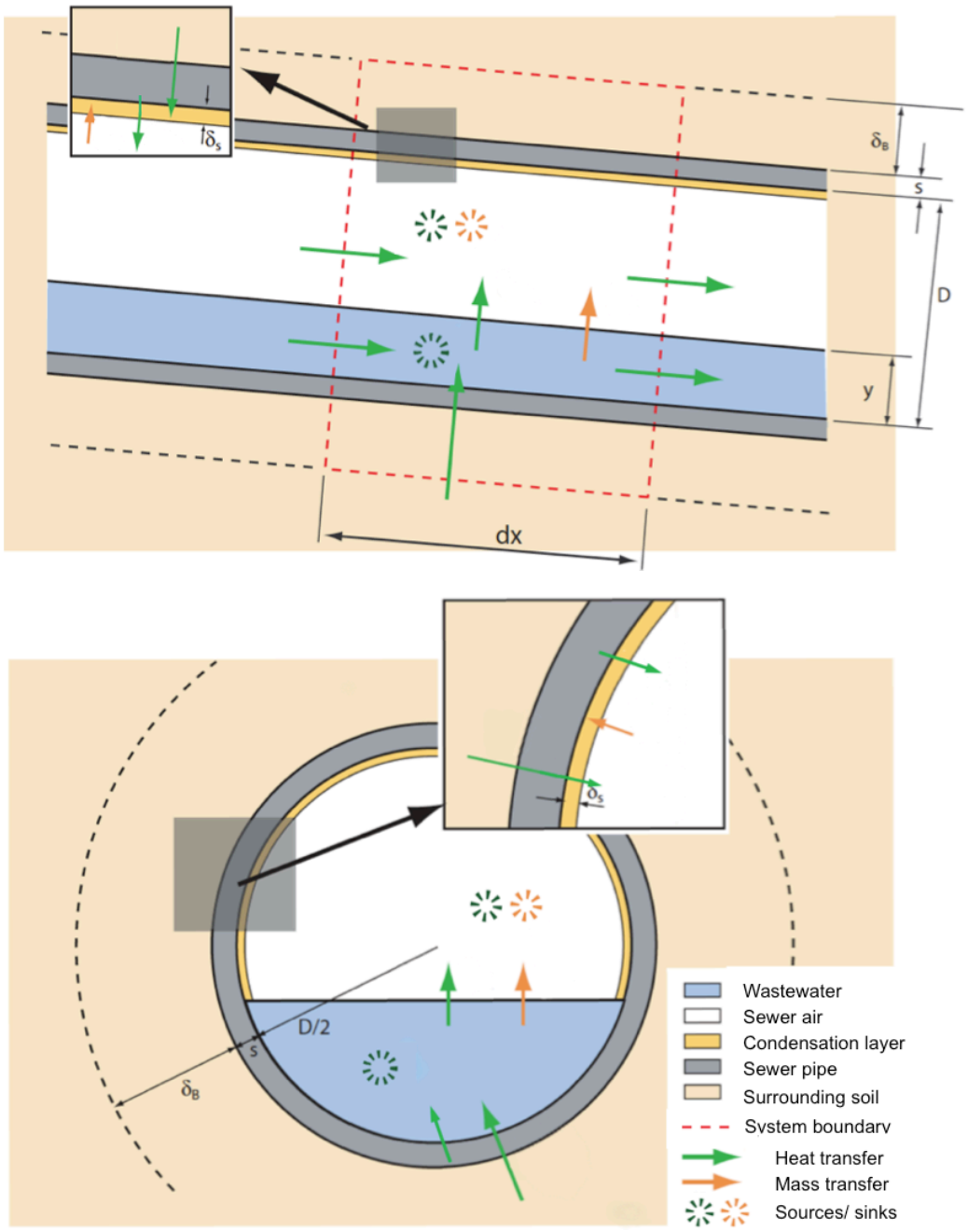

Figure 2.2: Schematic representation of a sewer pipe. In the upper view a control volume is defined (red dashed outline) and five compartments: *sewage*, *sewer air*, *condensation layer*, located *pipe wall* and *soil*. The lower view is a section through this control volume with transfer processes: orange arrows indicate mass transfer and green arrows denote heat transfer. (Adapted from Dürrenmatt & Gujer, 2006).

2.2.1 Mass balance

The mass balance in the model has established with the same principle as the continuity equation previously mentioned in Section 2.1. The rate at which mass enters a system is equal at which mass leaves the system.

According to Dürrenmatt and Gujer (2006) it is essential to take into account the possibility of transverse mass flow and if there are sources or sinks. The temporal exchange of mass is given by the further equation:

$$\frac{\partial(\rho A \Delta x)}{\partial t} = (\rho Q)_x - (\rho Q)_{x+\Delta x} + L_t l_t \Delta_x + A\Delta_x \sum r \tag{2.5}$$

where ρ is density (kg.m^{-3}), A cross sectional area (m^2), Δx length of the control volume (m), Q flow rate (m^3.s^{-1}), L_t length (m), l_t transverse mass flow (kg.m^{-2}s^{-1}) and finally $\sum r$ represents sources or sinks. Assuming density is constant:

$$\frac{\partial(\rho A \Delta x)}{\partial t} = (\rho Q)_x - (\rho Q)_{x+\Delta x} + L_t l_t \Delta_x + A\Delta_x \sum r \tag{2.6}$$

$$\rho \frac{\partial(A \Delta x)}{\partial t} = \rho Q_x - (\rho Q_x + \rho \Delta Q_x) + L_t l_t \Delta_x + A\Delta_x \sum r \tag{2.7}$$

$$\rho \frac{\partial A}{\partial t} = -\rho \frac{\Delta Q_x}{\Delta x} + L_t l_t \Delta_x + A\Delta_x \sum r \tag{2.8}$$

$$\lim_{\Delta x \to 0} \quad \rho \frac{\partial A}{\partial t} = -\rho \frac{\Delta Q_x}{\Delta x} + L_t l_t \Delta_x + A\Delta_x \sum r \tag{2.9}$$

$$\frac{\partial A}{\partial t} = -\frac{\partial Q}{\partial x} + \frac{1}{\rho}\left(L_t l_t \Delta_x + A\Delta_x \sum r\right) \tag{2.10}$$

$$\boldsymbol{\frac{\partial A}{\partial t} = -\frac{\partial Q}{\partial x} + \frac{1}{\rho}\left(L_t l_t \Delta_x + A\Delta_x \sum r\right)} \tag{2.11}$$

For the **water mass balance** the evaporation or condensation, referred as j_{vP}, is the transverse mass flow ($l_t = j_t = j_{vP}$), and has a negative sign since the flux vector is from the control volume, **Figure 2.2**. The exchange process takes place on the water surface with width B ($L_t = B$), and neither sources nor sinks are available in the pipe ($\sum r = 0$) (Dürrenmatt & Gujer, 2006).

Employing the terms of Equation 2.11, the mass balance water equation becomes:

$$\frac{\partial A_w}{\partial t} = -\frac{\partial Q_w}{\partial x} - \frac{1}{\rho_w} j_{vP} B \tag{2.12}$$

$$\frac{\partial A_w}{\partial t} + \frac{\partial Q_w}{\partial x} + \frac{1}{\rho_w} j_{vP} B = 0 \tag{2.13}$$

here, A_w is the water cross sectional area, Q_w is the water flow rate and ρ_w is the water density.

The mass balance for the air channel and water vapour can be derived in the same way as the mass balance in water, equation. In **air mass balance**, A_l is the air cross sectional area, Q_l is the air flow rate, and neither transverse mass flow nor sources/sinks are available (Dürrenmatt & Gujer 2006). The equation yields:

$$\frac{\partial A_l}{\partial t} + \frac{\partial Q_l}{\partial x} = 0 \tag{2.14}$$

Regarding the **water vapour mass balance**, let us consider X is the fraction of water vapour by dry air [kg/kg]). The principal to write the water vapour mass balance is the same as the previous equations. The transverses mass flows here are two, the evaporation or condensation mass flow (j_{vP}) and the mass flow due to condensation in the pipe (condensation layer) (j_{kP}), these process occurs in wetted perimeter air. Finally as sink the condensation oversaturation or condensation in the air volume due to (under)cooling in air (j_{kL}) (Dürrenmatt & Gujer, 2006). Above there is the equation:

$$A_l \frac{\partial X}{\partial t} + Q_l \frac{\partial X}{\partial x} - \frac{1}{\rho_l}(j_{vP} B - \zeta j_{kP} L_l - \xi j_{kL} A_l) = 0 \tag{2.15}$$

The control parameters ζ *(Zeta)* and ξ *(Xsi*) control the condensation and can take values of either zero or one:

At the condensation layer only steam vapour can be condensated, but water cannot evaporate , therefore $\zeta = 1$ if $p_L > p_{sat}(T_{pl})$ and $\zeta = 0$ in all the other cases. p_L is the partial pressure of steam vapour and $p_{sat}(T_{pl})$ is the saturated vapour pressure at the surface of the condensation layer at temperature T_{pl} (Dürrenmatt & Gujer, 2006).

If the absorbed moisture/loading with steam vapour of the sewer air X is bigger than the saturated loading with steam vapour X_{sat}, steam vapour condensates over the whole air cross sectional area. Hence $\xi = 1$ if $X > X_{sat}$ and $\xi = 0$ if $X \leq X_{sat}$ (Dürrenmatt & Gujer, 2006).

For the complete expression of Equation 2.13, Equation 2.14 and Equation 2.15 consult **Table A.1** and **Table A.3** of Appendix A.

2.2.2 Heat balance

In the conduit there are two types of heat exchange: first concerns with a balance inside the conduit and for this one, there are water heat balance and air heat balance, the second type could be classified as a heat transfer between the water/air part with the physical pipe and soil.

Water heat balance and air heat balance

The heat balance equations in the model were developed with the same principal as the continuity equation in Section 2.1, with some modifications. The rate at which heat enters a system is equal at which heat leaves the system. **Table 2.1** considers mass and heat relationships.

Table 2.1: Relation between mass and heat.

Mass		Heat	
ρ	$(kg.m^{-3})$	$C_p\rho T$	$(J.m^{-3})$
ρQ	$(kg.s^{-1})$	$C_p\rho QT$	$(J.s^{-1})$
$l_t = j_t$	$(kg.m^{-2}s^{-1})$	$l_t = \dot{q}_t$	$(J.m^{-2}s^{-1})$
$\sum r$	$(-)$	$\sum r = \dot{q}$	$(-)$

Using information from the above table into the Equation 2.11, a new equation emerges, the general heat balance equation:

$$\frac{\partial C_p\rho TA}{\partial t} = -\frac{\partial C_p\rho QT}{\partial x} + L_t\dot{q}_t + A\dot{q} \tag{2.16}$$

In the **water heat balance**, the transverse mass flow are heat flux pipe to water ($\dot{q}_{Rw}$) and occurs in the wetted perimeter (L_w), the heat that is lost to the air duct (heat flux water to air) ($\dot{q}_{wl}$) and evaporation or condensation ($\dot{q}_{vP}$) from the water. As a source, there is the biochemical activity ($\dot{q}_w$) produced in the wastewater (Dürrenmatt & Gujer, 2006). The next equations represent the exchange process:

$$\frac{\partial C_{p,w}\rho_W A_w T_w}{\partial t} = -\frac{\partial C_{p,w}\rho_w Q_w T_w}{\partial x} + L_t\dot{q}_t + A_w\dot{q}_w \tag{2.17}$$

$$\frac{\partial A_w T_w}{\partial t} = -\frac{\partial Q_w T_w}{\partial x} + \frac{1}{C_{p,w}\rho_W}(\dot{q}_{Rw}L_w - \dot{q}_{wl}B - \dot{q}_{vP}B + A_w\dot{q}_w) \tag{2.18}$$

$$\frac{\partial A_w T_w}{\partial t} + \frac{\partial Q_w T_w}{\partial x} - \frac{1}{C_{p,w}\rho_W}(\dot{q}_{Rw}L_w - \dot{q}_{wl}B - \dot{q}_{vP}B + A_w\dot{q}_w) = 0 \tag{2.19}$$

Regards **air heat balance**, the heat transfer to air comes from the pipe ($\dot{q}_{Rl}$) and water ($\dot{q}_{wl}$), as a source there is a heat flux oversaturation ($\dot{q}_{kl}$).

$$A_l\frac{\partial T_l}{\partial t} + Q_l\frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl}L_l + \dot{q}_{wl}B + \xi\dot{q}_{kl}A_l) = 0 \tag{2.20}$$

Table A.2 and **Table A.3** in Appendix A lists the variables thoroughly the expression in Equation 2.19 and Equation 2.20.

Pipe heat balance and soil heat balance

The pipe heat balance and soil heat balance can be achieved from Equation 2.4 in Section 2.1 with two simple assumptions. First of all, the heat conduction is considered only in the radial direction since the cylinder is long enough to neglected the top and bottom effects (Bergman et al., 2011). Applying this assumptions to the heat conductive equation the equation become:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(k\,r\frac{\partial T}{\partial r}\right)+\dot{q}'''=\rho c_p\frac{\partial T}{\partial t} \tag{2.21}$$

As second assumption the energy generated (thermal) within element is neglected ($\dot{q}'''=0$), the heat conductive equation yields:

$$\frac{1}{r}\frac{\partial}{\partial r}\left(k\,r\frac{\partial T}{\partial r}\right)=\rho c_p\frac{\partial T}{\partial t} \tag{2.22}$$

Rearrange the previous equation and substituting k for λ, the heat conductive equation using in the model is:

$$\frac{\partial T}{\partial t}-\frac{\lambda}{rC_p\rho}\frac{\partial}{\partial r}\left(r\frac{\partial T}{\partial r}\right)=0 \tag{2.23}$$

The model equation for pipe heat balance and soil heat balance are listed below.

Pipe heat balance in water part:

$$\frac{\partial T_{pw}}{\partial t}-\frac{\lambda_p}{rC_{p,p}\rho_p}\frac{\partial}{\partial r}\left(r\frac{\partial T_{pw}}{\partial r}\right)=0 \tag{2.24}$$

here, T_{pw} as pipe temperature for water part, $C_{p,p}$ the specific heat capacity for the pipe, and finally ρ_p the density of pipe.

Pipe heat balance in air part:

$$\frac{\partial T_{pl}}{\partial t}-\frac{\lambda_p}{rC_{p,p}\rho_p}\frac{\partial}{\partial r}\left(r\frac{\partial T_{pl}}{\partial r}\right)=0 \tag{2.25}$$

here, T_{pl} as pipe temperature for the air part, $C_{p,p}$ the specific heat capacity for the pipe, and finally ρ_p the density of pipe.

Soil heat balance in water part:

$$\frac{\partial T_{sw}}{\partial t}-\frac{\lambda_s}{rC_{p,s}\rho_s}\frac{\partial}{\partial r}\left(r\frac{\partial T_{sw}}{\partial r}\right)=0 \tag{2.26}$$

here, T_{sw} as soil temperature for water part, $C_{p,s}$ the specific heat capacity for the soil, and finally ρ_p the density of soil.

Soil heat balance in air part:

$$\frac{\partial T_{sl}}{\partial t}-\frac{\lambda_s}{rC_{p,s}\rho_s}\frac{\partial}{\partial r}\left(r\frac{\partial T_{sl}}{\partial r}\right)=0 \tag{2.27}$$

here, T_{sl} as soil temperature for air part , $C_{p,s}$ the specific heat capacity for the soil, and finally ρ_p the density of soil.

For further information about the variables in Equation 2.24, Equation 2.25, Equation 2.26 and Equation 2.27, please refer to **Table A.2** of Appendix A.

2.2.3 Water momentum balance

The water momentum balance equation can be achieved from Equation 2.2 by developing the terms for the gravity force and friction (third and fourth terms). However a few assumptions are made beforehand.

Beginning with the gravity force, the water depth could be represented by h instead of y. Additionally the bottom slope (S_0) is equal to $\sin\theta$ and can be approximated to $S_0 \approx \tan\theta = -\partial z_0/\partial x$. According to assumption 4 stated in section 2.1, the bottom slope is small enough to satisfy the approximation $\cos\theta \approx 1,\ \sin\theta \approx \tan\theta \approx \theta$. By substituting the third term comes that:

$$gA\frac{\partial(y+z)}{\partial x} = gA\frac{\partial y}{\partial x} + gA\frac{\partial z}{\partial x} = gA\frac{\partial h}{\partial x} + gA\frac{\partial z_0}{\partial x} = gA\frac{\partial h}{\partial x} - gAS_0 \tag{2.28}$$

Now regarding the friction tern in the dynamic laws, the wetted perimeter Ω can be replaced by P. Furthermore the wall shear stress is equal to $\tau = \rho g\frac{ALS_0}{PL}$ and, for the steady uniform flow, the friction and gravity forces are in balance: $S_0 = S_f$. Substituting this into the fourth term, the friction term becomes:

$$\frac{1}{\rho}\rho g\frac{ALS_0}{PL}P = gAS_0 = gAS_f \tag{2.29}$$

Replacing these finals substitutions into the Equation 2.2 yields:

$$\frac{\partial Q_w}{\partial t} + \frac{\partial}{\partial x}\left(\frac{{Q_w}^2}{A_w}\right) + gA_w\frac{\partial h}{\partial x} - gA_wS_0 + gA_wS_f = 0 \tag{2.30}$$

Changing the notation for S_0 and S_f by i_b and i_f, the final equation of momentum balance is:

$$\frac{\partial Q_w}{\partial t} + \frac{\partial}{\partial x}\left(\frac{{Q_w}^2}{A_w}\right) + gA_w\frac{\partial h}{\partial x} - gA_w\left(i_b - i_f\right) = 0 \tag{2.31}$$

2.3 Boundary and initial conditions

2.3.1 Boundary conditions

The succeeding boundary conditions are imposed on the upstream boundary: water temperature (T_w), air temperature (T_l), water flow rate (Q_w), air flow rate (Q_l) and humidity (ϕ). An stationary relation between water and air flow is assumed in order to calculated the air flow rate using semi-empirical formulas:

$$Q_l = 0.8560\frac{B}{P_l}u_{w,c}A_l \tag{2.32}$$

where, $u_{w,c}$ designated the water velocity at the interface water and air:

$$u_{w,c} = U_w \left(1 + \frac{\sqrt{c_f}}{k_{VK}} \left(\frac{3}{2} + 2.30 \log \frac{2h}{D} \right) \right) \tag{2.33}$$

where, U_w is the velocity of water (m.s^{-1}), c_f is the friction factor of pipe wall, k_{VK} is the Von Karman constant (≈ 4).

2.3.2 Initial conditions

For the initial conditions the stationary solution is assumed. For the water flow rate, the upstream boundary condition is assumed. The air flow rate is calculated from the water flow rate. The surface are is solved iteratively by ($i_b = i_f$):

$$A_w = \left(\frac{c_f P_w Q_w^2}{g i_b} \right)^{(1/3)} \tag{2.34}$$

Regarding the temperature and water vapour, the stationary equations for the water balance of water vapour and heat balance for water, air, pipe and soil need to be solved, which from eight coupled differential equations (from Equation 2.35 to Equation 2.41). The stationary equations represent the stationary solution that is independent of time, therefore the time derivatives can be set as zero.

$$\frac{dX}{dx} = \frac{1}{Q_l \rho_l} (j_{vP} B - \zeta j_{kP} L_l - \xi j_{kL} A_l) \tag{2.35}$$

$$\frac{dT_w}{dx} = \frac{1}{Q_w c_{p,w} \rho_w} (\dot{q}_{Rw} L_w - \dot{q}_{wl} B - \dot{q}_{vP} B + A_w \dot{q}_w) \tag{2.36}$$

$$\frac{dT_l}{dx} = \frac{1}{Q_l c_{p,l} \rho_l} (\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) \tag{2.37}$$

These tree equations are solved numerically, Equation 2.35 to Equation 2.37. View Section 2.4.

The equations for the temperature in pipe and soil are:

$$\frac{d}{dr} \left(r \frac{dT_{pw}}{dr} \right) = 0 \tag{2.38}$$

$$\frac{d}{dx} \left(r \frac{dT_{pl}}{dr} \right) = 0 \tag{2.39}$$

$$\frac{d}{dr} \left(r \frac{dT_{sw}}{dr} \right) = 0 \tag{2.40}$$

$$\frac{d}{dr} \left(r \frac{dT_{sl}}{dr} \right) = 0 \tag{2.41}$$

These four equations can be solved analytically:

$$T_{pw}(r) = T_w + (T_{s0} - T_w)k_{rw}\left(\frac{D}{2\lambda_p}\ln\left(\frac{2r}{D}\right) + \frac{1}{\alpha_{rw}}\right) \quad (2.42)$$

$$T_{sw}(r) = T_{s0} + (T_{s0} - T_w)k_{rw}\frac{D}{2\lambda_p}\ln\left(\frac{2r}{D + 2D_p + \delta_s}\right) \quad (2.43)$$

$$T_{pl}(r) = T_l + (T_{s0} - T_l)k_{rl}\left(\frac{D}{2\lambda_p}\ln\left(\frac{2r}{D}\right) + \frac{1}{\alpha_{rl}}\right) \quad (2.44)$$

$$T_{sl}(r) = T_{s0} + (T_{s0} - T_l)k_{rl}\frac{D}{2\lambda_p}\ln\left(\frac{2r}{D + 2D_p + \delta_s}\right) \quad (2.45)$$

the heat transfer flowing water α_{rw} and air α_{rl}, k_{rw} and k_{rl} are listed in **Table A.3** of Appendix A. T_{s0} is the bottom temperature (steady state), D is the pipe diameter, D_p the thickness of pipe, and δ_s the Influence distance of soil.

The numerical methods begin with initial and boundary conditions. At time $t = 0$, the uniform steady flow conditions are specified al all the location. (Mujumdar, 2001). In the present work it is called stationary solution and is used as initial conditions. Following, an introduction to the numerical solvers is present.

2.4 Numerical solvers

The many differential equations do not have closed form solution and therefore must be approximated numerically. A **finite difference method** creates a particular discretized system through finite differences that approximate the differential equation in order to solve it (Patty, 2010). In detail, this method consists in approximating the differential operator by replacing the derivatives in the equation using differential quotients. The function is partitioned in space and in time and approximations of the solution are computed at the space or time points (Pascal & De Buhan, 2008).

The **Lax-Wendroff method** is a numerical method based on finite differences for solving approximately hyperbolic conservation laws (partial differential equations). Peter Lax and Burton Wendroff proposed this technique in 1960 (Grove, 1999):

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0\,, \quad u(x,t) \quad (2.46)$$

The fundamental concept of this method is to extend $u(x,t)$ for a fixed x using a Taylor series with second order accuracy. Using the partial differential equations to replace the time derivatives with spatial derivatives and employing the central differences to approximate the resulting spatial derivatives to second order.

In practice the Lax-Wendroff method is implemented as two step method that is identical with the above equation for linear fluxes.

$$u_{i+1/2}^{n+1/2} = \frac{u_{i+1}^{n} + u_{i}^{n}}{2} - \frac{1}{2} - \frac{\Delta t}{\Delta x}(f_{i+1}^{n} - f_{i}^{n}) \tag{2.47}$$

$$f_{i+1/2}^{n+1/2} = f\left(u_{i+1/2}^{n+1/2}\right) \tag{2.48}$$

$$u_{i}^{n+1} = u_{i}^{n} - \frac{\Delta t}{\Delta x}(f_{i+1/2}^{n+1/2} - f_{i-1/2}^{n+1/2}) \tag{2.49}$$

In order to solve the partial differential equations numerically by finite differences methods a condition for convergence is needed. In the present case the Courant-Friedrich-Levy (CFL) condition is considered. The time step for each iteration must be less that a certain time. For the stability, the Courant number as defined in the next equation must be smaller than one.

$$CFL = v\,\frac{\Delta t}{\Delta x} \tag{2.50}$$

Where v is described as the maximum velocity of the water plus propagation of waves.

Regarding gas dynamics and in case of discontinuities, this method produces oscillations that can destroy the integrity of the computation. In order to inhibit these oscillations, an artificial viscosity could be added to the numerical method, for instance the linear artificial viscosity and the lapidus artificial viscosity. These two modifications to the Lax-Wendroff fluxes have been proven to be useful in practice (Grove, 1999).

Linear artificial viscosity is implemented by adding a term that simulates the diffuse term. For the lapidus artificial viscosity, in order to increase the artificial viscosity in regions of large gradient whilst reducing in smooth regions, an adaptive method was developed by Lapidus (Grove, 1999).

Chapter 3 will show how these equations could be implemented in the numerical framework.

Chapter 3

Methods

This work employs a software previously developed in MATLAB® by Bas Wols at KWR Water Research Institute (Nieuwegein, The Netherlands). This model aims to predict temperatures in a sewer system and steady flow, in order to find the best locations for heat recovery and help to understand the underlying physics of the dynamic system.

The methodology in this project is organized in two subsections. Section 3.1 describes the previously developed algorithm, data structures and result visualization. A few adaptions were made to these routines, like the implementation of unsteady regimes for flow rate and temperature of water.

For both steady and unsteady situations, the impact of each term in the equations for water and air heat balance equation (Equations 2.19 and 2.20) was studied. A script was developed to process all the combinations of terms and compare these results with the situation where all terms of the equations are included called general case. This process is present in Section 3.2.

3.1 Implementation of model

There is a wide variety of data types in MATLAB® and the ones employed in this project are described in Subsection 3.1.1. Furthermore in Subsection 3.1.2 the concepts of conduits and nodes are defined, plus a list of few assumptions there is presented. Subsection 3.1.3 describes the workflow required to initiate, process and visualize the analysis of a model. Modifications made to the code in order to implement unsteady flow regimes are present in Subsection 3.1.4 as well a comparison between both situation, steady and unsteady situations.

Please bear in mind that the developed model does not seek to quantify relations between wastewater discharge, airflow, water vapour and temperatures and discharge stimulation. For now it allows a qualitative examination of the physical process along the conduit.

3.1.1 Data types

MATLAB® is a high-level language that uses matrix data structures, a wide collection of functions and offers the user the possibility to develop user defined functions and scripts. The variable that stores the properties of modelled sewer system is mentioned as model and is based in two basic components known as conduits and nodes. The data structures employed in this project were array, structures, cells and variables. **Figure 3.1** shows a scheme of the model, stored as a structure, and the various fields that are enclosed in it.

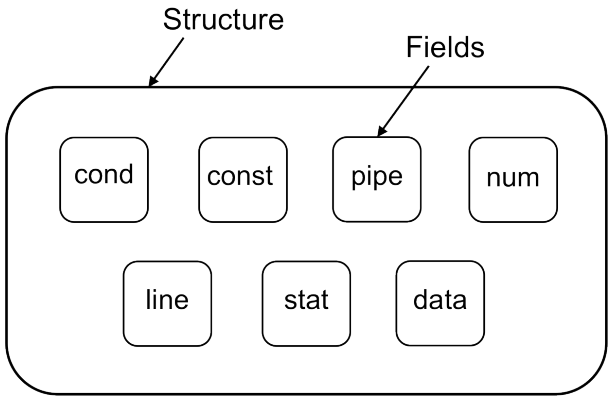

Figure 3.1: Representation of model structure with the seven fields: constant conditions (cond), constants (const), conditions of pipe (pipe), numerics (num), conditions of the network (line), stationary solution (stat) and results data (data).

A description of each one of the seven fields, the type of data, parameters and the respective value is made as follows.

Constant conditions

The 'constant conditions' are defined as a structure (*model.cond*) with variables enclosed, namely the temperature of air and soil, air pressure and humidity. **Table 3.1** contains those variables and the respective considered value for the simulations.

Table 3.1: Values for each variable of 'constant conditions' (*model.cond*).

Parameter	Symbol	Value
Air temperature	Tair	283 K
Soil temperature	Tsoil	280 K
Ambient air pressure	ψ	1000 mbar
Relative humidity	ϕ	0.8

Constants

The structure 'constants' (*model.const*) contains variables and structures of variables. The values considered in our analyses for these variables are listed in the subsequent table.

Table 3.2: Values for each parameter of 'constants' (*model.const*).

Parameter	Symbol	Value
Gravity acceleration	g	9.81 $m^3.s^{-1}$
Saturation pressure	p_{s0}	1.73E09 mbar
Saturation temperature	T_{s0}	311 K
Evaporation enthalpy	hfg	2453.3E03 $J.kg^{-1}$
Density water	ρ_w	1000 $kg.m^{-3}$
Density air	ρ_l	1.188 $kg.m^{-3}$
Specific heat capacity water	$c_{p,w}$	4.1813E03 $Jkg^{-1}.K^{-1}$
Specific heat capacity air	$c_{p,l}$	1007 $Jkg^{-1}.K^{-1}$
Kinematic viscosity water	ν_w	1E-06 $m^2.s^{-1}$
Kinematic viscosity air	ν_l	1.533E-05 $m^2.s^{-1}$
Thermal diffusivity water	α_w	1.4E0-7 $m^2.s^{-1}$
Thermal diffusivity air	α_l	2.216E0-5 $m^2.s^{-1}$
Thermal diffusivity soil	α_s	0.74E-06 $m^2.s^{-1}$
Thermal conductivity water	λ_w	0.6 $Wm^{-1}.K^{-1}$
Thermal conductivity air	λ_l	0.02569 $Wm^{-1}.K^{-1}$
Thermal conductivity soil	λ_s	2.2 $Wm^{-1}.K^{-1}$
Prandtl number water/air	Pr	$\mathrm{Pr} = \nu/\alpha$
Binary variables water	f_{Rw}, f_{wl}, f_{vP}	1 or 0
Binary variables air	f_{Rl}, f_{kl}	1 or 0

Conditions of conduit

The 'conditions of the conduit' or 'conditions of the pipe' are defined as a structure (*model.pipe*) with several variables – **Table 3.3**.

Table 3.3: Values for each parameter of 'conditions of the pipe' (*model.pipe*).

Parameter	Symbol	Value
Shape	"pipe"	-
Pipe diameter	D	0.235 m
Thickness of pipe	D_p	0.04 m
Pipe length	L	10 m
Bottom slope	i_b	2E-03 $m.m^{-1}$
Friction constant	c_f	2.4E-03
Density pipe	ρ_p	2000 $kg.m^{-3}$
Specific heat capacity pipe	c_p	0.84E003 $Jkg^{-1}.K^{-1}$
Influence distance of soil	δ_s	3 m
Thermal conductivity pipe	λ_p	2.3 $Wm^{-1}.K^{-1}$
Spatial grid distance	$\mathrm{Dx} = L/\mathrm{n}$	-
Thermal diffusivity pipe	$\alpha_p = \lambda_p c_p/\rho_p$	-

Numerics parameters

The 'numerics' structure (*model.num*) contains the parameters that concern the numeric simulation as shown in **Table 3.4**.

Table 3.4: Values for each parameter of 'numeric' (*model.num*).

Parameter	Symbol	Value
Number of time steps	nstep	1000
Grid size	n	100
Courant number criterium	CFL	0.8
Number of plots	nplots	5
Store solution nsaves times	nsaves	1000
Number of layers in pipe	$\mathrm{nlayers}$	5
Number of layers in soil	nsoil	5

Conditions of the network

The 'conditions of the network' or 'conditions of the line' are stored in a structure (*model.line*). The user can choose the connectivity of the pipe network using array *ind.* For the present project a simple conduit with two nodes that represents the extremities of the pipe. Initial conditions were set at the first node and no value was setting for the second node (*NaN).* The following table presents the values employed in the simulations.

Table 3.5: Values for each variables of 'conditions of the network (line)' (*model.line*).

Parameter	Symbol	Value
Connections between conduits and nodes	ind	[1 2]
Temperature of water	T_w	283 K
Temperature of air	T_l	Initialized at 'Tair'
Water flow rate	Q_w	0.001 $m^3.s^{-1}$
Air flow rate	Q_l	NaN
Water vapour fraction	X	NaN
Water cross sectional area	A_w	NaN

For the Q_l, X and A_w the values are computed by the algorithm. The same happens for T_l, however this value is initialized with the value of air temperature (Tair) from **Table 3.1**.

Stationary solution

As described in Subsection 3.1.3, one of the first steps in the analysis procedure is the determination of the stationary solution. The stationary solution represents the initial conditions at the first node. All variables associated with the stationary solution are stored in a structure (*model.stat*). The variables are: position along the pipe (x), bottom height (z_b), air cross sectional area (A_l), water cross sectional area (A_w), water depth (h), water flow rate (Q_w), air flow rate (Q_l), temperature of water (T_w), temperature of air (T_l), water vapour fraction (X), relativity humidity (ϕ), saturation pressure (p_{sat}), temperature in pipe water part (T_{pw}), temperature in pipe in air part (T_{pl}), temperature soil water part (T_{sw}) and temperature soil air part (T_{sl}).

Results data

The data structures were the results of the PDEs solutions are stored in a cell of structures (*model.data*). Each structure refers to a store time instant and contains a set of arrays corresponding to the estimated values for all the fields mentioned in the Stationary solution. Each data cell has one line per pipe. In this case the data cell has one line and the number of

columns is the same as the number of saved steps (time). Each array has the value of the respective field for all the grid positions. An example for a data cell is displayed in Figure 3.2, where the fields T_w and Q_w are shown:

$$\text{model.data} = \{\text{results for } t = 0\} \dots \{\text{results for } t = t_{final}\}$$

$$= \left\{ \begin{bmatrix} T_w(x_1, t=0) \\ T_w(x_2, t=0) \\ \dots \\ T_w(x_n, t=0) \end{bmatrix} \begin{bmatrix} Q_w(x_1, t=0) \\ Q_w(x_2, t=0) \\ \dots \\ Q_w(x_n, t=0) \end{bmatrix} \dots \right\} \dots \left\{ \begin{bmatrix} T_w(x_1, t=t_{final}) \\ T_w(x_2, t=t_{final}) \\ \dots \\ T_w(x_n, t=t_{final}) \end{bmatrix} \begin{bmatrix} Q_w(x_1, t=t_{final}) \\ Q_w(x_2, t=t_{final}) \\ \dots \\ Q_w(x_n, t=t_{final}) \end{bmatrix} \dots \right\}$$

Figure 3.2: Example for a data cell with T_w and Q_w as fields.

Here, x_n is the position along the pipe for grid size *n* and t_{final} is the number of steps in time.

3.1.2 Conceptual model

Definitions

The model is based in two basic components entitled conduits and nodes. A conduit is assumed to represent a prismatic pipe with circular cross-section and without discontinuity. Here wastewater discharge, airflow, water vapour and temperatures are functions of time and space, and are modelled by the balance equations described in Section 2.2 (Dürrenmatt & Wanner, 2008).

Nodes define discontinuities originated by lateral inflows, headspace openings, unexpected changes of the sewer geometry or material properties, and are modelled by continuity conditions. To clarify, a conduit could be a line and the extremities the nodes, so one line have two nodes. The boundary conditions of the system are defined in these nodes (Dürrenmatt & Wanner, 2008; Dürrenmatt & Gujer, 2006).

Two system boundaries can be distinguished; the first one is a conceptual boundary and the other a physical one. The conceptual boundary defines the part of the drainage system (therefore includes all sewers pipes, manholes, etc.) and is referred as node. The physical system boundary defines the balance of a volume in a pipe section which is required to derive the mathematical equations that describe the operations of heat transfer (Dürrenmatt & Gujer, 2006).

Assumptions and simplifications

These considerations are divided in physical constants, materials and topology, flow and heat balance.

Physical constants: (1) the physical constants of the pipe do not change with temperature (pipe or soil will not change of shape or size along the length); (2) thermally and hydraulically wastewater behaves as water; (3) water and air are classified as incompressible flows, *i.e.*

density is constant; (4) the specific enthalpy of water vapour is equal to the evaporation enthalpy h_{fg} (Dürrenmatt & Gujer, 2006).

Materials and topology: (5) nodes have no size, they are represented as zero-dimensional objects; (6) the pipe defines a line section, which is free of discontinuities (these are taken into account in the node); (7) the pipe is defined as a line segment in wich the material propreties and topology are constant in space and time; (8) sewers pipes have a circular cross section; (9) the soil is isotropic (uniform in all orientations) and homogeneous, therefore a constant thermal conductivity is considered; (10) interaction between sewer pipe and soil is limited to a layer around the pipe with a thickness of δ_s; (11) the temperature in the exterior of the layer with thickness of δ_s, is spatially and temporally constant (Dürrenmatt & Gujer, 2006).

Flow: (12) the connections (or places where water is discharged) are small enough to neglect the cross sectional area and use the one-dimensional property of the model; (13) the wastewater flow is not affected by the air in the pipe. But on the other side, the air flow is affected by the wastewater flow (not a reciprocal interaction between both streams); (14) in the calculation of steady-state, a uniform discharge is assumed. The unsteady flow can be modeled as a diffusive wave using the Saint-Venant equations; (15) returns flow can be neglected (Dürrenmatt & Gujer, 2006).

Heat balance: (16) heat created by internal and external friction is neglected; (17) the diffusive transport of heat in the longitudinal direction is neglected, only the advective heat transport is taken into account; (18) the heat balance between soil and pipe takes place only in the radial direction; (19) in condensation layer water vapour can condense, whereas water does not evaporates; (20) the condensation layer is massless; (21) the mass transport can be treated analogously to the heart transport. The mass transfer coefficient is equal to heat transfer coefficient divided by h_{fg} (with assumption 4) (Dürrenmatt & Gujer, 2006).

3.1.3 Analytical model

The code begins with function *fun_start* where a sewer model is initialized. Subsequently in function *sewer_temp_model* the model is processed. Finally results visualization is computed with function *fun_plot*, which plots the results along the pipe length for sample times of the analysis. Moreover, for the same results a function called *fun_plot_t* was developed in order to visualize the results in time instead of position. The flowchart in **Figure 3.3** displays shows a scheme of this algorithm.

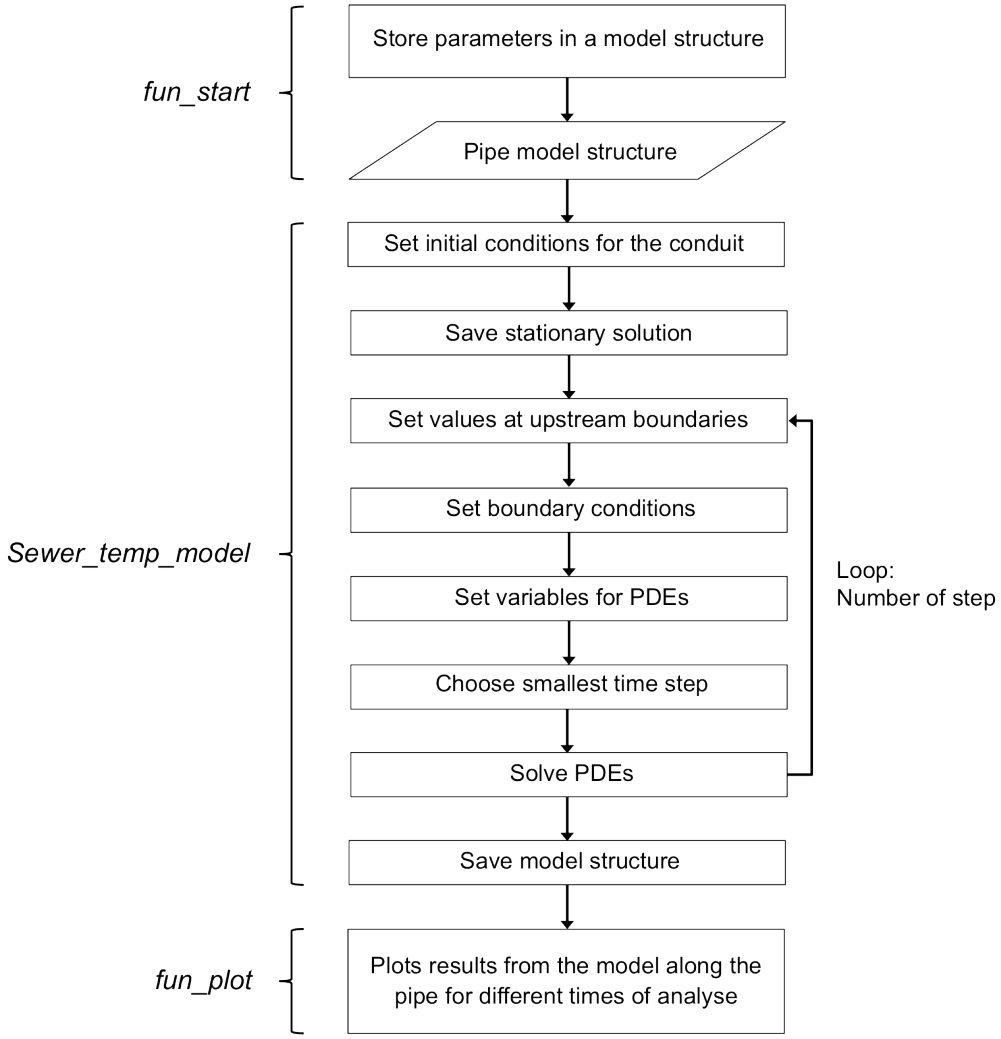

Figure 3.3: Previously developed program in MATLAB® in KWR Water Research Institute (Nieuwegein, The Netherlands) by Bas Wols, containing 3 functions: *fun_start*, *sewer_temp_main* and *fun_plot*. Number of steps defined by the user.

Start model

The model is initializing with function *fun_start*, this function stores the parameters (constant conditions, constants, conditions of conduit or pipe, numerics parameters, conditions of the network) in an initial version of the model structure. This function was modified in order to help to process different PDEs term combinations as described in Section 3.2. A binary array was added as an argument. This binary array indicates which terms in Equation 3.1 and Equation 3.2 are considered in the analysis. This will be clarified in Section 3.2.

Model analyses

In order to compute the numerical model, function *sewer_temp_model* is called. This function receives as argument a model structure initialized from function *fun_start*. It starts by defining the initial conditions for the conduit or pipe, and calculating and storing the stationary solution. The stationary solution holds the solution of the PDEs for time instant $t = 0$. Subsequently a loop spans across all step iterations and calculates the solution for each time step. For each

iteration it starts by initializing the conditions at the upstream boundaries, setting the variables for the PDEs, and calculating the smallest time interval for the respective step to solve the PDEs (according to Δt as described in Section 2.4). The most important part of the algorithm follows, the calculation of the solution of the PDEs at every pipe (n this work, only one pipe was modelled) by employing the Lax-Wendroff method for equations 2.13, 2.14, 2.15, 2.19, 2.20 and 2.31. For equations 2.24, 2.25, 2.26 and 2.27, these equations are analytically resolved. The loop finalizes by storing the calculated results in field *data* for the steps defined in the variable *num.nsave*.

Print model results in position

Function *fun_plot* is the default function for output visualization. This routine prints the obtained predictions for water height (h), water flow rate (Q_w), water temperature (T_w), and air temperature (T_l) along the length of the pipe for the time instants defined in the variable *num.nplot*.

Print model results in time

Function *fun_plot* allows the representation of the estimations in the length of the pipe. The author developed a new function named *fun_plot_t* to change the way results are arranged and plotted. For a given set of positions in the length of the path, this function plots the results as a function of time. In order to obtain the variables of the results cell *data* in time, function *datatime* was developed.

Function *datatime* receives as arguments a model structure and an array of positions along the pipe, and returns a cell named *datatt*. The model structure has a field called *data* that consists in a cell that encloses all the results obtained (*model.data*). This cell has as many positions as the number of time steps (*nstep*). Each position has a structure with fields that correspond to the calculated variables, for the corresponding time step. For each saved time step the routine gets the value of each variable at the position specified by the user. In conclusion there are two routines, one routine capable of displaying the calculated variables in time and another that shows them in position of the pipe.

Print model results in colour map

To visualise the results, colour maps representations were developed in function *fun_plot_color*. Colour maps are able to represent the system variables as function of time (abscissa) and position along the pipe (ordinate), with a colour gradient representing the magnitude of this variable in question.

In order to implement this function the data needed to be compiled from the cell '*model.data*' and arranged in a matrix where rows are the values of the variable in question over the time and the columns are the values the position in the pipe. This matrix was then given as an argument to default MATLAB function *colormap*.

3.1.4 Boundary conditions

Previously these routines were only allowing discharge cases for a steady flow was considered. Therefore this would correspond to constant boundary conditions (BC) in time.

Subfunction *fun_BC* defines the values at the boundary conditions. Some modifications were made at this level to allow the code to consider an unsteady situation, where some wastewater discharge flow properties varied with time. Unsteady discharges were simulated with time dependent boundary conditions for water flow rate (Q_w) and temperature in water (T_w). Discharges were considered to have a Gaussian function as shown in **Figure 3.4**.

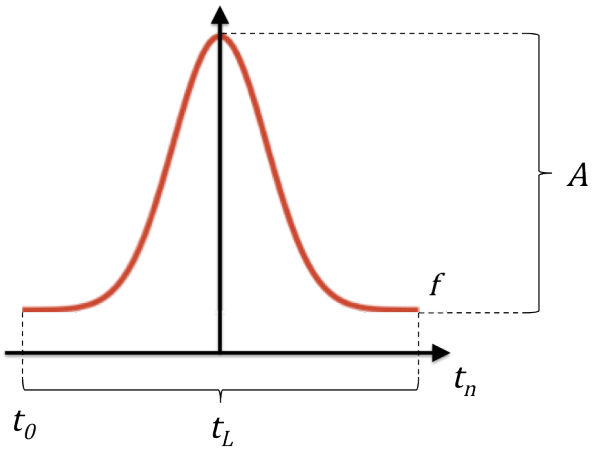

Figure 3.4: A Gaussian function (f) starts at t_0 over the numerical time t_n, the duration of the discharge is t_L and the peak has amplitude A.

The modified boundary conditions for this case are described in **Table 3.6**, where the Gaussian is added to a constant flow of water $Q_{w,bc}$ and to the water temperature $T_{w,bc}$.

Table 3.6: Time dependent boundary conditions for water flow rate (Q_w) and water temperature (T_w). Where t_0 correspond to start time, t_n is the numerical time, t_L the duration of the function and A' is the amplitude. For the water flow rate a Q was added to the variables and a T for the water temperature.

Variable	Steady situation	Unsteady situation
Water flow rate	$Q_w(x=0,t) = Q_{w,bc}$	$Q_w(x=0,t) = Q_{w,bc} + Q_{w,bc}.A'_Q.\exp\left(-\frac{(t_n - t_{0Q})^2}{0.5t_{LQ}^2}\right)$
Water temperature	$T_w(x=0,t) = T_{w,bc}$	$T_w(x=0,t) = T_{w,bc} + T_{w,bc}.A'_T.\exp\left(-\frac{(t_n - t_{0T})^2}{0.5t_{LT}^2}\right)$

3.2 Assess each terms in heat balance equations

Along with the implementation of time varying boundary condition for water flow and water temperature, one of the main goals of this project is to study the impact of each term in the

equations for heat balance in water and air (Equations 2.19 and 2.20) defined in Section 2.2. A script was developed to compute a model where all terms of the equations are included, that we shall call general case, and compare it with several models that respectively correspond to a combination of terms in the heat balance equations, hereby mentioned as modified cases (MC). The developed model is presented in a flowchart, following **Figure 3.5**.

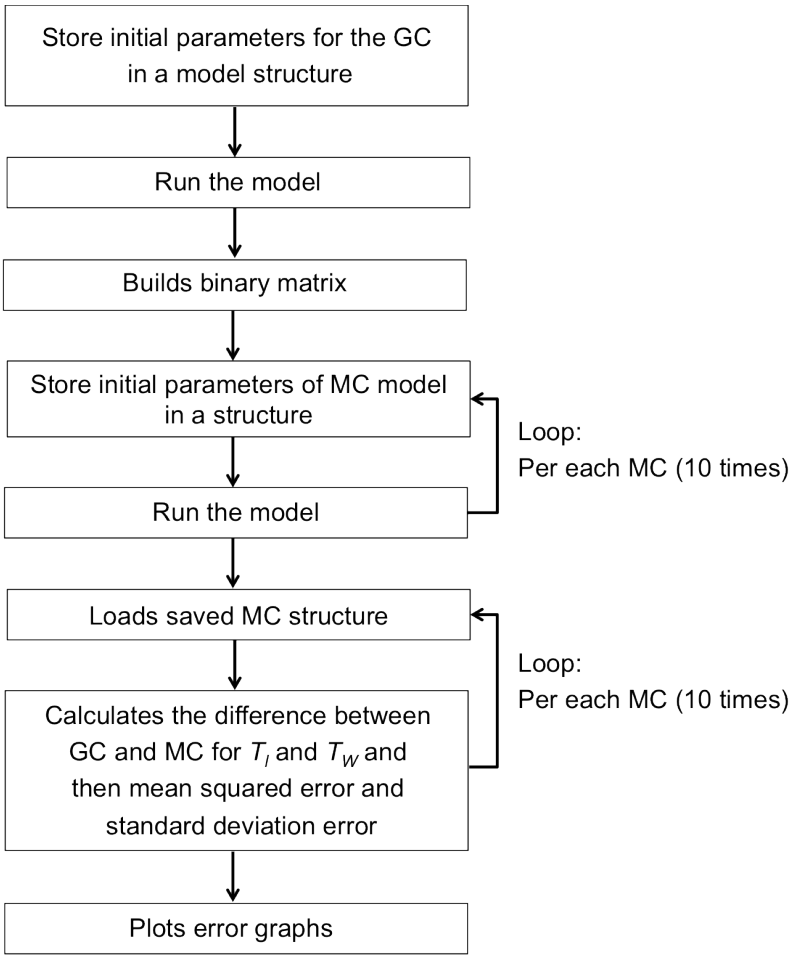

Figure 3.5: Developed model in MATLAB® with the principal phases.

First of all some modifications were made to function *fun_start* compared to the previous model from Bas Wols. This function receives as argument a binary array and each position of the array corresponds to a specific term in the heat balance equations. The terms will be included or not in the calculation of the PDEs.

Regarding **Figure 3.5**, this figure represents the principal phases of the developed script. It starts by initializing the model structure for the general case. Here function *fun_start* receives as argument a binary array containing a non-false value (1) in all its positions. The *sewer_temp_model* for the general case is executed and store.

The next phase concerns about the modified cases. A binary matrix is assembled with all combinations of equations parameters included in it. Each line corresponds to a modified case model and it is described in Subsection 3.2.1. For each line of the matrix, the initial parameters for the modified case are stored in a new model structure and processed by running the model. Finally the comparison between the general case and each modified case is made by calculating the deviation as described in Subsection 3.2.2 and these deviations are plotted – Subsection 3.2.3.

3.2.1 Matrix and modified cases

In order to determine the contribution of each term of the water heat balance equation (Equation 2.19) and air heat balance equation (Equation 2.20), all combinations of terms are considered and compared to a general case that will include all of them. Binary coefficients were introduced for each heat flux for both the equations. The biochemical activity term ($\dot{q}_w$) was assumed to be not significant and therefore was previous neglected in the heat balance water equation.

The following equations have the binary coefficients $\boldsymbol{f}$ in bold. The equation for the water heat balance, as adapted from Equation 2.19 is

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\boldsymbol{f_{Rw}} \cdot \dot{q}_{Rw} L_w - \boldsymbol{f_{wL}} \cdot \dot{q}_{wL} B - \boldsymbol{f_{vP}} \cdot \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \qquad (3.1)$$

where $\boldsymbol{f_{Rw}}$ is the binary coefficient for heat flux pipe to water, $\boldsymbol{f_{wL}}$ is the binary coefficient for the heat flux water to air and $\boldsymbol{f_{vP}}$ is the binary coefficient for evaporation or condensation. Equation 2.20 that corresponds to the air heat balance was modified as follows

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\boldsymbol{f_{Rl}} \cdot \dot{q}_{Rl} L_l + \boldsymbol{f_{wL}} \cdot \dot{q}_{wl} B + \boldsymbol{f_{kl}} \cdot \xi \dot{q}_{kl} A_l) = 0 \qquad (3.2)$$

where $\boldsymbol{f_{Rl}}$ is the binary coefficient for the heat flux pipe to the air, and finally $\boldsymbol{f_{kl}}$ is the binary coefficient for heat flux due oversaturation.

Furthermore, a matrix was developed with ten lines and five columns that represent the ten possible modified cases (MC) holding the combinations of inclusion of the five studies under study. The next table represents the matrix, with the first line corresponding to the general case (GC):

Table 3.7: Binary matrix containing the general case and 10 modified cases.

Cases	f_{Rw}	f_{wl}	f_{vP}	f_{Rl}	f_{kl}
GC	1	1	1	1	1
MC 1	0	0	0	1	1
MC 2	0	0	1	1	1
MC 3	0	1	0	1	1
MC 4	0	1	1	1	1

MC 5	1	0	0	1	1
MC 6	1	0	1	1	1
MC 7	1	1	0	1	1
MC 8	1	1	1	0	0
MC 9	1	1	1	0	1
MC 10	1	1	1	1	0

Modified cases MC 1 till MC 7 considered the equation for air heat balance complete, combining the terms for the water heat balance equation. The last modified cases MC 8, MC 9 and MC 10 represent the possible combinations for the air heat balance equation, without changes in the water heat balance equation. The general and modified cases are listed below:

General case

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw}L_w - \dot{q}_{wL}B - \dot{q}_{vP}B + \dot{q}_w A_w) = 0 \quad (3.3)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl}L_l + \dot{q}_{wl}B + \xi \dot{q}_{kl}A_l) = 0 \quad (3.4)$$

Modified case 1

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw}L_w - \dot{q}_{wL}B - \dot{q}_{vP}B + \dot{q}_w A_w) = 0 \quad (3.5)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl}L_l + \dot{q}_{wl}B + \xi \dot{q}_{kl}A_l) = 0 \quad (3.6)$$

Modified case 2

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw}L_w - \dot{q}_{wL}B - \dot{q}_{vP}B + \dot{q}_w A_w) = 0 \quad (3.7)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl}L_l + \dot{q}_{wl}B + \xi \dot{q}_{kl}A_l) = 0 \quad (3.8)$$

Modified case 3

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw}L_w - \dot{q}_{wL}B - \dot{q}_{vP}B + \dot{q}_w A_w) = 0 \quad (3.9)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl}L_l + \dot{q}_{wl}B + \xi \dot{q}_{kl}A_l) = 0 \quad (3.10)$$

Modified case 4

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \quad (3.11)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \quad (3.12)$$

Modified case 5

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \quad (3.13)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \quad (3.14)$$

Modified case 6

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \quad (3.15)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \quad (3.16)$$

Modified case 7

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \quad (3.17)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \quad (3.18)$$

Modified case 8

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \quad (3.19)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \quad (3.20)$$

Modified case 9

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \quad (3.21)$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \quad (3.22)$$

Modified case 10

$$\frac{\partial(A_w T_w)}{\partial t} + \frac{\partial(Q_w T_w)}{\partial x} - \frac{1}{c_{p,w}\rho_w}(\dot{q}_{Rw} L_w - \dot{q}_{wL} B - \dot{q}_{vP} B + \dot{q}_w A_w) = 0 \tag{3.23}$$

$$A_l \frac{\partial T_l}{\partial t} + Q_l \frac{\partial T_l}{\partial x} - \frac{1}{c_{p,l}\rho_l}(\dot{q}_{Rl} L_l + \dot{q}_{wl} B + \xi \dot{q}_{kl} A_l) = 0 \tag{3.24}$$

3.2.2 Deviations

At this point, the general and each modified case were saved in respective structures. In order to compare all modified cases to the general case, an deviation measure was calculated for each. The deviation measure consisted in calculating the difference in all grid positions for temperature of air (T_w) and temperature of water (T_l) between general case (T_w^{GC} and T_l^{GC}) and modified case (T_w^{MC} and T_l^{MC}). These differences of values are done for all the time steps (and expressed in percentage), resulting a T_w deviation matrix and T_l deviation matrix with time steps as a row and the grid position as a columns (*Deviation*T_W and *Deviation*T_l). Afterwards, the average value of matrix elements (using function $\mathrm{mean2}$) and the standard deviation (function $\mathrm{std2}$) are calculated for both parameters T_w and T_l resulting in the *meanDeviation*T_W, *meanDeviation*T_l, *stdDeviation*T_W and *stdDeviation*T_l. Since there are ten modified cases this routine has a looping for them, ten times. Expressing this procedure in equations yields:

$$\mathrm{Deviation}T_w = \frac{T_w^{MC} - T_w^{GC}}{T_w^{GC}} \cdot 100 \tag{3.25}$$

$$\mathrm{Deviation}T_l = \frac{T_l^{MC} - T_l^{GC}}{T_l^{GC}} \cdot 100 \tag{3.26}$$

$$\mathrm{meanDeviation}T_w = \mathrm{mean2}(\mathrm{Deviation}T_w) \tag{3.27}$$

$$\mathrm{meanDeviation}T_l = \mathrm{mean2}(\mathrm{Deviation}T_l) \tag{3.28}$$

$$\mathrm{stdDeviation}T_w = \mathrm{std2}(\mathrm{Deviation}T_w) \tag{3.29}$$

$$\mathrm{stdDeviation}T_l = \mathrm{std2}(\mathrm{Deviation}T_l) \tag{3.30}$$

3.2.3 Plot deviation results

To show the deviation results would be helpful the results for each modified cases in the same figure to facility the comparison between each modified case.

The chart chosen was a bar chart using the functions *deviationbar* and *bar.* The vertical bars represent the mean value or average for T_w and T_l (*meanDeviation*T_W and *meanDeviation*T_l) and the deviation bars the standard deviation (*stdDeviation*T_W and *stdDeviation*T_l).

Chapter 4

Results

This chapter can be separated in two main approaches. The first one encloses the results for the steady situation, described in Section 4.1. These results show a simulation from the original code by Bas Wols as in Chapter 3 (Subsection 3.1.3) and contain the output variables along the length of the pipe and over time for a fixed position. In addition, the deviations associated with the removal of some terms in the Equations 2.19 and 2.20 are presented.

Section 4.2 reports the results for the implemented unsteady situation. The same pipe modelled in Section 4.1 is employed. Colour maps for water temperature (T_w) and water flow rate (Q_w) as functions of pipe position and time are shown for the general case and the modified cases.

Since the results in Section 4.2 verified some unexpected oscillatory behaviour, Section 4.3 concludes with an extra situation where the time step ($dt = dt_{CFL}/8$) is reduced, in order to improve the outcome.

4.1 Steady situation

4.1.1 Model results in position

The original code provided post-processing routines that showed the model results along the length of the pipe for a number of time instants defined with variable *num.nplot.*

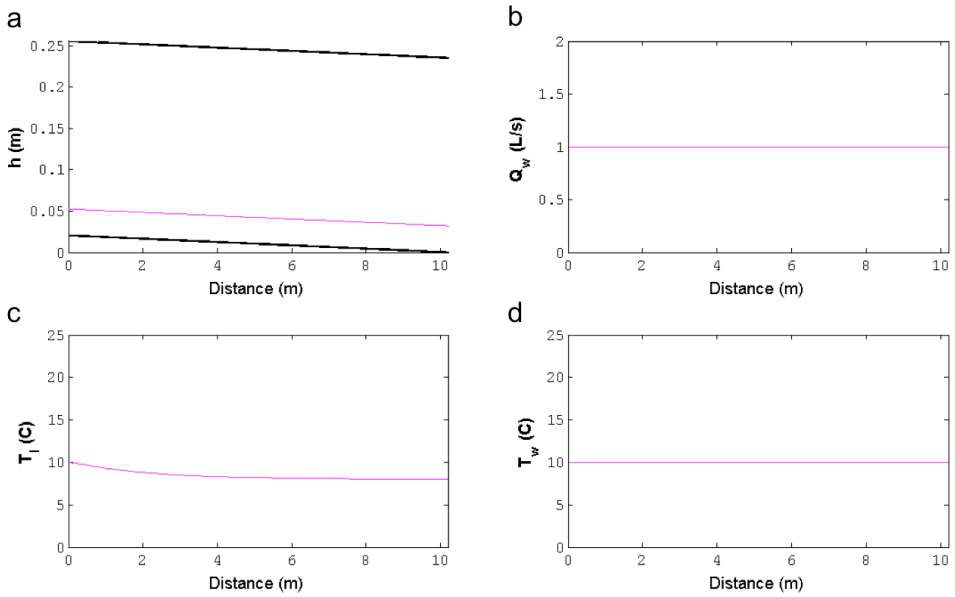

Figure 4.1: Representation in position of the pipe water depth (h) (4.1a), water flow rate (Q_w) (4.1b), air temperature (T_l) (4.1c) and water temperature (T_w) (4.1d), for a steady situation of the general case. Line colour denotes different time instants (five times depicted, however coincidental).

Figure 4.1 shows the predicted results for four variables: water depth (h), water flow rate (Q_w), air temperature (T_l), and water temperature (T_w), as function of pipe position, Different times of analyse (five times) are represented but since time variation is negligible, curves lay on top of each other.

Figure 4.1a shows h (purple line) and the bold lines represent the pipe walls position. Water height is less than 5cm. Regarding **Figure 4.1**b, Q_w is 1L/s and constant as expected. Air temperature T_l, shown in **Figure 4.1**c, is expected to suffer a slight reduction from 10ºC to approximately 8ºC. In **Figure 4.1**d displays T_w and hold a constant value of 10ºC along the length of the pipe and for different time instants.

The most important parameters here are T_w and Q_w, and they are constant as expected in a steady regime.

4.1.2 Model results in time

The same results from the previous subsection are shown here as function of time instead of position using the developed function *datatime*, that was described in Subsection 3.1.3.

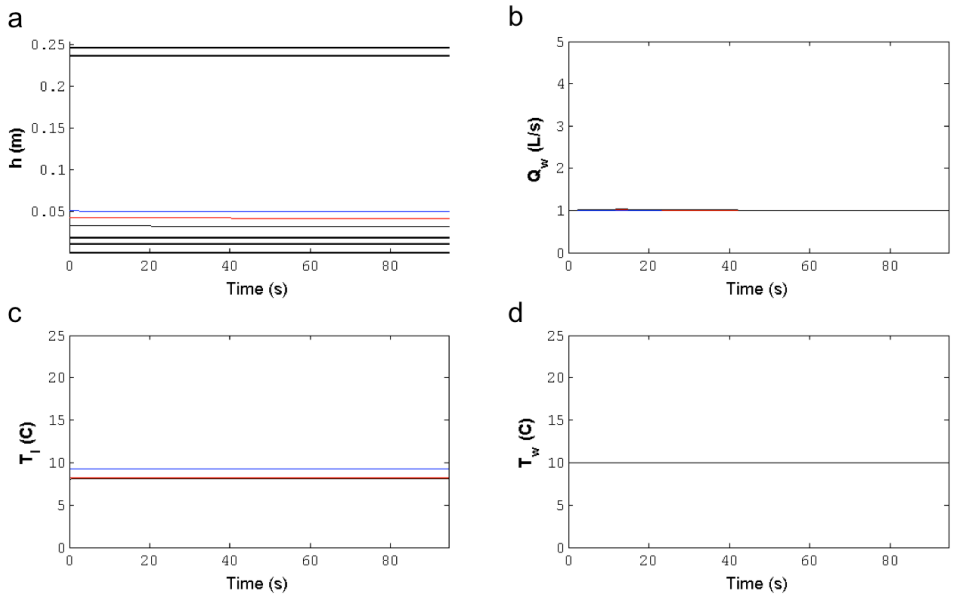

Figure 4.2: Representation in time for unsteady situation and general case for three positions in the pipe, $x = 1\text{m}$ in blue, $x = 5\text{m}$ in red and $x = 10\text{m}$ in black: water depth (h) (4.2a), water flow rate (Q_w) (4.2b), air temperature (T_l) (4.2c) and water temperature (T_w) (4.2d).

Figure 4.2 shows the same parameters of **Figure 4.1** (water depth (h), water flow rate (Q_w), air temperature (T_l) and water temperature (T_w)), however here is represented the variation in time for a fixed length or position in the pipe (x). In this case the values of x, they are: $x = 1\text{m}$, $x = 5\text{m}$ and $x = 10\text{m}$ were considered, depicted respectively with blue, red and black colours.

It is possible to observe in Figure 4.2a a reduction in h with time that follows the lower boundary of the pipe wall. Water flow rate Q_w is represented in Figure 4.2b where some visible variation observed, however negligible. Regarding T_l, in Figure 4.2c, the same slight reduction as reported in **Figure 4.1**c is present here. For T_w, Figure 4.2d, only the black line ($x = 10\text{m}$) is visible since the others two do not vary, laying on top of each other.

The potential of the developed routine *datatime* is not tangible in this section, since the results we obtained for the steady proved to have fairly constant trends in time and position. However Section 4.2.2.

4.1.3 Deviation results

Simulations for the steady case were performed for the modified cases, *i.e.*, for the different combinations of terms for the equations water heat balance and air heat balance as stated in Section 3.2. The difference in the output for the general and modified cases is shown in **Figure 4.3**.

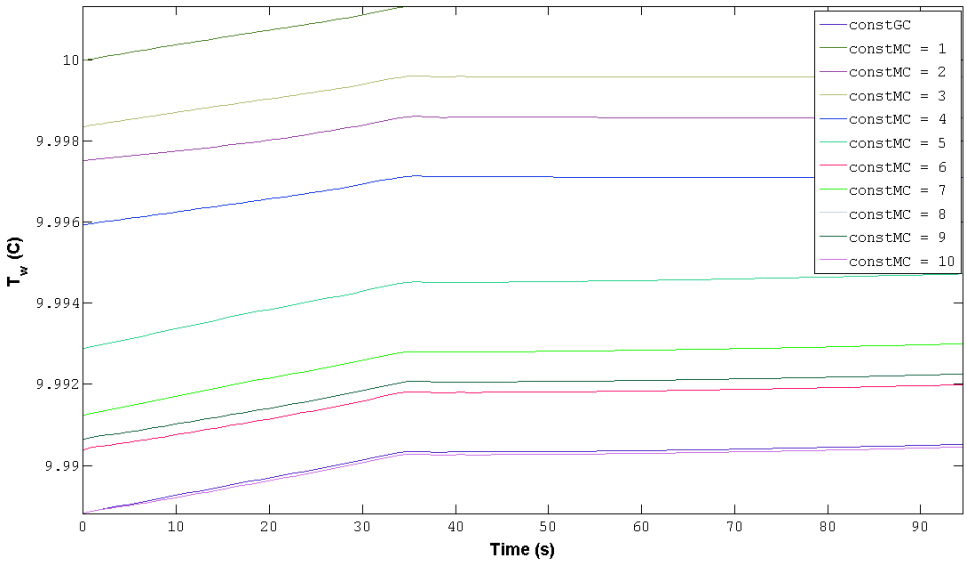

Figure 4.3: Water temperature (T_w) over the time for a fixed length $x = 10\text{m}$. Each line represents the 10 modified cases for steady situation.

Here, in **Figure 4.3** it is possible to see that there are slight variations in the predicted values of T_w, although looking at the scale, these values vary approximately between 10.002ºC and 9.989ºC. The water temperature for each modified cases is fairly the same as the general case with a maximum difference of 0.01 ºC. This shows that the dynamics in the system are consistent and completely disregarding the terms for the equation of water heat balance or the equation of air heat balance does not make an impact in our predictions. The same will not be verified in Section 4.2, where the unsteady case is considered.

In order to quantify the difference in the predictions from the modified cases to the general case, the deviation measure described in Subsection 3.2.2 was calculated for each one of them.

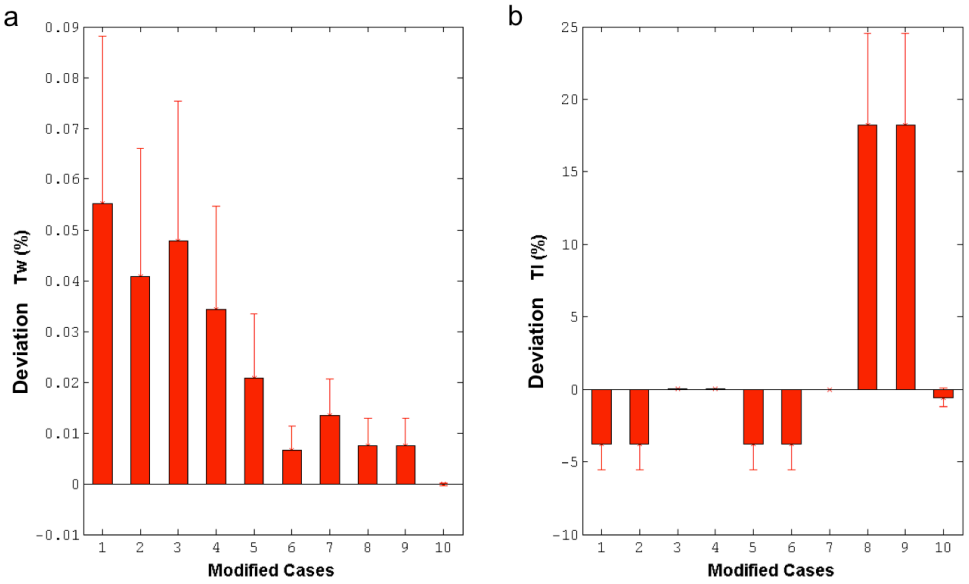

Figure 4.4: Difference between the general case and each modified case for the water temperature (T_w) (4.4a) and air temperature (T_l) (4.4b) and than the mean and the standard deviation. Steady situation.

Figure 4.4 shows these measures for the ten modified cases. The vertical bars represent the mean values for T_w and T_l, and the deviation bars the standard deviation. In order to remembering, in the modified cases MC 1 till MC 7 it was considered all the terms for air heat balance equation (Equation 2.20) and the possible combinations for the water heat balance equation (Equation 2.19). The opposite happens for the MC 8 to MC 10.

In **Figure 4.4**a, the modified cases with the highest deviations for T_w correspond to MC 1 to MC 4 where the term *heat flux pipe to water* ($\dot{q}_{Rw}$) was neglected. For modified cases MC 5 to MC7 the highest deviations occur when both the terms for *heat flux water to air* ($\dot{q}_{wL}$) and *evaporation/ condensation* ($\dot{q}_{vP}$) were neglected. The remaining modified cases (MC 8 to MC 10) correspond to the terms where all water heat balance equation were considered. The *heat flux pipe to air* seems to be more important than the *heat flux due to oversaturation*.

Regarding T_l, **Figure 4.4**b, the largest deviations occur for changes in the equation of heat balance in air (Equation 2.19), namely in MC 8 and MC 9 where the term *heat flux pipe to air* ($\dot{q}_{Rl}$) was not included and the deviations correspond to an overvalued of the air temperature. In the MC 10 these terms were assumed and the term *heat flux due oversaturation* ($\dot{q}_{kl}$) was neglected, the deviations indicate an undervalued situation for the air temperature. For the others modified cases (MC1 to MC7) the deviations were higher for the ones that did not include the term *heat flux water to air* ($\dot{q}_{wL}$) was neglected, *i.e.*, MC 1, MC 2, MC 5 and MC 6. The rest of the modified cases are related to the water heat balance and they do not have impact in the air temperature.

4.2 Unsteady situation

This section presents the results for an unsteady regime, as described in Subsection 3.1.4, and it is arranged in six subsections. The first two, similarly as in the previous section, report the results found for our model variables as function of time and position respectively in the general case. Colour maps were developed in order to present the model results for the water temperature (T_w) and water flow rate (Q_w) for the general case in both time and distance – Subsection 4.2.3. The same type of representation is employed for the modified cases for the parameter water temperature in Subsection 4.2.4. Subsection 4.2.5 subsequently reports the calculated difference deviations for these modified cases.

4.2.1 Model results in position

The following figure illustrated the model results as a function of position, for the unsteady situation.

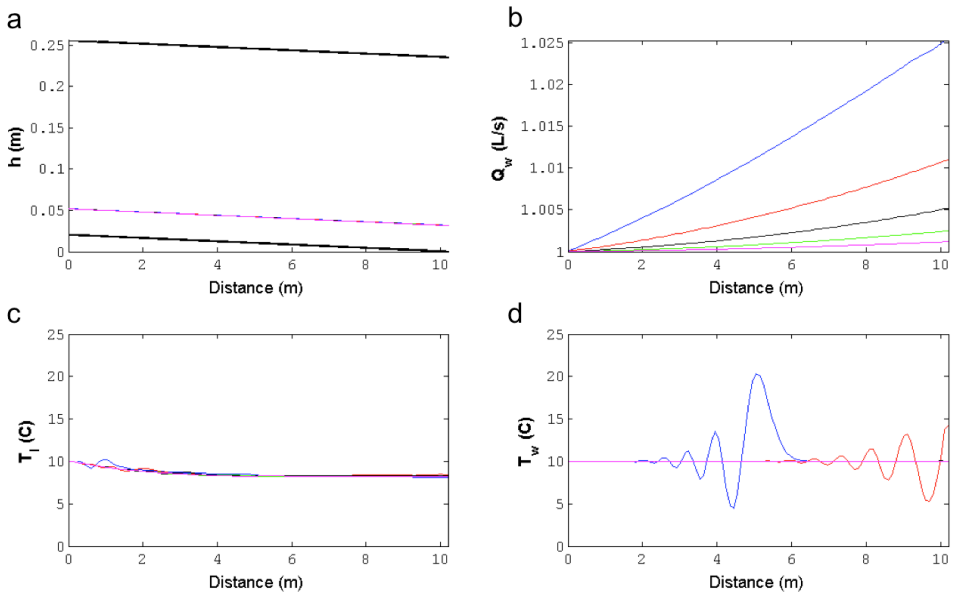

Figure 4.5: Representation in position of the pipe water depth (h) (4.5a), water flow rate (Q_w) (4.5b), air temperature (T_l) (4.5c) and water temperature (T_w) (4.5d), for an unsteady situation of the general case. Line colour denotes different time instants (five times depicted, blue, red, black, green and purple in ascending order).

Figure 4.5 shows four parameters as function of pipe position: water depth (h), water flow rate (Q_w), air temperature (T_l), and water temperature (T_w). Five time instants of the analysis are represented in different colour lines.

Figure 4.5a shows the water depth h along the pipe length, with a practically constant value along the length of the pipe for all different time instants. In a discharge, such behaviour is not expected. As it will be shown in Section 4.3, this might be due to a problem in the estimation of the minimum time step dt as stated in Section 2.4.

The water flow rate Q_w increases non-linearly along the pipe for all time instants – **Figure 4.5**b. The highest values achieved corresponded to the blue line, *i.e.*, the first time instant of the analysis, where a 2.5% increase is verified. The magnitude of each peak for Q_w decrease over the time, as the discharge passes though the pipe.

For T_l, it is visible in **Figure 4.5**c, that its profile is different for each reported times. However in general the water temperature tends to decrease along the pipe length. Regarding **Figure 4.5**d, the highest peak of T_w corresponds to the discharge, however extras oscillations are present. These could arise due to several reasons, such as course grid or large time step. For the last time instant (purple line) T_w seems to be constant. The green line and black line are coincident with the purple line.

4.2.2 Model results in time

The same results from the previous subsection are shown here as function of time, instead of position, using the developed function *datatime*.

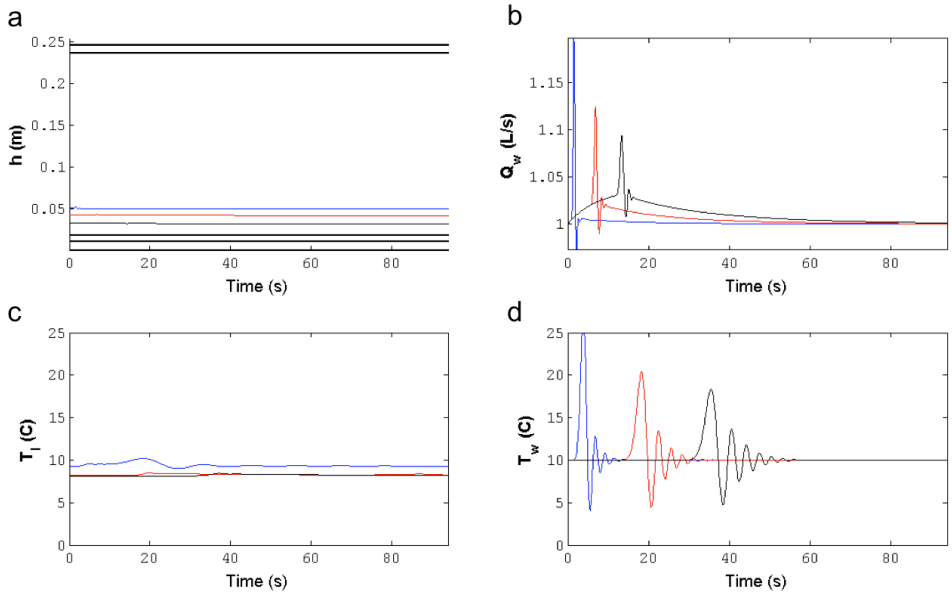

Figure 4.6: Representation in time for unsteady situation and general case for three positions in the pipe, $x = 1\text{m}$ in blue, $x = 5\text{m}$ in red and $x = 10\text{m}$ in black: water depth (h) (4.6a), water flow rate (Q_w) (4.6b), air temperature (T_l) (4.6c) and water temperature (T_w) (4.6d).

Figure 4.6 shows the same parameters of **Figure 4.5** (water depth (h), water flow rate (Q_w), air temperature (T_l) and water temperature (T_w)), however here these are represented as function of time for a fixed position in the pipe (x). The same values of x from Section 4.1.2 were considered: $x = 1\text{m}$, $x = 5\text{m}$ and $x = 10\text{m}$, respectively blue, red and black colours.

Figure 4.6a displays h following the lower boundary of the pipe wall, maintaining its value constant as reported in the previous section (again, not a reasonable result. Water depth

fluctuations are negligible and do not accurately represent the discharge). For Q_w, **Figure 4.6**b, it is visible some initial steep peak followed by smaller oscillations. As mentioned in the previous subsection, time step and grid size could be the reason of such behaviour. Post-peak wakes are also present in the water temperature, in **Figure 4.6**d. The evolution of the position of discharge wave is visible similarly for the plots regarding Q_w and T_w. The initial peak and further oscillations increase their position in time, while reducing the envelope, *i.e.*, reducing water flow rate and temperature. This is probably due to thermal exchange with the pipe, we will confirm in Section 4.2.5.

The potential of the developed routine *datatime* is evident in this section. The results we obtained for the unsteady situation, where a discharge is considered, have a far more complex evolution with time. A routine that allows the user to see the profile of system variables with time helped in the detection of the wave ripples artefacts that were obtained. However a full picture of the results will be better obtained with a colour map, described as follows.

4.2.3 Colour map for general case

In order to visualize the variables in interest for both time and position, the model results for the general case were plotted in a colour map, as shown in **Figure 4.7**.

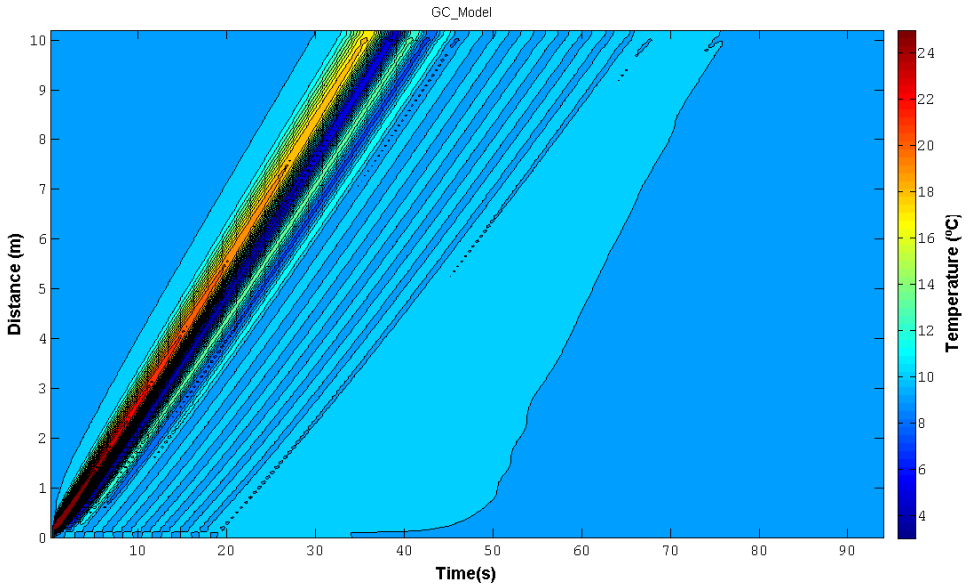

Figure 4.7: Colour map representation of water temperature (T_w) for the general case of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

The **Figure 4.7** illustrates a colour representation of the water temperature (T_w) for the general case of steady situation. The vertical axis represents the position along the pipe, the horizontal axis represents time and the colours define the temperature of water in Celsius degrees. In all

colour maps warm colours represent high values for the variable, whereas cold colour represent low values.

The warmer coloured peak corresponds to water temperatures that result from the main discharge. It is possible to realise that the temperature wave position covers the full length of the pipe in less than 40 seconds. As predictable, the oscillations reported in **Figure 4.5** and **Figure 4.6** are visible in this figure. The previously mentioned decrease in water temperature is also perceptible here, with the peak colour varying from dark red to yellow while the discharge goes through the pipe.

This type representation allows the user to have the full picture of the pipe in terms of spatial and time behaviour. However, *sewer_temp_model* and *datatime* are essential to accurately inspect the profile of the variables.

The following figure represents the colour map for the obtained water flow rate (Q_w) predictions.

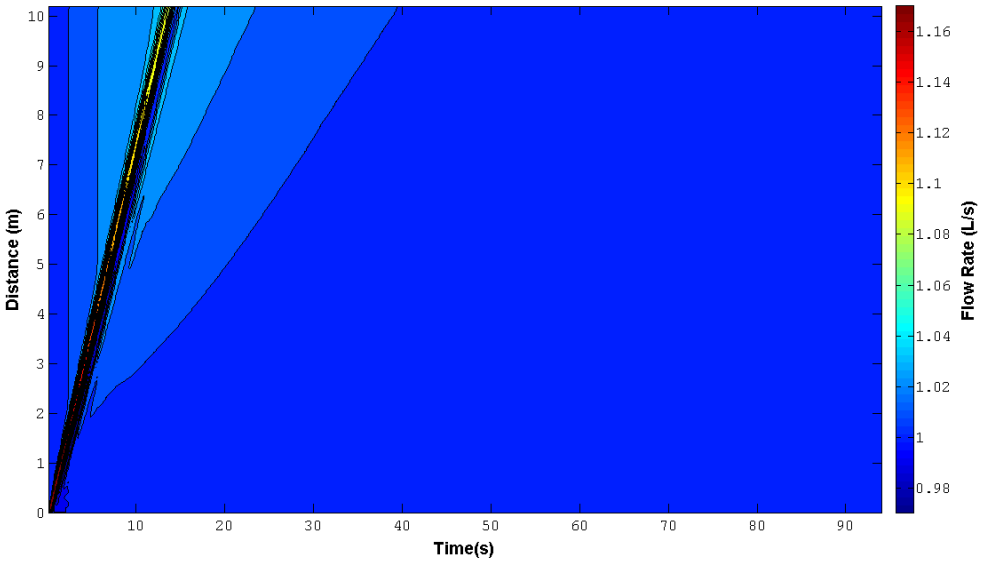

Figure 4.8: Colour map representation of water flow rate (Q_w) for the general case of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

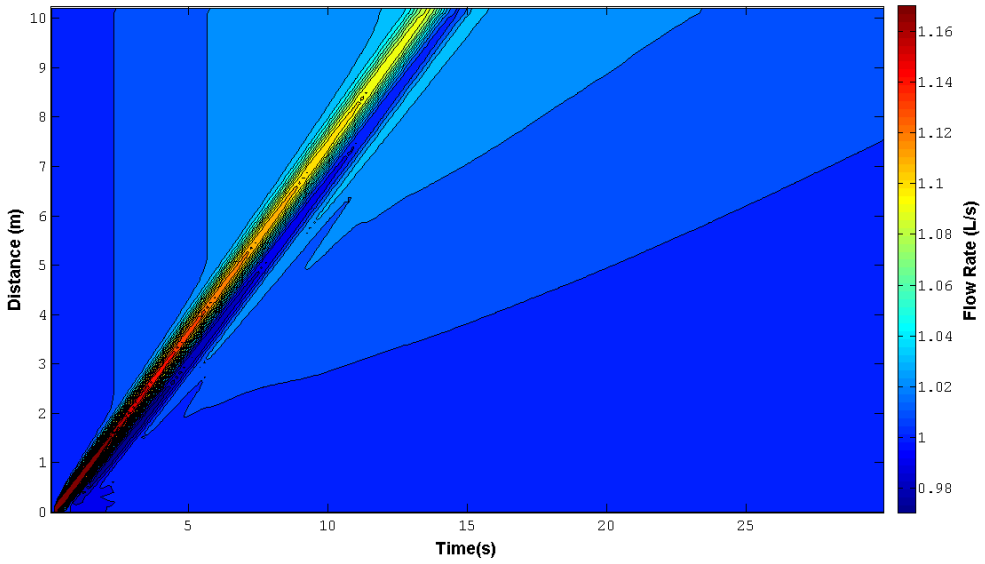

Figure 4.9: Zoom of the previous figure. Water flow rate (Q_w) for the general case of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

A similar behaviour as observed for T_w is present here, **Figure 4.9**. Initial high water flow rate values are reduced while the discharge position increases, that is, losing momentum throughout the pipe. The discharge takes about 15 seconds to cover the whole distance. This contrasts with the 40 seconds of the temperature wave, most likely due to the different boundary conditions given for T_w and Q_w at the inlet, as described in Subsection 3.1.4.

The next two subsections – Sections 4.2.4 and 4.2.5 – will show what happens when the terms of the water heat balance and air heat balance equations are removed. With the help of the colour maps and deviation calculation, we will be able to determine which equation terms are the most relevant for simulations of unsteady flow conditions.

4.2.4 Colour maps for modified cases

In the next figures, the ten modified cases are considered by displaying the results for water temperature (T_w) in the unsteady situation.

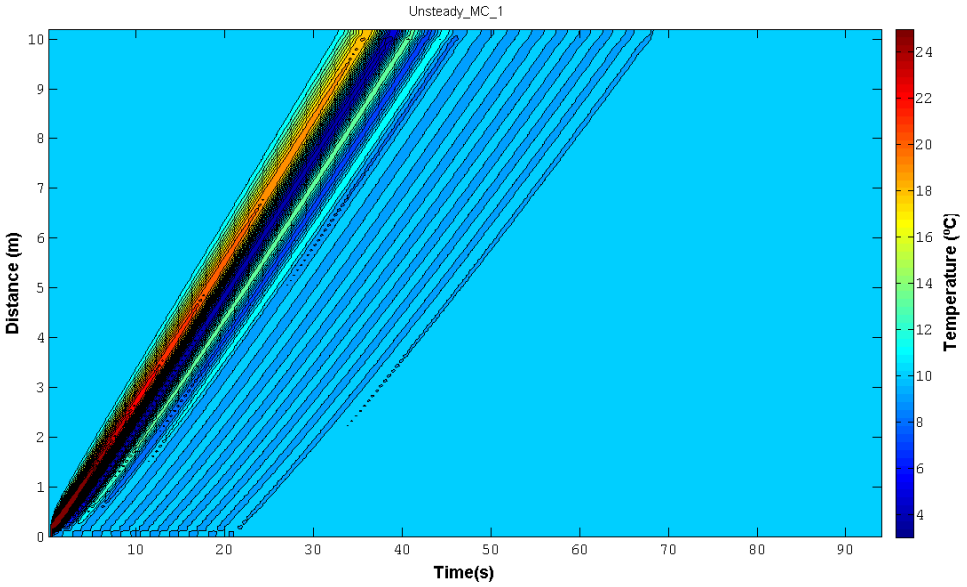

Figure 4.10: Colour map representation of water temperature (T_w) for the modified case number 1 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

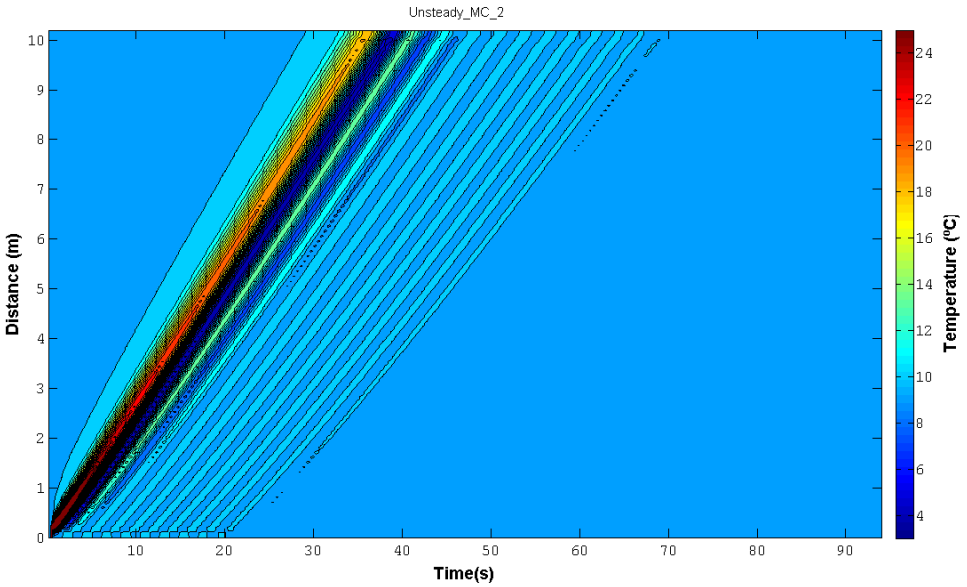

Figure 4.11: Colour map representation of water temperature (T_w) for the modified case number 2 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

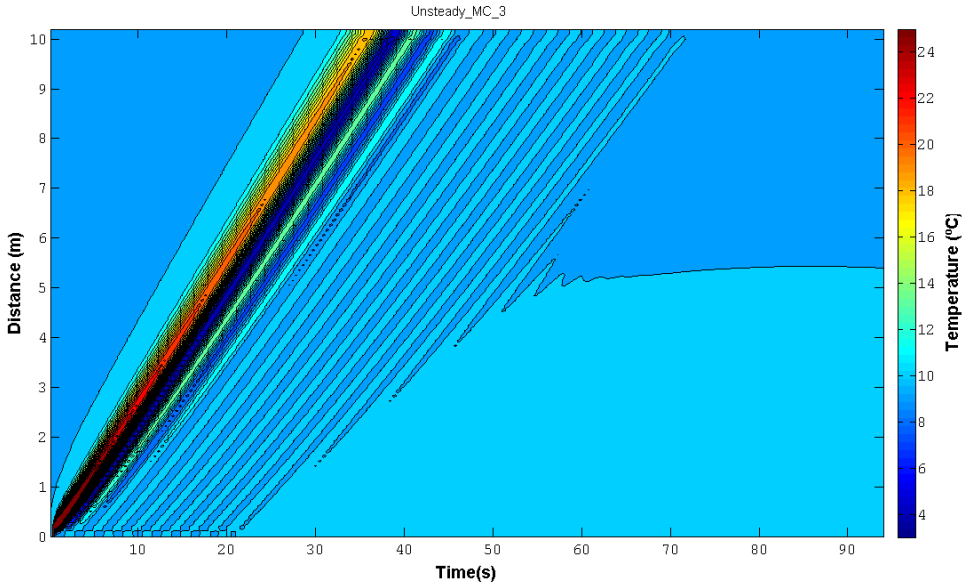

Figure 4.12: Colour map representation of water temperature (T_w) for the modified case number 3 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

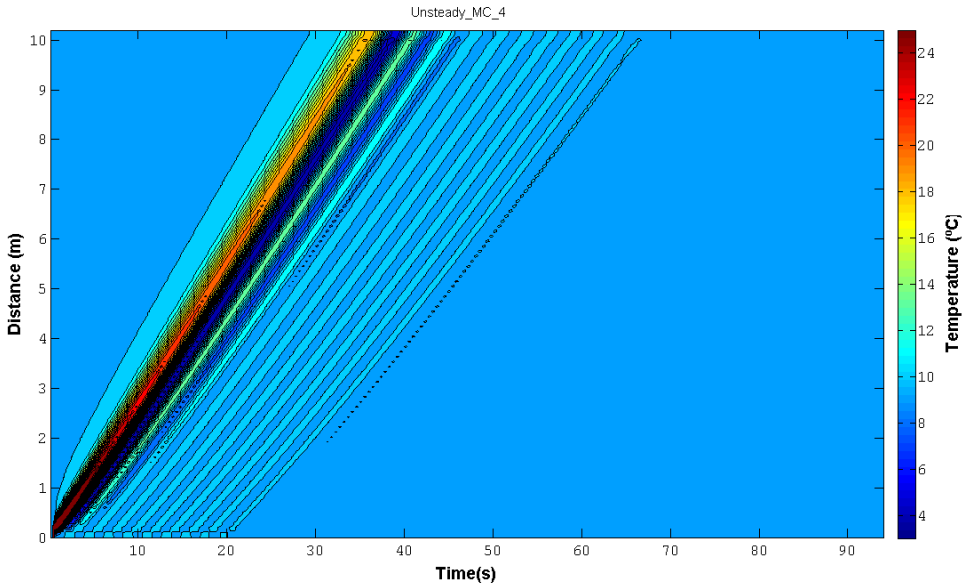

Figure 4.13: Colour map representation of water temperature (T_w) for the modified case number 4 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

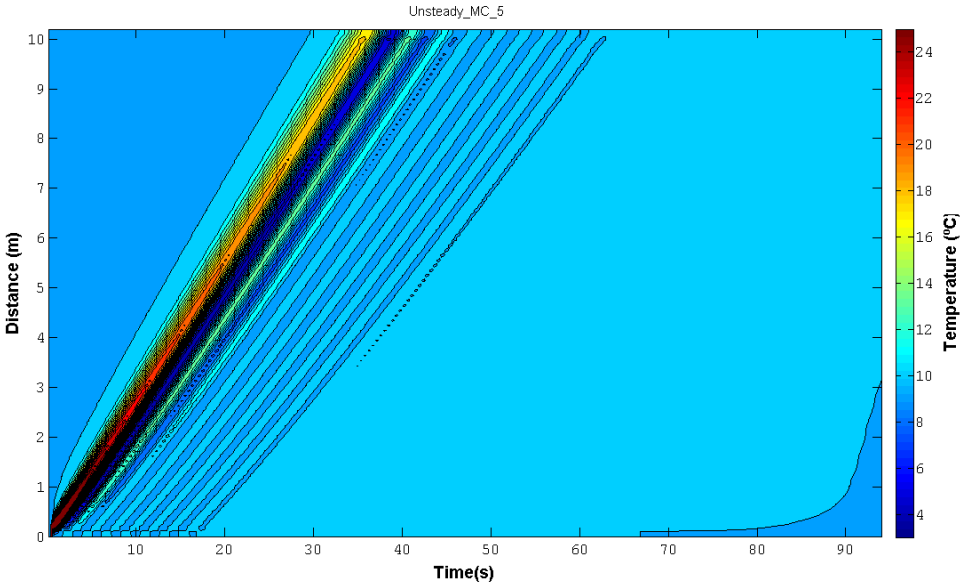

Figure 4.14: Colour map representation of water temperature (T_w) for the modified case number 5 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

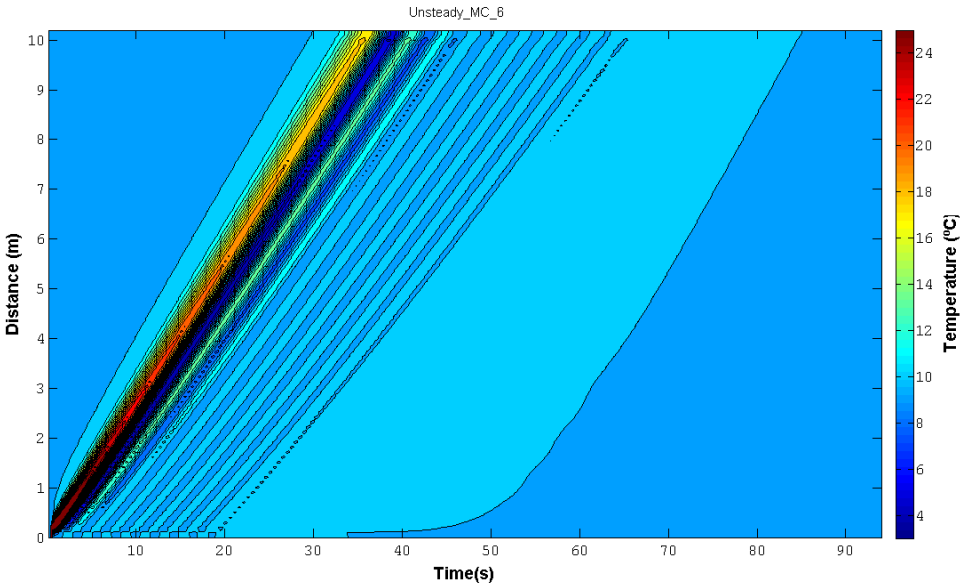

Figure 4.15: Colour map representation of water temperature (T_w) for the modified case number 6 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

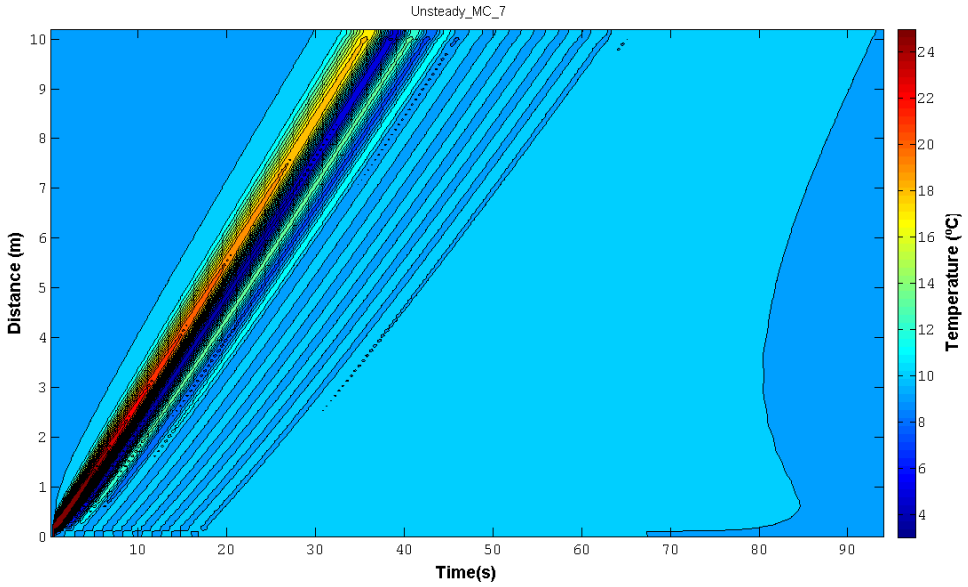

Figure 4.16: Colour map representation of water temperature (T_w) for the modified case number 7 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

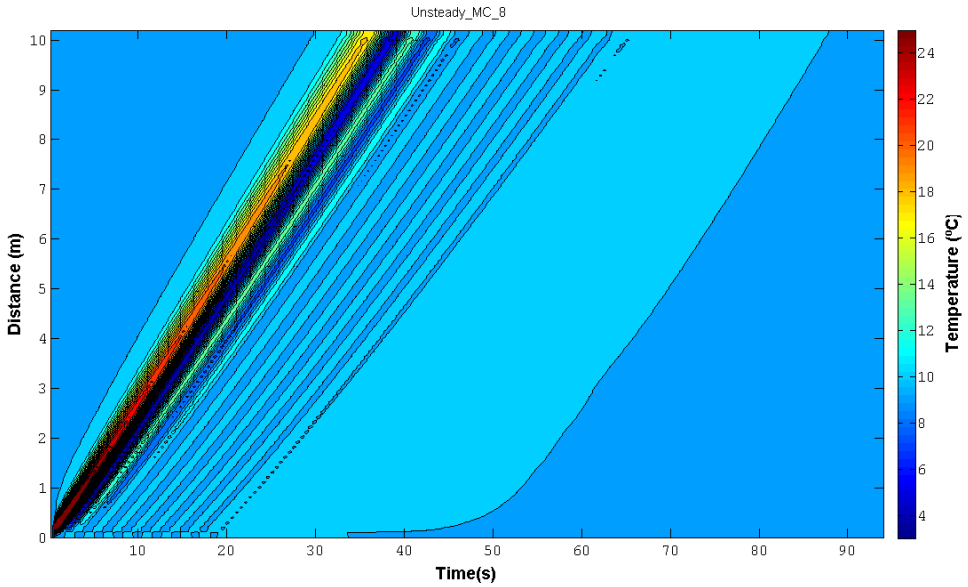

Figure 4.17: Colour map representation of water temperature (T_w) for the modified case number 8 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

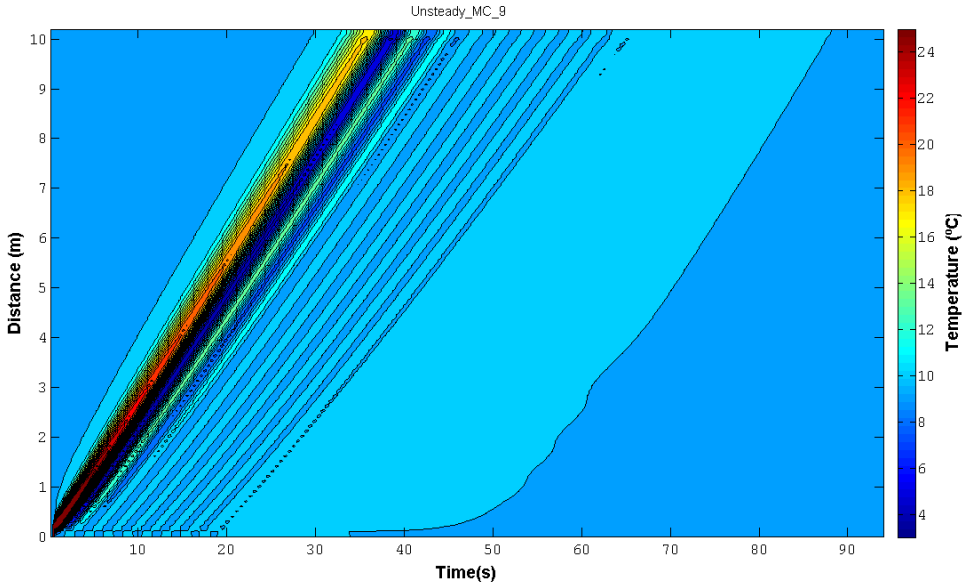

Figure 4.18: Colour map representation of water temperature (T_w) for the modified case number 9 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

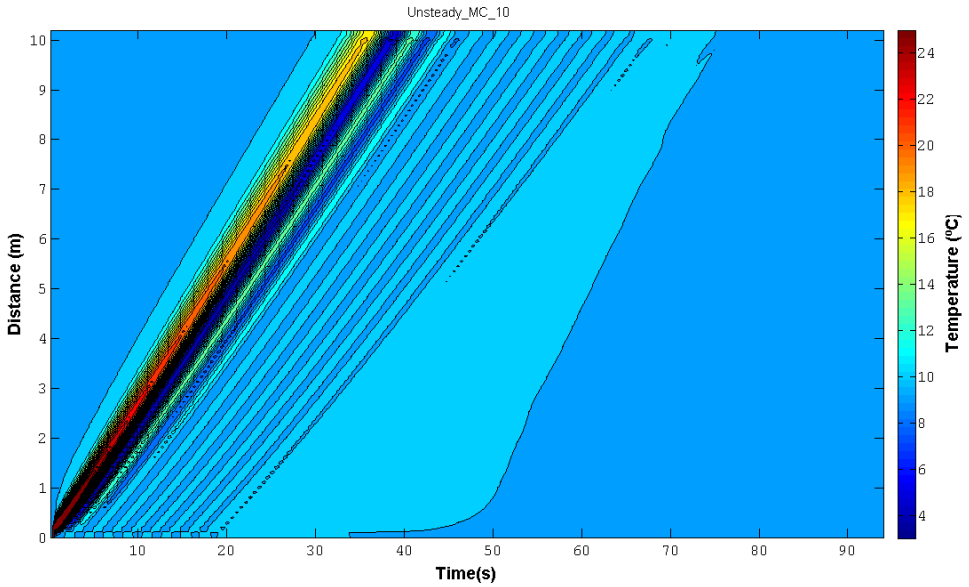

Figure 4.19: Colour map representation of water temperature (T_w) for the modified case number 10 of the unsteady situation. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

Colour representations of water temperature (T_w) are presented from **Figure 4.10** until **Figure 4.19** corresponding to modified cases MC1 till MC10 respectively. Remember that the vertical axis represents the position of pipe, the horizontal axis represents time and the colours define the temperature of water in Celsius degrees.

Analysing each figure is notable the differences between each modified cases on the water temperature T_w, however these differences are not measurable only for observing the figures. The scale for water temperature was maintained in order to ease the comparison. However it is difficult to compare all of these results to the ones obtained in the general case, as shown in **Figure 4.7**. In order to quantify these differences, the same deviations measures from the previous section are presented for the unsteady case in the following one.

4.2.5 Deviation results

In order to quantify the importance of the various terms of the water heat balance and air heat balance equations, the deviation difference in relation to the general case, as described in Section 3.2.2, was calculated for the predictions of the modified cases. The deviations obtained for comparisons in the water temperature $T_w$and air temperature T_l are displayed in **Figure 4.20**.

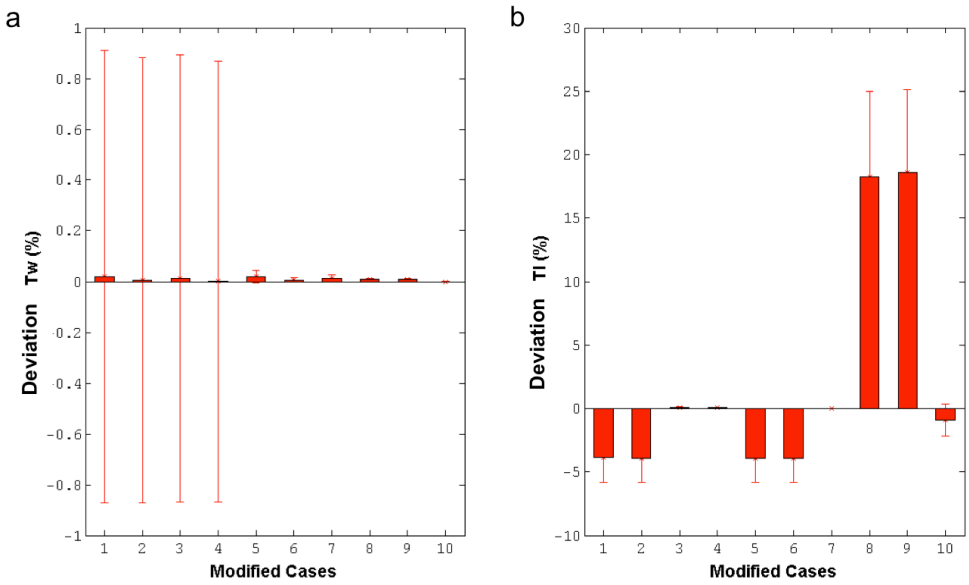

Figure 4.20: Difference between the general case and each modified case for the water temperature (4.20a) and air temperature (4.20b). The difference mean is displayed by the vertical bars and the standard deviation by the deviation bars.

The vertical bars represent the mean values and the deviation bars the standard deviation for T_w and T_l respectively in **Figure 4.20**a and b. To remembering, in the modified cases MC 1 till MC 7 it was considered all the terms for air heat balance equation (Equation 2.20) and the possible combinations for the water heat balance equation (Equation 2.19). The opposite happens for the MC 8 to MC 10.

In the case of T_w, **Figure 4.20**a, larger deviations were obtained for MC 1 to MC 4, since the term *heat flux pipe to water* ($\dot{q}_{Rw}$) was neglected for these four cases. Regarding the other cases where the terms associated with water heat balanced were modified, MC 5, MC 6 and MC 7, the deviation were lower since the $\dot{q}_{Rw}$ was considered. For these three modified cases

the deviation was lightly higher for the situation where the term for *evaporation/condensation* ($\dot{q}_{vP}$) was neglectd (MC 5 and MC 7) comparing the situation when it was assumed in MC 6 and the *heat flux water to air* ($\dot{q}_{wL}$) neglected. Between these two terms the $\dot{q}_{vP}$ seems to have a main role when compared with the $\dot{q}_{wL}$, however this diference is quite small. For MC 8 to MC 10 the deviations were lower since the all the water air terms were considered.

Regarding T_l, **Figure 4.20**b, the deviations behaviour is the same as the **Figure 4.4**b for the steady situation. Here the term *heat flux pipe to air* ($\dot{q}_{Rl}$) seems to be essential flowed by *heat flux water to air* ($\dot{q}_{wL}$).

The magnitude of the deviations in T_w is higher for the unsteady situation than to the steady situation, therefore the deviations for the unsteady situation are more relevant. In conclusion, for the water temperature the crucial term is the *heat flux pipe to water* ($\dot{q}_{Rw}$) and for the air temperature changes in *heat flux pipe to air* ($\dot{q}_{Rl}$) and *heat flux water to air* ($\dot{q}_{wL}$) are the ones with the most significant impact. The term *heat flux water to air* ($\dot{q}_{wL}$) appears in both equations and, despite of the fact that it does not have an impact in water temperature, it should be considered since it has an influence in air temperature.

4.3 Unsteady situation for smaller time step

The numerical artefacts present in Section 4.2 are likely to be associated with an overestimation of the minimum time step for the finite differences method (Lax-Wendroff method), leading to estimations that are unrealistic. In order to prove this hypothesis and with the intention of reducing the oscillations in the results, a smaller dt was chosen for the unsteady general case. To do this, the minimum time step defined in Section 2.4 was divided by an arbitrary integer, in this case eight. Therefore $dt = dt_{CFL}/8$, where dt_{CFL} is the minimum time step estimated with the Courant-Friedrich-Levy condition. **Figure 4.21** is a representation in position of the pipe, for water depth (h), water flow rate (Q_w), air temperature (T_l) and water temperature (T_w). Five time instances are represented for the colour lines, similarly to what was done in previous sections.

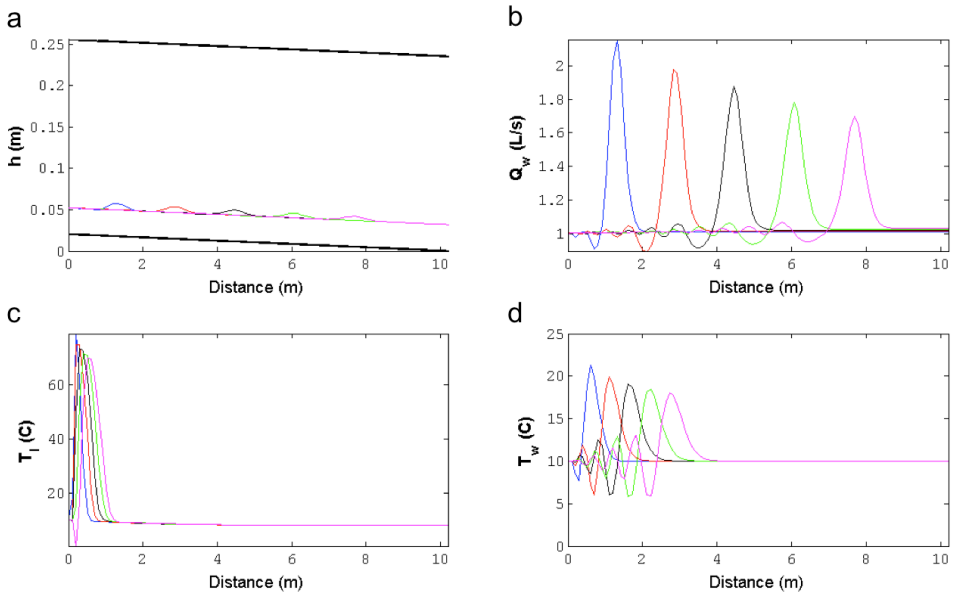

Figure 4.21: Representation in position of the pipe for water depth (h)(4.21a), water flow rate (Q_w) (4.21b), air temperature (T_l) (4.21c) and water temperature (T_w) (4.21d) the lines colours denote different times analyse. Unsteady situation and step time $dt = dt_{CFL}/8$, where dt_{CFL} is the minimum time step estimated with the Courant-Friedrich-Levy condition.

For, **Figure 4.21**a, the water height along the pipe is no longer constant comparing with **Figure 4.5**. A wave of fluid is now noticeable, being able to simulate the discharge variation in fluid content, and the position of this wave changes with time, clarifying the discharge position in the pipe. An even smaller dt might be necessary in order to obtain a reasonable value for h.

Figure 4.21b shows the estimations for Q_w. The oscillations were smoothed and after the main discharge peak a constant line was achieved. The highest values obtained were for the blue line (first time instant displayed) and as in previous simulations the value of Q_w decreases over the time, as momentum is lost along the pipe.

Regarding **Figure 4.21**c, a similar profile for each time instants is obtained for T_l. However it is noticeable that the distance covered is smaller than for Q_w and T_w (in **Figure 4.21**d) and in the simulations in Section 4.2. This, again, could be related to the boundary conditions defined in the inlet of the pipe. A more reasonable Gaussian, with more realistic parameters, should be defined for both Q_w and T_w. Nevertheless the same profile shape is present for Q_w, T_w and T_l.

Figure 4.22 shows the same parameters of **Figure 4.21** however here is represented the variation in time for a fixed length or position in the pipe (x), plotted with the function *datatime*. In this the values of x, they are: $x = 1\text{m}$, $x = 5\text{m}$ and $x = 10\text{m}$ were considered, depicted respectively with blue, red and black colours.

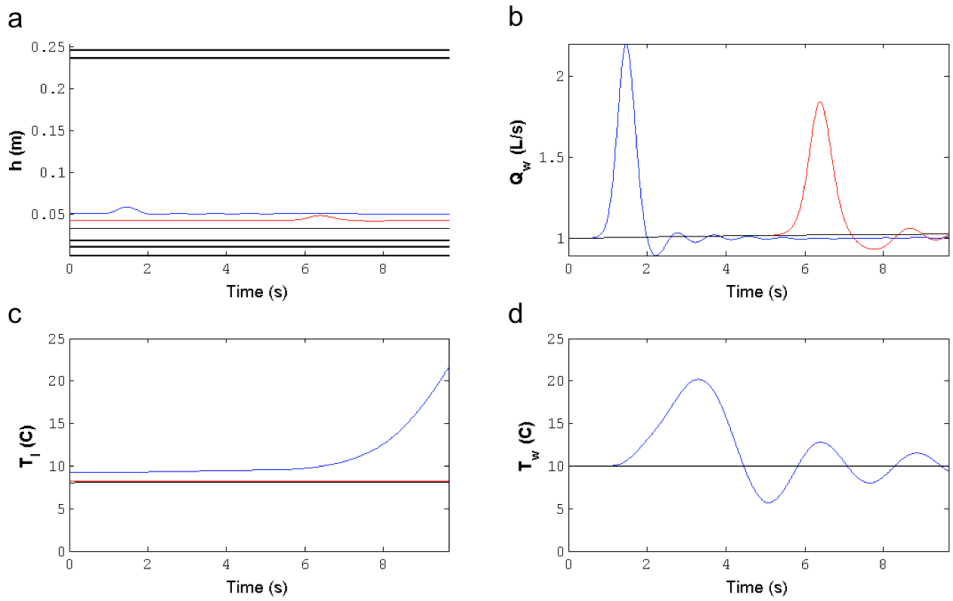

Figure 4.22: General case. Representation in time for three analyses in time, x = 1m in blue, x = 5m in red and x = 10m in black, for water depth (h) (4.22a), water flow rate (Q_w) (4.22b), air temperature (T_l) (4.22c) and water temperature (T_w) (4.22d), for unsteady situation and step time $dt = dt_{CFL}/8$, where dt_{CFL} is the minimum time step estimated with the Courant-Friedrich-Levy condition.

Figure 4.22a represents the water depth (h) and it is visible for different values of x the water height is varying over the time, as reported in the previous figure. The wave can be distinguished for $x = 1\text{m}$ (blue line) and $x = 5\text{m}$ (red line). At $x = 10\text{m}$ no variation is visible since the discharge does not reach this point. This is due to the fact that dt was reduced 8-fold, but the number of steps was kept constant. Therefore only 10 seconds of analysis were processed. For the water flow rate Q_w, as in **Figure 4.22**b, it is visible two peaks and smooth oscillations, however the situation for $x = 10\text{m}$ (black line) is constant again, for the same reasons as previously. Considering the **Figure 4.22**c, air temperature (T_l) has a curious pattern due to the fact that the amount of time analysed was not enough to enclose the passage of the discharge.

Finally, in **Figure 4.22**d, for the last parameter, water temperature (T_w), an unsteady regime can be distinguished for $x = 1\text{m}$. It is not possible distinguished the red line since it should be overlaid at 10 ºC.

Colour map representations for T_w and Q_w, in this extra scenario, are present in **Figure 4.23** and **Figure 4.24** respectively. Again the vertical axis represents the position in the pipe and the horizontal axis represents time. The colours define the temperature of water in Celsius degrees.

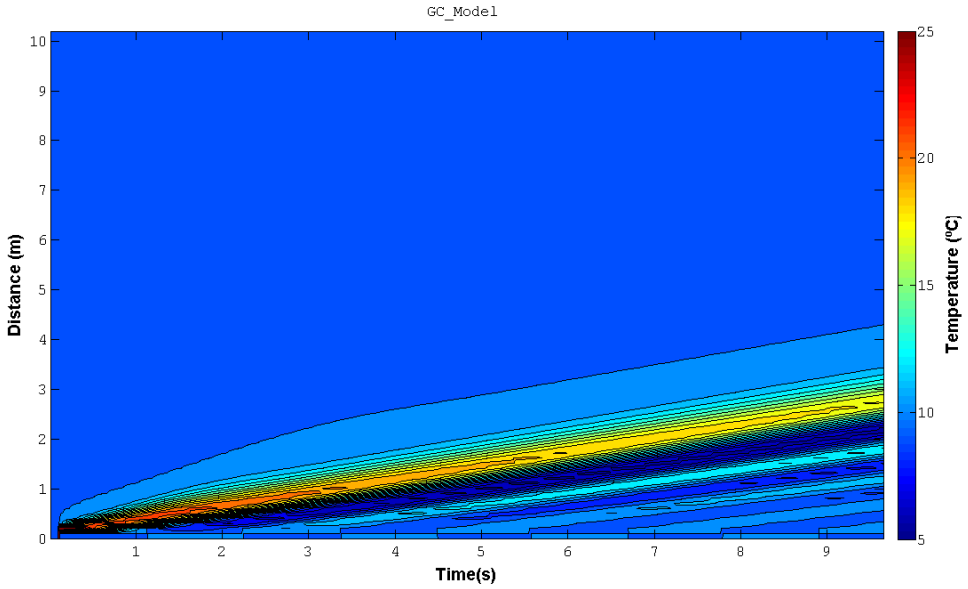

Figure 4.23: Colour map representation of water temperature (T_w) for unsteady situation of general case and step time $dt = dt_{CFL}/8$, where dt_{CFL} is the minimum time step estimated with the Courant-Friedrich-Levy condition. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

By comparing **Figure 4.23** with **Figure 4.7** one can notice that the oscillations are still present. Therefore the reduction in the duration of the time step was not enough to overcome this point. A further reduction in dt and/or an increase in the grid size might succeed. Another more obvious observation is the fact that the temperature fluctuations that are created from the discharge do not covers the whole length of the pipe, confirming the observations in **Figure 4.21** and **Figure 4.22**.

As mentioned, **Figure 4.24** is the colour map representation for the water flow rate Q_w, expressed in litres per second. The first observation is that the reduction of dt duration improved the profile of the predictions, since the steep profile variations visible in **Figure 4.6**b are not present here, with a reduction in magnitude of the post-peak wave.

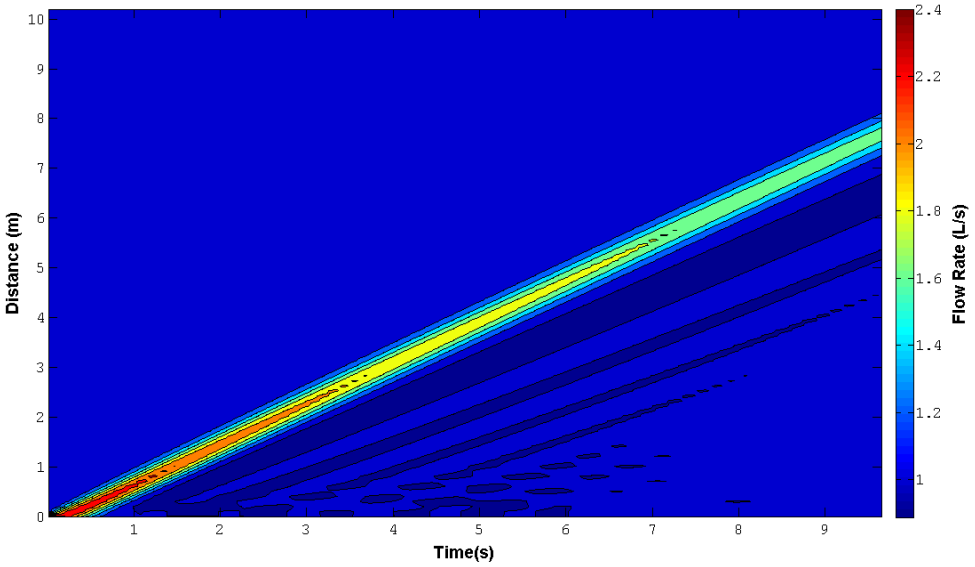

Figure 4.24: Colour map representation of water flow rate (Q_w) for unsteady situation of general case and step time $dt = dt_{CFL}/8$, where dt_{CFL} is the minimum time step estimated with the Courant-Friedrich-Levy condition. Vertical axis represent position along the pipe and horizontal axis represent time, the gradient of colour represents the temperature in Celsius degree.

The reduction of the time step contributed to improve the obtained results. However this improvement was not marked for all situations. A similar behaviour was found overall, however the extra oscillations are smoother, namely for water temperature. This algorithm will need further inspection through sensitivity studies to the grid size and time step parameters. These have a significant impact in the accuracy of the system solutions and require additional examination. Also, the boundary conditions defined for the unsteady water flow and water temperature should be refined, in order to feed the algorithm will realistic input.

Chapter 5

Discussion

The main topic of this work is heat recovery from wastewater and it consists in numerical modelling of energy changes in a pipe. A special focus on water temperature (T_w) and water flow rate (Q_w) was given. Representations of air temperature (T_l) and water depth (h) were also included in our results.

Results for the steady condition show the initial status of the model and code, at the beginning of this work, as developed by Bas Wols at KWR Water Research Institute (Nieuwegein, The Netherlands). In order to simulate a continuous flow condition, the boundary conditions (BC) were constant in time. As expected, the dynamics of this system is not the most interesting and obviously far from a real discharge situation. The predictions for the variables were constant when represented in length and in time. Therefore, the convenience of routine *datatime* developed in this project was not obvious at this point understandably.

The calculated difference deviations in the steady situation for T_l had the highest values for modified cases MC8 and MC9, when the binary coefficient $\boldsymbol{f_{Rl}}$ was zero, *i.e.* when *heat flux pipe to air* ($\dot{q}_{Rl}$) was neglected. In terms of impact on the predictions for the steady case, this term was followed by the term *heat flux water to air* ($\dot{q}_{wL}$). The calculated deviations were very small, due to the fact that the boundary conditions were constant. Therefore the removal of these terms did not have a major impact. However, the highest deviation values for T_w were obtained for MC1 to MC4, when the binary coefficient $\boldsymbol{f_{Rw}}$ was zero. This corresponded to the systems without the term *heat flux pipe to water* ($\dot{q}_{Rw}$), and therefore affecting more importantly the water temperature.

At this point we can state that in the steady case the terms heat flux pipe to air and heat flux pipe to water term are respectively the most relevant terms for the calculation of T_l and T_w respectively, although for the T_w the magnitude of the deviations were likely to be too small to

be considered. The modelling of steady flow conditions is therefore not very realistic and relevant in the field of wastewater heat recovery modelling.

In order to simulate a discharge, the boundary conditions applied at the begging of the pipe were modified and a Gaussian function was imposed to a baseline for T_w and Q_w with the intention of having varying water temperature and water flow rate. The unsteady flow situation was successfully implemented with fluctuations observed for Q_w and T_w and, consequently in T_l and h consequently.

A main wave in both Q_w and T_w is visible, created from the changing BC and the expected cooling-down effect was noticed. Reductions in both variables along the pipe proved that the algorithm successfully modelled energy conduction in the pipe, since heat and mass balance were reduced. This is consistent with previous studies that stated that wastewater temperature in the sewer is mostly affected by the heat exchange from water and the air duct, water evaporation and heat transfer through the pipe walls (Bischofberger and Seyfied (1984) in Dürrenmatt & Gujer, 2006).

The rate at which the reduction in both variables happens is a matter of further debate and needs validation in future developments of this work. At this point the routine *datatime* proved to be very useful. The plotting routines that were implemented previously were not suitable to understand the evolution in time of our variables, since you have to choose a limited amount of time intervals to be plotted.

In terms of the deviations of the modified cases in the unsteady situation, when inspecting the differences in water temperature, the crucial term is the *heat flux pipe to water* ($\dot{q}_{Rw}$). For the air temperature the terms that correspond to changes in *heat flux pipe to air* ($\dot{q}_{Rl}$) and *heat flux water to air* ($\dot{q}_{wL}$) are the ones with the most significant impact. Note that the term *heat flux water to air* ($\dot{q}_{wL}$) appears in both equation and even this term does not have a impact in water temperature it should be considered since he has a influence in air temperature.

The magnitude of the deviations for T_l was higher when compared to the magnitude of the deviations for T_w in both situations (steady and unsteady). As a matter of fact, the T_l deviation magnitudes for all MC were the same for these two situations. In air heat balance equation (Equation 2.20) the air cross-sectional area (A_l) and the air flow rate (Q_l) are not included of the partials derivatives (∂t and ∂x, respectively) however in air mass balance (Equation 2.14) theses variables are included. On top of that, the A_l is calculated from the water cross-sectional area (A_w) and the A_w is dependent of time as we can see in continuity equation from SVE (Equation 2.1). This might contribute to have no difference in the T_l deviation between steady and unsteady situations.

The water depth variations were not meaningful and in addition the obtained results verified some unexpected oscillations. These are likely to result from the use of a coarse grid and large dt. The dt value was predicted for each step from the Courant-Friedrich-Levy (CFL) condition. In order to study how dt duration could affect the results its value was reduced it in an eight-fold

factor and it different responses were obtained for our system. Namely water height variations were perceptible and relative size of the oscillations was smaller. However the *nstep* was kept constant, and the simulations only processed a fraction of the total time of the previous analyses. Nevertheless this was enough to compare it with the previous unsteady simulations.

A colour map representation was also implemented in this work for all systems and proved to be a very practical way of observing the overall behaviour of the system in time and distance, even if the profile of the obtained variables is not immediately understood.

Another point that deserves to be mentioned is the fact that the Q_w and T_w waves were out-of-phase and could be due to a lack of precision in the parameters of the Gaussian's given as input. The BC can therefore be improved in order to simulate more accurately the behaviour of a real discharge, by having a proper relation between the Q_w and T_w boundary conditions. By using real discharge experimental measurements, as exemplified in **Figure 5.1** and **Figure 5.2** this could be improved. These data can be used to provide realistic boundary conditions and also to validate the predictions of T_w, instead of using an idealised Gaussian function.

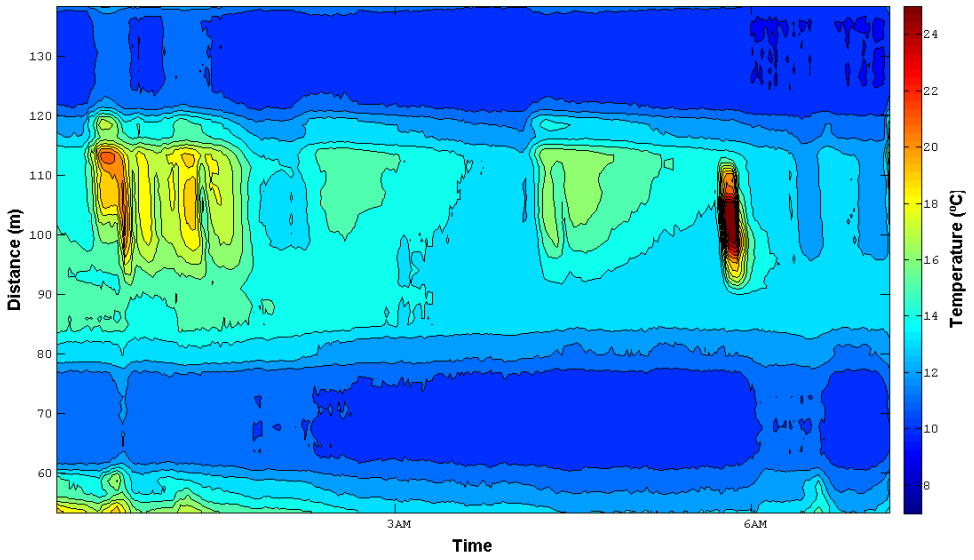

Figure 5.1: Real data for water temperature at 6th of March 2012 from Waternet.

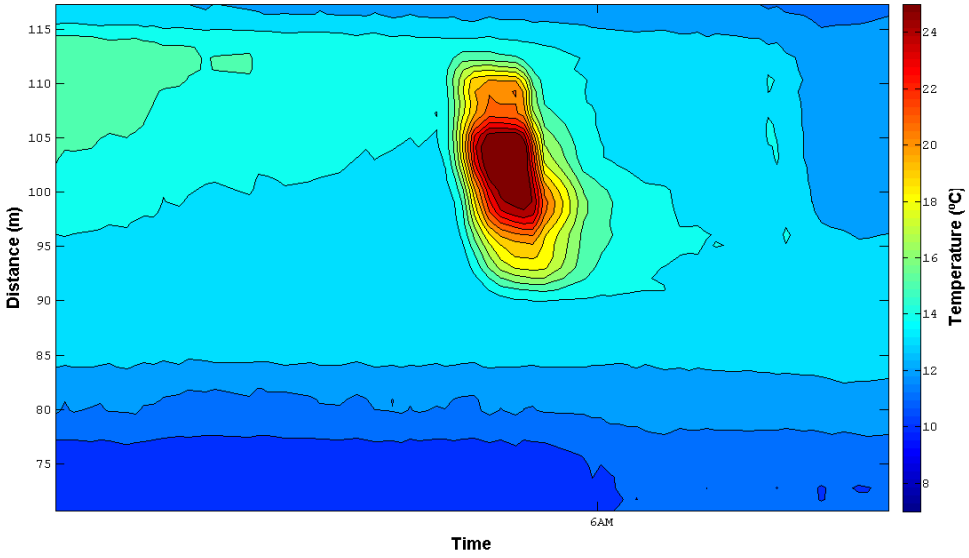

Figure 5.2: Zoom from previous figure. Real data for water temperature at 6th of March 2012 from Waternet.

The routines of this work proved to be a very versatile and promising way of modelling heat exchanges from wastewaters and the pipe network. A complex dynamical system was modelled employing mass balance equations, heat balance equations, momentum equation and heat conduction equations.

The computational time was not significative and further developments in more complex sewer networks can be solved in a normal laptop in a matter of hours. However a quantitative accuracy was not verified in this work, since these models need further work and validation in the aspects mentioned in this chapter. The next subsection addresses some of those points.

Chapter 6

Conclusions and future developments

6.1 Conclusions

Wastewater contains a significant amount of energy. This energy (thermal energy) could be recovered through a heat pump and a heat exchanger installed in sewers. A computer model of wastewater heat recovery could be very usefully to study the behaviour of this complex system, dispensing complicated measurement campaigns.

The work developed in this thesis uses a MATLAB® code developed by Bas Wols model at KWR Water Research Institute (Nieuwegein, The Netherlands), that simulates mass and thermal balances in pipe network for steady flow conditions. The first goal of this project was the implementation of a case of unsteady conditions for the water temperature and water flow rate, in order to simulate a discharge. As a second goal, this work aimed to assess the significance of the terms present in the equations that describe the physics of the model: water and air heat balance equations (Equation 2.19 and Equation 2.20).

In order to simulate a discharge, the boundary conditions applied at the begging of the pipe were modified and a Gaussian function was imposed to a baseline for T_w and Q_w with the intention of having varying water temperature and water flow rate. Regarding the second goal, binary coefficients were introduced for each term in both water heat balance equation and air heat balance equations and all the possible combination (ten modified cases) were computed. In order to compare each modified case to the general case, an deviation measure was calculated for each.

To visualize the results we added to the already existing routine that displays the predictions along the length of the pipe for a given set of time instants, a function to plot the results as a function of time and a colour map plot to represent system variables T_w and Q_w as function of both time and position along the pipe with a colour gradient representing the magnitude of this variable in question. For the deviations measure, it was chosen to use a bar chart with the information of the mean value (vertical bars) and the standard deviation (deviation bars) for T_w and T_l.

For the steady situation the predictions for the variables were fairly constant when represented in length and in time. Regarding the calculated difference deviations for the water temperature, removing all the terms in water heat balance equation does not have considerate impact, since the magnitude of the deviations were likely to be to small. However for the air temperature the *heat flux pipe to air* ($\dot{q}_{Rl}$) and the *heat flux water to air* ($\dot{q}_{wL}$) should be considered, since they create large deviations. In addition, the modelling of steady flow conditions is therefore not very realistic and relevant in the field of wastewater heat recovery modelling.

The unsteady flow situation was successfully implemented with fluctuations observed for Q_w and T_w and consequently in T_l and h. A main wave in both Q_w and T_w is visible, created from the Gaussian function that was imposed to a baseline for T_w and Q_w and the expected cooling-down effect was verified. Reductions in both variables along the pipe proved that the algorithm successfully modelled the mass and heat balance phenomena. The routine *datatime* proved to be very useful for unsteady conditions.

The deviations associated with the modified cases in the unsteady situation in water temperature predictions, point to the fact that the crucial term is the *heat flux pipe to water* ($\dot{q}_{Rw}$). For the air temperature the terms that correspond to changes in *heat flux pipe to air* ($\dot{q}_{Rl}$) and *heat flux water to air* ($\dot{q}_{wL}$) are the ones with the most significant impact. The magnitude of the deviations for air temperature was higher when compared to the magnitude of the deviations for T_w in both situations (steady and unsteady).

The obtained Q_w and T_w waves were out-of-phase in terms of progression along the pipe. This could be due to a lack of precision in the parameters of the Gaussian functions given as input. The BC can therefore be improved in order to simulate more accurately the behaviour of a real discharge, by having a proper relation between the Q_w and T_w boundary conditions.

In order to study how time step dt duration could affect the results, its value was reduced it in an eight-fold factor and it different responses were obtained for our system. Namely water height variations were perceptible and relative size of the oscillations was smaller. However the *nstep* parameter (number of steps) was kept constant, and the simulations only processed a fraction of the total time of the previous analyses.

6.2 Future developments

The model was tested for the parameters and variables presented in Chapter 3, however an extra analysis should be run for different parametric values, namely for water temperature and air temperature (**Table 3.5**). A detailed analysis for water flow rate could be useful in order to understand the limitations underlying this variable. The simulations presented in Chapter 4 were carried for a water flow rate of 1 L/s. The author also tested for 10L/s and the system behaviour was quite similar, however the deviation magnitude was less for T_w and higher for T_l.

Since this type of pipes have the capacity to fit flow rates of 100L/s even 150L/s, it would be relevant to test the model for a higher flow rate, to inspect if the behaviour would be similar and what would be the effect in the deviations for water temperature and air temperatures.

In order to smooth the oscillations present in the calculations of T_w and Q_w for the unsteady situation a few things could be done. The reduction of dt could be optimized and an increase in the density of the grid (*num.n*) could improve the results. A filter could also be used after each time step to inhibit oscillations produced by waves from the St. Venant equations (Dürrenmatt & Wanner, 2008). Since in case of discontinuities the Lax-Wendroff method produces oscillations that can destroy the integrity of the computation, an artificial viscosity could be added to the numerical method. Two examples of artificial viscosity are the linear artificial viscosity and the lapidus artificial viscosity (Grove, 1999). Although these methods could reduce the significance of the solution, since there is a trade-off: the peaks will be smeared out in order to remove the oscillations but the peaks associated with the discharge would also be reduced.

To study fluid motion and heat exchange in sewer pipes other numerical methods could be employed, such as computational fluid dynamics (CFD). However CFD is a technique that is applicable to analyse local phenomena and not a complex sewer network. This technique required higher computational times since it consists in a more sophisticated and detailed approach.

In order to improve the discharge behaviour, as exemplified in **Figure 5.1** and **Figure 5.2**, real discharge experimental measurements could be used. These data can be used to provide realistic boundary conditions, instead of using an idealised Gaussian function, and also to validate the predictions of T_w.

References

Amsterdam Climate Office, 2009. Amsterdam: A Different Energy - 2040 Energy Strategy.

Bergman, T.L. et al., 2011. *Fundamentals of Heat and Mass Transfer* 7th ed., Wiley.

Chanson, H., 2004. *Environmental Hydraulics for Open Channel Flows* 1st ed., Butterworth-Heinemann.

Dürrenmatt, D. J. & Gujer, W., 2006. *Berechnung des Verlaufs der Abwassertemperatur im Kanalisationsrohr*. ETH, Swiss Federal Institute of Technology Zurich. Available at: http://e-collection.ethbib.ethz.ch/eserv/eth:30918/eth-30918-01.pdf [Accessed August 19, 2012].

Dürrenmatt, D. J. & Wanner, O., 2008. *TEMPEST - Computer Program for Simulation of the Wastewater Temperatures in Sewers - User Manual*, Dübendorf, Switzerland: Eawag.

Dürrenmatt, David J. & Wanner, Oskar, 2008. Simulation of the wastewater temperature in sewers with TEMPEST. *Water Science & Technology*, 57(11), p.1809.

Edwini-Bonsu, S. & Steffler, P.M., 2004. Air flow in sanitary sewer conduits due to wastewater drag: A computational fluid dynamics approach. *Journal of Environmental Engineering and Science*, 3(5), pp.331–342.

Edwini-Bonsu, S. & Steffler, P.M., 2006a. Dynamics of air flow in sewer conduit headspace. *Journal of Hydraulic Engineering*, 132(8), pp.791–799.

Edwini-Bonsu, S. & Steffler, P.M., 2006b. Modeling ventilation phenomenon in sanitary sewer systems: A system theoretic approach. *Journal of Hydraulic Engineering*, 132(8), pp.778–790.

Elias Maxil, J.A. & Rietveld, L., 2011. Energy reduction in the urban water cycle: A review. In Internationale Symposium “Re-Water Braunschweig”. Braunschweig, Germany, pp. 195–207.

Fox, R.W., Pritchard, P.J. & McDonald, A.T., 2008. *Introduction to Fluid Mechanics* 7th ed., Wiley.

Hollmuller, P., 2003. Analytical characterisation of amplitude-dampening and phase-shifting in air/soil heat-exchangers. *International Journal of Heat and Mass Transfer*, 46(22), pp.4303–4317.

Holman, J., 2009. *Heat Transfer* 10th ed., McGraw-Hill Science/Engineering/Math.

IPCC, 2007. *Climate Change 2007 - The Physical Science Basis: Working Group I Contribution to the Fourth Assessment Report of the IPCC*, Cambridge University Press.

Krarti, M. & Kreider, J.F., 1996. Analytical model for heat transfer in an underground air tunnel. *Energy Conversion and Management*, 37(10), pp.1561–1574.

Kurpaska, S. & Slipek, Z., 1996. Mathematical model of heat and mass exchange in a garden subsoil during warm-air heating. *Journal of Agricultural Engineering Research*, 65(4), pp.305–311.

Liu, L., Fu, L. & Jiang, Y., 2010. Application of an exhaust heat recovery system for domestic hot water. *Energy*, 35(3), pp.1476–1481.

Meggers, F. & Leibundgut, H., 2011. The potential of wastewater heat and exergy: Decentralized high-temperature recovery with a heat pump. *Energy and Buildings*, 43(4), pp.879–886.

Mujumdar, P.P., 2001. Flood wave propagation. *Resonance*, 6(5), pp.66–73.

Nauffal, S., 2011. *Energy in the Urban Water Cycle: A Case Study of Heat Recovery in the Sewer System of Amsterdam*. Politecnico di Torino and Delft University of Technology.

Pascal, F. & De Buhan, M., 2008. The numerical simulation of complex PDE (MA691): Lecture Notes. Facultad de Ciencias F__sicas y Matem_aticas, Universidad de Chile.

Patty, S., 2010. Finite Difference Methods and Solving the Level Set Equations Numerically.

PBL, 2010. *Greenhouse Gas Emissions in the Netherlands 1990-2008*, PBL Netherlands Environmental Assessment Agency.

Pothof, I., 2012. Fundamentals of Urban Drainage (CIE4491): Lecture Notes. Watermanagement, Delft University of Technology. Available at: http://collegerama.tudelft.nl/mediasite/SilverlightPlayer/Default.aspx?peid=430031d538764bd6bc66a4e53fd08e171d.

R. Maidment, D. & Merwade, V., 2005. Distributed Flow Routing: Lecture Notes. Center for Research in Water Resources, University of Texas.

Schmid, F., 2009. Sewage water: interesting heat source for heat pumps and chillers. Swiss Energy Agency for Infrastructure Plants, Switzerland.

US EPA, C.C.D., 2012. Information on the basics of climate change. Available at: http://www.epa.gov/climatechange/basics/ [Accessed September 17, 2012].

W. Grove, J., 1999. Hydrodynamic Methods: Lecture Notes. Applied Theoretical and Computational Physics Division, Los Alamos National Laboratory. Applied Mathematics, Statistics University at Stony Brook.

Wanner, O, Panagiotidis, V. & Siegrist, H., 2004. Wärmeentnahme aus der Kanalisation – Einfluss auf die Abwassertemperatur. *Korrespondenz Abwasser*, 51(5), pp.489–495.

Waternet, 2012. About Waternet.

Wong, L.T., Mui, K.W. & Guan, Y., 2010. Shower water heat recovery in high-rise residential buildings of Hong Kong. *Applied Energy*, 87(2), pp.703–709.

Appendix A

A.1 Exchange processes in mass balance equations

Table A.1: Exchange processes in mass balance equations.

	Process	Equation
Water mass balance Eq. 2.13	Condensation pipe	$j_{kP} = \frac{\alpha_{kP}}{h_{fg}}\left(p_L - p_{sat}\left(T_{pl}^{(1)}\right)\right)$
Air mass balance Eq. 2.14	Surface area of air	$A_l = \frac{\pi}{4}D^2 - A_w$
Water vapour mass balance Eq. 2.15 [1]Emperical relation, which in case of oversaturation reduces the water vapor content to the satured value with relaxation time τ.	Condensation pipe	$j_{kP} = \frac{\alpha_{kP}}{h_{fg}}\left(p_L - p_{sat}\left(T_{pl}^{(1)}\right)\right)$
	Condensation oversaturation[1]	$j_{kL} = \frac{\rho_l}{\tau}(X - X_{sat})$
	Steering condensation pipe	$\zeta = \begin{cases} 1, & p_L > p_{sat}\left(T_{pl}^{(1)}\right) \\ 0, & p_L \leq p_{sat}\left(T_{pl}^{(1)}\right) \end{cases}$
	Steering oversaturation	$\xi = \begin{cases} 1, & X > X_{sat} \\ 0, & X \leq X_{sat} \end{cases}$

A.2 Heat fluxes in the heat balance equations

Table A.2: Heat fluxes in the heat balance equations.

	Process	Equation	
Water heat balance Eq. 2.19	Heat flux pipe to water	$\dot{q}_{Rw} = \alpha_{Rw}\left(T_{pw}^{(1)} - T_w\right)$	
	Heat flux water to air	$\dot{q}_{wl} = \alpha_{wl}(T_w - T_l)$	
	Evaporation or condensation	$\dot{q}_{vP} = \alpha_{vP}(p_{sat}(T_w) - p_L)$	
	Biochemical activity	$\dot{q}_w = e_{COD} r_{COD}$	
Air heat balance Eq. 2.20 [2]Emperical relation, which in case of oversaturation increases the heat of the air due to condensation with relaxation time τ.	Heat flux pipe to air	$\dot{q}_{Rl} = \alpha_{Rl}\left(T_{pl}^{(1)} - T_l\right)$	
	Heat flux oversaturation[2]	$\dot{q}_{kl} = h_{fg}\frac{\rho l}{\tau}(X - X_{sat})$	
Pipe heat balance in water part Eq. 2.24	BC heat flux pipe to water	$\dot{q}_{pw} = \alpha_{Rw}\left(T_{pw} - T_w\right)$	$r = D/2$
	BC temperature soil-pipe (water part)	$T_{pw} = T_{sw}$	$r = D/2 + D_p$
Pipe heat balance in air part Eq. 2.25	BC heat flux pipe to air	$\dot{q}_{pl} = \alpha_{Rl}\left(T_{pl} - T_l\right)$	$r = D/2$
	BC temperature soil-pipe (air part)	$T_{pl} = T_{sl}$	$r = D/2 + D_p$
Soil heat balance in water part Eq. 2.26 T_{s0} (steady state) temperature of bottom.	BC heat flux pipe to water	$\dot{q}_{pw} = -\dot{q}_{sw}$	$r = D/2 + D_p$
	BC temperature soil (water part)	$T_{sw} = T_{s0}$	$r = D/2 + D_p + \delta_s$
Soil heat balance in air part Eq. 2.27	BC heat flux pipe to air	$\dot{q}_{pl} = -\dot{q}_{sl}$	$r = D/2 + D_p$
	BC temperature soil (air part)	$T_{sl} = T_{s0}$	$r = D/2 + D_p + \delta_s$
Heat flux	Pipe to water / air	$\dot{q}_{pw} = -\left(\lambda_p \frac{\partial T_{pw}}{\partial r}\right)$	$\dot{q}_{pl} = -\left(\lambda_p \frac{\partial T_{pl}}{\partial r}\right)$
	Soil to water /air	$\dot{q}_{sw} = -\left(\lambda_s \frac{\partial T_{sw}}{\partial r}\right)$	$\dot{q}_{sl} = -\left(\lambda_s \frac{\partial T_{sl}}{\partial r}\right)$

A.3 Coefficients and others

Table A.3: Coefficients and others.

Process	Equation
Heat transfer water-air interface	$\alpha_{wl} = 5.85\sqrt{\lvert U_l - U_w \rvert}$
Heat transfer flowing water	$\alpha_{wl} = \frac{Nu_w \lambda_w}{R_w}$
Heat transfer flowing air	$\alpha_{rl} = \frac{Nu_l \lambda_l}{l}$
Heat transfer condensation pipe	$\alpha_{kP} = 8.75\sqrt{\lvert U_l \rvert}$
Heat transfer condensation water-pipe	$\alpha_{vP} = 8.75\sqrt{\lvert U_l - U_w \rvert}$
Thermal transmittance pipe	$K_P = \frac{\lambda_p}{D_p}$
Saturation pressure	$p_{sat} = p_{s0} exp\left(\frac{T_{s0}}{T}\right)$
Relative humidity	$\phi = \frac{X}{0.622 + X} \frac{p_{sys}}{p_{sat}(T_l)}$
Partial pressure water vapor	$p_L = \phi p_{sat}(T_l)$
Saturation water vapour	$X_{sat} = 0.622 \frac{p_{sat}(T_l)}{p_{sys} - p_{sat}(T_l)}$
Constant k_{rw}	$\frac{1}{k_{rw}} = \frac{1}{\alpha_{rw}} + \frac{D}{2\lambda_p} \ln\left(1 + \frac{2D_p}{D}\right) + \frac{D}{2\lambda_p} \ln\left(1 + \frac{2\delta_s}{D + 2D_p}\right)$
Constant k_{rl}	$\frac{1}{k_{rl}} = \frac{1}{\alpha_{rl}} + \frac{D}{2\lambda_p} \ln\left(1 + \frac{2D_p}{D}\right) + \frac{D}{2\lambda_p} \ln\left(1 + \frac{2\delta_s}{D + 2D_p}\right)$

Appendix B

MATLAB® code for unsteady conditions (case with $dt/8$).

B.1 fun_start

```
function [model] = fun_start(row)

%% Numerics
% Number of time steps
num.nstep               = 1000;
% Grid size
num.n                   = 100;
% Courant number criterium
num.CFL_crit            = 0.8; %0.9;
% Number of plots
num.nplot               = 5;
% Store solution nsave times
num.nsave               = 1000;
num.nlayer              = 5;
num.nsoil               = 5;

%% Constant conditions
% Air temperature
cond.Tair               = 283;
% Temperatures
cond.Tsoil              = 280;              % K
% Ambient air pressure
cond.psys               = 1000;              % mbar
% Relative humidity
cond.phi                = 0.8;              % -

%% Conditions of network (line)
% Connections between conduits and nodes
line.ind      =  [1 2];%[1 3; 2 3; 3 4; 4 5]; %[1 3;2 3;3 4]; %
% Initial conditions at nodes
% Temperatures at nodes
line.node.Tw  = [283 NaN];%[296.5 296.5 NaN 293]; %[20 10 NaN NaN]+273; %
line.node.Ta  = repmat(cond.Tair,1,max(max(line.ind))); %[25 25 NaN NaN]+293;
% Initial water flow rate
line.node.Qw  = [.001 NaN];%[0.00015 0.00016 NaN 293]; % [0.1 0.05 NaN NaN]; %
% m3/s
% Initial air flow rate
line.node.Ql  = NaN(1,max(max(line.ind)));
% Initial water vapour fraction
line.node.X   = NaN(1,max(max(line.ind)));
% Initial area of water in pipe
line.node.Aw   = NaN(1,max(max(line.ind)));
% Calculate parameters at the conduits
line          = fun_line(line);

%% Conditions of conduit (pipe)

% Shape
pipe.shape              = ['pipe'];
% Width / diameter of channel
pipe.D                  = 0.235;                % m
% Thickness of pipe
pipe.Dp                 = 0.04;             % m
% Pipe length
pipe.L                  = 10;             % m
% Bottom slope
pipe.i_bottom           = 2E-3;             % m/m
% Friction constant
pipe.cf                 = 2.4E-3;             % -
% Density pipe
```

```
pipe.rho                = 2000;             % kg/m^3
% Heat capacity pipe
pipe.Cp                 = 0.84E3;              % J/kg/K
% Influence distance of soil
pipe.delta_s            = 3;              % m
% Heat conductivity
pipe.lambda             = 2.3;                % W/m/K;
% Spatial grid distance
pipe.dx                 = pipe.L/num.n;    % m
% Thermal diffusivity
pipe.alfa               = pipe.lambda/pipe.Cp/pipe.rho;
% Store for all pipe circuits
pipe(1:line.n_cnd)      = pipe(1);
%% Constants
% Gravitational constant
const.g                 = 9.81;             % m^2/s
% Saturation pressure
const.ps0               = 1.73E9;           % mbar
const.Ts0               = -5311;            % K
const.hfg               = 2453.3E3;         % J/kg

% Density
const.water.rho         = 1000;             % kg/m^3
const.air.rho           = 1.188;            % kg/m^3

% Heat capacity
const.water.Cp          = 4.1813E3;         % J/kg/K
const.air.Cp            = 1007;              % J/kg/K

% Kinematic viscosity
const.water.nu          = 1E-6;             % m2/s;
const.air.nu            = 1.533E-5;          % m2/s;

% Thermal diffusivity
const.water.alfa        = 1.4E-7;           % m2/s;
const.air.alfa          = 2.216E-5;           % m2/s;
const.soil.alfa         = 0.74E-6;

% Heat conductivity
const.water.lambda      = 0.6;                % W/m/K;
const.air.lambda        = 0.02569;                % W/m/K;
const.soil.lambda       = 2.2;                % W/m/K;

% Prandtl number
const.water.Pr          = const.water.nu./const.water.alfa;
const.air.Pr            = const.air.nu./const.air.alfa;

%% Sources
const.sources.water.fRw = row(1);
const.sources.water.fwl = row(2);
const.sources.water.fvP = row(3);
const.sources.air.fRl = row(4);
const.sources.air.fkl = row(5);

num.nsave               = max(num.nsave,num.nplot);
%% Store paramaters in model structure
model.cond              = cond;
model.const             = const;
model.pipe              = pipe;
model.num               = num;
model.line              = line;

%%
function line = fun_line(line)
% function to calculate the variables in the conduits from the values at
% the nodes
% Input:
% line          = line-structure with line network information
% Output:
% line          = line-structure with line network information
```

```
n_cnd    = size(line.ind,1);
n_node   = max(max(line.ind));

% Calculate variables at the inner nodes from the outer upstream nodes
for n = 1 : n_node
    % Index of conduits with same downstream node as upstream node for n
    % (connected pipes)
    ind_cnd      = find(line.ind(:,2)==n);
    % Index of upstream nodes for connected pipes
    ind_node     = line.ind(ind_cnd,1);
    if ~isempty(ind_node)
        % Set variables
        line.node.Qw(n) = sum(line.node.Qw(ind_node));
        line.node.Tw(n) =
sum(line.node.Qw(ind_node).*line.node.Tw(ind_node))/sum(line.node.Qw(ind_node)
);
        line.node.Ql(n) = sum(line.node.Ql(ind_node));
        %line.cnd.Ta(n) = line.node.Ta(ind_node);
    end
end

% Calculate variables at the conduits from the upstream nodes
for n = 1 : n_cnd
    % Index of upstream node for conduit n
    ind_node      = line.ind(n,1);
    % Set variables
    line.cnd(n).Qw = sum(line.node.Qw(ind_node));
    line.cnd(n).Ql = sum(line.node.Ql(ind_node));
    line.cnd(n).Tw =
sum(line.node.Qw(ind_node).*line.node.Tw(ind_node))/sum(line.node.Qw(ind_node)
);
    line.cnd(n).Ta = line.node.Ta(ind_node);
    line.cnd(n).X  = line.node.X(ind_node);
    line.cnd(n).Aw  = line.node.Aw(ind_node);
end
line.n_cnd  = n_cnd;
line.n_node = n_node;
```

B.2 fun_sources

```
function [S_Aw,S_alfa_w,S_alfa_l,S_X,alfa_Rw,alfa_Rl] =
fun_sources(cnd,pipe,const,cond,nstep)
% nlayer          = num.nlayer;
% n               = num.n;

% S_Tpl           = ones(n+2,nlayer);
% S_Tpw           = ones(n+2,nlayer);
       r0       = pipe.Dp + pipe.D/2;

% Heat transfer
alfa_wl   = 5.85*sqrt(abs(cnd.Ul-cnd.Uw));                          % W/m^2/K
alfa_vP   = 8.75*sqrt(abs(cnd.Ul-cnd.Uw));                          %
W/m^2/mbar
alfa_kP   = 8.75*sqrt(abs(cnd.Ul));
alfa_Rw   = cnd.Nu_w.*const.water.lambda./(cnd.Aw./cnd.Pw);
alfa_Rl   = cnd.Nu_l.*const.air.lambda./(cnd.Al./cnd.Pl);

% Steering parameters
zeta      = sign(cnd.pL - cnd.psat_p);  zeta(zeta<0) = 0;
ksi       = sign(cnd.X  - cnd.Xsat);    ksi(ksi<0)   = 0;

if nstep == 0
    kw        = 1./((pipe.D/(2*pipe.lambda))*log(1+2*pipe.Dp/pipe.D) +
(pipe.D/(2*const.soil.lambda))*log(1+pipe.delta_s/r0) + 1./alfa_Rw); % +
```

```
1./cnd.k_RS);
    kl        = 1./((pipe.D/(2*pipe.lambda))*log(1+2*pipe.Dp/pipe.D) +
(pipe.D/(2*const.soil.lambda))*log(1+pipe.delta_s/r0) + 1./alfa_Rl);
%1./(1./alfa_Rl + 1./cnd.k_RS);
    q_Rw  =  kw.*(cond.Tsoil-cnd.Tw);
    q_Rl  =  kl.*(cond.Tsoil-cnd.Tl);
    tau   = 1e-6;
else
    q_Rw  = alfa_Rw.*(cnd.Tpw(:,1)-cnd.Tw);
    q_Rl  = alfa_Rl.*(cnd.Tpl(:,1)-cnd.Tl);
    tau   = 1;
end
%    zeta  = zeta*0;
    ksi   = ksi*tau;

% Sources

fRw = const.sources.water.fRw;
fwl = const.sources.water.fwl;
fvP = const.sources.water.fvP;
fRl = const.sources.air.fRl;
fkl = const.sources.air.fkl;

S_Aw      = (1./const.water.rho).*(-alfa_vP/(const.hfg).*(cnd.psat-
cnd.pL).*cnd.B);
S_X       = (1./const.air.rho).*((alfa_vP/const.hfg).*(cnd.psat-cnd.pL).*cnd.B
- zeta.*(alfa_kP/const.hfg).*(cnd.pL-cnd.psat_p).*(cnd.Pl-cnd.B) -
ksi.*const.air.rho.*(cnd.X-cnd.Xsat).*cnd.Al);
S_alfa_w  = (1./(const.water.Cp*const.water.rho)).*(fRw.*q_Rw.*(cnd.Pw-cnd.B)
- fwl.*alfa_wl.*(cnd.Tw-cnd.Tl).*cnd.B - fvP.*alfa_vP.*(cnd.psat-
cnd.pL).*cnd.B); %
S_alfa_l  = (1./(const.air.Cp*const.air.rho)).*(fRl.*q_Rl.*(cnd.Pl-cnd.B) +
fwl.*alfa_wl.*(cnd.Tw-cnd.Tl).*cnd.B  +
fkl.*ksi.*const.hfg.*const.air.rho.*(cnd.X-cnd.Xsat).*cnd.Al);
%kl.*(cond.Tsoil-cnd.Tl)
```

B.3 sewer_temp_model

```
function [model] = sewer_temp_model(model);
% 1D Model for temperature in sewer lines
%
% Lax-Wendroff finite difference method.
% Reflective boundary conditions.

% Read paramete rs from model-structure
cond            = model.cond;
const           = model.const;
pipe            = model.pipe;
num             = model.num;
line            = model.line;

tic
% Set initial conditions for all conduits
for n = 1 : line.n_cnd
    [line.cnd(n).Aw,line.cnd(n).Ql,line.cnd(n).X] =
fun_equilibrium(line.cnd(n).Qw,line.cnd(n).Ta,pipe(n),cond,const,num);
    cnd(n)            = fun_init(line.cnd(n),pipe(n),cond,const,num);
    cnd(n)            = fun_stationary(n,cnd(n),line,pipe(n),cond,const,num);
end
% Save stationary solution
model           = fun_save(model,cnd,line,num,0);
fprintf('Stationary solution, time elapsed = %3.1f s\n',toc);


% Inner loop, time steps.
for nstep = 1 : num.nstep
```

```
    %fprintf('Step %i\n',nstep);
    % Repeat for each conduit
    for n = 1 : line.n_cnd
        % Set values at upstream boundaries
        BC       = fun_BC(nstep,n,cnd,line,cond,num);
        % Set boundary conditions
        cnd(n)   = fun_boundary(BC,cnd(n),pipe(n),const,cond,num);
        % Calculate variables
        cnd(n)   = fun_variable(cnd(n),pipe(n),cond,const);
        % Time steps at different conduits
        dt(n)    = num.CFL_crit*pipe(n).dx/cnd(n).Umax;
    end
    % Choose smallest time step
    dt         = (min(dt/8));

    % Solve pde for each conduit
    for n = 1 : line.n_cnd
        cnd(n)      = fun_pde(dt,cnd(n),pipe(n),num,const,cond);
    end

    % Printing to screen
    if mod(nstep,100) == 0
         fprintf('Step %i, dt = %f s, time elapsed = %3.1f s\n',nstep,dt,toc);
    end

    % Plotting
    model.num.dt(nstep) = dt;
    num.dt(nstep) = dt;

    % Saving
    model = fun_save(model,cnd,line,num,nstep);

end
% Plotting
%fun_plot(model);
%% fun_init
function cnd = fun_init(cnd_eq,pipe,cond,const,num);
% function to set initial conditions
% Input:
% cnd_eq        = structure with equilibrium conditions
% pipe          = structure with pipe information
% cond          = structure with conditions
% const         = structure with constants
% num           = structure with numerical information
% Output:
% cnd           = structure with all variables

% Numerical information
% Number of pipe segments
n                   = num.n;
% Number of pipe layes
nlayer              = num.nlayer;
% Length pipe segment
dx                  = pipe.dx;

% Conductivity pipe/soil
cnd.k_RS            =
1./((pipe.D/(2*const.soil.lambda))*log(1+2*pipe.delta_s/(pipe.D+2*pipe.Dp)));

% Coordinates soil
cnd.r               =
repmat(linspace(pipe.D/2+pipe.Dp,pipe.D/2+pipe.Dp+pipe.delta_s,num.nsoil),n+2,
1);
cnd.dr              = cnd.r(1,2) - cnd.r(1,1);

cnd.r_pipe          = repmat(linspace(pipe.D/2,pipe.D/2 +
pipe.Dp,num.nlayer),n+2,1);
cnd.dr_pipe         = cnd.r_pipe(1,2) - cnd.r_pipe(1,1);
% Coordinates pipe
cnd.x               = linspace(0,dx*(n+2),n+2)';
```

```
% Bottom
cnd.zb              = -pipe.i_bottom*cnd.x + pipe.i_bottom*cnd.x(end);

% Initial conditions
cnd.Aw              = ones(n+2,1)*cnd_eq.Aw;
cnd.Qw              = ones(n+2,1)*cnd_eq.Qw;
cnd.Ql              = ones(n+2,1)*cnd_eq.Ql;
cnd.Tw              = ones(n+2,1)*cnd_eq.Tw;
cnd.Tl              = ones(n+2,1)*cnd_eq.Ta;
cnd.X               = ones(n+2,1)*cnd_eq.X;
cnd.Tpw             = ones(n+2,nlayer)*cond.Tsoil;
cnd.Tpl             = ones(n+2,nlayer)*cond.Tsoil;

cnd.Tsw             = ones(n+2,num.nsoil)*cond.Tsoil;
cnd.Tsl             = ones(n+2,num.nsoil)*cond.Tsoil;

% Shape of the pipe
[cnd.h,cnd.B,cnd.Pw,cnd.Pl,cnd.Al,cnd.Apw,cnd.Apl,cnd.Lpw,cnd.Lpl] =
fun_shape(cnd.Aw,pipe,num);

% For time iteration
cnd.Al_old          = cnd.Al;
cnd.Aw_old          = cnd.Aw;

% Declare other variables
cnd.i_fric          = zeros(n+2,1);
cnd.Uw              = zeros(n+2,1);
cnd.Ul              = zeros(n+2,1);
cnd.Umax            = zeros(1  ,1);
cnd.psat            = zeros(n+2,1);
cnd.psat_l          = zeros(n+2,1);
cnd.psat_p          = zeros(n+2,1);
cnd.Xsat            = zeros(n+2,1);
cnd.phi             = zeros(n+2,1);
cnd.pL              = zeros(n+2,1);
cnd.Re_w            = zeros(n+2,1);
cnd.Re_l            = zeros(n+2,1);
cnd.Nu_w            = zeros(n+2,1);
cnd.Nu_l            = zeros(n+2,1);

%% fun_BC
function BC = fun_BC(nstep,n,cnd,line,cond,num)
% Function to determine values at boundary conditions
% Input:
% nstep         = time step
% cnd_eq        = structure with equilibrium conditions
% cond          = structure with conditions
% num           = structure with numerical information
% Output:
% BC            = structure with upstream boundary conditions

% Equilibrium values at conduits
cnd_eq      = line.cnd(n);

% Nodes indices of conduit n
ind_cnd     = line.ind(n,:);
% Find if upstream node is present at downstream nodes of other conduits
% (connecting conduints)
ind_cnd_cn  = find(line.ind(:,2)==ind_cnd(1));

if isempty(ind_cnd_cn)
    % Start conduit

    if isfield(num, 'dt')
        timeT = sum(num.dt);
    else
        timeT = 0;
    end

    BC.Qw       = cnd_eq.Qw + fun_BC_time(timeT ,.1,.5,1.5*cnd_eq.Qw,'Gauss');
```

```
    BC.Tw       = cnd_eq.Tw + fun_BC_time(timeT,.1,.001,10*cnd_eq.Tw,'Gauss');
    BC.Tl       = cnd_eq.Ta ;
    BC.X        = cnd_eq.X ;
    BC.Ql       = cnd_eq.Ql ;

else

    % Inner conduits
    for i = 1 : length(ind_cnd_cn)
        Qw_cn(i)= cnd(ind_cnd_cn(i)).Qw(end);
        Tw_cn(i)= cnd(ind_cnd_cn(i)).Tw(end);
        Tl_cn(i)= cnd(ind_cnd_cn(i)).Tl(end);
        Ql_cn(i)= cnd(ind_cnd_cn(i)).Ql(end);
        X_cn(i) = cnd(ind_cnd_cn(i)).X(end);
    end

    BC.Qw       = sum(Qw_cn);
    BC.Tw       = sum(Qw_cn.*Tw_cn)./sum(Qw_cn);
    BC.Ql       = sum(Ql_cn);
    BC.Tl       = sum(Ql_cn.*Tl_cn)./sum(Ql_cn);
    BC.X        = sum(Ql_cn.*X_cn)./sum(Ql_cn);
end
    BC.Tsoil    = cond.Tsoil;

%% fun_BC_time
function ft = fun_BC_time(tn,t0,tL,A,type)
% Function to set time dependent boundary conditions
% Input:
% tn            = time
% t0            = start time
% tL            = duration
% A             = amplitude
% type          = type of function
% Output:
% ft            = function value
switch type
    case 'Gauss'
        ft = A*exp(-((tn-t0).^2)./(0.5*tL^2));
    case 'constant'
        ft = A;
    case 'step'
        ft = A*0.5*(1+erf((tn-t0)/(tL*sqrt(0.5))));
    case 'ramp'
    %    ft =
end

%% fun_boundary
function cnd = fun_boundary(BC,cnd,pipe,const,cond,num)
% Function to set boundary conditions
%
% Input:
% BC            = structure with upstream boundary conditions
% cnd           = structure with all variables
% num           = structure with numerical information
%
% Output:
% cnd           = structure with all variables

% Number of pipe segments
n               = num.n;

% Upstream boundary conditions
% Reflective boundary conditions for Aw
cnd.Aw (1,:)    = cnd.Aw(2,:);
cnd.Qw (1,:)    = BC.Qw;
cnd.Ql (1,:)    = BC.Ql;
cnd.Tw (1:2,:)  = BC.Tw;
cnd.Tl (1:2,:)  = BC.Tl;
cnd.X  (1,:)    = BC.X;
cnd.Tpw(1,:)    = cnd.Tpw(2,:);
```

```
cnd.Tpl(1,:)    = cnd.Tpl(2,:);

%
[S_Aw,S_alfa_w,S_alfa_l,S_X,alfa_Rw,alfa_Rl] =
fun_sources(cnd,pipe,const,cond,1);
cnd.Tpw(:,1)    = (alfa_Rw.*cnd.Tw +
(pipe.lambda/cnd.dr_pipe)*cnd.Tpw(:,2))./(alfa_Rw+pipe.lambda/cnd.dr_pipe);
cnd.Tpl(:,1)    = (alfa_Rl.*cnd.Tl +
(pipe.lambda/cnd.dr_pipe)*cnd.Tpl(:,2))./(alfa_Rl+pipe.lambda/cnd.dr_pipe);

% Downstream boundary conditions (reflective)
cnd.Aw (n+2,:)  = cnd.Aw (n+1,:);
cnd.Qw (n+2,:)  = cnd.Qw (n+1,:);
cnd.Ql (n+2,:)  = cnd.Ql (n+1,:);
cnd.Tw (n+2,:)  = cnd.Tw (n+1,:);
cnd.Tl (n+2,:)  = cnd.Tl (n+1,:);
cnd.X  (n+2,:)  = cnd.X  (n+1,:);
cnd.Tpw(n+2,:)  = cnd.Tpw(n+1,:);
cnd.Tpl(n+2,:)  = cnd.Tpl(n+1,:);

% Boundary conditions for soil
cnd.Tsw(:,end)  = BC.Tsoil;
cnd.Tsl(:,end)  = BC.Tsoil;

cnd.Tsw(:,1)    =
((pipe.lambda/const.soil.lambda)*(cnd.dr/cnd.dr_pipe)*cnd.Tpw(:,end-1) +
cnd.Tsw(:,2))/(1+(pipe.lambda/const.soil.lambda)*(cnd.dr/cnd.dr_pipe));
cnd.Tsl(:,1)    =
((pipe.lambda/const.soil.lambda)*(cnd.dr/cnd.dr_pipe)*cnd.Tpl(:,end-1) +
cnd.Tsl(:,2))/(1+(pipe.lambda/const.soil.lambda)*(cnd.dr/cnd.dr_pipe));

% In the pipe (outer boundary)
cnd.Tpw(:,end)  = cnd.Tsw(:,1);
cnd.Tpl(:,end)  = cnd.Tsl(:,1);

%% fun_variable
function cnd = fun_variable(cnd,pipe,cond,const)
% Input:
% cnd           = structure with all variables
% pipe          = structure with pipe information
% cond          = structure with conditions
% const         = structure with constants

% Output:
% cnd           = structure with all variables
% Variables

% Friction
cnd.i_fric    = pipe.cf*cnd.Qw.^2.*cnd.Pw./(const.g*cnd.Aw.^3);

% Velocities
cnd.Uw        = cnd.Qw./cnd.Aw;
cnd.Ul        = cnd.Ql./cnd.Al;
% Maximum velocity water
cnd.Umax      = max(abs(cnd.Uw)+sqrt(const.g*cnd.h));

% Saturation pressure
cnd.psat      = const.ps0.*exp(const.Ts0./cnd.Tw);                  % mbar
cnd.psat_l    = const.ps0.*exp(const.Ts0./cnd.Tl);
cnd.psat_p    = const.ps0.*exp(const.Ts0./cnd.Tpl(:,1));
cnd.Xsat      = 0.622*cnd.psat_l./(cond.psys-cnd.psat_l);
% Humidity
cnd.phi       = (cnd.X./(0.622+cnd.X)).*cond.psys./cnd.psat_l;
% Partial water vapor pressure
cnd.pL        = cnd.phi.*cnd.psat_l;
% Reynolds number
cnd.Re_w      = abs(cnd.Uw).*pipe.D./const.water.nu;
cnd.Re_l      = abs(cnd.Ul).*pipe.D./const.air.nu;
% Nusselt number
cnd.Nu_w      = 0.023*cnd.Re_w.^(4/5).*const.water.Pr^(1/3);
```

```
cnd.Nu_l     = 0.023*cnd.Re_l.^(4/5).*const.air.Pr^(1/3);
%% fun_stationary
function cnd = fun_stationary(ncond,cnd,line,pipe,cond,const,num)


% Numerical parameters
dx              = pipe.dx;
n               = num.n;
nlayer          = num.nlayer;
nsoil           = num.nsoil;

       r0       = pipe.Dp + pipe.D/2;
       rpipe    = cnd.r_pipe;
       r        = cnd.r;
eps = 1;
while eps > 1E-5
       cnd      = fun_variable(cnd,pipe,cond,const);
       % Set values at upstream boundaries
       BC       = fun_BC(1,ncond,cnd,line,cond,num);

       % Set boundary conditions
       cnd      = fun_boundary(BC,cnd,pipe,const,cond,num);

       % New values
       X        = cnd.X; %zeros(n+2,1);
       Tw       = cnd.Tw; %zeros(n+2,1);
       Tl       = cnd.Tl;%zeros(n+2,1);

       % Sources
       [S_Aw,S_alfa_w,S_alfa_l,S_X,alfa_Rw,alfa_Rl] =
fun_sources(cnd,pipe,const,cond,0);

        kw       = 1./((pipe.D/(2*pipe.lambda))*log(1+2*pipe.Dp/pipe.D) +
(pipe.D/(2*const.soil.lambda))*log(1+pipe.delta_s/r0) + 1./alfa_Rw); % +
1./cnd.k_RS);
        kl       = 1./((pipe.D/(2*pipe.lambda))*log(1+2*pipe.Dp/pipe.D) +
(pipe.D/(2*const.soil.lambda))*log(1+pipe.delta_s/r0) + 1./alfa_Rl);
%1./(1./alfa_Rl + 1./cnd.k_RS);

        % Heat flux
        q_Sw     = kw.*(cond.Tsoil - cnd.Tw);
        q_Sl     = kl.*(cond.Tsoil - cnd.Tl);

        % Solution for temperature in pipe
        cnd.Tpw  = repmat(Tw,1,nlayer) +
repmat(q_Sw,1,nlayer).*((pipe.D/(2*pipe.lambda)) .* log(2*rpipe/pipe.D) +
(1./repmat(alfa_Rw,1,nlayer)));
        cnd.Tpl  = repmat(Tl,1,nlayer) +
repmat(q_Sl,1,nlayer).*((pipe.D/(2*pipe.lambda)) .* log(2*rpipe/pipe.D) +
(1./repmat(alfa_Rl,1,nlayer)));


       % Start solving ode
       i        = 1:n+1;

       X(i+1)   = cnd.X(i)   + dx*S_X(i)./cnd.Ql(i);
       Tw(i+1)  = cnd.Tw(i)  + dx*S_alfa_w(i)./cnd.Qw(i);
       Tl(i+1)  = cnd.Tl(i)  + dx*S_alfa_l(i)./cnd.Ql(i);

       % Difference
       eps      = abs(cnd.Tw - Tw) + abs(cnd.Tl - Tl) + abs(cnd.X - X);
       eps      = mean(eps(end-1));

       cnd.X    = X;
       cnd.Tw   = Tw;
       cnd.Tl   = Tl;
end
% Analytical solution for temperature in pipe and soil
% Source terms
[S_Aw,S_alfa_w,S_alfa_l,S_X,alfa_Rw,alfa_Rl] =
```

```
fun_sources(cnd,pipe,const,cond,0);

% Solution for temperature in soil
cnd.Tsw  = repmat(cond.Tsoil,n+2,nsoil) +
repmat(q_Sw,1,nsoil).*((pipe.D/(2*const.soil.lambda)) .*
log(r./(r0+pipe.delta_s)));
cnd.Tsl  = repmat(cond.Tsoil,n+2,nsoil) +
repmat(q_Sl,1,nsoil).*((pipe.D/(2*const.soil.lambda)) .*
log(r./(r0+pipe.delta_s)));
%
figure(2)
subplot(231)
hold on
plot(cnd.x,cnd.Tw)
subplot(232)
hold on
plot(cnd.x,cnd.Tl)
subplot(233)
hold on
plot(cnd.x,cnd.X)
subplot(234)
hold on
plot(cnd.x,cnd.Tpw)
subplot(235)
hold on
plot(cnd.x,cnd.Tpl)
subplot(236)
hold on
plot(cnd.x,cnd.Tsw)

%% fun_pde
function cnd = fun_pde(dt,cnd,pipe,num,const,cond)
% Input:
% cnd           = structure with all variables
% pipe          = structure with pipe information
% const         = structure with constants
% num           = structure with numerical information
% Output:
% cnd           = structure with all variables

% r-coordinate (for soil)
r               = cnd.r;
r_pipe          = cnd.r_pipe;

% Numerical parameters
dx              = pipe.dx;
dr              = cnd.dr;
dr_pipe         = cnd.dr_pipe;
nlayer          = num.nlayer;
n               = num.n;

% Declare variables (for Lax-Wendroff scheme)
Awx             = zeros(n+1,1);
Qwx             = zeros(n+1,1);
Twx             = zeros(n+1,1);
Tlx             = zeros(n+1,1);
Xx              = zeros(n+1,1);

% % Temperature fluxes at soil-pipe interface
% q_Sw     = -const.soil.lambda*(cnd.Tsw(:,2)-cnd.Tsw(:,1))/dr;
% q_Sl     = -const.soil.lambda*(cnd.Tsl(:,2)-cnd.Tsl(:,1))/dr;

% Source terms
[S_Aw,S_alfa_w,S_alfa_l,S_X,alfa_Rw,alfa_Rl] =
fun_sources(cnd,pipe,const,cond,1);


% Heat transfer soil
j = 2:num.nsoil-1;
cnd.Tsw(:,j) = cnd.Tsw(:,j) + (const.soil.alfa)*(dt/dr^2).*(cnd.Tsw(:,j+1)-
```

```
cnd.Tsw(:,j) - (r(:,j-1)./r(:,j)).*(cnd.Tsw(:,j)-cnd.Tsw(:,j-1)));
cnd.Tsl(:,j) = cnd.Tsl(:,j) + (const.soil.alfa)*(dt/dr^2).*(cnd.Tsl(:,j+1)-
cnd.Tsl(:,j) - (r(:,j-1)./r(:,j)).*(cnd.Tsl(:,j)-cnd.Tsl(:,j-1)));

% Heat transfer pipe
j = 2:num.nlayer-1;
cnd.Tpw(:,j) = cnd.Tpw(:,j) + (pipe.alfa)*(dt/dr_pipe^2).*(cnd.Tpw(:,j+1)-
cnd.Tpw(:,j) - (r_pipe(:,j-1)./r_pipe(:,j)).*(cnd.Tpw(:,j)-cnd.Tpw(:,j-1)));
cnd.Tpl(:,j) = cnd.Tpl(:,j) + (pipe.alfa)*(dt/dr_pipe^2).*(cnd.Tpl(:,j+1)-
cnd.Tpl(:,j) - (r_pipe(:,j-1)./r_pipe(:,j)).*(cnd.Tpl(:,j)-cnd.Tpl(:,j-1)));

%% First half step
i        = 1:n+1;

% Mass balance
Awx(i)   = (cnd.Aw(i+1)+cnd.Aw(i))/2 - dt/(2*dx)*(cnd.Qw(i+1)-cnd.Qw(i)); % +
S

% New water depths etc in pipe
[hx,Bx,Pwx,Plx,Alx] = fun_shape(Awx,pipe,num);

% Momentum balance
Qwx(i) = (cnd.Qw(i+1)+cnd.Qw(i))/2 -  ...
         dt/(2*dx)*((cnd.Qw(i+1).^2./cnd.Aw(i+1)) - ...
                    (cnd.Qw(i).^2./cnd.Aw(i))) - ...
         dt/(2*dx)*const.g*cnd.Aw(i).*(cnd.h(i+1)-cnd.h(i)); % + ...

% Heat transfer
Twx(i) = ((cnd.Tw(i+1).*cnd.Aw(i+1)+cnd.Tw(i).*cnd.Aw(i))/2 -  ...
         dt/(2*dx)*((cnd.Qw(i+1).*cnd.Tw(i+1)) - ...
                    (cnd.Qw(i).*cnd.Tw(i)))./Awx(i);

% Heat transfer
Tlx(i) = (cnd.Tl(i+1)+cnd.Tl(i))/2 -  ...
         dt/(2*dx)*cnd.Ql(i).*(cnd.Tl(i+1)-cnd.Tl(i))./cnd.Al(i);

% Heat transfer
Xx(i) = (cnd.X(i+1)+cnd.X(i))/2 -  ...
         dt/(2*dx)*cnd.Ql(i).*(cnd.X(i+1)-cnd.X(i))./cnd.Al(i);

%% Second half step
i           = 2:n+1;

% Mass balance water
cnd.Aw(i)   = cnd.Aw(i) - (dt/dx)*(Qwx(i)-Qwx(i-1)) + dt*S_Aw(i);
%Ql(i) = Ql(i-1) - (dx/dt)*(2*Alx(i-1)-Al(i)-Al(i-1));

% Update geometric properties
[cnd.h,cnd.B,cnd.Pw,cnd.Pl,cnd.Al,cnd.Apw,cnd.Apl,cnd.Lpw,cnd.Lpl] =
fun_shape(cnd.Aw,pipe,num);

% Momentum balance
cnd.Qw(i)   = cnd.Qw(i) - ...
              (dt/dx)*((Qwx(i).^2./Awx(i)) - ...
                       (Qwx(i-1).^2./Awx(i-1))) - ...
              (dt/dx)*const.g*Awx(i-1).*(hx(i)-hx(i-1)) + ...
              dt*const.g*cnd.Aw(i).*(pipe.i_bottom-cnd.i_fric(i));% + S

% Mass balance air
cnd.Ql(i)   = cnd.Ql(i-1) - (dx/dt)*(cnd.Al(i)-cnd.Al_old(i));

% Heat transfer
cnd.Tw(i)   = (cnd.Tw(i).*cnd.Aw_old(i) - ...
              (dt/dx)*((Qwx(i).*Twx(i)) - ...
                       (Qwx(i-1).*Twx(i-1))) + ...
               dt*S_alfa_w(i))./cnd.Aw(i);

% Heat transfer
cnd.Tl(i)   = cnd.Tl(i) - ...
              (dt/dx)*cnd.Ql(i).*(Tlx(i)-Tlx(i-1))./cnd.Al(i) + ...
```

```
              dt*S_alfa_l(i)./cnd.Al(i);

% Mass transfer water vapor
cnd.X(i)    = cnd.X(i) - ...
              (dt/dx)*cnd.Ql(i).*(Xx(i)-Xx(i-1))./cnd.Al(i) + ...
              dt*S_X(i)./cnd.Al(i);

% % Heat transfer pipe (water part)
% cnd.Tpw(i,:) = (cnd.Tpw(i,:).*cnd.Apw(i,:) + ...
%                 dt*S_Tpw(i,:))./cnd.Apw(i,:);
%
% % Heat transfer pipe (air part)
% cnd.Tpl(i,:) = (cnd.Tpl(i,:).*cnd.Apl(i,:) + ...
%                 dt*S_Tpl(i,:))./cnd.Apl(i,:);

% Store old surfaces
cnd.Al_old   = cnd.Al;
cnd.Aw_old   = cnd.Aw;

%% fun_equilibrium
function [Aw_eq,Ql_eq,X_eq] = fun_equilibrium(Qw,Tl,pipe,cond,const,num)
% Function to calculate equilibrium water area, air flow rate and water
% vapor
% Input:
% Qw            = water flow rate
% Tw            = water temperature
% pipe          = structure with pipe information
% const         = structure with constants
% num           = structure with numerical information
% Output:
% Aw_eq         = equilibrium water area
% Ql_eq         = air flow rate equilibrium
% X_eq          = water vapour equilibrium

% Initial conditions for iteration
Aw_eq = 1;
eps   = 1;
while eps>1E-3
    [h,B,Pw,Pl,Al] = fun_shape(Aw_eq,pipe,num);
    Aw_eq0 = Aw_eq;
    % Equilibrium between bottom slope and friction
    Aw_eq = (Qw^2*Pw*pipe.cf/(pipe.i_bottom*const.g))^(1/3);
    eps = abs(Aw_eq0 - Aw_eq);
end

% Mean water velocity
Uw = Qw./Aw_eq;
Uwstar = Uw*sqrt(pipe.cf);
kappa = 0.4;

% Velocity at water surface (for pipe)
Uwc = Uw + (Uwstar/kappa)*(3/2 + 2.3*log10(2*h/pipe.D));

% Equilibrium discharge of air
Ql_eq = 0.8560*(B/(Pl))*Uwc*Al;

% Equilibrium water vapour content
psat0       = const.ps0.*exp(const.Ts0./Tl);                  % mbar
X_eq        = 0.622*psat0*cond.phi/(cond.psys-psat0*cond.phi);

%% fun_shape
function [h,B,Pw,Pl,Al,Apw,Apl,Lpw,Lpl] = fun_shape(varargin)
% Function to calculate the geometric properties of the water area
% Input:
% Aw            = cross-sectional area of water
% pipe          = structure with properties of pipe
% num           = structure with numerical information
% Output:
% h             = water depth
% B             = width of water surface
```

```
% Pw             = perimeter of water
% Pl             = perimeter of air
% Al             = cross-sectional area of air
% Apw            = cross-sectional area of water part of pipe layer
% Apl            = cross-sectional area of air part of pipe layer
% Lpw            = wetted perimeter of water
% Lpl            = wetted perimeter of air

% Read variables
Aw      = varargin{1};
pipe    = varargin{2};
num     = varargin{3};

% Number of layes
nlayer  = num.nlayer;
% Declare variables
h = Aw*0; B = Aw*0; Pw = Aw*0; Pl = Aw*0; Al = Aw*0;
Lpw     = zeros(length(Aw),nlayer);
Lpl     = zeros(length(Aw),nlayer);
Apw     = zeros(length(Aw),nlayer);
Apl     = zeros(length(Aw),nlayer);

% For different shapes of conduit
switch pipe.shape
    % Pipe shape
    case {'pipe'}
        % Maximum area
        Aw_max = pi*pipe.D^2/4;
        Aw(Aw>Aw_max) = Aw_max;
        % Initialize
        gamma = Aw*0 + pi;
        eps = 1;
        eps_crit = 1E-3*length(Aw);
        % Determine angle by iteration
        while eps>eps_crit;
            dgamma = (gamma-sin(gamma)-(8*Aw/pipe.D^2))./(1-cos(gamma));
            gamma = gamma - dgamma;
            eps = sum(dgamma);
        end
        % Determine parameters
        h = (pipe.D/2)*(1-cos(gamma/2));
        B = 2*sqrt(pipe.D*h-h.^2);
        Pw = B + pipe.D*gamma/2;
        Pl = B + pipe.D*(pi-gamma/2);
        Al = Aw_max - Aw;
    case {'square'}
        Aw_max = pipe.D^2;
        h = Aw./pipe.D;
        B = h*0 + pipe.D;
        Pw = 2*h + 2*pipe.D;
        Pl = 2*(pipe.D-h) + 2*pipe.D;
        Al = Aw_max - Aw;
    case {'triangle'}
        h = sqrt(2*Aw);
        B = h;
        Pw = h*(1+sqrt(3));
end
% Determine pipe layer areas and perimeter for each pipe layer
for j = 1 : nlayer
       Lpw(:,j) = (Pw-B).*((j./nlayer)*pipe.Dp + pipe.D)/pipe.D;
       Lpl(:,j) = (Pl-B).*((j./nlayer)*pipe.Dp + pipe.D)/pipe.D;
       Apw(:,j) = (0.5*Lpw(:,j) + 0.5*(Pw-B).*(((j-1)./nlayer)*pipe.Dp +
pipe.D)/pipe.D).*pipe.Dp;
       Apl(:,j) = (0.5*Lpl(:,j) + 0.5*(Pl-B).*(((j-1)./nlayer)*pipe.Dp +
pipe.D)/pipe.D).*pipe.Dp;
end

%% fun_save
function model = fun_save(model,cnd,line,num,nstep)
```

```
if mod(nstep,num.nstep/num.nsave) == 0
    for n = 1 : line.n_cnd
       data.x  = cnd(n).x; data.zb = cnd(n).zb;
       data.Al = cnd(n).Al; data.Aw = cnd(n).Aw; data.h = cnd(n).h; data.Qw =
cnd(n).Qw; data.Ql = cnd(n).Ql; data.Tw = cnd(n).Tw;
       data.Tl = cnd(n).Tl; data.X = cnd(n).X; data.phi = cnd(n).phi;
data.psat = cnd(n).psat; data.Tpw = cnd(n).Tpw; data.Tpl = cnd(n).Tpl;
       data.Tsw = cnd(n).Tsw; data.Tsl = cnd(n).Tsl;
       if nstep == 0
           % Stationary solution
           model.stat = data;
       else
           % Time dependent solution
           model.data{n,num.nsave*nstep/num.nstep} = data;
       end
    end
end
```

B.4 datatime

```
% Only for one row

function timecell = datatime(model, x)

timecell = cell(length(x), 1);

for i = 1:length(x)

    xi = x(i);

    data = model.data;
    stat = model.stat;
    num = model.num;

    sizeSteps = size(data,2)+1;

    zb_t = zeros(sizeSteps, 1);
    Al_t = zeros(sizeSteps, 1);
    Aw_t = zeros(sizeSteps, 1);
    h_t  = zeros(sizeSteps, 1);
    Qw_t = zeros(sizeSteps, 1);
    Ql_t = zeros(sizeSteps, 1);
    Tw_t = zeros(sizeSteps, 1);
    Tl_t = zeros(sizeSteps, 1);
    X_t = zeros(sizeSteps, 1);
    phi_t = zeros(sizeSteps, 1);
    psat_t = zeros(sizeSteps, 1);
    Tpw_t = zeros(sizeSteps, num.nlayer);
    Tpl_t = zeros(sizeSteps, num.nlayer);
    Tsw_t = zeros(sizeSteps, num.nlayer);
    Tsl_t = zeros(sizeSteps, num.nlayer);

    Tw_t(1) = stat.Tw(xi);
    Tl_t(1) = stat.Tl(xi);
    Qw_t(1) = stat.Qw(xi);
    Ql_t(1) = stat.Ql(xi);
    h_t(1) = stat.h(xi);
    phi_t(1) = stat.phi(xi);
    zb_t(1) = stat.zb(xi);
    Al_t(1) = stat.Al(xi);
    Aw_t(1) = stat.Aw(xi);
    X_t(1) = stat.X(xi);
    psat_t(1) = stat.psat(xi);
    Tpw_t(1,:) = stat.Tpw(xi,:);
    Tpl_t(1,:) = stat.Tpl(xi,:);
    Tsw_t(1,:) = stat.Tsw(xi,:);
```

```
    Tsl_t(1,:) = stat.Tsl(xi,:);

    for lineStep = 1:size(data,2)

        Tsw_t  (lineStep+1,:) = data{lineStep}.Tsw(xi,:);
        Tsl_t  (lineStep+1,:) = data{lineStep}.Tsl(xi,:);
        Tw_t   (lineStep+1) = data{lineStep}.Tw(xi);
        Tl_t   (lineStep+1) = data{lineStep}.Tl(xi);
        Tpw_t  (lineStep+1,:) = data{lineStep}.Tpw(xi,:);
        Tpl_t  (lineStep+1,:) = data{lineStep}.Tpl(xi,:);
        Qw_t   (lineStep+1) = data{lineStep}.Qw(xi);
        Ql_t   (lineStep+1) = data{lineStep}.Ql(xi);
        h_t    (lineStep+1) = data{lineStep}.h(xi);
        phi_t  (lineStep+1) = data{lineStep}.phi(xi);
        zb_t   (lineStep+1) = data{lineStep}.zb(xi);
        Al_t   (lineStep+1) = data{lineStep}.Al(xi);
        Aw_t   (lineStep+1) = data{lineStep}.Aw(xi);
        X_t    (lineStep+1) = data{lineStep}.X(xi);
        psat_t(lineStep+1) = data{lineStep}.psat(xi);

    end

    dt = num.dt;
    t = zeros(sizeSteps, 1);

    for iStep = 1:sizeSteps
        t(iStep) = sum(dt(1:(iStep-1)*num.nstep/num.nsave));
    end

    datat.xi = xi;
    datat.t = t;

    datat.zb = zb_t;
    datat.Al = Al_t;
    datat.Aw = Aw_t;
    datat.h = h_t;
    datat.Qw = Qw_t;
    datat.Ql = Ql_t;
    datat.Tw = Tw_t;
    datat.Tl = Tl_t;
    datat.X = X_t;
    datat.phi = phi_t;
    datat.psat = psat_t;
    datat.Tpw = Tpw_t;
    datat.Tpl = Tpl_t;
    datat.Tsw = Tsw_t;
    datat.Tsl = Tsl_t;

    timecell{i} = datat;

end

end
```

B.5 fun_plot

```
function fun_plot(model)
close all

linecolor = {'b','r','k','g','m'};

for n = 1 : model.line.n_cnd+1
    figure(n)
    for i = 1 : model.num.nplot;
        if n > model.line.n_cnd
            data = model.stat;
            t    = 0;
```

```
            pipe = model.pipe(1);
        else
            pipe = model.pipe(n);
            ii   = floor(i*model.num.nsave/model.num.nplot);
            data = model.data{n,ii};
            t    = sum(model.num.dt(1:ii*model.num.nstep/model.num.nsave));
        end
        subplot(221)
        hold on
        plot(data.x,data.h+data.zb,'color',linecolor{i},'linewidth',1)
        % plot(data.x,pipe.D-data.h+data.zb,'--
','color',linecolor{i},'linewidth',1)
        plot(data.x,data.zb,'k','linewidth',2)
        plot(data.x,data.zb+pipe.D,'k','linewidth',2)
        ylim([0 pipe.D+max(data.zb)])
        %title('Water depth (h [m])')
        xlim([0 pipe.L])
        axis('tight');
        set(gca,'FontSize', 17,'FontName','Courier');
        ylabel('h (m)','FontSize',19,'FontWeight', 'bold','FontName','Arial');
        xlabel('Distance (m)','FontSize',19,'FontName','Arial');

        subplot(222)
        plot(data.x,data.Qw*1000,'color',linecolor{i},'linewidth',1)
        hold all
        %ylim([0 0.5])
        %title('Flow rate water (Q_w [L/s])')
        xlim([0 pipe.L])
        axis('tight');
        set(gca,'FontSize', 17,'FontName','Courier');
        ylabel('Q_w (L/s)','FontSize',19,'FontWeight',
'bold','FontName','Arial');
        xlabel('Distance (m)','FontSize',19,'FontName','Arial');


        subplot(224)
        plot(data.x,data.Tw-273,'color',linecolor{i},'linewidth',1)
        %plot(data.x,data.zb,'k','linewidth',2)
        hold all
        %title('Temperature water (T_w [C])')
        xlim([0 pipe.L])
        axis('tight');
        set(gca,'FontSize', 17,'FontName','Courier');
        ylabel('T_w (C)','FontSize',19,'FontWeight',
'bold','FontName','Arial');
        xlabel('Distance (m)','FontSize',19,'FontName','Arial');
        ylim([0 25])

        subplot(223)
        plot(data.x,data.Tl-273,'color',linecolor{i},'linewidth',1)
        hold all
        %title('Temperature air T_l [C]')
        xlim([0 pipe.L])
        axis('tight');
        set(gca,'FontSize', 17,'FontName','Courier');
        ylabel('T_l (C)','FontSize',19,'FontWeight',
'bold','FontName','Arial');
        xlabel('Distance (m)','FontSize',19,'FontName','Arial');

    end
end
```

B.6 fun_plot_t

```
function fun_plot_t(model,datatx)
%close all
```

```
linecolor = {'b','r','k','g','m','b','r','k','g','m','b','r','k','g','m'};

for plotI = 1:length(datatx)

    %figure;
    hold on;

    datat = datatx{plotI};
    pipe  = model.pipe(model.line.n_cnd);
    timeMax = max(datat.t);

    subplot(221)
    hold on
    plot(datat.t,datat.h+datat.zb,'color',linecolor{plotI},'linewidth',1)
    plot(datat.t,datat.zb,'k','linewidth',2)
    plot(datat.t,datat.zb+pipe.D,'k','linewidth',2)
    ylim([0 pipe.D+max(datat.zb)])
    %title('Water depth (h [m])')
    xlim([0 timeMax])
    axis('tight');
    set(gca,'FontSize', 17,'FontName','Courier');
    ylabel('h (m)','FontSize',19,'FontWeight', 'bold','FontName','Arial');
    xlabel('Time (s)','FontSize',19,'FontName','Arial');

subplot(222)
    plot(datat.t,datat.Qw*1000,'color',linecolor{plotI},'linewidth',1)
    hold all
    %title('Flow rate water (Q_w [L/s])')
    xlim([0 timeMax])
    ylim([0 4E-3])
    axis('tight');
    set(gca,'FontSize', 17,'FontName','Courier');
    ylabel('Q_w (L/s)','FontSize',19,'FontWeight', 'bold','FontName','Arial');
    xlabel('Time (s)','FontSize',19,'FontName','Arial');

    subplot(224)
    plot(datat.t,datat.Tw-273,'color',linecolor{plotI},'linewidth',1)
    %plot(data.x,data.zb,'k','linewidth',2)
    hold all
    %title('Temperature water (T_w [C])')
    xlim([0 timeMax])
    axis('tight');
    set(gca,'FontSize', 17,'FontName','Courier');
    ylabel('T_w (C)','FontSize',19,'FontWeight', 'bold','FontName','Arial');
    xlabel('Time (s)','FontSize',19,'FontName','Arial');
    ylim([0 25])

    % ylim([0 1])
    subplot(223)
    plot(datat.t,datat.Tl-273,'color',linecolor{plotI},'linewidth',1)
    hold all
    %title('Temperature air T_l [C]')
    xlim([0 timeMax])
    axis('tight');
    set(gca,'FontSize', 17,'FontName','Courier');
    ylabel('T_l (C)','FontSize',19,'FontWeight', 'bold','FontName','Arial');
    xlabel('Time (s)','FontSize',19,'FontName','Arial');
    ylim([0 25])

end
```

B.7 fun_plot_color

```
function fun_plot_color(model)
%% Calculates time array

data = model.data;
```

```
num = model.num;

t = zeros(size(data,2), 1);
X = data{1}.x;

no_times = length(t);
no_points = length(X);

Tw = zeros(no_times, no_points);
%Qw = zeros(no_times, no_points);

for i = 1:no_times
    t(i) = sum(num.dt(1:i*num.nstep/num.nsave));
    Tw(i,:) = data{i}.Tw-273;
    %Qw(i,:) = data{i}.Qw;
end

%% Plot
figure;

Twrange=0:1:25;

contourf(t,X,Tw',Twrange);
%contourf(t,X,Qw')

%surf(t,X,Tw',Twrange);

colbar = colorbar;

set(gca,'FontSize', 17);
set(gca,'FontName','Courier');
ylabel(colbar, 'Temperature (∫C)','FontSize', 19,'FontWeight',
'bold','FontName','Arial');
xlabel('Time(s)','FontSize', 19, 'FontWeight', 'bold','FontName','Arial');
ylabel('Distance (m)', 'FontSize', 19,'FontWeight',
'bold','FontName','Arial');
% xlim([0,50]);
```

B.8 mcdeviations

```
writeFiles = 0;

%% Calculates general case
gcModel       = fun_start(ones(1,5));
gcModel       = sewer_temp_model(gcModel);

fun_plot_color(gcModel);
title('GC\_Model');

%% Builds modified cases matrix
% MC matrix
nVar1 = 3;
nVar2 = 2;
mVarf =
[fun_mat(nVar1),ones(2^nVar1,nVar2);ones(2^nVar2,nVar1),fun_mat(nVar2)];
% deletes general case entries
mVarf(sum(mVarf,2) == size(mVarf,2), :) = [];

%% Calculates modified cases
% runs and saves models
if writeFiles
    for i = 1:size(mVarf,1)
        row = mVarf(i,:);
        model = fun_start(row);
        model = sewer_temp_model(model);
        save(['Dynamic_MC_',num2str(i)],'model');
    end
```

```
end

%% Calculates mean squared deviation and std deviation for each MC
meanDeviationTw = zeros(size(mVarf,1), 1);
meanDeviationTl = zeros(size(mVarf,1), 1);
stdDeviationTw = zeros(size(mVarf,1), 1);
stdDeviationTl = zeros(size(mVarf,1), 1);

for i = 1:size(mVarf,1)
    model = load(['Dynamic_MC_',num2str(i)]);
    model = model.model;
    fun_plot_color(model);
    title(['Dynamic\_MC\_',num2str(i)]);
    data = model.data;

    % only considering one conduit
    deviationDataTw = zeros(size(data,2),size(data{1}.x,1));
    deviationDataTl = zeros(size(data,2),size(data{1}.x,1));

    for dataI = 1:size(data,2)
        deviationDataTw(dataI,:) = (data{dataI}.Tw -
gcModel.data{dataI}.Tw)./(gcModel.data{dataI}.Tw-273)*100;
        deviationDataTl(dataI,:) = (data{dataI}.Tl -
gcModel.data{dataI}.Tl)./(gcModel.data{dataI}.Tl-273)*100;
    end

    meanDeviationTw(i) = mean2(deviationDataTw);
    meanDeviationTl(i) = mean2(deviationDataTl);
    stdDeviationTw(i) = std2(deviationDataTw);
    stdDeviationTl(i) = std2(deviationDataTl);

%     plot(mean(deviationDataT));

end
%% Plots deviation graphsm
figure;

subplot(121)
%plot(abs(meanDeviationTw), 'rx-');
deviationbar([1:10], meanDeviationTw, stdDeviationTw, 'rx')
hold on;
bar([1:10], meanDeviationTw,0.5,'r');
set(gca,'FontSize', 15,'FontName','Courier');
set(gca,'XTick',[1:10]);
ylabel('Deviation (Tw)','FontSize',15,'FontWeight',
'bold','FontName','Arial');
xlabel('Modified Cases','FontSize',15,'FontWeight',
'bold','FontName','Arial');
xlim([0.5,10.5]);

subplot(122)
%plot(abs(meanDeviationTl), 'rx-');
deviationbar([1:10], meanDeviationTl, stdDeviationTl, 'rx')
hold on;
bar([1:10], meanDeviationTl,0.5,'r');
set(gca,'FontSize', 15,'FontName','Courier');
set(gca,'XTick',[1:10])
ylabel('Deviation (Tl)','FontSize',15,'FontWeight',
'bold','FontName','Arial');
xlabel('Modified Cases','FontSize',15,'FontWeight',
'bold','FontName','Arial');
xlim([0.5,10.5]);
```