NETWORK MODELLING
OF
TRANSIENT HEAT EXCHANGER PERFORMANCE

Jacobus C Olivier
B.Eng. (Mechanical)

Thesis submitted in partial fulfilment of the requirements for the degree
Magister Engineering (Mechanical)
School of Mechanical and Materials Engineering
at the
Potchefstroom University for Christian Higher Education

Supervisor: Prof. G.P. Greyvenstein
Potchefstroom
2005

ABSTRACT

This study investigates the applicability of the thermal-fluid network approach to the modelling of transient heat exchanger performance.

Two different solution algorithms, namely the Implicit Pressure Correction Method (IPCM) and the Runge Kutta method with Trapezoidal Damping (RKTD) for the solution of the one-dimensional governing equations in thermal-fluid network problems are presented. The advantages and disadvantages of two types of numerical discretisation schemes used in thermal-fluid network problems are discussed and the discretised one-dimensional governing equations for the staggered grid discretisation scheme used in the IPCM and RKTD method is presented. The RKTD method is used as a time integration scheme for the generalised thermal-fluid network solver Xnet. Several test cases are introduced and the basic primitive elements available in Xnet are compared to the commercial thermal-fluid network code, Flownex (which uses the IPCM), for both steady-state and transient conditions.

Two different network topologies are introduced for the discretisation of heat exchangers when a network approach is followed and the thermal-fluid network solver Xnet is applied to a basic parallel and counter flow configured pipe-in-pipe type heat exchanger to investigate the effect on the type of discretisation scheme used. The results obtained are compared to primitive element models in Flownex as well as the composite RX element in Flownex.

The extent to which thermal-fluid network solvers are able to predict transient heat exchanger performance are further investigated by modelling a complex shell-and-tube heat exchanger using Xnet and comparing the steady-state and transient results to both a primitive element model in Flownex as well as the composite STX element in Flownex. This contributes to the validation of Flownex's heat exchanger models by using a different approach than Flownex.

The results showed that the explicit method used in Xnet is capable of solving large arbitrary structured thermal-fluid networks with a high level of accuracy. The result of Flownex compares very well with that of Xnet, which proves (verifies) that the solution algorithm is correctly implemented in both codes. Even though the explicit thermal-fluid network code, Xnet, can accurately predict fast transients, a drawback of this method is the large computational time required to simulate transient heat exchangers with large thermal masses.

UITTREKSEL

Hierdie studie ondersoek die geldigheid van die termovloei netwerkbenadering vir die modellering van transiënte hitteruiler werkverrigting.

Twee verskillende oplos-algoritmes, naamlik die Implisiete Druk-korreksie Metode (IPCM) en die Runge Kutta metode met Trapesoïdale Demping (RKTD) om die eendimensionele behoudsvergelykings in termovloeinetwerkprobleme mee op te los, word bespreek. Die voor-en nadele van twee tipes numeriese diskretiseringsmetodes wat gebruik word in termovloei netwerkprobleme, word bespreek en die gediskretiseerde een-dimensionele behoudsvergelykings vir 'n verspringende-rooster-diskretiseringskema word verduidelik vir beide die IPCM en die RKTD metode. Die RKTD metode word gebruik as die tydintegrasieskema vir 'n algemene termovloei netwerkoplosser, Xnet. Talle toetsgevalle word aangebied en die basiese primitiewe elemente wat beskikbaar is in Xnet word vergelyk vir beide gestadigde en ongestadigde vloei met die kommersiële termovloei kode Flownex (wat gebruik maak van die IPCM).

Twee verskillende netwerk topologieë word aangetoon om hitteruilers mee te diskretiseer wanneer 'n netwerkbenadering gevolg word. Die termovloei netwerkoplosser Xnet word ook toegepas op 'n basiese parallel en teenvloei pyp-in-pyp tipe hitteruiler om die effek van die tipe netwerk topologie te ondersoek. Die resultate wat verkry is, word vergelyk met primitiewe element modelle in Flownex asook die saamgestelde RX element in Flownex.

Die vermoë van termovloei netwerk kodes om transiënte hitteruilerwerkverrigting te voorspel is verder ondersoek deur die modellering van komplekse hitteruilers en is sodoende toegepas op 'n buis-en-huls-hitteruiler en die resultate wat verkry is, vir gestadigde en ongestadigde toestande is vergelyk met 'n primitiewe element model wat opgestel is in Flownex en ook die saamgestelde STX element in Flownex. Hierdie studie het sodoende bygedra tot die validasie van Flownex se hitteruiler modelle deur van 'n totaal verskillende benadering gebruik te maak.

Die resultate toon dat die eksplisiete metode wat in Xnet gebruik word, in staat is om groot arbitrêre termovloei netwerke op te los met 'n hoë vlak van akkuraatheid. Die resultate van Flownex vergelyk baie goed met dié van Xnet. Dit lewer bewys (verifikasie) van die feit dat die oplosalgoritme korrek geïmplimenteer is in beide kodes. Alhoewel die eksplisiete termovloei netwerk kode vinnige transiente baie akkuraat kan voorspel/oplos, is die berekeningstyd wat nodig is om hitteruilers met groot termiese massas transiënt te simuleer, baie groot.

ACKNOWLEDGEMENTS

- Our heavenly father who gave me the talent and granted me the opportunity to achieve my goals in life.

- I would like to express my gratitude to my supervisor, professor Gideon Greyvenstein, for his contribution to this study and his constant motivation and support.

- To my family who supported me through my years of study. The example you set, gave me the inspiration to succeed and achieve my best.

- Jineane, who helped with the proofreading and who were always there when I needed motivation and support.

- My colleagues, for their support and insight throughout this study.

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE AND ABBREVIATIONS

NOMENCLATURE

A = Area

CS = Control surface

CV = Control volume

C = Constant in Dittus-Boelter equation, Product of $\dot{m}$ and c_p, Courant number

c_p = Specific heat at constant pressure

c = Velocity of sound

D_H = Hydraulic diameter

E = Energy

e = Energy per unit mass, Roughness

f = Friction factor

g = Gravitational acceleration

h = Heat transfer coefficient

h_0 = Total enthalpy

k = Thermal conductivity

L = Length

M = Mach number

m = Constant in Dittus-Boelter equation

$\dot{m}$ = Mass flow

n = Constant in Dittus-Boelter equation, Number of time steps, Increment number

N_u = Nusselt number

p = Static pressure

$\overline{P}$ = Linear momentum

p_0 = Total pressure

Pr = Prandtl number

Q = Nett heat / Volumetric flow rate

$\dot{q}$ = Heat transfer rate

R = Gas constant

Re = Reynolds number

s = Compressibility factor

T = Static temperature

T_0 = Total temperature

t = Time

u = Internal energy

V = Velocity

v = Volume

W_N = Nett work

x = Length in the direction of the flow

y = Solution to differential equation

z = Elevation above some datum

GREEK SYMBOLS

α = Time step factor

Δp = Pressure drop

Δq = Effective temperature difference in LMTD method

Δt = Length of time step / Change in time

ΔT = Change in temperature

Δx = Distance between nodes in x-direction

Δy = Distance between nodes in y-direction

ε = Effectiveness

ρ = Density

θ = Temperature difference in LMTD method

ω = Damping factor

γ = Ratio of the specific heats

SUBSCRIPS, SUPERSCRIPS

i = Control volume index / Inlet

c = Cold

e = East

eff = Effective

h = Hot

n = Time step

max = Maximum

min = Minimum

o = Outlet

t = Time

w = West

ABBREVIATIONS

CAD	=	Computer Aided Design
CF	=	Counter flow
CFD	=	Computational Fluid Dynamics
CHT	=	Conductive Heat Transfer
E-NTU	=	Effectiveness-Number Of Transfer Units
FEM	=	Finite Element Method
FN	=	Flownex
GUI	=	Graphical User Interface
HPC	=	High Pressure Compressor
HPT	=	High Pressure Turbine
IC	=	Inter cooler
LMTD	=	Log Mean Temperature Difference
LPC	=	Low Pressure Compressor
LPT	=	Low Pressure Turbine
LW	=	Lax Wendroff
MOC	=	Method Of Characteristics
NTU	=	Number Of Transfer Units
ODE	=	Ordinary Differential Equations
PC	=	Pre cooler
PCU	=	Power Conversion Unit
PF	=	Parallel flow
PBMR	=	Pebble Bed Modular Reactor
PBR	=	Pebble Bed Reactor
PIPEDG	=	Duct with Area Change Pipe
PIPEDW	=	Darcy-Weisbach Pipe
PIPEHX	=	Heat Exchanger Pipe
PIPEPB	=	Pebble Bed Pipe
RX	=	Recuperator
RK	=	Runge Kutta
STX	=	Shell-And-Tube Heat Exchanger

CHAPTER 1 - INTRODUCTION

1.1 Introduction

Simulation has become increasingly important in the design of complex thermal-fluid systems such as power plants. Simulation of thermal-fluid systems entails the determination of pressures, temperatures and mass flow rates of the components within the thermal-fluid network. Thermal-fluid networks consist of several subsystems and heat exchangers are a vital part of these complex systems. Even though heat exchangers play such an important role in the performance of thermal-fluid systems, very little attention has been given to the transient simulation of heat exchangers as part of a larger system. The reason for this is the complexity of the dynamic modelling of heat transfer in heat exchangers and larger networks and therefore very few thermal-fluid analyses on these complex integrated systems are done on this complex level.

With the ever-increasing importance of the simulation of thermal-fluid systems such as the Pebble Bed Modular Reactor (PBMR), the need arose for methods to accurately predict transient behaviour of these integrated and complex systems.

1.2 Background

The PBMR is an excellent example of a complex thermal-fluid system where heat exchangers play a vital role in the performance of the overall system. The PBMR is a new type of high temperature gas nuclear power plant based on a closed cycle three-shaft recuperative Brayton cycle. The following is a description of the thermodynamic cycle of the PBMR.

In Figure 2.1 helium enters the reactor (PBR) and gains heat from the hot fuel spheres that are heated by the nuclear reaction, as the helium passes through the packed bed. The helium then leaves the reactor and expands through the high-pressure turbine (HPT), which drives the high-pressure compressor (HPC). This turbine forms part of the high-pressure turbo-unit. Next, the helium is further expanded through the low-pressure turbine (LPT), which is part of the low-pressure turbo-unit. The LPT also drives the low-pressure compressor (LPC). The helium then expands through the power turbine (PT) driving the generator. At this point, the helium is still at a high temperature. It then flows through the primary side of the recuperator (RX), where it transfers heat to the low temperature gas returning to the reactor.

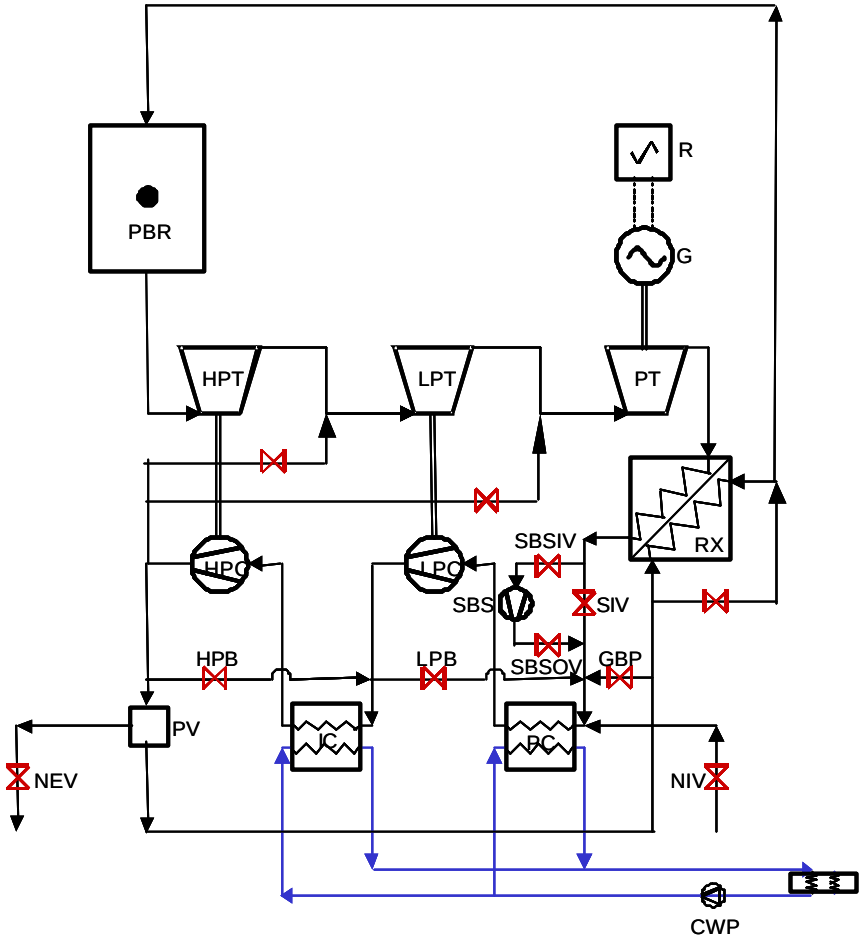


Figure 2.1: Schematic layout of the PBMR thermal cycle.

After leaving the recuperator the helium is further cooled in the pre-cooler, before entering the LPC. The helium is then compressed by the LPC and cooled in the inter-cooler (IC). This process increases the density and improves the efficiency of the compressor. The HPC compresses the helium to an even higher pressure. The high-pressure helium stream then flows through the RX where it is pre-heated, after which it returns to the reactor.

The PBMR makes use of different control strategies to adapt to changing electrical load demand. For the system to adapt to the changing load demand, it is necessary to do load rejection in the case of a sudden decrease in load. Load rejection is done by a reduction of the flow through the three turbines by means of a control valve that diverts high-pressure gas which has left the HPC to a point just before the pre-cooler.

When this happens, pressure and mass flow transients occur in the system, and this affects the behaviour and performance of the individual components, and therefore the performance of the system as a whole. The heat exchangers usually have large thermal inertias and do not react immediately to such transients. In order to model the overall performance of the system as a whole, accurate prediction of the transient performance of the heat exchangers in the cycle is essential.

1.3 Problem statement

Although there are several thermal-fluid network codes available on the market, only a few can model transient problems and even fewer can deal with heat exchangers as distributed systems or sub-networks. The distributed approach is necessary to accurately predict the transient behaviour of heat exchangers. One such code is Flownex, developed within the School of Mechanical and Materials Engineering at Potchefstroom University for Christian Higher Education. Flownex is a powerful thermal-fluid network analysis code that is able to model transient heat exchanger performance in a distributed manner or as part of a larger network.

Flownex is used by the company PBMR (Pty) Ltd to model the Power Conversion Unit (PCU) of the PBMR, and as part of the verification and validation of Flownex the need exists to use or to develop a new code that is able to model transient heat exchanger performance in a discretised manner (distributed approach) to aid in the validation of Flownex. Since this code will be used as a validation tool, it is necessary to follow a totally different approach from Flownex. An attractive approach will be to use an explicit code, since explicit codes yield very accurate results. To ensure the accuracy, stability and speed of the method used, choices will have to be made between different numerical discretisation schemes and solution algorithms. Other issues are the physical representation of three-dimensional thermal-fluid networks with certain numerical discretisation schemes, and the physical discretisation of the thermal model at hand. These issues indicate the complexity of deciding which type of method to use.

1.4 Objective of study

The objective of this study is to validate Flownex's capability to simulate steady-state and transient heat exchanger performance by comparing it to another code. Since no other codes that can simulate the transient performance of heat exchangers as part of a larger network could be found, a new code will have to be developed. This code, together with Flownex, will then be used to investigate the transient performance of heat exchangers. A good comparison of results will prove that the governing equations and solution algorithm are correctly implemented in both codes. This is an important step in the verification process of the codes.

1.5 Layout of thesis

In the following chapter, the different methods used to predict general heat exchanger performance will be discussed. A brief survey on previous work done on the simulation of heat exchanger performance will be given, as well as a discussion on two general approaches used to simulate thermal-fluid problems. A discussion on the differences between explicit and implicit solution algorithms will be given and three explicit solution algorithms used in thermal-fluid modelling are discussed. The chapter concludes with a brief overview of existing thermal-fluid network codes.

In Chapter 3, two thermal-fluid network solution methods, namely the *Implicit Pressure Correction Method (IPCM)* and the *Runge Kutta method with Trapezoidal Damping (RKTD)* will be discussed in sufficient detail to show the reader the difference between the two methods.

In Chapter 4 a general description of the generalised network solver Xnet will be given, as well as a description of the applicable elements available in the code for this study. The rest of the chapter will be dedicated to an extensive comparison of Xnet's and Flownex's basic flow and heat transfer elements.

In Chapter 5 an annular type tube-in-tube parallel and counter flow configured heat exchanger will be used to investigate the effect of different network topologies on the accuracy of the heat exchanger results. The purpose of this specific study will be to determine which type of network topology would yield the most accurate results for a certain level of discretisation.

Chapter 6 will give a brief overview of complex heat exchangers and an in-depth investigation into the capability of Xnet and Flownex in predicting transient heat exchanger performance.

Chapter 7 will conclude with a summary and recommendations for further research.

CHAPTER 2 - LITERATURE SURVEY

2.1 Introduction

The process of heat exchange between two fluids that are at different temperatures and separated by a solid wall occurs in many engineering applications. The device used to implement this heat exchange is known as a heat exchanger, and specific applications may be found in space heating, air-conditioning, power production, waste heat recovery and in the chemical processing industry.

In this chapter, the different methods used to predict general heat exchanger performance will be discussed. A brief survey of previous work done on the simulation of heat exchanger performance will be given, as well as a discussion of two general approaches used to simulate thermal-fluid problems. A discussion of the differences between explicit and implicit solution algorithms will be given and three explicit solution algorithms used in thermal-fluid modelling will be discussed. The chapter will be concluded with a brief overview of existing thermal-fluid network codes.

2.2 Simulation of heat exchanger performance

Prediction of heat exchanger performance can be done for both steady-state and transient conditions. Two approaches for steady-state heat exchanger performance analysis exist. These two approaches are the lumped approach and the distributed (or discretised) approach. In the case of the lumped approach, there are two analytical methods that could be used for a steady-state heat exchanger performance analysis. For transient heat exchanger performance analysis, the only approach that could be used is the distributed approach. Figure 2.1 schematically represents the process of heat exchanger performance described above. The lumped and distributed approaches will be discussed in the sections to follow.

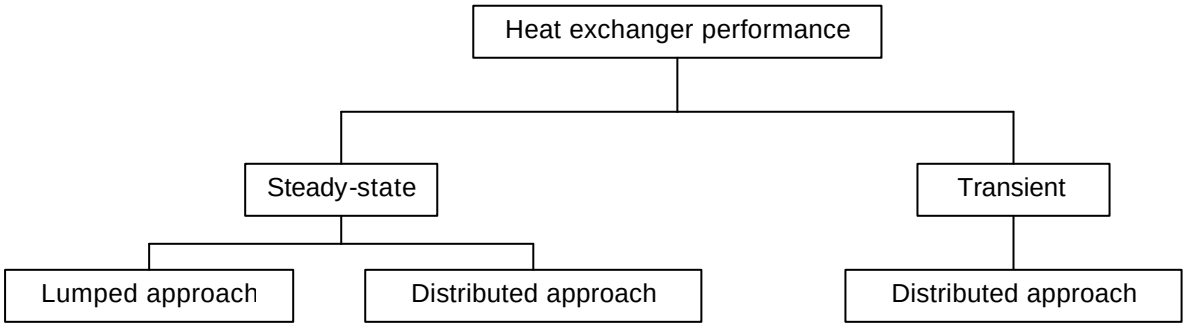


Figure 2.1: Heat exchanger performance analysis process.

2.2.1 Lumped system approach

The lumped system approach describes the heat exchanger as a lumped model. This means that it is assumed that the density, r, and thermal conductivity, k, of the fluid, as well as the surface heat transfer coefficient, h, stays constant through the heat exchanger. Two analytical lumped methods exist, namely the Log Mean Temperature Difference Method (LMTD), and the Effectiveness-Number Of Transfer Units (E-NTU) method. In the LMTD approach, the geometry is described by a form factor, F, and is used in the general heat exchanger equation, Equation (2.1) to calculate the heat transfer. The given parameters necessary for simulation with the lumped approach, would therefore be the inlet conditions of both the hot and cold fluid as well as the overall heat transfer coefficient, U, the heat transfer area, A, and the form factor, F (Figure 2.2).

Figure 2.2: Schematic representation of the lumped system approach.

The general heat exchanger equation is written in terms of the effective temperature difference between the hot and cold fluid, q_{eff}, as:

$$Q = U A_0 F \Delta q_{eff} \tag{2.1}$$

This equation, combined with the First Law equations, defines the energy flows for a heat exchanger. For the hot fluid the First Law equation is written in terms of the temperature change that the hot fluid undergoes, ΔT_h, as:

$$Q = -\dot{m}_h c_{p_h} \Delta T_h \tag{2.2}$$

For the cold fluid, the First Law equation may be written in terms of the cold fluid temperature change, ΔT_c, as:

$$Q = \dot{m}_c c_{p_c} \Delta T_c \tag{2.3}$$

In the case of the LMTD method, the driving force for the heat transfer between the hot and cold streams is the temperature difference across the tube wall. As seen in Figure 2.3, the difference will vary with the axial location, so that it may be referred to in terms of the effective or integrated average temperature differences.

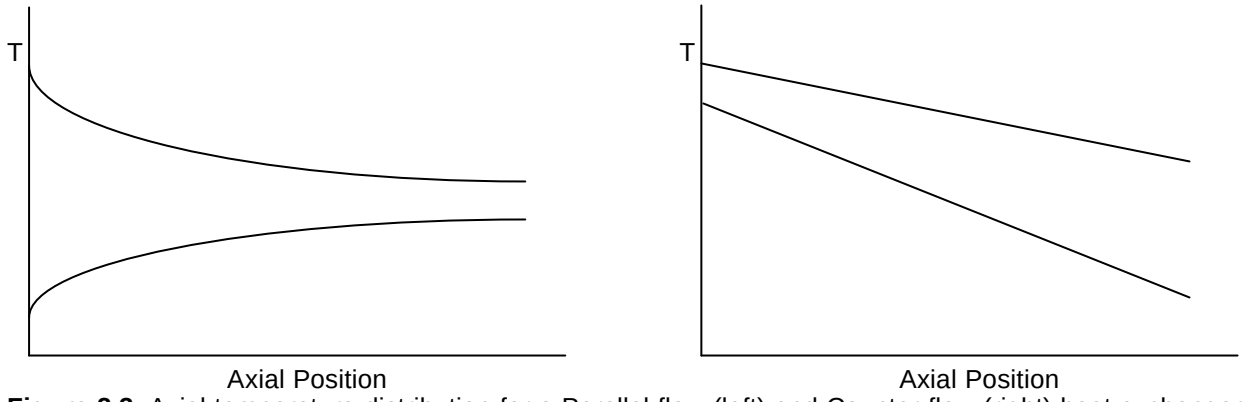


Figure 2.3: Axial temperature distribution for a Parallel flow (left) and Counter flow (right) heat exchanger.

By using Equations (2.1), (2.2) and (2.3) it may be illustrated that the integrated average temperature difference for either parallel or counterflow may be written as (Incropera and De Witt, 1996):

$$\Delta q_{eff} = LMTD = \frac{q_1 - q_2}{\ln\left(\frac{q_1}{q_2}\right)} \tag{2.4}$$

The effective temperature difference, Δq_{eff}, is also known as the log mean temperature difference (LMTD). Equation (2.4) applies to all types of heat exchangers, whilst the shape factor, F, in Equation (2.1) is different for different types of heat exchangers such as parallel flow, counter flow and cross flow heat exchangers.

It is a simple matter to use the LMTD method for heat exchanger analysis when the fluid temperatures are known and the outlet temperatures are specified, or readily determined from the energy balance expressions. The value of Δq_{eff} for the exchanger may then be determined. However, if only the inlet temperatures are known, use of the LMTD method requires an iterative procedure (Incropera and De Witt, 1996). In a situation such as this, the Effectiveness-NTU method is more convenient to use.

The maximum possible heat transfer rate that could possibly be delivered by the heat exchanger, is defined as:

$$Q_{\max} = C_{\min}\left(T_{h,i} - T_{c,i}\right) \tag{2.5}$$

where $C_{\min}$ is the minimum value of $(\dot{m}_h c_{p_h}, \dot{m}_c c_{p_c})$. The effectiveness, e, of the heat exchanger is then defined as:

$$e \equiv \frac{Q}{Q_{\max}} \tag{2.6}$$

Substitution of Equations (2.5) and (2.2) into Equation (2.6) leads to the following expression for effectiveness:

$$e \equiv \frac{C_h \left(T_{h,i} - T_{h,o} \right)}{C_{\min} \left(T_{h,i} - T_{c,i} \right)} \tag{2.7}$$

By definition, the effectiveness must be in the range $0 \le e \le 1$. The actual heat transfer could then be determined from Equations (2.5) and (2.6). For any type of heat exchanger, it can be shown that (Kays and London, 1984):

$$e = f\left(NTU, \frac{C_{\min}}{C_{\max}} \right) \tag{2.8}$$

where

$$C_{\min} = \min(\dot{m}_h c_{p_h}, \dot{m}_c c_{p_c})$$
$$C_{\max} = \max(\dot{m}_h c_{p_h}, \dot{m}_c c_{p_c})$$

The number of transfer units is a dimensionless parameter that is defined as:

$$NTU \equiv \frac{UA}{C_{\min}} \tag{2.9}$$

Specific forms of Effectiveness-NTU relations for different types and configurations of heat exchangers may be found in Kays and London (1984).

2.2.2 Discretised approach

In the discretised approach, the heat exchanger is subdivided into a number of interconnected elemental heat exchangers as shown in Figure 2.4. Differential equations describing conservation of mass, momentum and energy are written for each elemental flow conduit, while the energy equation is applied to the mass of metal separating the hot and cold streams. The system of equations is then solved with an appropriate method. With the discretised approach, knowledge of the internal configuration is therefore essential.

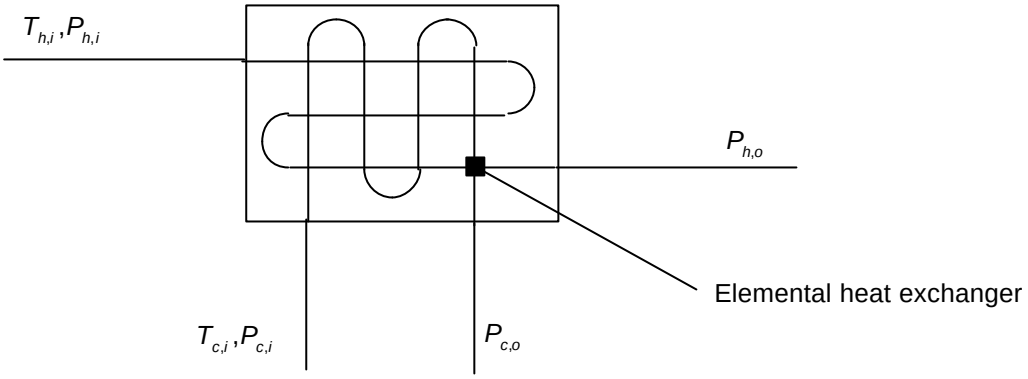


Figure 2.4: Schematic representation of the discretised approach.

2.2.3 Need for the discretised approach

When a steady-state analysis is done on heat exchanger performance and the density, r, thermal conductivity, k, and viscosity, m, of the fluid, as well as the surface heat transfer coefficient, h, stays constant throughout the heat exchanger, the lumped approach produces quick and accurate results that are not dependant on the physical size of the heat exchanger. When these properties vary significantly throughout the heat exchanger, the discretised approach needs to be used to properly take the effect of varying properties into account. For a transient heat exchanger performance analysis, the lumped approach cannot be used. The reason for this is that the density, r, thermal conductivity, k, and viscosity, m, of the fluid and the surface heat transfer coefficient, h, varies throughout the heat exchanger and the thermal inertia of the heat exchanger needs to be taken into account. The lumped models do not take thermal inertia into account, which is critical in transient heat exchanger performance analysis. A discretised approach is therefore essential if one wants to model transient heat exchanger performance.

2.3 Survey of previous work on simulation of heat exchanger performance

An extensive literature survey was conducted into the field of transient or dynamic modeling of complex heat exchangers. The focus area of the survey was mainly simulation models for the prediction of transient heat and momentum transfer in heat exchangers, and the approach used to simulate heat exchanger performance.

The literature that were found were evaluated on the basis of the following:

- the method used to represent the actual physical geometry of the heat exchanger,
- the extent to which the heat exchanger is modeled transient, i.e. which parameters is time dependent,
- the accuracy of the results obtained by the models that exists,
- the conclusions that could be drawn from the work.

Steady-state modelling of heat exchanger performance has been done extensively in the past, whilst dynamic modelling of heat exchanger performance has been done to a lesser extent.

The two main types of models found in the literature, related to the modelling of heat transfer devices like heat exchangers, are lumped models and discretised models (Section 2.2.1 and 2.2.2).

The first type of model gives good results when standard heat exchangers are used (Paynter and Takahashi, 1983; Tan and Spinner, 1984), but is too complex when a solution is required for most industrial systems (Williams and Morris, 1961; Masukuchi, 1960), since the system is rarely uncoupled from its environment, being a subsystem of a larger system.

Williams and Morris (1961) studied literature on heat exchanger dynamics and control up to 1960 and came to the conclusion that the distributed parameter model seems entirely adequate for concentric tube exchangers and the tube side of shell-and-tube heat exchangers, and that an adequate representation of shell side flow in shell-and-tube heat exchangers has not yet been obtained.

Gaddis and Schlunder (1979) have shown that an effective method for modelling a multipass heat exchanger is to break it up into several elements or cells. With this method, a number of linear algebraic equations are obtained which can be solved by using either direct or indirect methods.

Correa and Marchetti (1987) developed a model where a multipass shell-and-tube heat exchanger is divided into several elements or cells. Each cell in the model is defined as a small lumped heat exchanger. The number of baffles and the number of tube passes in the shell determines the number of these elemental parts. These cells have to be connected according to the actual heat exchanger structure and the arrangement of the inlet and outlet connections. An energy balance is set up for this arrangement, which included the following contributions:

the rate of heat flowing in across the cell surface by means of convection,
the rate of heat energy flowing out across the cell surface by means of convection,
the rate of heat flowing in (or out) across the surface (which is the heat exchange area between the shell-side and tube-side), by conduction or convection, i.e. the heat exchanged between the fluids and
the heat accumulation rate.

Another assumption that was made is that the heat transfer coefficient in every cell is constant. This is only valid for the case of low flow rates. Several first order differential equations must then be solved simultaneously to predict the transient behaviour. This was done by means of applying an iterative procedure, or solving the equivalent matrix equations. This model only predicts transient behaviour of the heat exchanger for varying inlet temperatures and flow variations and the results have shown good agreement to the results produced by Roppo and Ganic (1983).

Stainthorp and Axon (1965) described the dynamic behaviour of a multipass heat exchanger, subject to steam temperature and steam flow perturbations by the modified one pass model, which describes only flow changes in either the shell side or tube side of the heat exchanger.

Forghieri and Papa (1978) set up 3 different models of a counter flow heat exchanger with temperance disturbances and step flow variations, and obtained the asymptotic solution to step variations of the flow rate by means of the Laplace transform. The models used are based on the conventional plug-flow model, i.e. no dispersion (or backmixing (Mecklenburgh, 1975)) occurs in the flow direction and the axial velocity of process fluid is uniform.

Xuan and Roetzel (1993) applied the shell-side dispersion model to predict dynamic response to both arbitrary temperature changes and step flow variations in parallel and counter flow heat exchangers which showed good agreement between the theoretical and experimental results.

Xuan and Roetzel presented another model in 1993. This derivation involves the influence of heat capacities of both fluids, and the capacities of shell and tube walls. This model can handle two different flow arrangements. The Laplace transform is used to convert partial differential equations into ordinary differential equations, and the temperature profiles in the time domain are obtained by means of numerical inversion. The results also showed good agreement to analytical results.

Lachi *et al.* (1997) developed a two-parameter method in the case of a double-pipe heat exchanger for laminar and turbulent flows. They then extended the model for the case of a shell-and-tube heat exchanger with parallel flow, when the flow rates vary simultaneously at the two inlets. This extended method did not include the use of the equivalence method. They set up an energy balance for the complete heat exchanger, and introduced a time constant for the exit temperature. The exit temperature has an exponential shape after a time lapse. The difference in the results obtained for varying fluid flow rates for both the double-pipe and shell-and-tube heat exchanger are less than 10 percent between the theoretical and experimental set-up.

An evaluation of the above literature reveals that little research has been done on transient heat exchanger performance and that most of this research focused on heat exchangers as standalone equipment. The need therefore exists for a general method that is able to model transient heat exchanger performance as part of a larger system simulation.

2.4 Simulation of thermal systems

Thermal-fluid systems are simulated by solving the mass, momentum and energy equations. The CFD approach is used to solve the detail flow, pressure and temperature field in complex geometries. The same approach is followed in the modelling of thermal-fluid networks representing complex systems, thus the name systems CFD. These two approaches will be discussed next, to point out the differences of the two approaches.

2.4.1 CFD approach

CFD entails the solution of the differential equations for the conservation of mass (also known as the continuity equation), momentum (also known as the Navier-Stokes equation), and energy equations (also known as the first law of thermodynamics) on a per unit volume basis. Figure 2.5 shows a typical two-dimensional control volume used in the CFD approach. It is assumed that the properties such as velocity, pressure and temperature vary smoothly over the control volume and that the properties of the control volume as a whole can be represented by the average values situated at the nodal point P within the control volume.

The differential forms of the conservation equations are integrated in a discrete manner so that the velocities, pressures and temperatures of the control volume may be written in relation to that of its nearest neighbours (E, W, N, S) and the mass, momentum and energy fluxes across the boundaries of the control volume. This enables the calculation of the distribution of velocities, pressures and temperatures throughout the flow field.

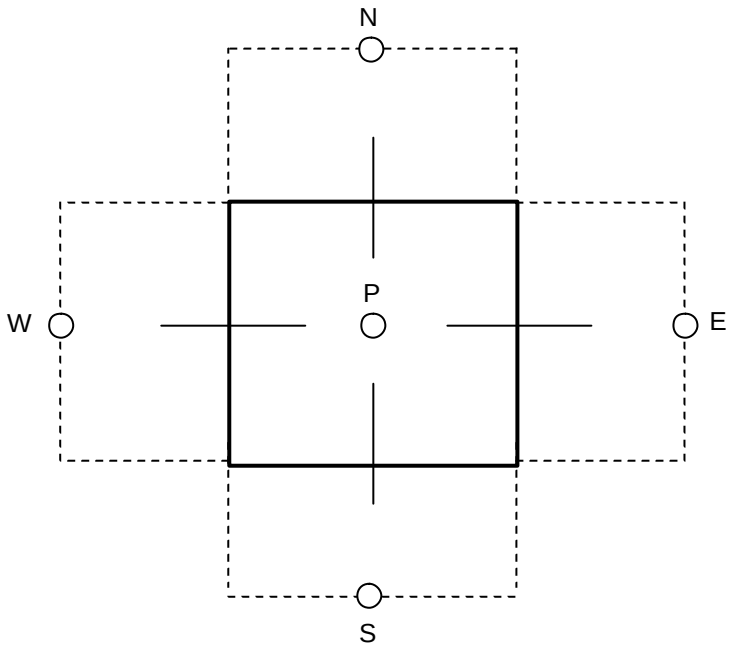


Figure 2.5: Typical control volume for a CFD approach.

It is important to note that the conservation of mass and energy is typically written for the control volume around point P, while the conservation of momentum is typically written for the flows over the boundaries at the control volume interfaces, i.e. for the dashed line control volumes. This is known as a staggered grid approach.

Even though CFD also entails the solution of the mass, momentum and energy equation, a CFD analysis usually involves a single component or a part of it. A CFD analysis requires a complex definition of the problem and is not well suited for a larger system where the designer needs to adjust and optimise certain variables to reach an optimum design. The CFD analysis would then be done in the end when the design is finalised and no more changes in the system is foreseeable.

2.4.2 Thermal-fluid network approach

The so-called network approach, on which several thermal-fluid software codes are based, makes use of a collection of one-dimensional elements connected at nodes in and unstructured manner, as shown in Figure 2.6. Circles denote the elements, while the nodes are denoted by squares. The elements can be any type of thermal-fluid component, such as a pipe, pump, fan, compressor, turbine or heat exchanger element. The nodes can either be a connection between two elements with no physical significance or it can have a volume in order to represent a reservoir or a tank.

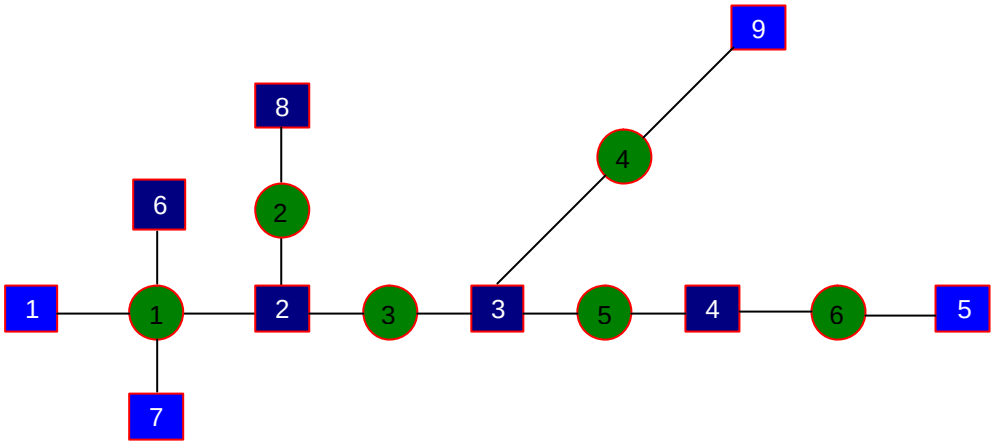


Figure 2.6: Typical node-element configuration for a network approach.

Similar to the CFD approach, it is assumed that the properties of the fluids in a node are represented by a single average value. Also similar to the CFD approach, the conservation of mass and energy are written for a node while the conservation of momentum is written for elements that serve as connections between nodes.

Although the assumption of one-dimensional flow may at first seem like a major restriction, it is important to note the any two or three dimensional flow field can also be built up with the network approach by assembling the correct combination of elements for the different coordinate directions. This is typically the approach followed in models for discretised heat exchangers and also the pebble bed reactor model (Greyvenstein *et al*., 2002). This approach is ideally suited for the simulation of large arbitrary structured thermal-fluid networks.

2.5 Solution algorithms

2.5.1 Explicit versus implicit numerical solution schemes

Numerical solution schemes are often referred to as being explicit or implicit. When a direct computation of the dependent variables can be made in terms of known quantities, the computation is said to be *explicit*. In contrast, when the dependent variables are defined by coupled sets of equations, and either a matrix or iterative technique is needed to obtain the solution, the numerical method is said to be *implicit*.

In CFD, the governing equations are non-linear and the number of unknown variables is typically very large. Under these conditions implicitly formulated equations are almost always solved using iterative techniques, even though some CFD codes utilise explicit methods.

Iterations are used to advance a solution through a sequence of steps from a starting state to a final, converged state. This is true whether the solution sought is either one step in a transient problem or a final steady-state result. In either case, the iteration steps resemble a time-like process. Of course, the iteration steps usually do not correspond to a realistic time-dependent behaviour. In fact, it is this aspect of an implicit method that makes it attractive for steady-state computations, because the number of iterations required for a solution is often much smaller than the number of time steps needed for an accurate transient that asymptotically approaches steady conditions (Flow 3D CFD Software, 1985).

On the other hand, it is also this "distorted transient" feature that leads to the question of what the consequences are of using an implicit versus an explicit solution method for a time-dependent problem. The answer to this question has two parts: one part has to do with numerical stability and the other with numerical accuracy (Flow 3D CFD Software, 1985).

The principal reason for using implicit solution methods, which are more complex to programme and require more computational effort for each time step, is to allow for large time-step sizes. A simple qualitative model will help to illustrate how this works (Flow 3D CFD Software, 1985). This model describes a point implicit method, rather than a matrix of implicit formulated equations that need to be solved simultaneously. Let P be a quantity whose value P^{n+1} we want to compute at time $t=(n+1)dt$, in terms of its value at time $t=ndt$, that is, $P^{n+1}=P^n+Sdt$, where S is the rate of change in P.

In an explicit numerical method S would be evaluated in terms of known quantities at the previous time step n. An implicit method, in contrast, would evaluate some or all of the terms in S in terms of unknown quantities at the new time step $n+1$. Since new quantities then appear on both the left and right side of the P equation, it is said to be an implicit definition of the new $n+1$ values. Usually a matrix or iterative solution must be used to compute the new quantities.

Numerical stability has to do with the behaviour of the solution as the time-step, dt, is increased. If the solution remains well behaved for arbitrarily large values of the time step, the method is said to be unconditionally stable. This situation never occurs with explicit methods, which are always conditionally stable, and limited by the Courant stability criteria (Fletcher, 1991).

A typical iterative solution for P^{n+1} is constructed by computing the $k+1$ iterate in terms of the k^{th} iterate value, where the first iterate is taken to be equal to P^n. The equation for P^{k+1} is often Newton's approximation (or similar approximation) having the form $P^{k+1}=P^k+a(P^n-P^k+S^kdt)$. In this expression

a is a relaxation factor and S^k is an approximation to S evaluated in terms of the k^{th} iterate. If a is chosen properly, successive iterates will eventually converge to P^{n+1}.

The relaxation coefficient a must have the form $a = 1/(1+Cdt)$ in order to insure the proper limits at small and large values of dt (Flow 3D CFD Software, 1985). That is, at very small time-step sizes the explicit equation is recovered, while at very large step sizes the equation has a limiting value independent of dt. The quantity C must be a positive coefficient characterising all the terms in the original equation (i.e., in S) that have been approximated implicitly. For example, if P were a velocity component governed by a momentum equation with implicit viscous terms, then C would be proportional to the kinematic viscosity divided by the square of the grid size. When dt is sufficiently small only one iteration is necessary for convergence, which leads to $P^{n+1} = P^n + dt/(1+Cdt)S^n$. This shows that the implicit formulation ads a smaller change to P in one time step than would occur in an explicit method because of the under-relaxation factor $a = 1/(1+Cdt)$ that multiplies the time step.

As a general rule, it can be shown that the condition $Cdt < 1$ is very nearly equivalent to the stability condition for an explicit approximation. Another general rule is that the time-step sizes for explicit stability and accuracy are usually equivalent. Thus, when $Cdt > 1$, an explicit method would be unstable, but implicit methods simply under-relax more to maintain the stability of the iterative solution. It is this increased damping, with the increase in time-step size, which produces inaccuracies in transient behaviour.

For an implicit method to have minimal under-relaxation (i.e. little damping), a time-step size much smaller than the stable, explicit value would have to be used. In fact, according to the above, at the explicit stability limit $Cdt = 1$the implicit approximation still has a significant under-relaxation factor of $a = 1/2$. To reduce this under-relaxation damping the time-step size would have to be much smaller than the explicit stability limit, but this makes little sense since an implicit method is not required in that case.

An elementary physical problem involving the propagation of a pressure wave can be used to illustrate the differences between implicit and explicit methods. Imagine a step increase in pressure is applied at one end of an organ pipe that is closed at the opposite end. We know that a pressure wave will move down the pipe and be reflected at the closed end. Given enough time, pressure waves will travel back and forth in the pipe many times before the pressure distribution settles down to the constant value applied at the open end. If only steady-state results are wanted, then an implicit solution scheme with lots of damping of the pressure waves should be used so that steady conditions will be reached as quickly as possible. In this case the damping incorporated in the implicit iteration method (i.e., the under-relaxation) is highly desirable. If, instead, the transient pressure waves are to be investigated, then we want the least amount of numerical damping so that many wave reflections can be accurately followed. This situation is best treated with an explicit solution method.

Explicit methods require a time-step size that limits the advance of the pressure step to less than one computational cell per time step. However, this restriction is related to accuracy because most difference equations involve quantities from neighbouring cells only. A pressure wave that propagates further than one cell in one time step would then be moving into regions that have no defined influence on the pressure. Not only is this physically unrealistic, it also leads to numerical instability.

Implicit methods, on the other hand, couple all the cells together through an iterative solution that allows pressure signals to be transmitted through a grid. The price for this communication between distantly located cells is a damping or smoothing of the pressure waves introduced by the under-relaxation needed to solve the coupled equations. The choice of whether an implicit versus explicit method should be used ultimately depends on the goal of the computation. When time accuracy is important, explicit methods produce greater accuracy with less computational effort than implicit methods.

In the following paragraphs three different explicit numerical solution algorithms will be presented, along with a comparison between them.

2.5.2 Explicit solution algorithms

2.5.2.1 Introduction

To perform an extensive evaluation of explicit methods is a study in itself. Fortunately, this has been done repeatedly in the past. Two explicit numerical methods, which are available to solve partial differential equations in fluid flow problems, namely the Method of Characteristics (MOC) and the Lax Wendroff (LW), have been described and used extensively in the past (Fletcher, 1991; Van Ravenswaay, 1997; (Sod, 1978; Liska & Wendroff, 2001) to solve systems of differential equations and will be discussed briefly in this section. The popular Runge Kutta (RK) time integration scheme that solves ordinary differential equations will also be discussed in this section.

2.5.2.2 Method Of Characteristics

The Method of Characteristics (MOC) is one of the best-known explicit finite difference methods for solving transient pipe flow problems. This method converts the partial differential equations (describing mass, momentum and energy conservation) to ordinary differential equations. The three partial differential equations are rewritten as two ordinary differential equations. These equations are only valid along characteristic lines (lines indicating propagation in time and space at the local speed of sound). When discretised on these lines, conditions at the next time step are calculated at points where these lines intersect. The solution of these total differential equations does, however, not enforce conservatism and are also first-order accurate in space and time.

This method is particularly suited for systems with complex boundary conditions, as each boundary and each pipe section are analysed separately during each time step. Unfortunately the MOC is restricted by the Courant condition, which limits the time step, and consequently makes this method time consuming and

expensive to use (Wylie and Streeter, 1993). However, the MOC, which is suited for isothermal problems, is easily implemented on a computer and this makes the method so attractive.

A detailed description of the MOC and its variants can be found in Wylie and Streeter (1993:80) and the implementation for pipe flow in Van Ravenswaay (1997).

2.5.2.3 Lax-Wendroff methods

The Lax-Wendroff Method (LW) has been a very effective and popular algorithm for solving the equations that govern inviscid, compressible flow (Fletcher, 1991). It can be shown that this family of finite difference schemes satisfies conservation of mass, momentum and energy over control volumes (Winterbone *et al*. 2000:91). Points where conservation quantities are calculated are taken to be at the centre of these control volumes. To calculate these quantities at a next time step, the flux term(s) for mass, momentum or energy conservation are integrated over time. This is done at the boundaries of each control volume. These methods differ in the way the control volume is defined, and the way the flux on the boundaries is calculated. In Winterbone *et al*. (2000:92-103), the following methods of this family are discussed.

The *single-step Lax Friedrichs scheme* is first-order accurate in space and time. This scheme is highly dissipative. For the single-step Lax-Wendroff method, fluxes are calculated using a Jacobian matrix. In a two-step version of the Lax-Wendroff method, these fluxes are calculated by solving conservation quantities on the control volume boundaries at an intermediate time step. For this method, evaluation of the Jacobian matrix is not required. The MacCormack method uses a predictor and then a corrector step. Fluxes on the boundaries are calculated using values of the predictor time step.

All three of the latter methods are second-order accurate in space and time. For all such higher order methods, oscillations occur in the vicinity of sharp gradients (discontinuities). A first attempt to resolve these oscillations was to use explicit artificial viscosity. When using artificial viscosity, a parameter that is problem specific needs to be specified. Only limited success was achieved with this solution.

Botha (2000) used the two-step LW method and derived a general scheme using the partial differential equations describing conservation of mass, momentum and energy. Two forms of the momentum equation were used to derive two variants of this scheme. The first variant uses static pressure in the momentum conservation equation, while the second variant uses total pressure. Using the general scheme, the method is then extended to handle boundary conditions as well as to incorporate non-inertial elements, junctions and tanks. These schemes were used to develop an elementary explicit pipe network simulation code using the programming language C++. The simulation code was used in several cases and compared to an implicit pressure correction method.

It is evident from the two-step LW method described by Botha that modelling the boundary conditions are tedious. When more complex networks are modelled with the LW method, this becomes a complex task.

For this reason, the LW method is not recommended for complex networks which contain several components that need to be solved with a more general solver.

2.5.2.4 4th Order Runge Kutta method

Several explicit methods similar to the MOC convert the partial differential equations to ordinary differential equations (ODE). To solve such equations, time integration schemes like the methods of Heun, Adams Bashforth or Runge Kutta are used (Kreyszig, 1993).

Since the Runge Kutta method is so easily implemented and yields very accurate results (although dispersion may occur), this method will be discussed here.

The classic Runge Kutta (RK) time integration scheme are used to solve first-order ODE of the form,

$$\frac{\partial y}{\partial x} = f(x, y) \tag{2.10}$$

The RK scheme iterates the x-values by simply adding a fixed step size of h at each iteration. The $y-$iteration is a weighted average of four values, k_1, k_2, k_3 and k_4. These four values are as follows (Kreyszig, 1993:1035):

$$k_1 = hf\left(x_n, y_n\right) \tag{2.11}$$

$$k_2 = hf\left(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right) \tag{2.12}$$

$$k_3 = hf\left(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}\right) \tag{2.13}$$

$$k_4 = hf\left(x_n + h, y_n + k_3\right) \tag{2.14}$$

The $y-$iteration is then given by

$$y_{n+1} = y_n + \frac{1}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right) \tag{2.15}$$

This equation is then used to solve all the ODE in the problem. When this is done, the solution is repeated for the next time step if required. This method can then be used to solve a system of ODE resulting from the application of the conservation laws to the nodes (mass and energy) and elements (momentum).

2.5.2.5 Choice of numerical method

Even though there are several different solution algorithms to choose from, some methods are better suited than others for certain applications. The parameters which influence the choice of method to be used will

not only be the accuracy, but the stability of the method as well as the speed in which a solution will be acquired. The dispersion which explicit numerical methods exhibit is also crucial in the choice. These aspects will be discussed briefly below.

Accuracy

The accuracy of a numerical solution is defined as the degree of conformity of numerically calculated value to its actual value. The LW method is of second-order accuracy in space and time, whilst the MOC is of third order. The 4^{th}-order RK method is of fourth-order accuracy in space and time, as indicated in its name. It should be noted that the accuracy of the method means nothing unless the method is stable.

Stability

Explicit methods such as the MOC and LW method are all limited by the same stability criterion, namely the famous Courant-Friedrichs-Lewy stability criterion (Fletcher, 1991), often simply called the *Courant condition*. In order for these methods to be stable, it is necessary that

$$c\Delta t \leq \Delta x \tag{2.16}$$

where c is the velocity of sound, and x the length of the grid spacing. As a result of this condition, the time step, Δt, must be kept very small to ensure stability in these methods.

Speed

The speed of explicit numerical methods depends on the time step, which is limited by the Courant condition. Since the MOC and LW method are both limited by the time step, the total computational time involved in the solution algorithm will determine the overall speed of the solution.

Other issues

There are several other issues that make some of the methods unattractive to use. The LW method smears out discontinuities and produces oscillations (dispersion) around discontinuities (Fletcher, 1991). Other explicit methods also have the tendency to produce dispersion around discontinuities. This phenomenon is beyond the scope of this thesis, and will not be discussed here. The MOC is also not suited for heat transfer problems, since it is only valid for isothermal problems (Van Ravenswaay, 1997).

2.6 Existing commercial codes

A brief survey of commercial thermal-fluid software codes shows that there are a wide variety of commercial software codes available. Botha (2000) did an extensive survey on the availability of commercial thermal-fluid codes, and generally considered codes that are able to simulate transient compressible flow. Botha sent a questionnaire to 35 companies which market commercial software. Less than a third replied, and some information was found on there websites.

Botha concentrated mainly on the following aspects, which are applicable to this thesis:

The method used to represent the network and solve the flow / heat transfer (for example finite difference or finite volume, explicit (MOC) or implicit Euler, LaGrange),
General capabilities (or models) (whether it can simulate two phase flow, transient flow, heat transfer, isothermal flow, real time, non-Newtonian fluids, pipe elasticity, flow in branches, gravitational effects, etc.),
Specific models and capabilities (Can it model; compressors, turbines, heat exchangers, controllers, user defined models?),
Whether heat exchangers are modelled as lumped or distributed systems. (In a distributed approach the heat exchanger is subdivided, is thermal capacitance included, and are temperature and heat transfer solved for the fluid and the metal in each subdivision, for instance),
Whether boundary conditions and model parameters can be changed during a transient simulation,
Applications (for example: simulation, design, plant training, simulator etc.).

From the ten companies that responded to the questionnaire, only five are able to simulate transient behaviour of fluid flow. Only four companies software model heat exchangers as a distributed system. Only two codes could model transient heat exchanger performance as a distributed system and only one company on the list could model transient heat exchanger performance (as a distributed system) as part of a larger simulation system. The code they used was Sinda/Fluint, but Flownex can also be used for this. These two codes will be discussed briefly in the next two sections.

2.6.1 Sinda/Fluint

Sinda/Fluint is a comprehensive, generalised tool for simulating complex thermal-fluid systems. Sinda/Fluint is capable of performing steady-state and transient analysis on single components as well as large systems. Sinda/Fluint simulates heat exchangers as lumped or distributed systems.

2.6.2 Flownex

Flownex is a thermal-fluid network analysis code that performs detail analyses and designs of complex thermal-fluid systems. Flownex uses nodes and elements to represent a thermal-fluid network. Elements are components such as pipes, valves, compressors or heat exchangers, while nodes are the end points of elements. Elements can be connected in any arbitrary way at common nodes to form a network. Flownex solves the momentum equation in each element and the continuity and energy equation at each node. This gives Flownex a pseudo CFD capability, which allows it to predict complex phenomena such as pressure and temperature waves in pipes and buoyancy effects in packed beds. Flownex also has the ability to perform both steady-state and dynamic analysis and predicts both fast and slow transients fast and accurately with its robust implicit solver. Flownex also enables the user to use standard composite models built into the code or to build advanced discretised models of complex components such as heat exchangers and gas-cooled nuclear reactors with its versatile model builder.

2.7 Summary

In this chapter the main approaches used to simulate heat exchangers were briefly discussed, and a survey of previous work done on heat exchangers was given. Two network approaches were presented and the

differences between them explained. The difference between explicit and implicit solution methods was discussed and three numerical methods were discussed and compared. Lastly a review of existing software codes was given.

In Chapter 3, thermal-fluid network codes will be discussed and an explicit method will be presented, which will serve as basis for a thermal-fluid network solver.

CHAPTER 3 - THE IMPLICIT PRESSURE CORRECTION METHOD AND THE RUNGE KUTTA METHOD WITH TRAPEZOIDAL DAMPING

3.1 Introduction

In this chapter two methods to solve thermal-fluid networks, namely the *Implicit Pressure Correction Method (IPCM)* and the *Runge Kutta with Trapezoidal Damping method (RKTD)* will be discussed in sufficient detail to show the reader the difference between the two methods.

To do this, the governing equations that describe the transient behaviour of fluid flow in arbitrary systems will be presented in integral form, as well as in a one-dimensional differential form. The numerical discretisation schemes used in thermal-fluid networks will be discussed briefly to give the necessary background when the governing equations are applied to unstructured networks in the IPCM and RKTD method.

The dynamic thermal-fluid network analysis code Flownex will be discussed briefly, as it makes use of the IPCM solution algorithm. This code will also be used to investigate the transient behaviour of heat exchangers using a network approach.

The main focus of this study will be an in-depth discussion on the application of the governing equations on unstructured networks found in thermal-fluid networks, which are solved with the IPCM and RKTD method. The explicit RKTD method will form the basis for a generalised thermal-fluid network solver Xnet that will be discussed and compared to the thermal-fluid network code Flownex in the next chapter. This code, Xnet, will also be used to investigate the transient behaviour of heat exchangers using a network approach and in turn be used to validate Flownex.

3.2 Governing equations

The equations that govern fluid flow and heat transfer in thermal-fluid systems are the equations that describe the conservation of mass, momentum and energy. These three equations will be discussed in the paragraphs to follow. In all the cases, the equations will first be presented in a general integral form for a control volume, whereafter the differential form applicable to a one-dimensional pipe or flow conduit will be given.

3.2.1 Conservation of mass

Application of the law of conservation of mass to a control volume leads to:

$$\frac{Dm}{Dt} = 0 = \oiint_{CS} (r\overline{V} \cdot d\overline{A}) + \frac{\partial}{\partial t} \iiint_{CV} r\, dv \tag{3.1}$$

where

CS is the control surface,

CV the control volume,

m the mass,

t the time,

r the density,

$\overline{V}$ the velocity,

$\overline{A}$ the area and

v the volume.

Equation (3.1) may then be rewritten as:

$$\oiint_{CS} (r\overline{V} \cdot d\overline{A}) = -\frac{\partial}{\partial t} \iiint_{CV} r\, dv \tag{3.2}$$

This equation shows that the net efflux rate of mass through the control surface equals the rate of decrease of mass inside the control volume. The differential form of Equation (3.2), for a variable area pipe element, is given by Shames (1992:29):

$$\frac{\partial r}{\partial t} = -\frac{1}{A} \frac{\partial (rVA)}{\partial x} \tag{3.3}$$

where t is the time, r the density, V the velocity, x the length dimension in the direction of the flow, $\dot{m}$ the mass flow, and A the flow area.

3.2.2 Conservation of momentum

The equation for the conservation of linear momentum is given by:

$$\frac{D\overline{P}}{Dt} = -\oiint_{CS} \overline{T} d\overline{A} - \frac{\partial}{\partial t} \iiint_{CV} \overline{B} r\, dv + \oiint_{CS} \overline{V} (r\overline{V} \cdot d\overline{A}) + \frac{\partial}{\partial t} \iiint_{CV} \overline{V} (r\, dv) \tag{3.4}$$

where $\overline{P}$ is the linear momentum, $\overline{T}$ the surface force distribution on the control surface, and $\overline{B}$ the body force distribution on the control volume. Equation (3.4) calculates the sum of these force distributions with the rate of efflux of linear momentum across the control surface plus the rate of increase of linear momentum inside the control volume (Shames, 1992:155). It can be shown (Shames, 1992) that Equation (3.4) may be rewritten in the following differential form:

$$\frac{\partial(rV)}{\partial t}+\frac{1}{A}\frac{\partial\left(rV^2A\right)}{\partial x}+\frac{\partial p}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{frV|V|}{2D_H}=0 \tag{3.5}$$

where p is the static pressure, g the gravitational acceleration, z the elevation, f the friction factor and D_H the hydraulic diameter. Equation (3.5) may be rewritten in the following form:

$$V\left(\frac{\partial p}{\partial t}+\frac{\partial(rVA)}{\partial x}\right)+r\frac{\partial V}{\partial t}+rV\frac{\partial V}{\partial x}+\frac{\partial p}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{frV|V|}{2D_H}=0 \tag{3.6}$$

Substitution of Equation (3.3) into Equation (3.6) leads to:

$$r\frac{\partial V}{\partial t}+rV\frac{\partial V}{\partial x}+\frac{\partial p}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{frV|V|}{2D_H}=0 \tag{3.7}$$

Equation (3.7) may be further manipulated into the following form:

$$r\frac{\partial V}{\partial t}+r\frac{\partial\left(\frac{1}{2}V^2\right)}{\partial x}+\frac{\partial p}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{frV|V|}{2D_H}=0 \tag{3.8}$$

For incompressible flow, Equation (3.8) may be written as:

$$r\frac{\partial V}{\partial t}+\frac{\partial\left(p+\frac{1}{2}rV^2\right)}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{frV|V|}{2D_H}=0 \tag{3.9}$$

Since the total pressure, p_0, for incompressible flows is defined by:

$$p_0=p+\tfrac{1}{2}rV^2 \tag{3.10}$$

Equation (3.9) may be written in the following final form:

$$r\frac{\partial V}{\partial t}+\frac{\partial p_0}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{f\,rV|V|}{2D_H}=0 \tag{3.11}$$

For compressible flow, Equation (3.7) is handled somewhat differently. It can be shown from principles of gas dynamics (Greyvenstein, 2002) that:

$$rV\frac{\partial V}{\partial x}+\frac{\partial p}{\partial x}=\frac{p}{p_0}\frac{\partial p_0}{\partial x}+\frac{rV^2}{2T_0}\frac{\partial T_0}{\partial x} \tag{3.12}$$

where p_0 is the total pressure and T_0 the total temperature. For compressible flow the total pressure is defined as:

$$p_0=p\left(1+\frac{g-1}{2}M^2\right)^{\frac{g}{g-1}} \tag{3.13}$$

where M is the Mach number and g the ratio of the specific heats for the gas under consideration. The total temperature is defined in terms of the static temperature T as:

$$T_0=T\left(1+\frac{g-1}{2}M^2\right) \tag{3.14}$$

Substitution of Equation (3.12) in Equation (3.7) leads to the following form of the compressible momentum equation:

$$r\frac{\partial V}{\partial t}+\frac{p}{p_0}\frac{\partial p_0}{\partial x}+rg\frac{\partial z}{\partial x}+\frac{\partial T_0}{\partial x}+\frac{\tilde{f}\,rV|V|}{2D_H}=0 \tag{3.15}$$

where $\tilde{f}$ is the effective friction factor which is given by:

$$\tilde{f}=f+\frac{D_H}{T_0}\frac{\partial T_0}{\partial x}\frac{V}{|V|} \tag{3.16}$$

The second term on the right of Equation (3.16) is usually much smaller than f and in the case of adiabatic flow it will be exactly zero. However, in the case of combusting flows, or flow through rotating blade passages, the term may become significant and will therefore be retained.

3.2.3 Conservation of energy

Application of the law of conservation of energy to a control volume leads to:

$$\frac{DE}{Dt} = \oiint_{CS}(e + rv)(r\overline{V}\cdot d\overline{A}) + \frac{\partial}{\partial t}\iiint_{CV}(e)(r\,dv) \tag{3.17}$$

where

$$\frac{DE}{Dt} = \frac{dQ}{dt} - \frac{dW_N}{dt} \tag{3.18}$$

and

E is the energy,

Q the net heat added to system,

W_N the net work done and

e the energy per unit mass.

The energy per unit mass is defined as the kinetic energy plus potential energy plus internal energy and is given by:

$$e = \frac{V^2}{2} + gz + u \tag{3.19}$$

where

V is the velocity,

g the gravitational acceleration,

z the elevation above some datum and

u the internal energy.

Equation (3.17) states that the net rate of energy transferred into the control volume by heat and work equals the rate of efflux of stored energy from the control volume plus the rate of the increase of stored energy per unit mass (Shames, 1992:226). Equation (3.17) could be rewritten in the following differential form (Shames, 1992):

$$\frac{\partial(ru_0)}{\partial t} + \frac{\partial\{(h_0 + gz)rV\}}{\partial x} - q = 0 \tag{3.20}$$

where h_0 is defined as $h + V^2/2$, and u_0 is defined as $u + V^2/2$.

In the case of ideal gasses the total enthalpy can be written in terms of the total temperature (Sonntag *et al*, 1998) as $h_0 = c_p T_0$, with c_p the specific heat at constant pressure. Therefore, Equation (3.20) can be written as:

$$\frac{\partial\left(r c_p T_0 - p\right)}{\partial t} + \frac{\partial\left\{\left(c_p T_0 + gz\right) rV\right\}}{\partial x} - q = 0 \qquad (3.21)$$

3.2.4 Equation of state

In the case of compressible flow an equation is also needed that expresses the pressure in terms of density and temperature, and which takes the compressibility in consideration. Such an equation is the equation of state, which is given by:

$$p = s\, r R T \qquad (3.22)$$

where

p is the static pressure,

R the gas constant,

s the compressibility factor and

T the static temperature.

The behaviour of many gases, such as air, oxygen and helium very closely approximates the ideal gas under most conditions and may, with good accuracy, be presented by the above equation of state (Shames, 1992:19) with $s = 1$.

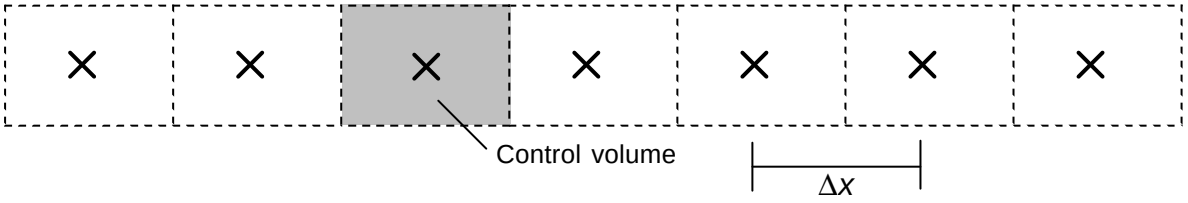
The equations governing incompressible flow are therefore given by Equations (3.3), (3.11), (3.21)and (3.22) , while the equations governing compressible flow are (3.3), (3.15) and (3.21). Although the equations presented in this section are valid for both liquid and gas flows, for the sake of brevity, only the compressible variant of the method will be discussed in the sections to follow. These equations are applied to each component (element) in the thermal-fluid network and thus form a set of equations that needs to be solved through an implicit (iterative) or explicit solution algorithm. To obtain such a set, the governing equations described above must first be discretised. The different discretisation schemes will be discussed in the next section.

3.3 Discretisation schemes

In the previous section, the one-dimensional governing equations for compressible and incompressible flow were presented. In order to solve a system of these equations, it is necessary to decide which numerical scheme to use and in turn to decide on the best-suited discretisation scheme for the chosen numerical method. The numerical grid defines the discrete locations at which the variables are to be calculated and is essentially a discrete representation of the geometric domain on which the problem is to be solved. It divides the solution domain into a finite number of subdomains, such as elements and control volumes. The two most common and widely used discretisation schemes are the co-located and staggered grid discretisation schemes (Ferziger and Peric, 2002). These two schemes will be discussed in the next section.

3.3.1 Co-located versus staggered grid discretisation schemes

The obvious choice is to define all the variables at the same set of grid points and to use the same control volumes for all variables. This type of grid is called a co-located grid and is shown in Figure 3.1. Since many of the terms in each of the equations are essentially identical, the number of coefficients that must be computed and stored are minimised and thus simplifying the programming.



× Grid points where velocity, V, pressure, p, temperature, T, and density, r, are defined

Figure 3.1: Schematic representation of the one-dimensional co-located grid.

The co-located arrangement also has significant advantages in complicated solution domains, especially when the boundaries have slope discontinuities or the boundary conditions are discontinuous. Other arrangements of the variables lead to some of the variables being located at singularities in the grid, which may lead to singularities in the discretised equations. The difficulty with pressure-velocity coupling and the occurrence of oscillations in pressure makes this method less favourable, unless improved pressure-velocity coupling algorithms are used.

There is no need for all the variables to be defined at the same grid point, and it is often more advantageous to use a different arrangement. In Cartesian coordinates, the staggered arrangement introduced by Harlow and Welsh (1965), shown in Figure 3.2, offers several advantages over the co-located arrangement. Several terms that require interpolation with the co-located arrangement, can be calculated without interpolation (Ferziger and Peric, 2002). Perhaps the biggest advantage of the staggered arrangement is the strong coupling between the velocities and the pressure. This helps to avoid some types of convergence problems and unwanted oscillations.

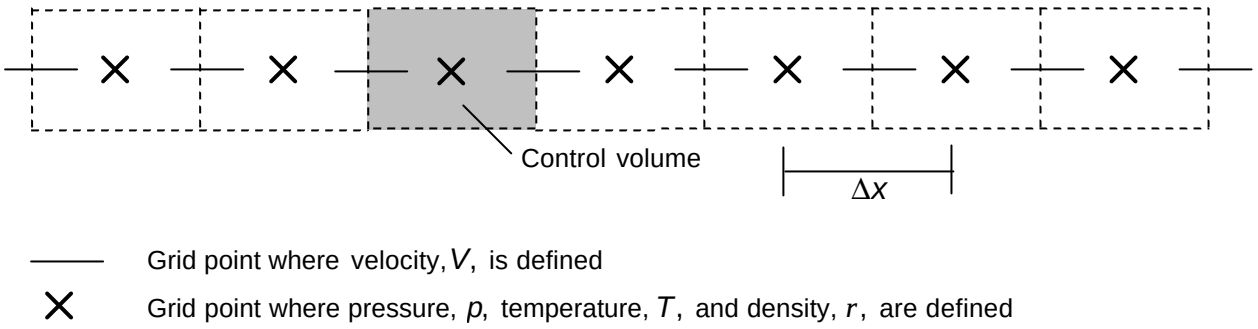


Figure 3.2: Schematic representation of the one-dimensional staggered grid.

3.3.2 Staggered grid and wall nodes

A wall node could be defined as any node that represents a solid mass, such as a pipe wall. The element connecting two solid nodes or a solid node and flow node, do not represent mass transfer (velocity), but rather heat transfer. An element between two solid nodes represents conductive heat transfer, while an element between a solid node and a flow node represents convective heat transfer. For this reason it is not necessary to include the wall nodes in the control volume of the flow grid. A simple flow network with convection elements as the heat transfer links to the wall nodes are shown in Figure 3.3.

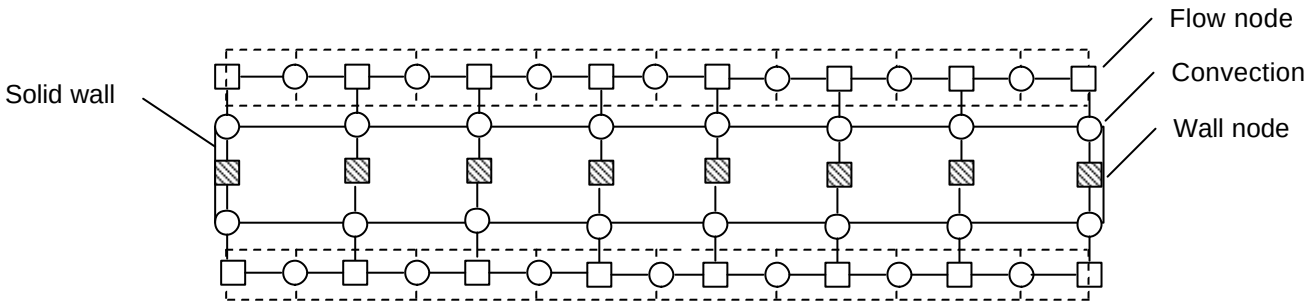


Figure 3.3: Schematic representation of a network with wall nodes.

3.4 The Implicit Pressure Correction Method (IPCM)

The IPCM is used by Flownex to solve the set of discretised equations for any arbitrary structured thermal-fluid network. The computational grid and the solution algorithm used by the IPCM to solve the discretised governing equations will be discussed in the following sections. It should be noted that the difference between the IPCM and the RKTD method which will be presented later, are not in the way in which the equations are discretised, but rather in the solution algorithm used by the methods.

3.4.1 Computational grid

The network representation of the staggered grid uses elements and nodes to represent the network. The flow elements are control volumes for momentum, whilst the nodes are the centre points of control volumes for mass and energy. The IPCM uses a staggered grid discretisation scheme for reasons pointed out in section 3.3.1. A simple flow network without junctions (for the sake of simplicity) is represented by Figure 3.4. Elements within the flow network are represented as circles and nodes as squares. The dotted lines represent the control volume boundaries. The index, i, identifies the control volume, and therefore the node, which is located at the centre of the control volume.

Pressure, temperature and density are defined at the flow nodes, which represent the centre of the control volume, while velocity and volumetric flow rates are defined at the centre of the flow elements, which in turn represent the faces of the control volume. Where a boundary node exists, which is defined as any node that is associated with only one flow element, there is half a control volume. These nodes and other nodes within the network, may have fixed boundary conditions.

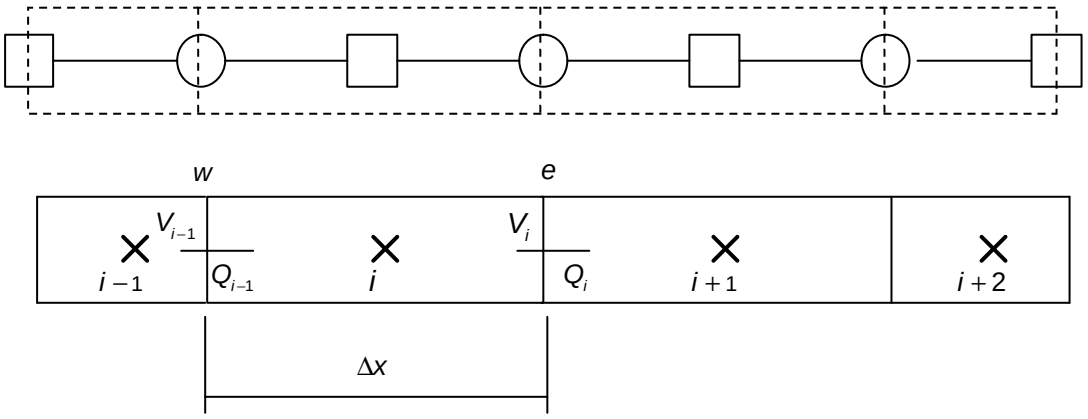


Figure 3.4: Schematic representation of a typical flow network together with the finite volume discretisation of the network.

3.4.2 Discretisation of governing equations for the IPCM

Finite volume discretisation, as shown in Figure 3.4, is used to transform the partial differential equations into a set of algebraic equations. Since the pressure, temperature and density are defined at the flow nodes and the velocity and volumetric flow rates at the centre of the flow elements, it is necessary to discretise the continuity and energy governing equation for a node and the momentum governing equation for an element.

Integration of the continuity equation over control volume i leads to:

$$\forall_i \frac{(r_i - r_i^o)}{\Delta t} + a\left(r_e Q_i - r_w Q_{i-1}\right) + (1-a)\left(r_e Q_i - r_w Q_{i-1}\right)^o = 0 \qquad (3.23)$$

where Q is the element's volumetric flow rate and $\forall$ is the volume of the control volume. The superscript o refers to the previous time level, t, while values or terms without any superscript are defined at the new time level $(t + \Delta t)$. a is a weighing factor between the previous and the present time levels and its value can range between 0 and 1. With $a = 0$ the scheme becomes fully explicit and with $a = 1$ the scheme becomes fully implicit. If $a = 0.50$ the time integration is equivalent to that of the Crank-Nicholson method (Roach, 1972). Subscripts e and w refer to values to the right and left at the control volume boundaries as shown Figure 3.4.

For more complex networks where more than two branch elements connect at the same node, as depicted in Figure 3.5 on the next page, Equation (3.23) may be expressed as

$$\forall_i \frac{(r_i - r_i^o)}{\Delta t} + a\left(\sum_{j=1}^{J} r_j Q_j s_j + d_i\right) + (1-a)\left(\sum_{j=1}^{J} r_j Q_j s_j + d_i\right)^o = 0 \qquad (3.24)$$

where s_j indicates the flow direction of element j with $s_j = 1$ if the positive flow direction of element j is from node i to node j and $s_j = -1$ if the positive flow direction of element j is from node j to node i.

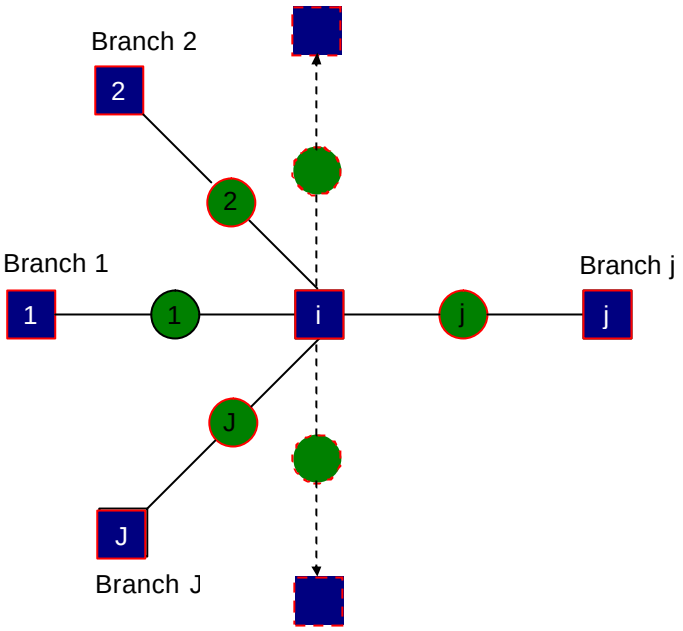


Figure 3.5: General node with neighbouring nodes connected through branch elements.

Integration of the momentum equation, Equation (3.15) over a control volume centred at interface e leads to:

$$\frac{r_e \Delta x}{A_e}\frac{(Q_i - Q_i^o)}{\Delta t} + a\left(\frac{p_e}{p_{0_e}}(p_{0_{i+1}} - p_{0_i}) + \frac{\tilde{f}\,\Delta x\, r_e Q_i |Q_i|}{2DA_e^2}\right) + (1-a)\left(\frac{p_e}{p_{0_e}}(p_{0_{i+1}} - p_{0_i}) + \frac{\tilde{f}\,\Delta x\, r_e Q_i |Q_i|}{2D_H A_e^2}\right)^o = 0 \quad (3.25)$$

where A_e is the cross sectional area at control volume (cell) face e.

Integration of the energy equation, Equation (3.21) (using total enthalpy) over the control volume i in Figure 3.4 leads to:

$$\forall_i\left(\frac{r_i h_{0_i} - r_i^o h_i^o}{\Delta t}\right) - \forall_i\left(\frac{p_i - p_i^o}{\Delta t}\right) + a(r_e Q_i h_{0_e} - r_w Q_{i-1} h_{0_w}) + (1-a)(r_e Q_i h_{0_e} - r_w Q_{i-1} h_{0_w})^o = \forall_i \dot{q} \quad (3.26)$$

If more than one branch exists as depicted in Figure 3.5, Equation (3.26) may be written as:

$$\forall_i\left(\frac{r_i h_{0_i} - r_i^o h_i^o}{\Delta t}\right) - \forall_i\left(\frac{p_i - p_i^o}{\Delta t}\right) + a\left(\sum_{j=1}^{J} s_j r_j Q_j h_{0_j}\right) + (1-a)\left(\sum_{j=1}^{J} s_j r_j Q_j h_{0_j}\right)^o = \forall_i \dot{q} \qquad (3.27)$$

Equations (3.24), (3.25) and (3.27) need to be solved simultaneously for all the unknowns p_{0_i}, Q_i, h_{0_i} and r_i at the new time level, given their values at the previous time level and at the boundaries. This implicit set of algebraic equations is solved using the IPCM. This method will be described briefly in the next section. A detailed description may be found in Greyvenstein (2002).

3.4.3 Solution algorithm

Flownex uses the IPCM solution algorithm to solve the discretised one-dimensional governing equations presented above. This solution algorithm provides a fast and accurate solution to the governing equations. The steps in the solution procedure are listed in Figure 3.6.

Flownex assigns initial pressures to nodes that do not have a fixed pressure. It then calculates the mass flow in each element using the known mass flow – pressure drop ($\dot{m} - \Delta p$) relation for each flow element in the network. After calculating the mass flows, Flownex tests for continuity at each node in the network. If continuity is not satisfied at all nodes, Flownex adjusts the pressures at the nodes to achieve continuity and in turn recalculates the mass flows for the elements. This procedure is repeated done until convergence, and then only is the energy equation solved. When the energy equation is solved, Flownex adjusts the pressure and repeats the process described until convergence is achieved. If it is a transient solution, Flownex advances to the next time step and repeats the whole process. Since Flownex employs a segregated solution algorithm where the governing equations and additional equations are solved sequentially, it is possible to control the solution procedure through various relaxation parameters (Greyvenstein, 2002).

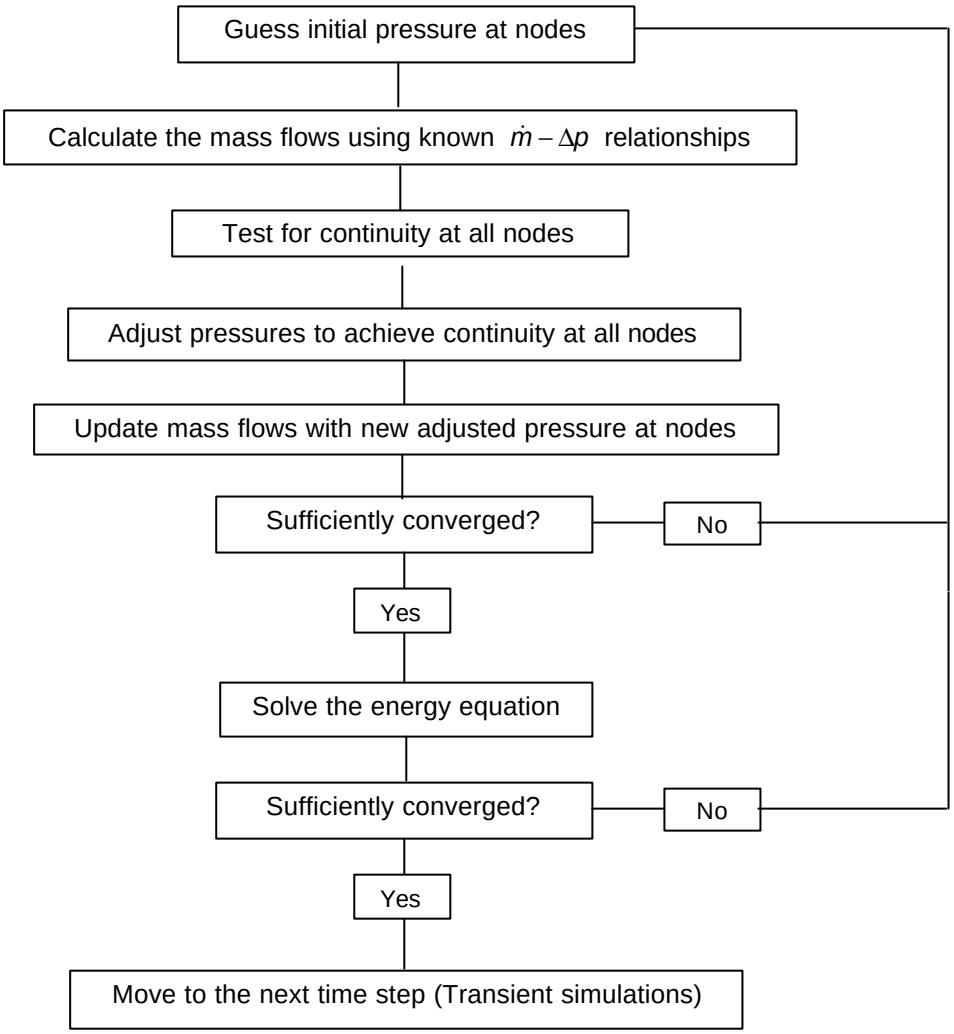


Figure 3.6: Solution procedure flow diagram for the thermal-fluid network code Flownex.

3.5 Runge Kutta with Trapezoidal Damping (RKTD)

In Chapter 2, different numerical solution algorithms were discussed as well as the difference between implicit and explicit methods. In section 2.5.1 the advantages and disadvantages for using an explicit method to solve systems of one-dimensional equations were given. In this section a staggered grid explicit method that uses the RK time integration scheme with trapezoidal damping for solving the one-dimensional governing equations will be given and discussed. It should be noted that the RK scheme solves ordinary differential equations. These equations are obtained by discretising the spatial derivatives of the governing equations using a finite volume discretisation scheme as shown in Figure 3.4. The computational grid used by the RKTD method is the same as the computational grid used by the IPCM and is discussed in Section 3.4.1.

3.5.1 Discretisation of governing equations (RKTD)

Finite volume discretisation, as shown in Figure 3.4, is used to transform the partial differential equations into a set of ordinary differential equations. Pressures, densities and temperatures are defined at control

volume centres while velocities and volumetric flow rates are defined at control volume boundaries. It is therefore necessary to discretise the continuity and energy governing equation for a node and the momentum governing equation for an element.

Integration of the continuity equation over control volume i leads to:

$$\forall_i\left(\frac{d\,r_i}{dt}\right)+\left(r_eQ_i-r_wQ_{i-1}\right)=0 \tag{3.28}$$

with

$$\left[\begin{array}{l} r_w=0.5\left(r_{i-1}+r_i\right) \\ r_e=0.5\left(r_i+r_{i+1}\right) \end{array}\right]^t$$

The subscripts e and w refer to values to the right and left at the control volume boundaries as shown in Figure 3.4. For more complex networks, where there exist more than one branch element at one node, as depicted in Figure 3.5, Equation (3.28) may be expressed as:

$$\forall_i\left(\frac{d\,r_i}{dt}\right)+\sum_{j=1}^{J} r_jQ_js_j+d_i=0 \tag{3.29}$$

where s_j indicates the flow direction of element j with $s_j=1$ if the positive flow direction of element j is from node i to node j and $s_j=-1$ if the positive flow direction of element j is from node j to node i.

Integration of the momentum equation, Equation (3.15), over a control volume centred at interface e leads to:

$$\frac{r_e\Delta x}{A_e}\left(\frac{dQ_i}{dt}\right)+\left(\frac{p_e}{p_{0_e}}(p_{0_{i+1}}-p_{0_i})+\frac{\tilde{f}\,\Delta x\,r_eQ_i\left|Q_i\right|}{2D_HA_e^2}\right)=0 \tag{3.30}$$

where A_e is the cross-sectional area at control volume (cell) face e.

Integration of the energy equation, Equation (3.21) (using total enthalpy) over the control volume i in Figure 3.4 leads to:

$$\forall_i\left(\frac{d\left(r_ih_{0_i}\right)}{dt}\right)-\forall_i\left(\frac{dp_i}{dt}\right)+\left(r_eQ_ih_{0_e}-r_wQ_{i-1}h_{0_w}\right)=\forall_i\dot{q} \tag{3.31}$$

For more complex networks where more than two branch elements connect at the same node, as depicted in Figure 3.5, Equation (3.26) may be written as

$$\forall_i\left(\frac{d\left(r_i h_{0_i}\right)}{dt}\right)-\forall_i\left(\frac{dp_i}{dt}\right)+\left(\sum_{j=1}^{J} s_j r_j Q_j h_{0_j}\right)=\forall_i \dot{q} \tag{3.32}$$

with

$$\begin{bmatrix} h_{0w}=0.5\left(h_{0i-1}+h_{0i}\right) \\ h_{0e}=0.5\left(h_{0i}+h_{0i+1}\right) \end{bmatrix}^t$$

Equations (3.28), (3.30) and (3.32) form a set of ordinary differential equations that can be solved with well-known explicit methods such as the RK method or the Euler method. In Xnet, these equations are solved with the RK time-integration scheme, which will be discussed briefly in the following section.

3.5.2 Integration scheme

The explicit method presented here makes use of the 4th-order RK integration scheme to solve the one-dimensional differential equations. Explicit numerical schemes, such as the RK scheme used for this method, exhibits dispersion, which are oscillations associated with rapid changes in the solution. The dispersion exhibited by explicit schemes is beyond the scope of this study and will not be discussed in detail here. Several authors have used trapezoidal damping in the past to reduce the effect of dispersion in explicit numerical schemes. Alvarado *et al.* (1983) used trapezoidal integration with damping to reduce the effect of dispersion in the solution of power transients. McFarland (1997) used the Bashforth 2nd-order integration (explicit) algorithm, followed by the trapezoidal integration algorithm (implicit) to model a closed-loop feedback control system.

To reduce the effect of dispersion "*Trapezoidal damping*" is used in Xnet. The technique of trapezoidal damping is implemented into explicit scheme as part of the RK integration scheme and is performed after the last RK step (See section 2.5.2.4). In the last step of the RK method, the variable, y^{n+1}, and $\left(dy/dt\right)^{n+1}$ is calculated. y^{n+1} is then recalculated with the following semi-implicit equation:

$$y^{n+1}=y^n+\Delta t\left[w\frac{dy^{n+1}}{dt}+\left(1-w\right)\frac{dy^n}{dt}\right] \tag{3.33}$$

where w is the trapezoidal damping factor. This damping factor is approximately 0.3. By using this technique, the effect of dispersion is greatly reduced. To visually illustrate the effect of trapezoidal damping, Van Der Merwe (2003) applied the RK method to a simple pipe model with two transient inputs. The effect of damping, on the outlet pressure of the pipe, is shown in Figure 3.7. The magenta line shows the RK

solution with damping and the blue line the RK solution without damping. It can be clearly seen that the RK solution with trapezoidal damping quickly settles on the solution.

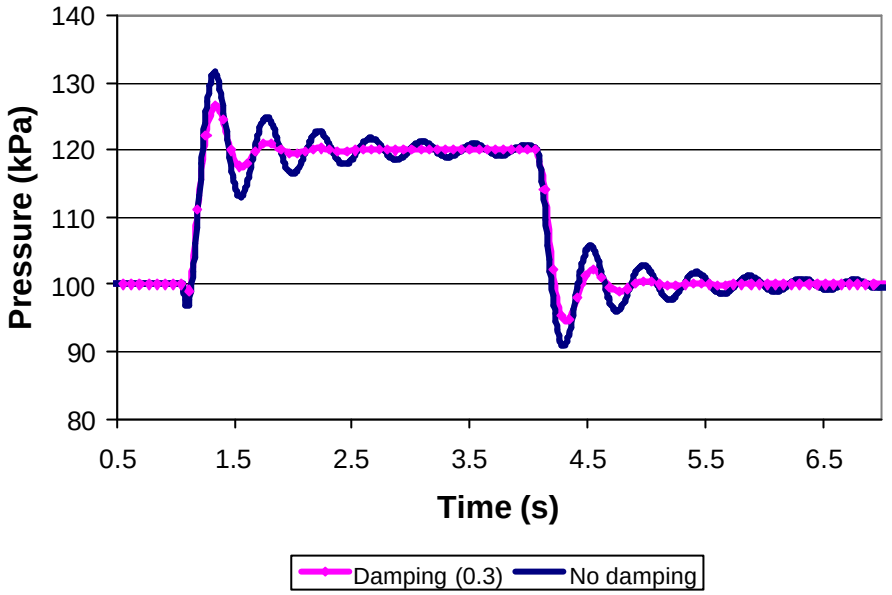


Figure 3.7: The effect of numerical damping on the dispersion posed by the RK scheme.

3.6 Summary

In this chapter two solution algorithms for the system of governing equations applicable to thermal-fluid networks were discussed, namely the IPCM and the RKTD methods. The one-dimensional governing equations that describe fluid flow and heat transfer in thermal-fluid networks and two common discretisation schemes used for the discretisation of the governing equations were presented. The solution algorithms employed by the two methods were also discussed. A technique known as trapezoidal damping was introduced to reduce the effect of dispersion present in the RK integration scheme used with the explicit method (RKTD). In the next chapter a newly developed thermal-fluid network code, Xnet, will be discussed. This thermal-fluid network code uses the RKTD method as basis for its solver. Flownex will also be compared to Xnet as part of the validation of Xnet's flow and heat transfer elements.

CHAPTER 4 - THERMAL-FLUID NETWORK SOLVER – XNET

4.1 Introduction

In Chapter 3 a basic explicit method with trapezoidal damping was discussed, namely the RKTD method. This method was developed and extended into a generalised network solver by the Faculty Of Engineering at the Potchefstroom University.

The objective of this chapter is to compare Xnet's results for some basic flow and heat exchange models to the results given by Flownex. Flownex's capability to solve pipe flow and heat exchange has been extensively validated in the past (Greyvenstein, 2001; Greyvenstein *et al.*, 2002). Van Der Merwe (2003) modelled a High Temperature Test Reactor (HTTR) using a discretised network model in Flownex and Xnet. The models in Flownex and Xnet consisted of a complex network model built with so-called primitive flow and heat transfer elements. Van der Merwe's model was used to investigate fluid flow and heat transfer within the reactor. This model would then also serve as a benchmark model in the validation process of the composite HTTR element in Flownex as well as a general benchmark of Flownex. The results produced by Van der Merwe's study compared well with that of the composite HTTR model in Flownex, and therefore shows the validity of the code for thermal-fluid problems. Van Der Merwe's study, however, was limited to cases that were applicable to his study, and the need therefore exists for an investigation into the validity of the applicable flow and heat transfer elements for use in general thermal-fluid network problems. Some basic models will be presented to compare the flow equations and the basic heat transfer equations used in Xnet to that of Flownex. The chapters to follow will deal with steady-state and transient heat exchanger performance analysis.

In this chapter a general description of the generalised network solver, Xnet, will be given, as well as a description of the applicable elements available in the solver for this study. A comparative study will then be presented to show how Xnet's fluid flow and heat transfer results for certain cases compare to Flownex. The chapter will be concluded with a summary.

4.2 Thermal fluid network code – Xnet

Xnet is a generalised thermal-fluid network code, which is capable of solving complex thermal networks. Xnet is capable of solving steady-state and transient heat and momentum transfer in complex networks. Xnet uses the RKTD method presented in Chapter 3 to solve the system of one-dimensional governing equations. The solution procedure used in Xnet to calculate and solve the element properties within a network was discussed briefly in Chapter 3. A more in-depth discussion of the solution procedure in Xnet, including the variable update routines, is beyond the scope of this study, but may be found in the Xnet 2.00 Theory and User Manual (Olivier, 2004).

4.3 General element types

Xnet utilises a range of so-called primitive elements to model thermal-fluid networks. Primitive elements represent the smallest network building blocks and cannot be discretised in smaller units. Examples are a pipe section, a convective heat transfer coefficient and a conductive heat transfer link. In the next section, the elements that are applicable to this study, namely the modelling of heat exchangers, will be discussed in more detail.

4.3.1 Darcy Weisbach pipe elements – PIPEDW

A Darcy Weisbach pipe element is described as a pipe element with the frictional pressure drop across the pipe, Δp_0, a function of the Darcy Weisbach friction factor, f. The Darcy Weisbach friction factor used in the momentum equation, Equation 3.15, is given by:

$$f = \begin{cases} \dfrac{64}{\mathrm{Re}} & \text{if Re} < 2300 \\ \dfrac{0.25}{\left(\log\left(\dfrac{e}{3.7D_h} + \dfrac{5.74}{\mathrm{Re}^{0.9}}\right)\right)^2} & \text{if Re} > 5000 \end{cases} \qquad (4.1)$$

where

Re is the Reynolds number,

Re < 2300 indicates laminar flow,

Re > 5000 indicates turbulent flow and

e is the mean inside wall roughness.

For the transitional region ($2300 \leq \mathrm{Re} < 5000$) the friction factor is calculated with linear interpolation between the two equations shown in Equation (4.1).

The Darcy-Weisbach pipe element is a constant area annulus, which is calculated with the user specified diameters. If the inside diameter is zero, the element becomes a typical round pipe. The roughness and loss factors for the calculation of the pressure drop across the element can be specified. Different secondary pressure drop loss factors could be specified for inlet-and-exit losses, as well as losses within the pipe element. The inlet losses are based on the inlet velocity, whereas the exit losses are based on the exit velocity. The losses within the pipe element are based on the average velocity within the pipe. It is also possible to specify the number of pipes in parallel in a single group. This group of pipes has a common inlet and outlet, as well as the same pipe geometry

4.3.2 Duct with area change pipe elements - PIPEDG

A primitive duct pipe element represents a general duct with any inlet shape, whether it is circular, square, triangular etc. The duct can either have the same inlet and outlet flow areas or different inlet and outlet flow

areas. For non-circular flow elements, the inlet (and outlet) area and perimeter is given and the hydraulic diameter calculated with:

$$D_{H_i} = \frac{4A_i}{P_i} \tag{4.2}$$

where A_i and P_i is the area and perimeter at the inlet (or outlet) respectively.

Xnet calculates the mean area of the inlet and outlets and base the mass flow through the element on this mean area. The frictional pressure drop is calculated with the Darcy-Weisbach friction factor given in the previous section and is also based on the mean velocity. The roughness and loss factors for the pressure drop calculation can be specified, as is also the case for the primitive Darcy-Weisbach pipe element. It is also possible to specify inlet and outlet loss factors, based on the inlet and outlet velocities respectively, for forward and backward flow. Loss factors based on mean velocities can also be specified for forward and backward flow within the element.

4.3.3 Heat exchanger pipe elements - PIPEHX

This type of element is used to model flow through complex heat exchanger passages such as the flow across a bank of finned tubes or inside a compact heat exchanger. For this element it is necessary to specify the hydraulic diameter, the frontal area and the area ratio defined as the minimum flow area to frontal area, for both the inlet and outlet. It is also necessary to specify the gas volume within the tube bank for transient simulations. Inlet and outlet loss factors can be specified. The friction factor for this type of element is a function of the element geometry. The friction factor can either be specified as a constant value or a user-defined curve, known as a heat exchanger chart. A typical chart is shown in Figure 4.1. The curve gives the friction factor as a function of the Reynolds number. This friction factor is then substituted in the momentum equation to give the pressure drop across the element.

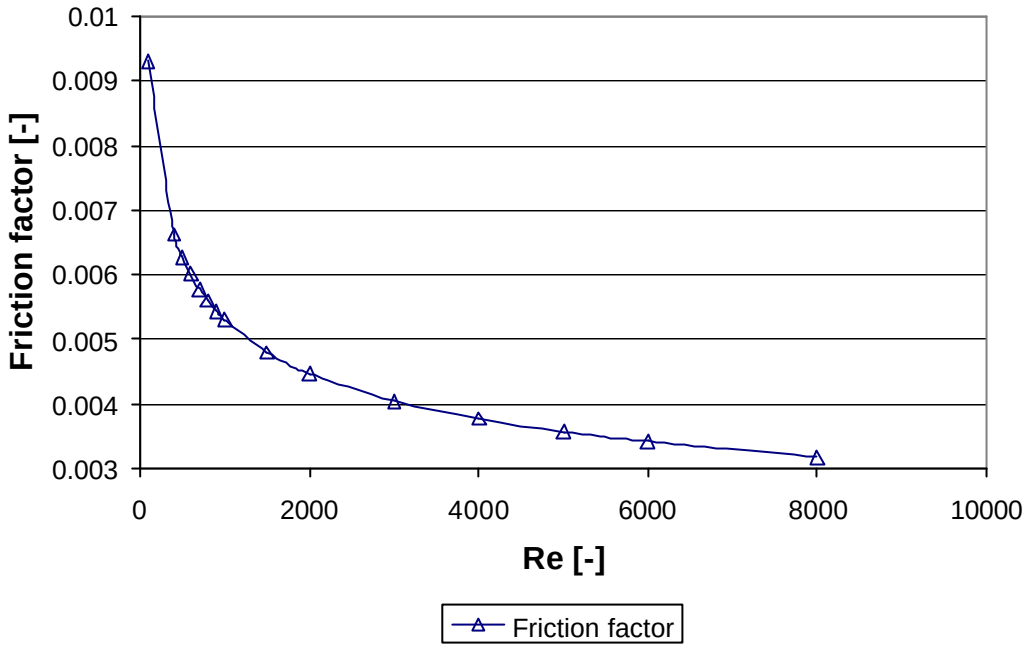


Figure 4.1: Typical heat exchanger chart showing the relation between the Reynolds number and the friction factor.

4.3.4 Pebble Bed pipe elements - PIPEPB

The Pebble Bed pipe element represents an element that could be used for the simulation of fluid flow through a porous medium similar to that of a pebble bed reactor. This element allows several different relations for flow through a packed bed. The two most important pressure drop relations are the Ergun relation (Ergun, 1952; McCabe, Smith and Harriott, 1993) and the Pebble Bed relation for the pressure drop through the bed. The Ergun relation is given by:

$$f = 2\left(\frac{150\mu(1-e)^2}{\rho d_p e^2 |V_i|} + \frac{1.75|V|}{|V_i|}\left(\frac{1-e}{e}\right)\right) \tag{4.3}$$

where

V is the actual velocity in the bed,

e is the void fraction,

d_p is the diameter of the pebbles and taken as the hydraulic diameter and

V_i is the average velocity in the in the element.

The Pebble Bed relation is given by:

$$f = 2\left[\frac{160\mu}{d_p \rho}\frac{(1-e)^2}{e^2|V_i|} + 3\frac{|V|(1-e)^{1.1}}{e|V_i|}\left(\frac{\mu}{\rho e|V|d_p}\right)^{0.1}\right] \tag{4.4}$$

4.3.5 Convection elements

Convection elements serve as the heat exchange link between a fluid and a solid wall. Convection elements can be connected between either a flow element or a flow node to either a solid element or a solid node. For a convection element, the user must specify the heat transfer area, and either a user-defined curve, which gives the relation between the Reynolds number and the Colburn *j* factor which is used to calculate the heat transfer coefficient. If no curve is specified, the relation

$$Nu = C\,\mathrm{Re}^{m}\,\mathrm{Pr}^{n} \tag{4.5}$$

holds. The constants C, m, n, are dependant of the geometry of the element.

4.3.6 Nodes

Nodes represent the end points of elements. Nodes can either be a solid node or a flow node. The flow nodes represent the end points of flow elements and the solid node the end point of a conduction of convection element. Conduction elements represent heat flow through a solid material. This implies that solid nodes represent a fixed mass of solid material. A solid node will for example represent the physical finned tube in a finned tube heat exchanger or a composite piece of isolation material.

4.4 Comparative study of fluid flow and heat transfer

An extensive comparative study was done between Xnet and Flownex to compare the results for fluid flow elements and heat transfer elements in thermal-fluid networks. The study entailed the individual comparison of fluid flow and heat transfer and several combinations of heat transfer and fluid flow. To avoid repetition of the results, only some of the cases will be presented in graphical form while the rest will be discussed. In all the cases helium (defined in Appendix A) was used as working fluid. A complete specification and discussion of the test cases are given in Appendix C.

4.4.1 Comparison of fluid flow elements

The comparison of fluid flow elements is done to verify (and validate) that the governing equations are correctly implemented and solved. The four flow elements discussed in Section 4.3 will be used. The steady-state comparison will comprise of plotting mass flow values for a given pressure drop across the element. Since the fluid velocity in a heat exchanger is relatively low, the Reynolds number in these test cases ranges between:

$$0 \le \mathrm{Re} \approx 40 \times 10^{4}$$

For the transient comparison, a step increase in pressure drop will be applied over the element with an expected increase in the mass flow. The steady-state and transient comparisons will also involve investigating the effect of the number of increments used when an element is discretised. This would indicate the minimum number of increments that would be necessary for an accurate result. Four levels of

discretisation will be used. A pipe element will subsequently be subdivided into 1 increment, and 5, 10 and 20 increments respectively. This validation process is schematically shown in Figure 4.2.

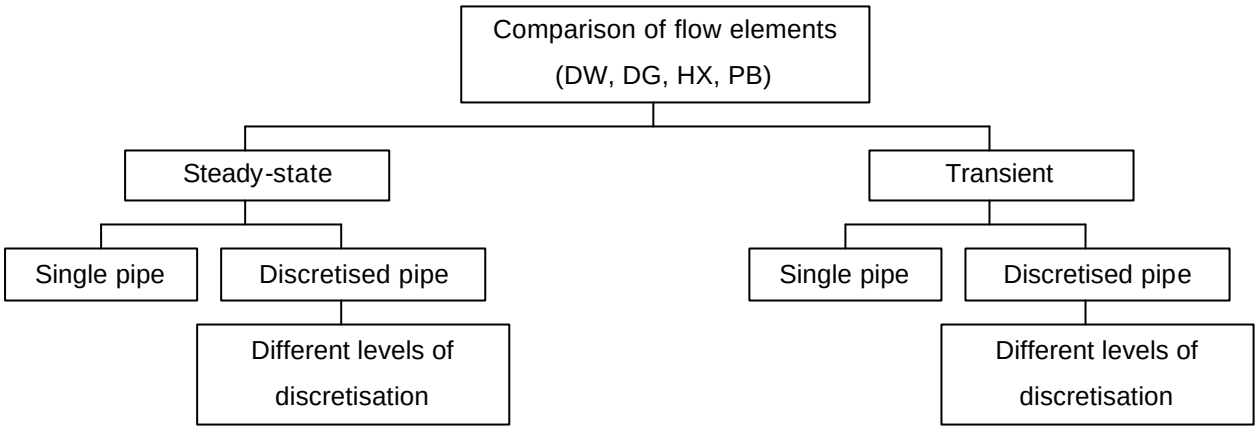


Figure 4.2: Schematic representation of the validation process of the flow elements in Xnet.

4.4.1.1 Darcy Weisbach flow element - PIPEDW

The Darcy Weisbach flow element that were used had the following inputs:

Variable	***Value(s)***
Fluid	Helium
L	10.0 [m]
$\varnothing$	0.10 [m]
p_i	150 [kPa]
Δp_o	1; 2; 5; 10; 15; 20; 25; 30; 35; 40; [kPa]
e	30 [µm]

Table 4.1: Inputs for the Darcy-Weisbach pipe element.

Steady-state results

Four levels of discretisation were investigated here. In Figure 4.3 the results for a single PipeDW flow element and a discretised PipeDW flow element with 20 increments ($n = 20$) are compared. The figure shows that with an increase in the mass flow, the velocity increases, which increases the frictional pressure loss. It is also evident from this figure that there is good agreement between the value predicted by Xnet with that predicted by Flownex. It is evident from the figure that only a single element is necessary to achieve an accurate enough result for a case where the velocity is low. The velocity in this case varied from 400 m/s to 560 m/s.

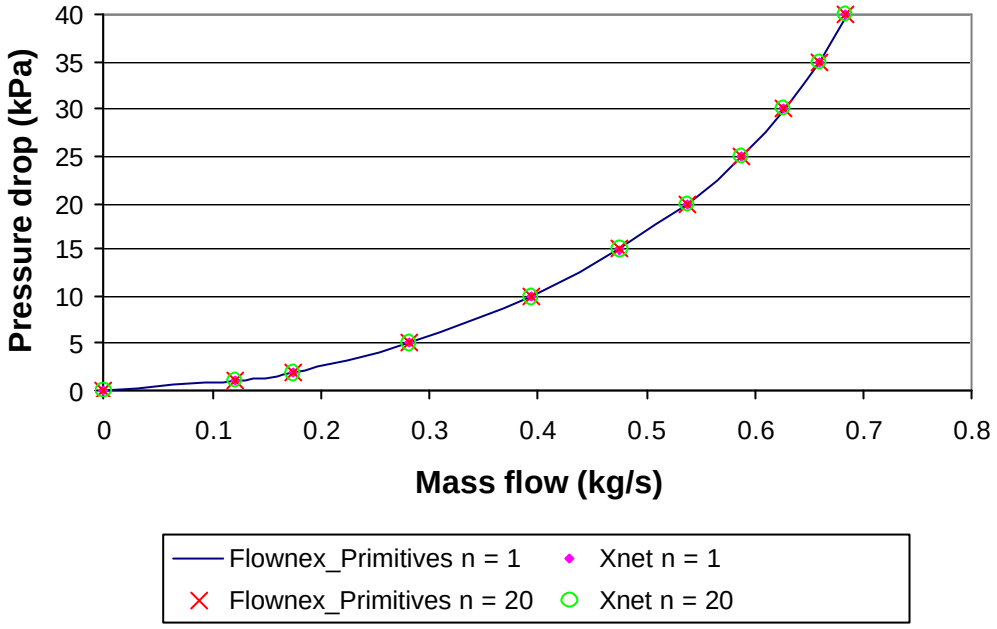


Figure 4.3: Steady-state comparison between the results for a single PipeDW flow element and a discretised PipeDW flow element with 20 increments.

Transient results

For the transient simulation, the PipeDW flow element was subjected to a 2 kPa step increase in pressure drop across the element. This was done by changing the outlet pressure, p_o, instantaneously from 149 kPa to 147 kPa. Four levels of discretisation were investigated, as explained above, and the result for 1 increment ($n = 1$) and 20 increments ($n = 20$) are shown in Figure 4.4 and Figure 4.5. As expected, an increase in the mass flow over a period of time resulted from the step increase in the pressure drop across the element. In Figure 4.4 Xnet produces a smooth curve for the mass flow increase over the time period, and shows good agreement with the result produced by Flownex. For this case, Xnet's time step, Δt, defined as:

$$\Delta t = a\,\Delta t_{Courant} \tag{4.6}$$

was 7.8727×10^{-3} with $a = 0.8$.

As the number of increments is increased, Xnet produces small oscillations in its prediction of the mass flow during the transient. It is illustrated that this problem tends to get larger as the number of increments are increased (Figure 4.5). As the number of increments are increased to 20, the time step, Δt, decreases to 3.9363×10^{-4}. If the time step factor, a, of Xnet is varied, there is no difference in the oscillations produced for the predicted mass flow. Investigation into increased damping and zero damping (Section 3.5.3), does not offer solutions to the oscillation problem either. This shows that this phenomenon is neither a numerical stability problem, nor the result of the damping process. To further investigate the oscillations, the two-step LW method described by Botha (2000) was used to simulate the same problem.

The two-step LW software code developed by Botha produced a similar oscillation on the predicted mass flow result and is shown in Figure 4.6, together with the result of the RKTD method. A logical explanation is that the oscillations are secondary pressure waves due to the sudden change in the inlet pressure. These pressure waves occur when a fast transient is superimposed on a slow transient. When Flownex's time step is made the same as the RKTD's time step, Flownex produces the same oscillation of the result and is compared to the RKTD in Figure 4.7. Flownex and the RKTD produce almost the exact same oscillation, and it is difficult to distinguish them in the figure. When Flownex's time step is chosen large, the fast transient is filtered and only the slow transient is seen. Since Xnet's time step is limited by the Courant condition, the secondary pressure waves in fast transient will always be seen in Xnet. However, since slow transients in heat exchangers are more important, this phenomenon will not be investigated further.

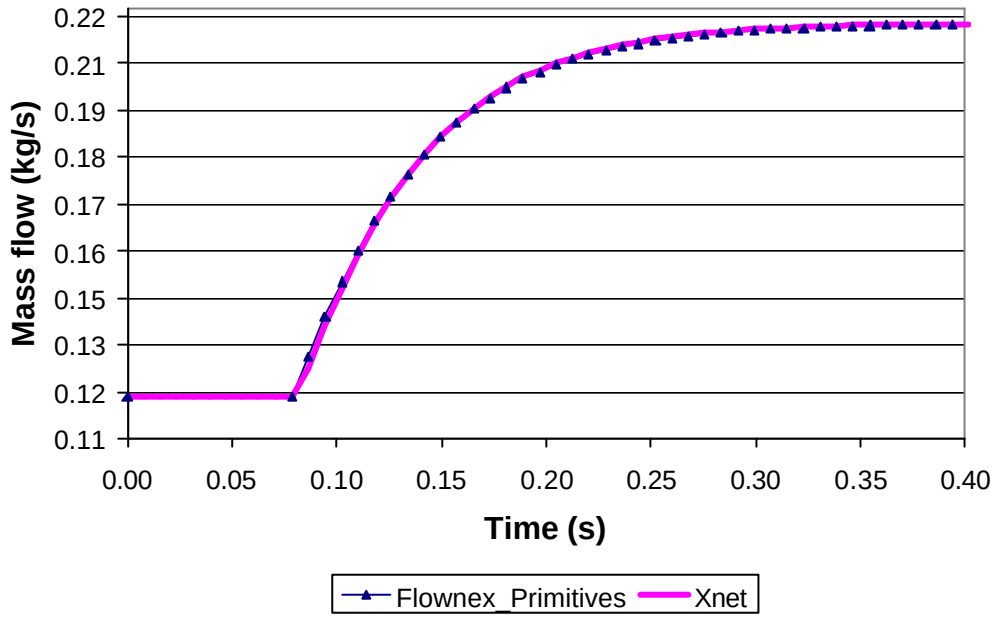


Figure 4.4: Transient $\dot{m}$ - t relationship for a single PipeDW flow element.

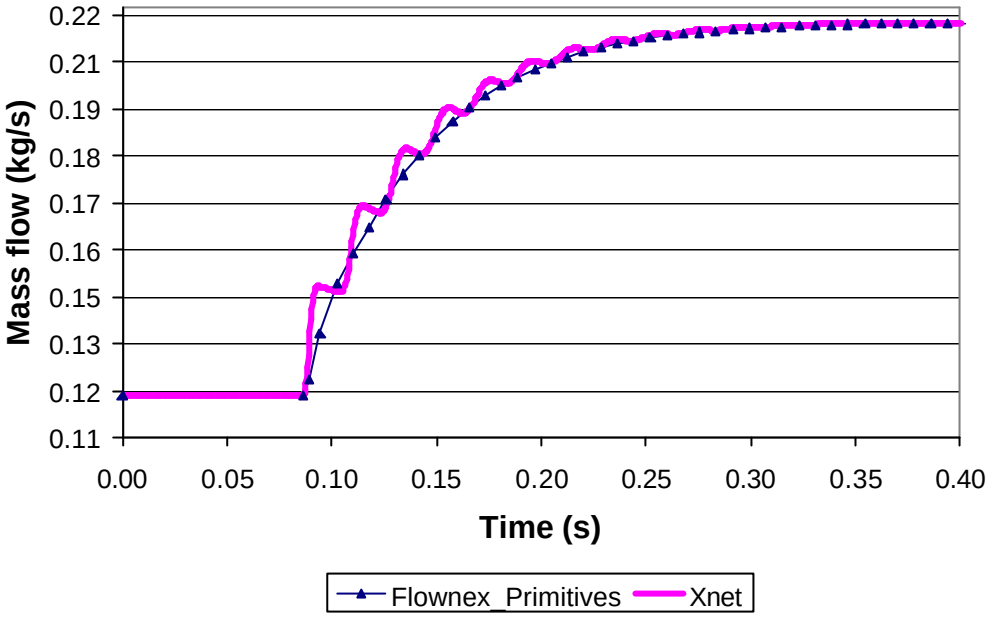


Figure 4.5: Transient $\dot{m}$ - t relationship for a discretised PipeDW flow element with $n = 20$.

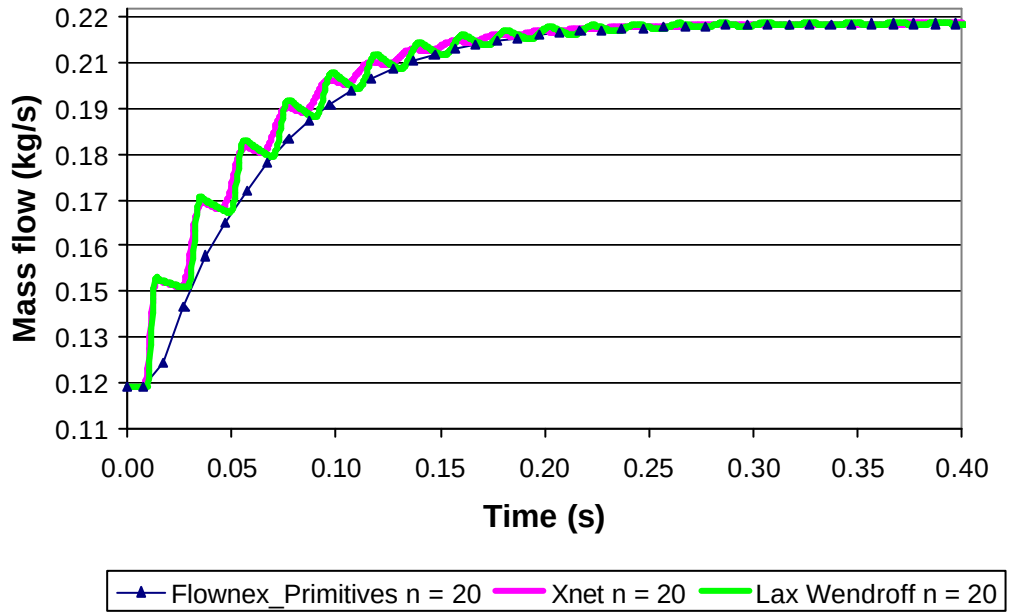


Figure 4.6: Transient $\dot{m}$ - t relationship comparison between Xnet, Flownex and the LW explicit scheme for a discretised PipeDW flow element with $n = 20$.

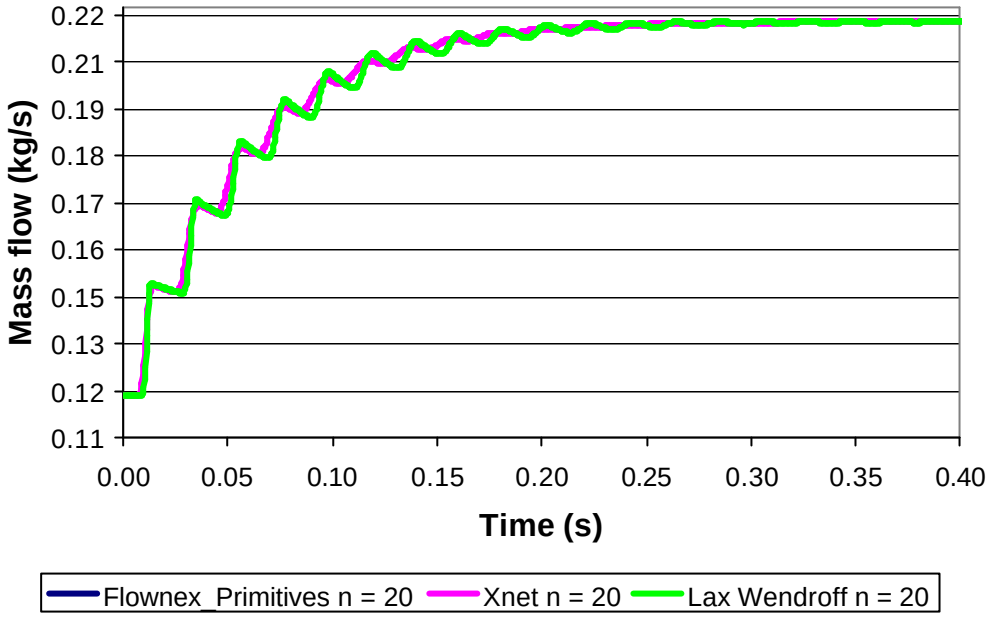


Figure 4.7 Transient $\dot{m}$-t relationship comparison between Flownex, Xnet and the LW explicit scheme for a discretised PipeDW flow element with $n = 20$ and the same time step.

4.4.1.2 Duct with area change flow element - PIPEDG

The duct with area change flow element that were used had the following inputs (Table 4.2):

Variable	*Value(s)*
Fluid	Helium
L	10.0 [m]
$\varnothing_i$	0.10 [m]
$\varnothing_o$	0.20 [m]
p_i	150 [kPa]
Δp_o	1; 2; 3; 4; 5; 10; 15; 20 [kPa]
e	30 [µm]

Table 4.2: Inputs for the duct with area change pipe element.

Steady-state results

Four levels of discretisation were investigated here. In Figure 4.8 the results for a single PipeDG flow element and a discretised PipeDG flow element with two levels of discretisation are compared. The figure shows that with an increase in the mass flow, the velocity increases, which increases the frictional pressure loss. It is also evident from this figure that there is good agreement of the value predicted by Xnet with that predicted by Flownex. It is evident from the figure that when a single element is used, the results differ to a great extent from the discretised elements results. When more increments are used, the result get more

accurate. Since this element has a varying area, more increments give a better approximation of the shape of the element. For an element with differing inlet and outlet areas it is necessary to discretise the element into smaller elements.

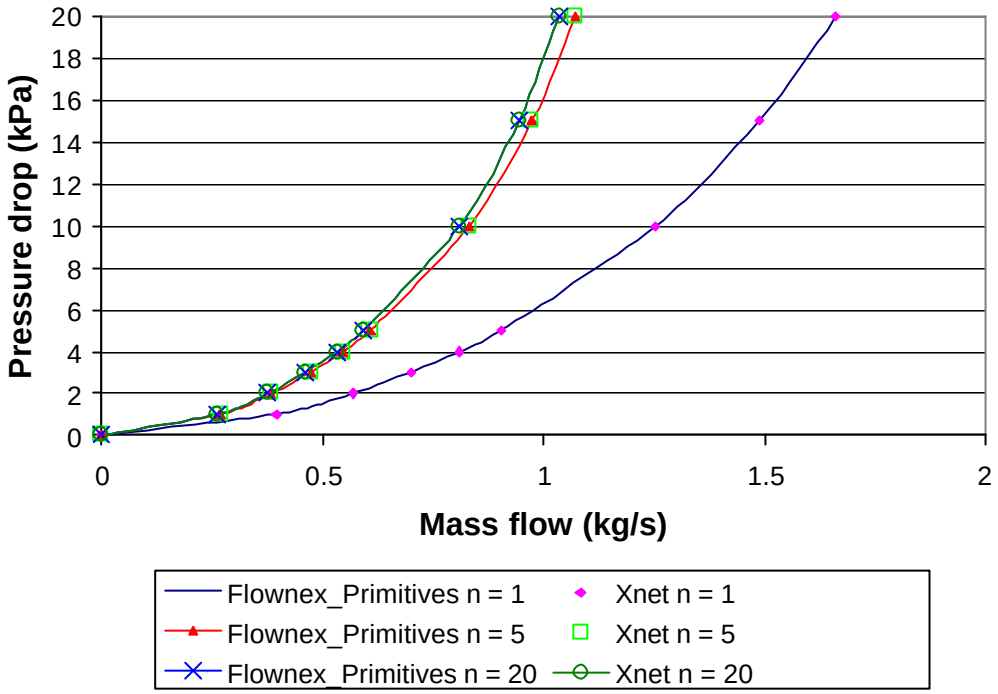


Figure 4.8: Steady-state comparison between the accuracy of the results for a single PipeDG flow element and a discretised PipeDG flow element with 5 and 20 increments respectively.

Transient results

For the transient simulation, the PipeDG flow element was subjected to a 2 kPa step increase in pressure drop across the element. This was done by changing the outlet pressure, p_o, instantaneously from 149 kPa to 147 kPa. Four levels of discretisation were investigated, as explained above, and the result for 1 increment ($n = 1$) and 20 increments ($n = 20$) are shown in Figure 4.9 and Figure 4.10. As expected, an increase in the mass flow over a period of time resulted from the step increase in pressure drop across the element. In Figure 4.9 Xnet produces a smooth curve for the mass flow increase over the time period and shows good agreement with the result produced by Flownex. As the number of increments is increased, Xnet produces small oscillations in its prediction of the predicted mass flow during the transient. It appears that this phenomenon tends to increase as the number of increments is increased (Figure 4.10). This phenomenon was discussed for the Darcy Weisbach flow element and will not be repeated here.

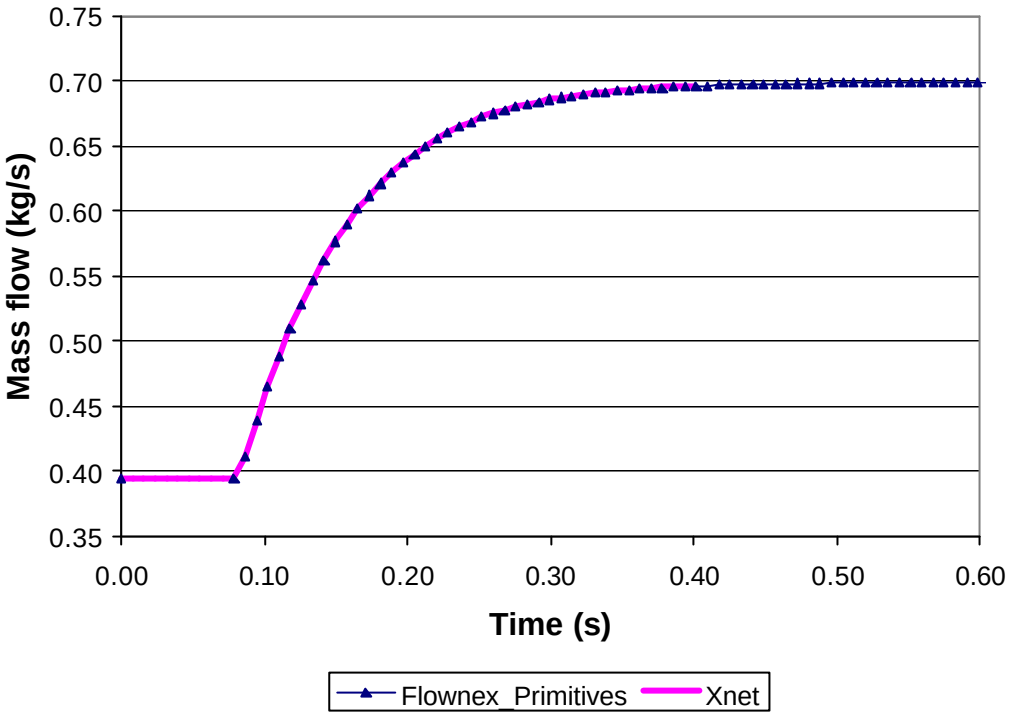


Figure 4.9: Transient $\dot{m}$ - t relationship for a single PipeDG flow element.

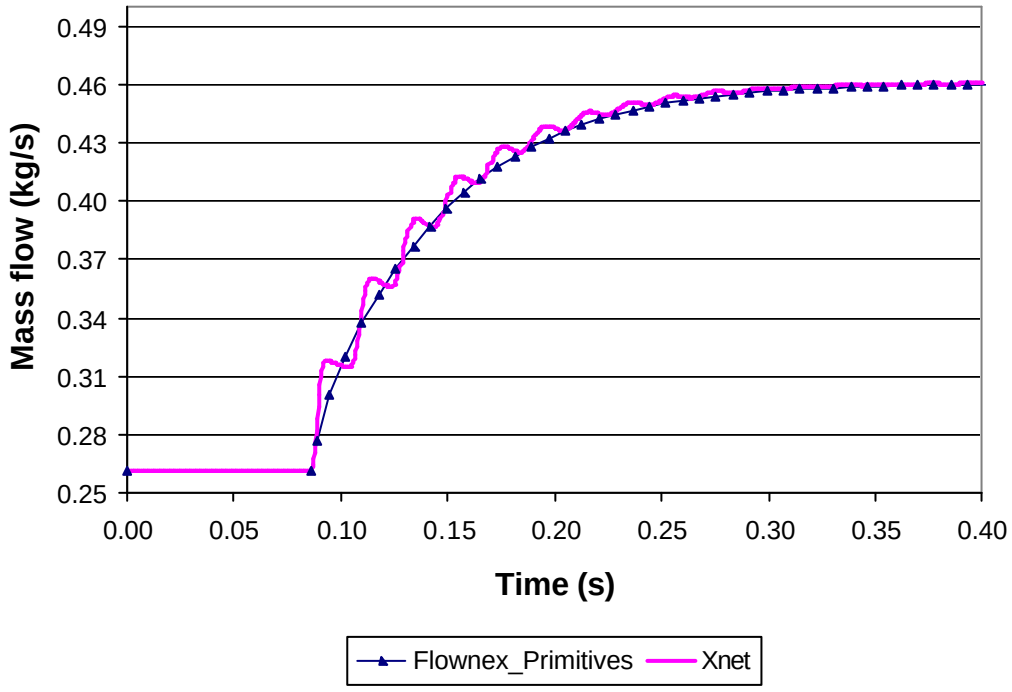


Figure 4.10: Transient $\dot{m}$ - t relationship for a discretised PipeDG flow element with $n = 20$.

4.4.1.3 Heat exchanger conduit flow element - PIPEHX

The heat exchanger conduit flow element that were used had the following inputs (Table 4.2):

Variable	*Value(s)*
Fluid	Helium
L	10.0 [m]
H	1.0 [m]
W	2.0 [m]
D_H	0.10 [m]
s	0.50 [-]
p_i	150 [kPa]
Δp_o	1; 2; 5; 10; 15; 20; 25; 30; 35; 40 [kPa]
f	0.05 [-]

Table 4.3: Inputs for the heat exchanger conduit pipe element.

Steady-state results

Four levels of discretisation were investigated here. In Figure 4.11 the results for a single PipeHX flow element and a discretised PipeHX flow element with two levels of discretisation are compared. The figure shows that with an increase in the mass flow, the velocity increases, which increases the frictional pressure loss. It is also evident from this figure that there is good agreement between the value predicted by Xnet and that predicted by Flownex and that only a single element is necessary to achieve a sufficiently accurate result.

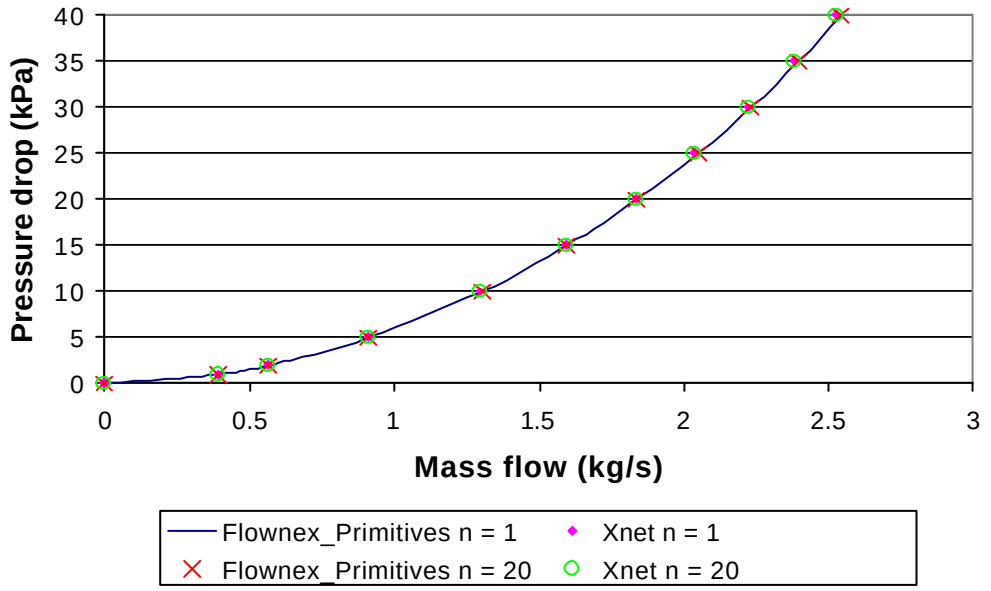


Figure 4.11: Steady-state comparison between the accuracy of the results for a single PipeHX flow element and a discretised PipeHX flow element with 20 increments.

Transient results

For the transient simulation, the PipeHX flow element was subjected to a step increase in pressure drop across the element. Four levels of discretisation were investigated here, and the result for 1 increment ($n = 1$) and 20 increments ($n = 20$) are shown in Figure 4.12 and Figure 4.13 respectively. As expected, an increase in the mass flow over a period of time resulted from the step increase in the pressure drop across the element. In Figure 4.12 Xnet produces a smooth curve for the mass flow increase over the time period, and shows good agreement with the result produced by Flownex.

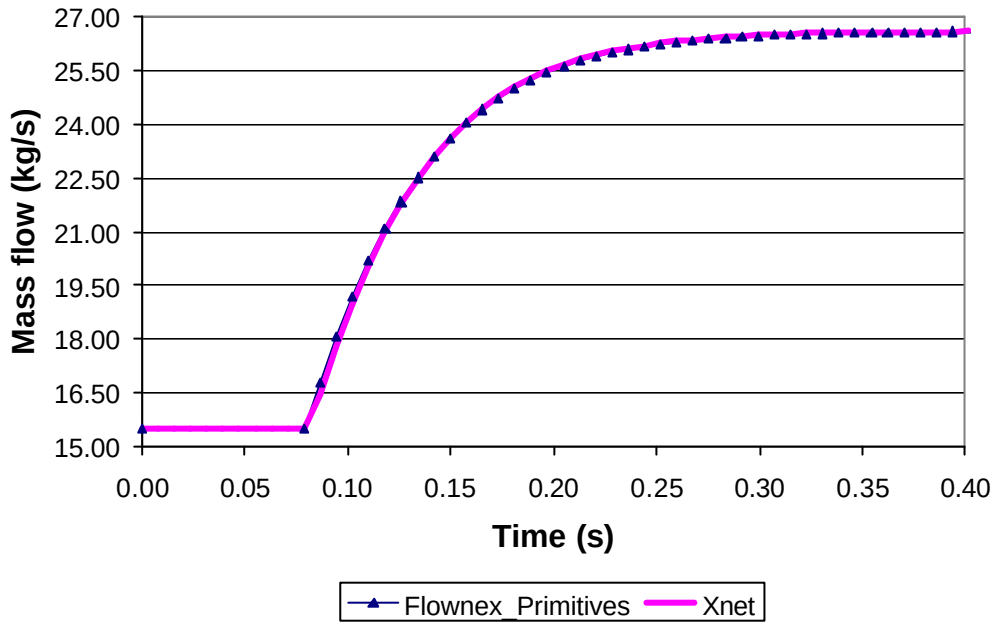


Figure 4.12: Transient $\dot{m}$-t relationship for a single PipeHX flow element.

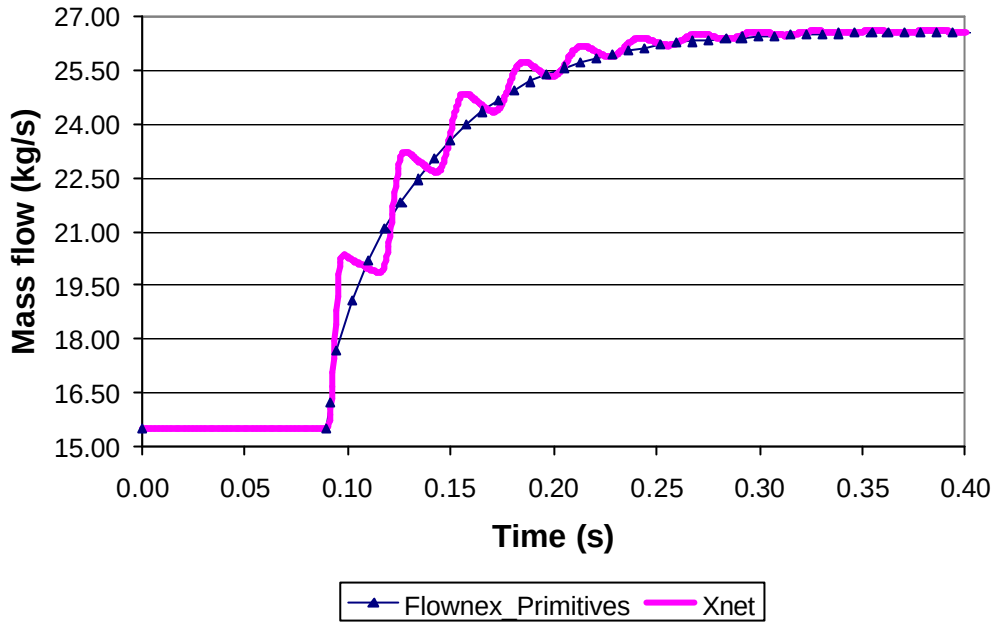


Figure 4.13: Transient $\dot{m}$-t relationship for a discretised PipeHX flow element with $n = 20$.

As the number of increments is increased, Xnet produces small oscillations in its prediction of the predicted mass flow during the transient. It is evident that this phenomenon tends to increase as the number of increments is increased (Figure 4.13). This phenomenon was discussed for the Darcy Weisbach flow element and will not be repeated here.

4.4.1.4 Porous Bed Flow Element - PIPEPB

The Porous Bed flow element that was used had the following inputs (Table 4.2):

Variable	***Value(s)***
Fluid	Helium
L	10.0 [m]
A_i	0.2 [m]
A_o	0.2 [m]
D_H	0.60 [m]
e	0.50 [-]
p_i	150 [kPa]
Δp_o	1; 2; 5; 10; 15; 20; 25; 30; 35; 40 [kPa]
f	Pebble Bed

Table 4.4: Inputs for the porous bed pipe element.

Steady-state results

Four levels of discretisation were investigated here. In Figure 4.14 the results for a single PipePB flow element and a discretised PipePB flow element with two levels of discretisation are compared. The figure shows that with an increase in the mass flow, the velocity increases, which increases the frictional pressure loss. It is also evident from this figure that there is good agreement between the value predicted by Xnet and that predicted by Flownex, and that only a single element is necessary to achieve a sufficiently accurate enough result.

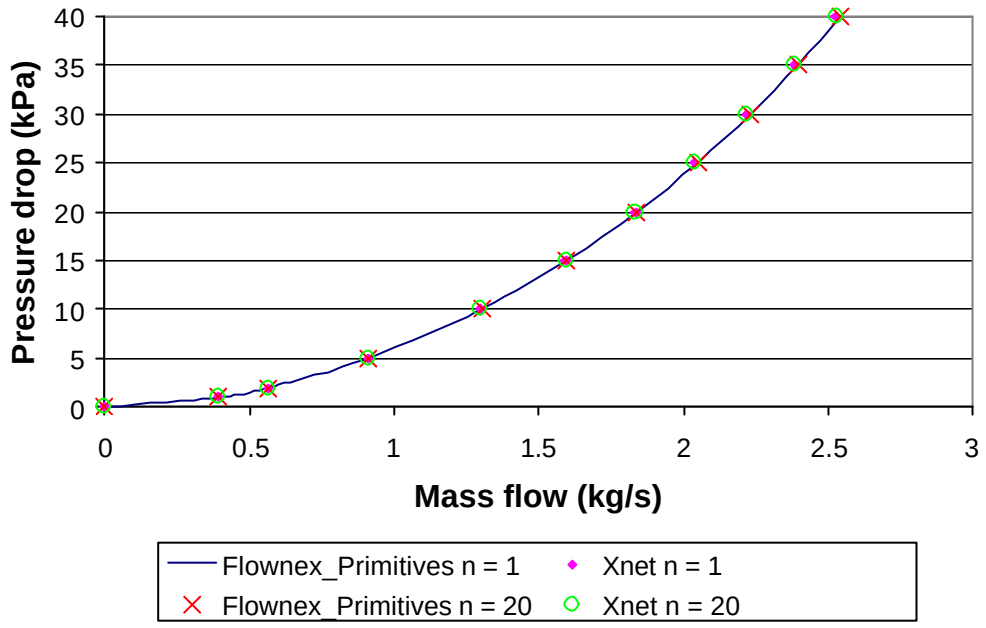


Figure 4.14: Steady-state comparison between the accuracy of the results for a single PipePB flow element and a discretised PipePB flow element with 20 increments.

Transient results

For the transient simulation, the PipePB flow element was subjected to a step increase in pressure drop across the element. Four levels of discretisation were investigated here, and the result for 1 increment ($n = 1$) and 20 increments ($n = 20$) are shown in Figure 4.15 and Figure 4.16 respectively. As expected, an increase in the mass flow over a period of time resulted from the step increase in the pressure drop across the element. In Figure 4.15 Xnet produces a smooth curve for the mass flow increase over the time period, and shows good agreement with the result produced by Flownex. As the number of increments is increased, Xnet produces small oscillations in its prediction of the predicted mass flow during the transient. It appears that this phenomenon tends to get larger as the number of increments is increased (Figure 4.16). This phenomenon was discussed for the Darcy Weisbach flow element and will not be repeated here.

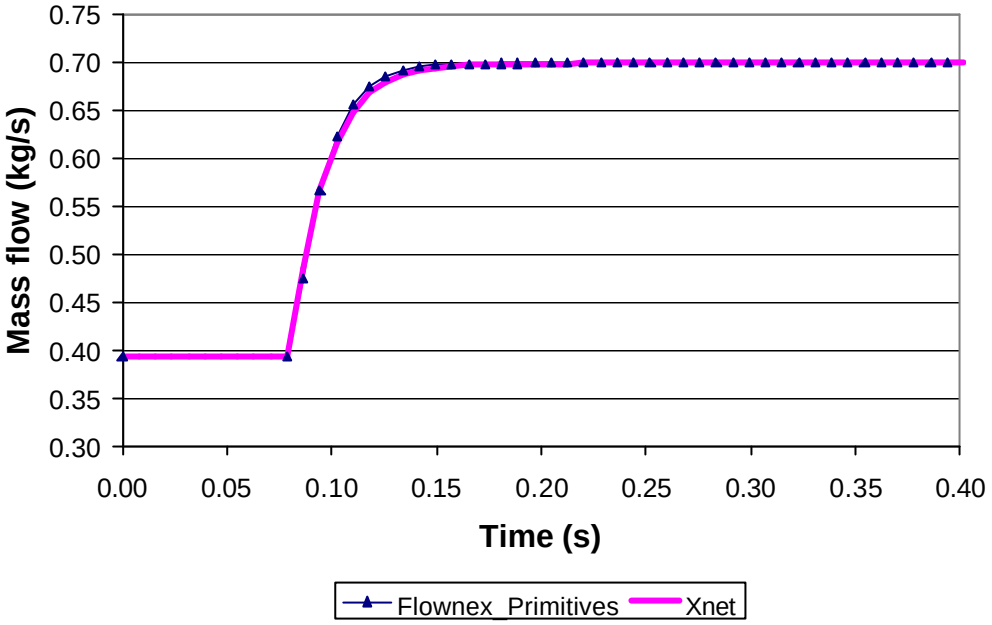


Figure 4.15 Transient $\dot{m}$ - t relationship for a single PipePB flow element.

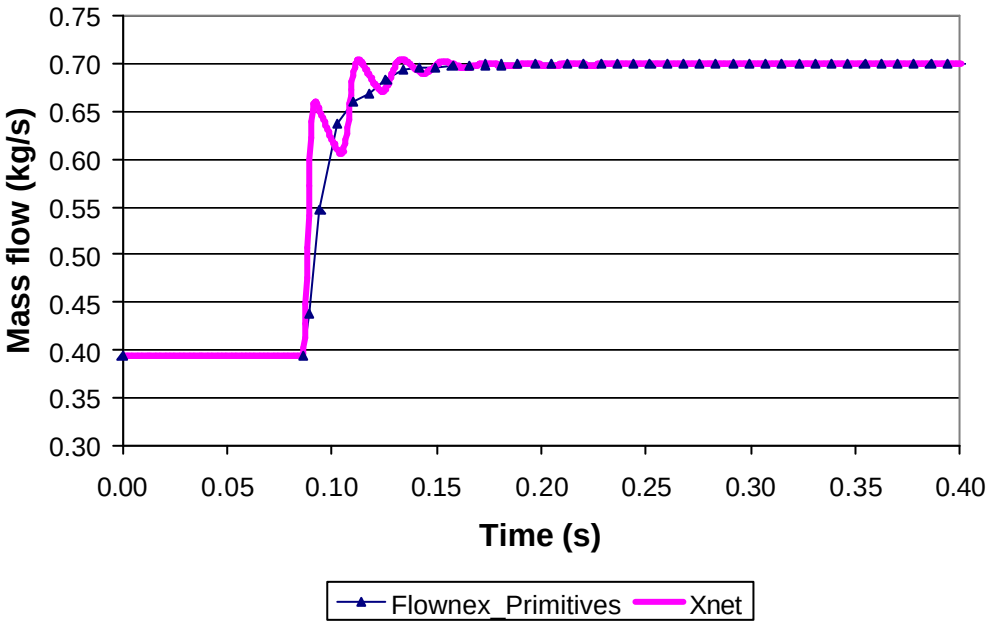


Figure 4.16: Transient $\dot{m}$ - t relationship for a discretised PipePB flow element with $n = 20$.

4.4.2 Comparison of heat transfer elements

To ensure that the heat transfer equations are correctly implemented, several different comparisons (test cases) were performed. The test cases comprised of steady-state and transient heat transfer. The first two test cases were for a fixed fluid inlet temperature, and the second two for a fixed wall temperature. These test cases used both the node-element and node-node network topologies. Transient simulations were only performed for cases where the node-element network topology were used. *It should be noted that since heat transfer in the direction of the flow (axial heat transfer) is not the focus of this study of heat exchangers, only heat transfer through convection was investigated*

The convective heat transfer connection topology implemented in Xnet that is applicable here will be discussed briefly in the following section to clarify the logic for the methodology used for the comparison of the convection element.

4.4.2.1 Convective heat transfer network topology

There are two different network topologies implemented in Xnet for convective heat transfer. These two topologies are needed since it is possible to connect the convection elements either to a flow node or a flow element. A detailed description of the topologies will only be presented in Chapter 5. It should be noted, however, that both types of topologies, whether it is a node-element connection or a node-node connection, will be used during the comparison process to ensure the correct implementation of the heat transfer equations.

Test case 1 – Steady-state

The first test case will consist of a PipeDW flow element with a convection element connected to it. This represents a pipe with heat transfer to the pipe wall, which is at an initial temperature of 25 ºC. The flow element will have a fixed pressure drop and a fixed temperature at the inlet. The steady-state temperature of the solid node (which represents the pipe wall), which is connected with a convection element at the *flow element*, will be compared to Flownex and the analytical result for a range of fixed fluid inlet temperatures. The test case is schematically shown in Figure 4.17 and the inputs given in Table 4.5 on the following page.

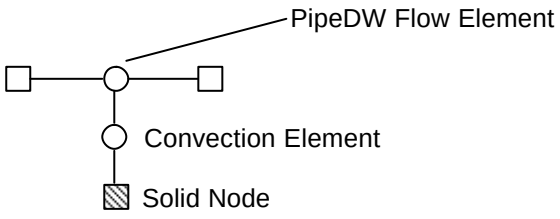


Figure 4.17: Schematic representation of test case 1.

Variable	*Value(s)*
Fluid	Helium
L	10.0 [m]
$\varnothing$	0.10 [m]
p_i	150 [kPa]
p_o	130 [kPa]
e	30 [μm]
A_H	3.142 [m^2]

Table 4.5: Inputs for the test case 1.

Figure 4.18 show that there is an increase in the solid node temperature as the fluid inlet temperature is increased. The results also show that the wall temperature is less than the fluid's total temperature. The results obtained by Xnet also show good agreement with that of Flownex and the analytical result.

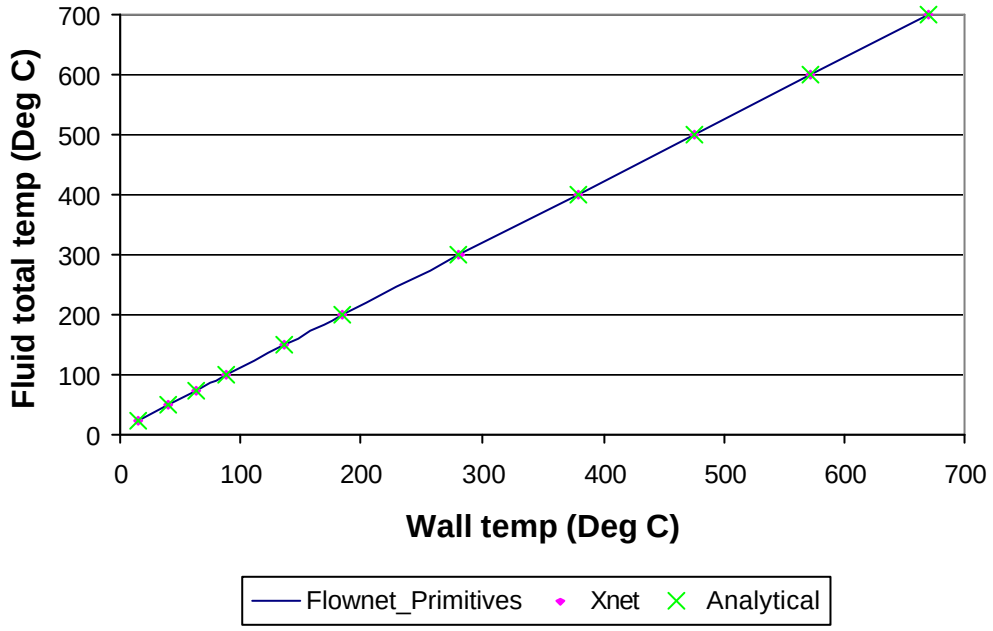


Figure 4.18: Test case 1 steady-state results.

Test case 1 – Transient

For the transient simulation the above problem was used with a fixed fluid inlet temperature of 200 °C with a step increase to 275 °C. Figure 4.19 shows the result obtained for the transient solution. The solid node temperature gradually increases after the fluid inlet temperature was subjected to a step input. The reason for the gradual increase is the fact that the solid node has thermal inertia and takes time to heat up. This result confirms the thermal inertia of the node and shows that the surface heat transfer coefficient, h, is correctly implemented.

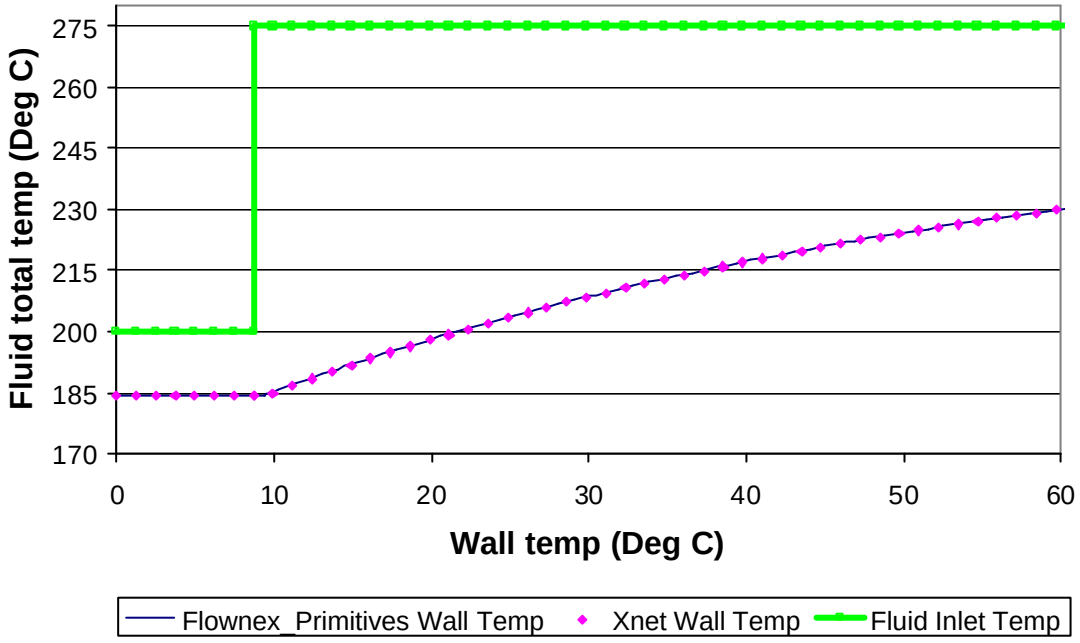


Figure 4.19: Transient result of test case 1.

Test case 2

The second test case will consist of two PipeDW flow elements with a convection element connected to the flow node between them. This case uses the same inputs as defined in Table 4.5. The difference between the two cases is that for this case, the pipe is divided into two increments and the convection element is connected at the flow node between the two elements. The test case is schematically shown in Figure 4.20. The flow elements will have a fixed pressure drop and a fixed temperature at the inlet. The steady-state temperature of the solid node, which is connected with a convection element at the *flow node*, will be compared to Flownex for a range of fluid inlet temperatures. Figure 4.21 show that there is an increase in the solid node temperature as the fluid inlet temperature is increased. The results obtained by Xnet also show good agreement with those of Flownex and the analytical result.

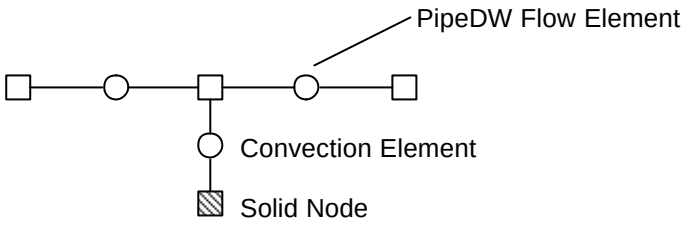


Figure 4.20: Schematic representation of test case 2.

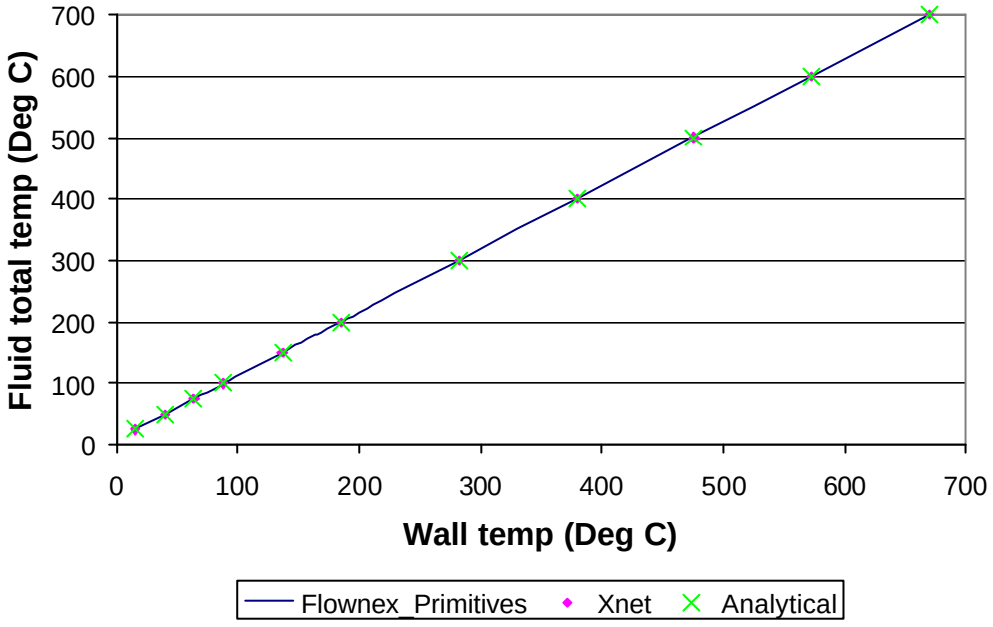


Figure 4.21: Test case 2 steady-state results.

Test case 3 – Steady-state

This test case consists of a PipeDW flow element with a convection element connected to it. This case uses the same inputs as defined in Table 4.5. The flow element will have a fixed pressure drop and a fixed temperature of 25 °C at the inlet. The temperature of the solid node, which is connected with a convection element at the *flow element*, is fixed and the steady state outlet temperature of the fluid compared to Flownex for a range of solid node temperatures. The test case is schematically shown in Figure 4.22. Figure 4.23 shows that there is an increase in the solid node temperature as the fluid inlet temperature is increased. The results obtained by Xnet also show good agreement with those of Flownex and the analytical result.

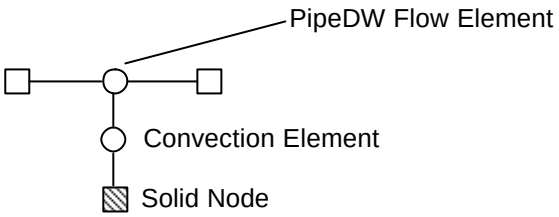


Figure 4.22: Schematic representation of test case 3.

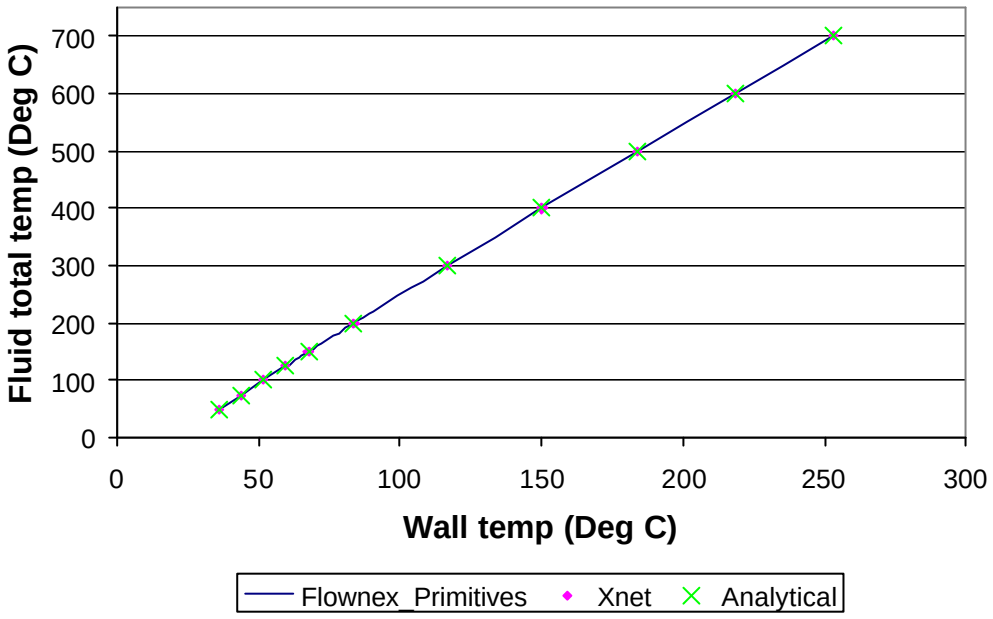


Figure 4.23: Test case 3 steady-state results.

Test case 3 – Transient

For the transient simulation the above problem were used with a fixed solid node temperature of 75 °C with a step increase to 175 °C. Figure 4.24 shows the result obtained for the transient solution. The solid node temperature almost immediately increases after the fluid inlet temperature was subjected to a step input. The reason for the sudden increase is the fact that the solid node's thermal inertia has no effect on the fluid's temperature and will cause the fluid temperature to rise almost instantaneously.

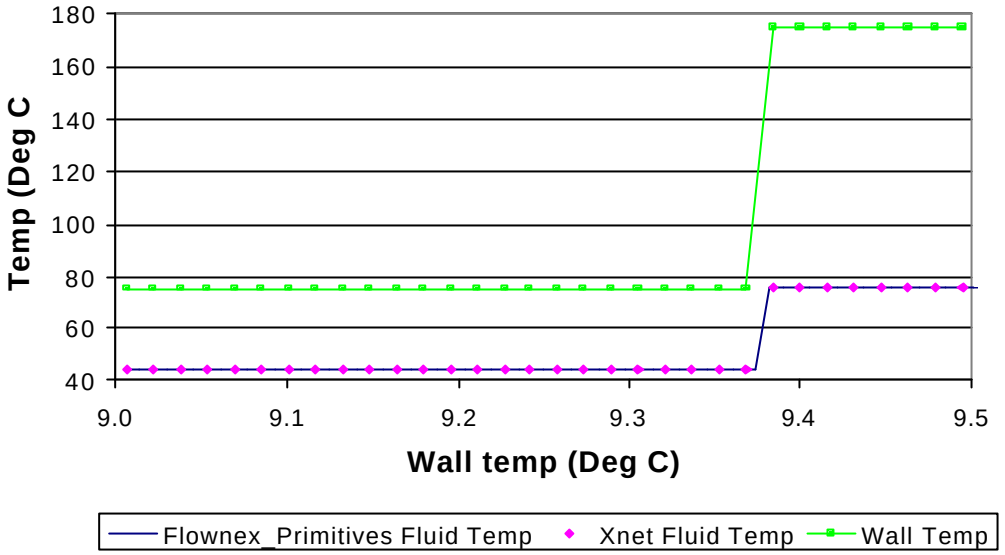


Figure 4.24: Transient results for test case 3.

Test case 4

This test case consists of two PipeDW flow elements with a convection element connected to the flow node between them. This case uses the same inputs as defined in Table 4.5. The difference between the two cases is that for this case, the pipe is divided into two increments and the convection element is connected at the flow node between the two elements. The flow element will have a fixed pressure drop and a fixed temperature of 25 °C at the inlet. The temperature of the solid node, which is connected with a convection element at the *flow node*, will be fixed and the steady-state outlet temperature of the fluid compared to Flownex for a range of solid node temperatures. The test case is schematically shown in Figure 4.25.

Figure 4.26 shows that there is an increase in the solid node temperature as the fluid inlet temperature is increased. The results obtained by Xnet also show good agreement with those of Flownex and the analytical result.

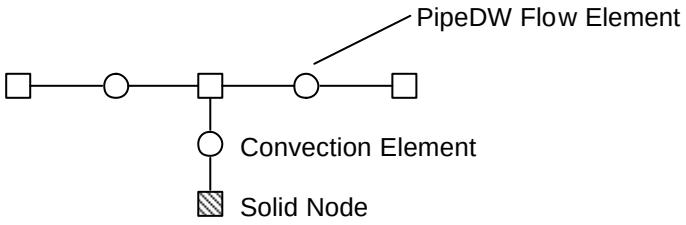


Figure 4.25 Schematic representation of test case 4.

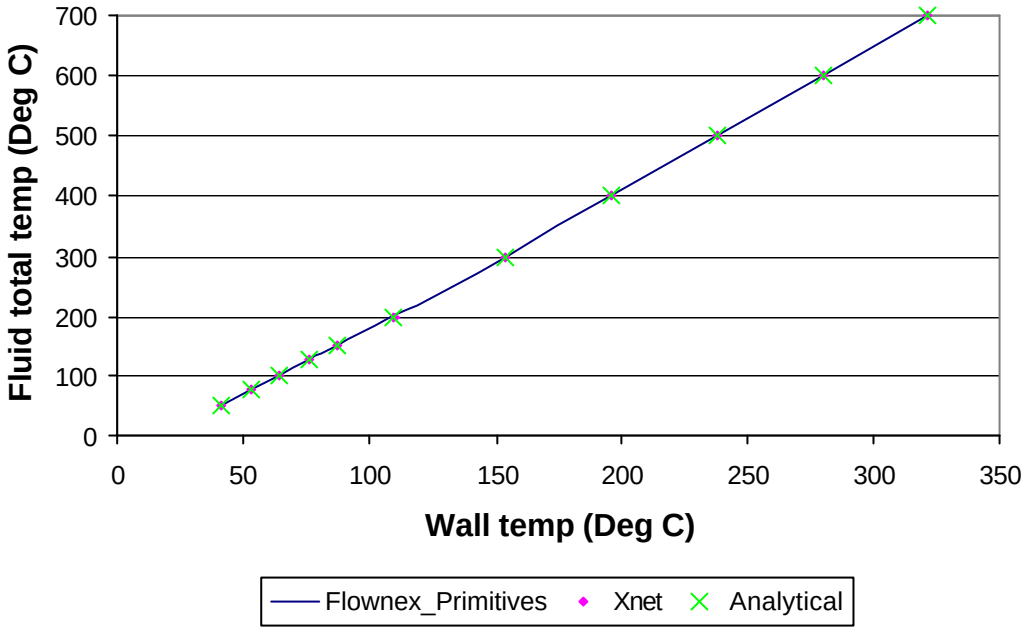


Figure 4.26: Test case 4 steady-state results.

4.4.3 Combinations of fluid flow and heat transfer

To further the comparison of basic fluid flow and heat transfer between Xnet and Flownex, a few other test cases will be presented. These cases combine the effect of fluid flow and heat transfer and will involve of the sudden closure of a valve at the end of a pipe and pressure equalisation of tanks.

Sudden valve closure at the end of a pipe

The model used here to compare Xnet to Flownex comprises of a single flow element, subdivided into 20 increments (SUBDIVIDEDPIPE) with fixed inlet conditions upstream, i.e. pressure and fluid temperature, and fixed pressure downstream. The flow was then solved to steady-state, after which the pressure at the downstream node (end of the pipe) was solved to simulate a sudden closure of a control valve downstream. Only the transient behaviour of the fluid in the pipe was investigated. The variables that will be shown graphically will be the element mass flow, as well as the downstream node pressure and temperature. The inputs for this case is given in Table 4.6

Variable	*Value(s)*
Fluid	Helium
L	100.0 [m]
$\varnothing$	0.10 [m]
p_i	300 [kPa]
T_i	15 [ºC]
p_o	100 [kPa]
e	30 [µm]
A_H	3.142 [m^2]

Table 4.6: Inputs for the sudden closure of a valve at the end of a pipe.

Figure 4.27 to Figure 4.29 show the results obtained for pressure, temperature and mass flow respectively for the sudden closure of a valve at the end of a pipe element. Figure 4.27 shows that the outlet pressure rises suddenly and then oscillates around the inlet pressure value until the steady-state conditions, which are the same as the inlet pressure, are reached. Figure 4.28 shows that the outlet temperature rises and oscillates as a result of the variation in the pressure at the outlet of the pipe.

The variation of the mass flow in the pipe is shown in Figure 4.29. The figure shows that the mass flow varies around zero and gradually reaches a steady-state condition of zero – since there is no flow anymore.

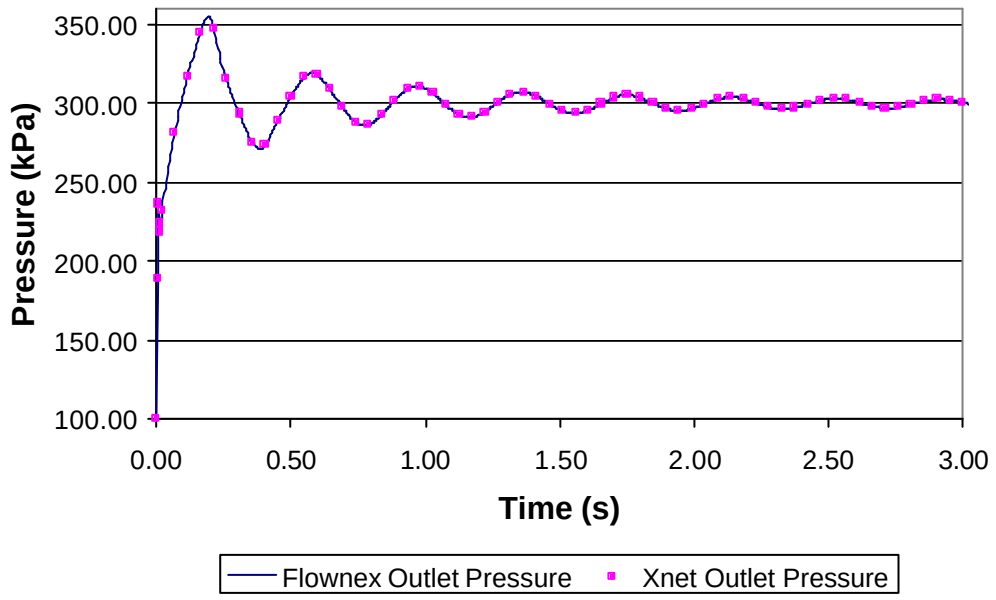


Figure 4.27: Pressure results obtained from Flownex and Xnet for a transient subdivided pipe.

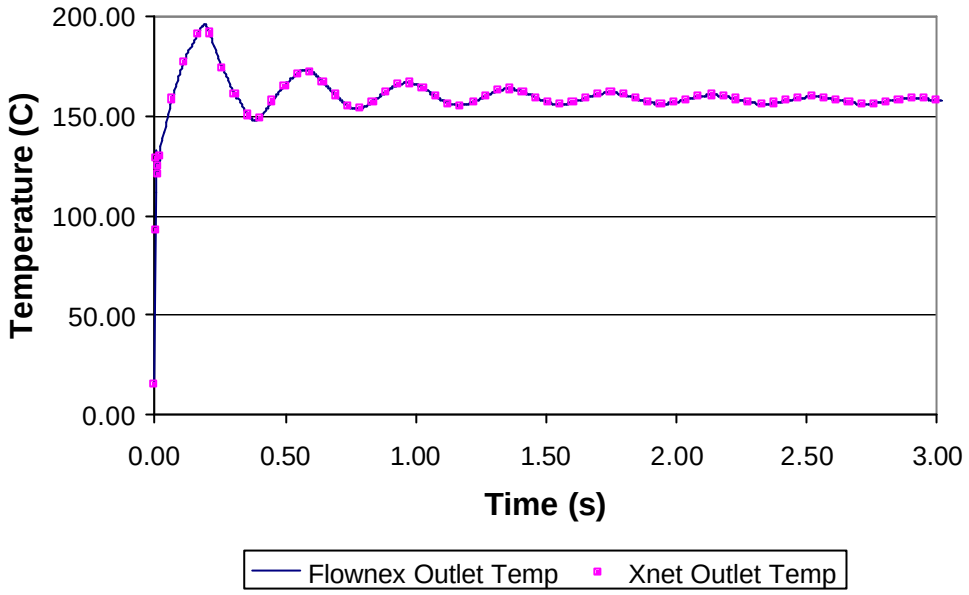


Figure 4.28: Temperature results obtained from Flownex and Xnet for a transient subdivided pipe.

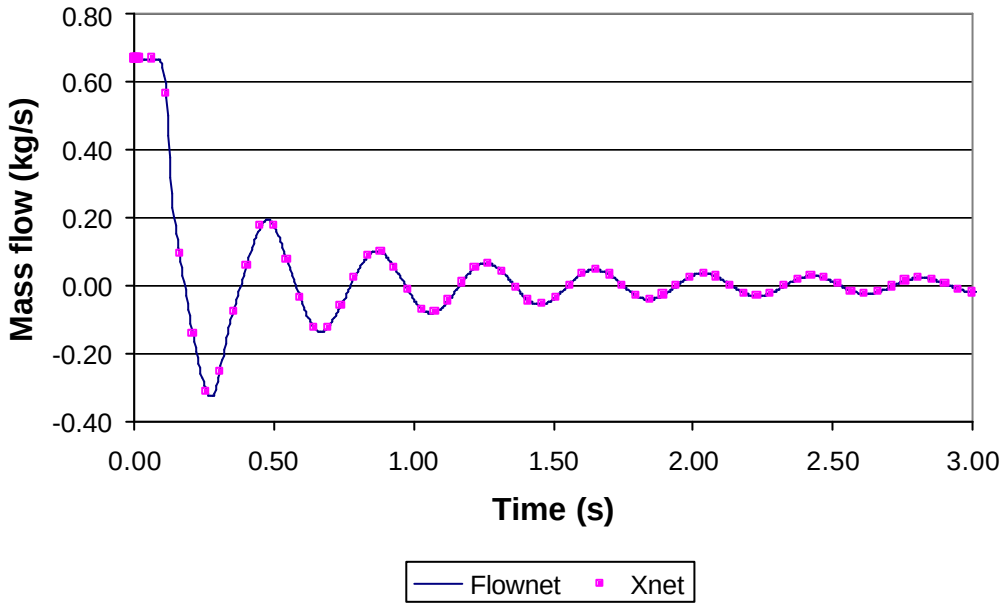


Figure 4.29: Mass flow results obtained from Flownex and Xnet for a transient subdivided pipe.

The small differences in the temperatures, pressures and mass flow for the subdivided pipe simulated in Xnet, in comparison to Flownex, shows that both Flownex and Xnet are capable of simulating very fast transients in pipes.

Pressure equalisation in tanks

To do an investigation into control volume conservation, a simple case of pressure equalisation was done between four tanks. The tanks were at the same elevation since helium is used, with each tank at a different pressure and temperature. The tanks will be represented as a node with a specified volume. Each node's initial values were specified at the start of the solution, after which the network was solved. The variables that will be shown graphically will be the node pressures and temperatures. The test case is schematically shown in Figure 4.30.

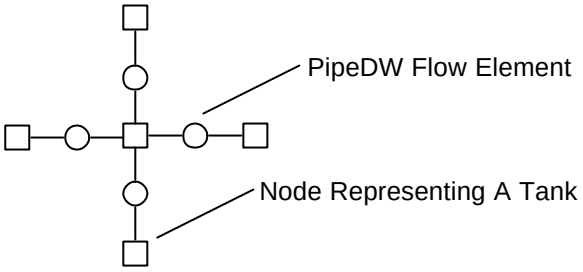


Figure 4.30: Schematic representation of the test case for pressure equalisation.

The pipe elements that connect the four tanks are 10 metres long and have a diameter of 0.10 metres each. The boundary conditions applicable to each tank are shown in Table 4.7

	TANK			
Variable	***Tank 1***	***Tank 2***	***Tank 3***	***Tank 4***
Tank Volume	10 [m^3]	10 [m^3]	10 [m^3]	10 [m^3]
Initial Temperature	90 [°C]	50 [°C]	120 [°C]	50 [°C]
Initial Pressure	100 [kPa]	80 [kPa]	85 [kPa]	65 [kPa]

Table 4.7: Boundary conditions for the pressure equalisation test case.

Figure 4.31 and Figure 4.32 show the pressure and temperature results obtained for the comparison of the pressure equalisation test between Xnet and Flownex. Figure 4.31 shows that the pressure of each tank gradually equalises to a steady-state value. The values predicted by Xnet also shows good agreement with that of Flownex.

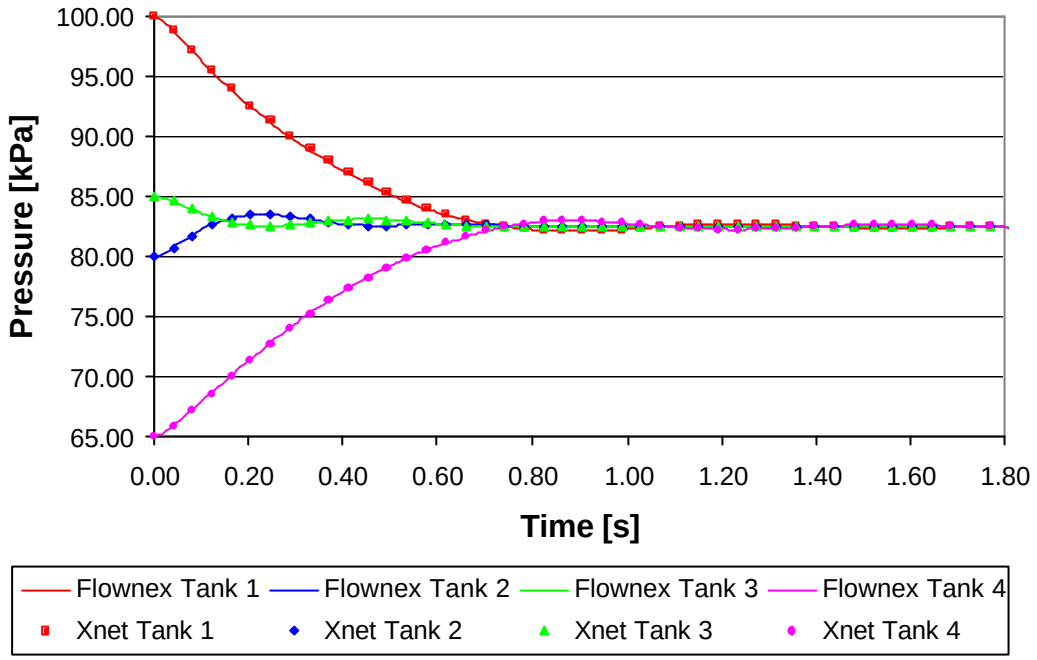


Figure 4.31: Pressure comparison between Flownex and Xnet during the pressure equalization in 4 tanks at different initial pressures and temperatures.

Figure 4.32 shows that the temperature of each tank. The value of the temperature of each tank either rises or lowers gradually as the pressure in the specific tank rises or lowers during pressure equalisation. The values predicted by Xnet also show good agreement with that of Flownex.

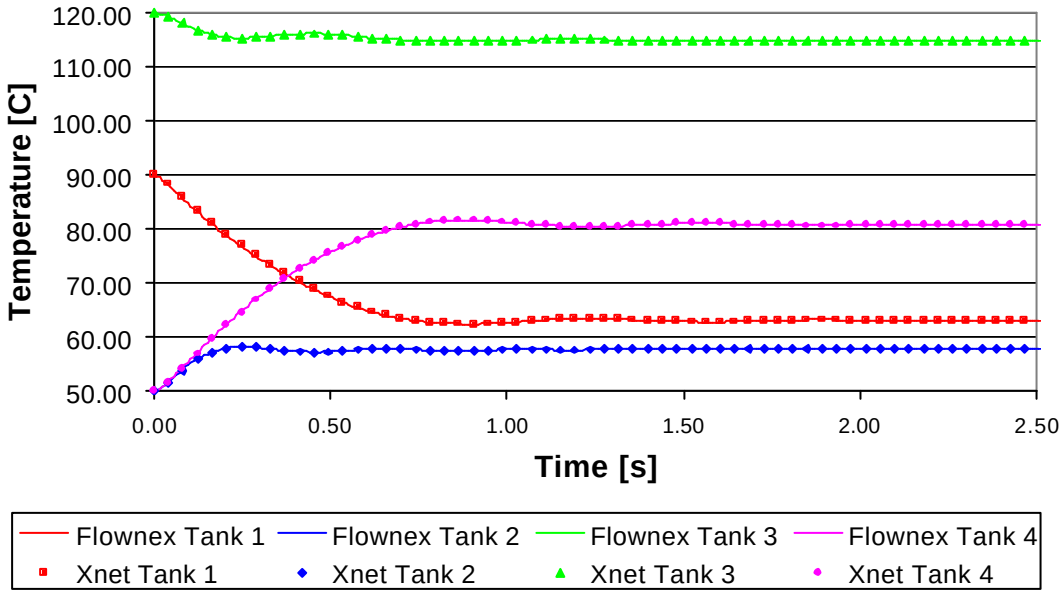


Figure 4.32: Temperature comparison between Flownex and Xnet during the pressure equalisation in 4 tanks at different initial pressures and temperatures.

4.5 Summary

In this chapter a generalised thermal-fluid network code was discussed. A brief discussion of the element types that are available in Xnet was presented and the differences between the elements where shown. The applicable elements needed for the modelling of heat exchangers in Xnet were compared to another thermal-fluid network code, namely Flownex which uses the IPCM. The results of the cases that were used to compare Flownex and Xnet have shown good comparison and it can therefore be concluded that the governing equations are correctly implemented in both codes. Xnet will now be used to model more complex coupled heat exchange networks by applying it to parallel flow and counter flow heat exchangers in the following chapter. This will be done to investigate the effect of the level of discretisation on the heat exchanger simulation results and will be applied to two different network topologies.

CHAPTER 5 - MODELLING OF PARALLEL FLOW AND COUNTER FLOW HEAT EXCHANGERS

5.1 Introduction

In this chapter an annular type tube-in-tube parallel flow and counter flow heat exchanger will be used to investigate the effect of different network topologies on the accuracy of the heat exchanger results. The reason for this choice is the relative simplicity of these types of heat exchangers. The purpose of this specific study will be to determine what type of network topology would yield the most accurate results for a certain level of discretisation.

In order to determine this, the two network codes Xnet and Flownex will be used together with the Engineering Equation Solver (EES) (F-Chart Software, 2003). The two network topologies used by Flownex and Xnet in primitive element networks will be implemented in EES as an additional benchmark. The results of the primitive element models in Flownex and Xnet will then be compared to the models in EES and a composite element in Flownex, namely the RX element.

In the Section 5.2 a brief introduction to parallel flow (PF) and counter flow (CF) heat exchangers will be given. In Section 5.3 a brief description on the Flownex RX element will be given and it will be described how primitive element models of heat exchangers are constructed in Xnet and Flownex. The results for the different types of network topologies will then be presented in Section 5.4. Section 5.5 will give a comparative study between primitive element models in Flownex and Xnet, as well as the composite RX element in Flownex.

5.2 Parallel flow and counter flow heat exchangers

Although counter flow heat exchangers have a simple layout, this type of heat exchanger is quite generally used in the industry. An example is the recuperator of the PBMR cycle. Parallel flow heat exchangers, on the other hand, are seldom used due to the fact that they are less efficient than counter flow heat exchangers. However, when the flow in one of the flow passages of the counter flow heat exchanger reverses, for instance in case of an accident, the heat exchanger becomes a parallel flow heat exchanger. Therefore, although parallel flow heat exchangers are seldom used as such, it is important to accurately model parallel flow heat exchanger performance. In parallel flow heat exchangers both the hot and cold streams enter the heat exchanger at the same end and travel to the opposite end in parallel streams. Energy is transferred along the length from the hot to the cold fluid so that the outlet temperatures asymptotically approach one another. In the counter flow heat exchanger the two streams enter at opposite ends of the heat exchanger and flow in opposite directions. Temperatures within the two streams tend to approach one another in a nearly linearly fashion resulting in a much more uniform heating pattern.

Figure 5.1 represents the axial temperature profiles for each type of configuration. The parallel flow configuration results in rapid initial rates of heat exchange but the rates rapidly decrease as the temperatures of the two streams approach one another. Counter flow provides relatively uniform temperature differences and, consequently, leads toward relatively uniform heat rates throughout the length of the unit.

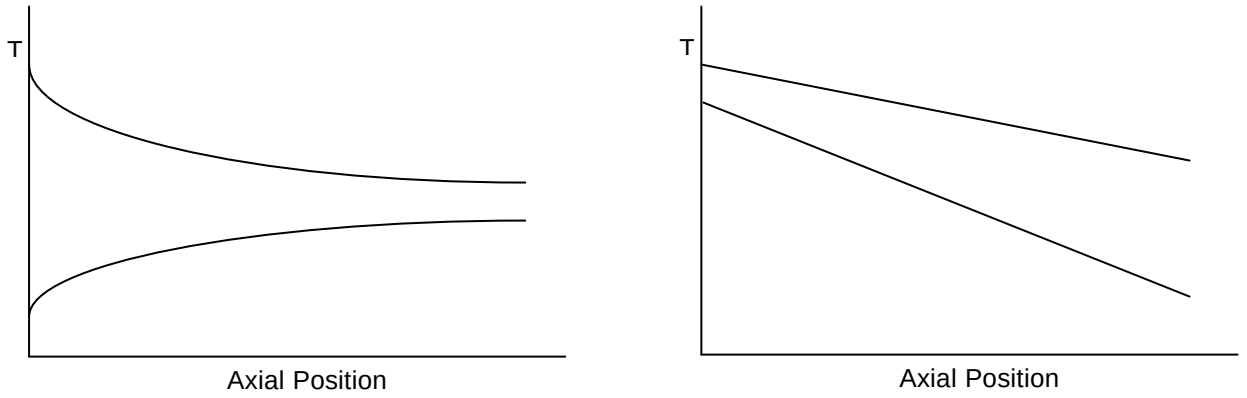


Figure 5.1: Axial temperature distribution for a PF (left) and CF (right) heat exchanger.

Since parallel flow and counter flow heat exchangers are the most straightforward type of heat exchanger available, this type of heat exchanger will be used to investigate the effect of the type of network topology on accuracy.

The models that will be used for the comparison between Xnet and Flownex's simulation will be discussed next.

5.3 Heat exchanger simulation models

5.3.1 Flownex RX element

The Flownex RX element is a composite element used for the modelling of counter flow and parallel flow heat exchangers in Flownex. The RX element in the Flownex Graphical User Interface (GUI) is depicted on systems level as two associated elements as shown in Figure 5.2.

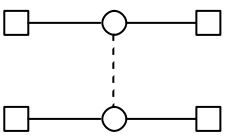


Figure 5.2: Recuperator depicted as a pair of associated elements at systems level.

Although the RX element is shown as a single element in the GUI (Figure 5.2), it is internally discretised in a number of increments as shown in Figure 5.3. Note the way in which the wall nodes are placed opposite the centres of the flow elements. Other network topologies are also possible and later in this chapter the different network topologies will be compared with each other to identify the most suitable one.

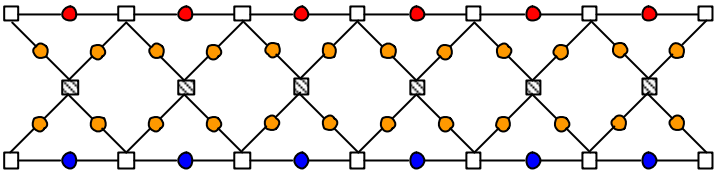


Figure 5.3: Discretisation of the recuperator.

The way in which the heat transfer in the RX element is calculated will be discussed later in this chapter (Section 5.3.2), since it is similar to the primitive networks heat transfer calculations. A complete description of the RX element in Flownex can be found in the Flownex 6.4 User Manual (Coetzee *et al*., 2003).

5.3.2 Flownex and Xnet primitive heat exchange models

The primitive elements used in Flownex and in Xnet are the basic elements used to construct discretised thermal-fluid components such as heat exchangers. Heat transfer between two fluids across a solid wall is schematically represented by Figure 5.4. The heat transfer is accomplished through convective heat transfer from the fluid to the solid wall or vice versa whichever one is hotter and the heat transfer in the axial direction is neglected.

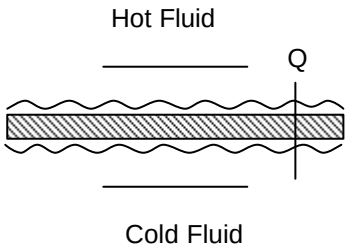


Figure 5.4: Schematic representation of heat transfer across a solid wall.

Two different network topologies can be used when heat exchangers are modelled using the network approach. The two different network topologies refer to the manner in which the convection elements are connected to the fluid flow elements. The convection nodes may either be connected to a flow node or a flow element, as was pointed out in Section 4.3.5. To avoid confusion, the connection to a flow element will be termed a *node-element connection* and a connection to a flow node will be termed a *node-node connection*. These two types of network topologies are shown in Figure 5.5 and Figure 5.6.

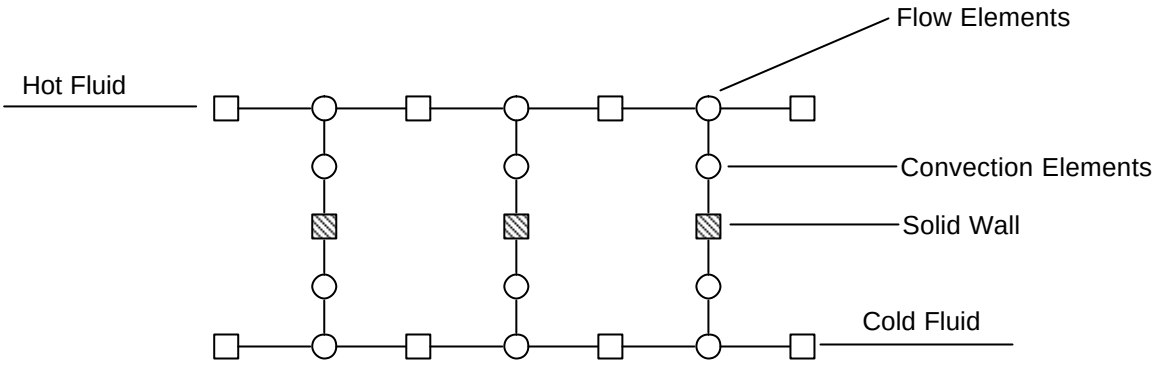


Figure 5.5: Schematic representation of the network topology where the convection elements are connected to the flow elements (node-element connection).

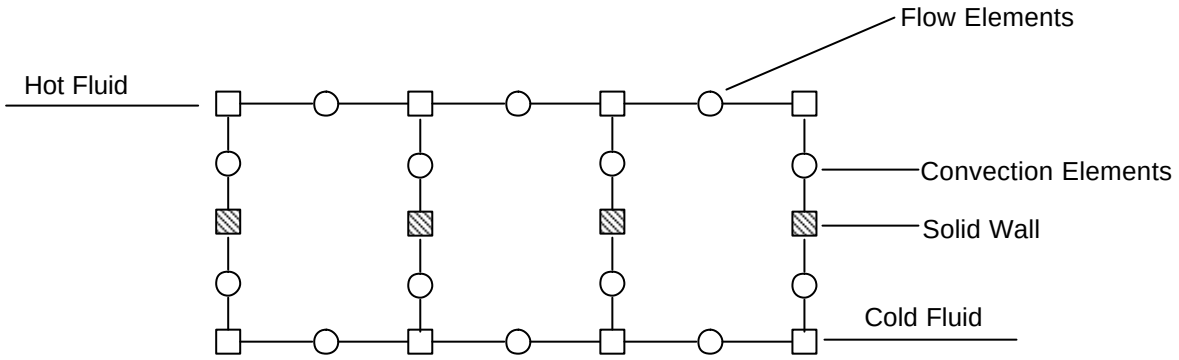


Figure 5.6: Schematic representation of the network topology where the convection elements are connected to the flow nodes (node-node connection).

When the convection elements are connected in the GUI to the flow elements as in Figure 5.5, the network topology is changed in the solver as shown in Figure 5.7. This is done because elements cannot be connected directly to each other in the solver. The single convection element between the flow elements and solid nodes shown in Figure 5.6 are now represented by two convection elements on either side of each flow element at the upstream and downstream node. These two convection elements actually represent only one heat transfer coefficient.

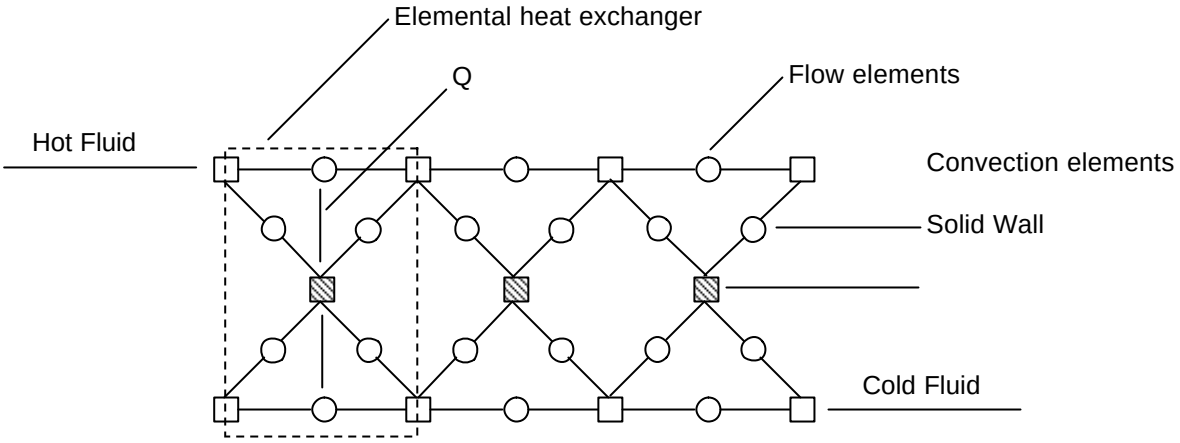


Figure 5.7: Flownex node-element representation of heat transfer across a solid wall.

Consider the network topology for an elemental heat exchanger depicted in Figure 5.8:

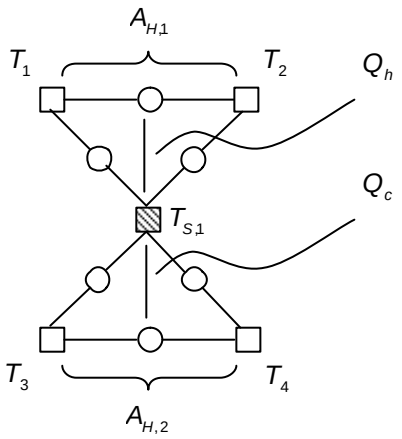


Figure 5.8 Elemental heat exchanger.

The heat transferred from the hot fluid to the solid node and the solid node to the cold fluid, Q_h,Q_c, are calculated by the following equation:

$$Q_h = h_1 A_{H,1}\left(\frac{T_1 + T_2}{2} - T_{S,1}\right) \tag{5.1}$$

and

$$Q_c = h_2 A_{H,2}\left(T_{S,1} - \frac{T_3 + T_4}{2}\right) \tag{5.2}$$

where

h_1 and h_2 is the surface heat transfer coefficient on the hot side and the cold side respectively,

$A_{H,1}$ and $A_{H,2}$ the heat transfer areas of the hot side and the cold side respectively and

T the respective static temperatures denoted in Figure 5.8.

The surface heat transfer coefficient, h, is calculated from the following equation derived from the definition of the Nusselt number,

$$h = \frac{N_u k}{D_H} \tag{5.3}$$

where

Nu is the Nusselt number, obtained from a semi-empiric correlation such as the Dittus Boelter equation (Holman, 1992),

k the thermal conductivity of the fluid and

D_H the hydraulic diameter of the flow conduit.

In the case were the convection elements are connected to the flow nodes, the heat transfer from the hot fluid to the first solid node and from the first solid node to the cold fluid, Q_h,Q_c, are calculated as follows,

$$Q_h = A_{H,1} h_1 \left(T_1 - T_{S1} \right) \tag{5.4}$$

and

$$Q_c = A_{H,3} h_2 \left(T_{S1} - T_3 \right) \tag{5.5}$$

where

h_1 and h_2 is the heat transfer coefficient of the hot fluid and the cold fluid respectively,

$A_{H,1}$ and $A_{H,2}$ the heat transfer areas of the hot fluid and the cold fluid respectively and

T the respective static temperatures denoted in Figure 5.9.

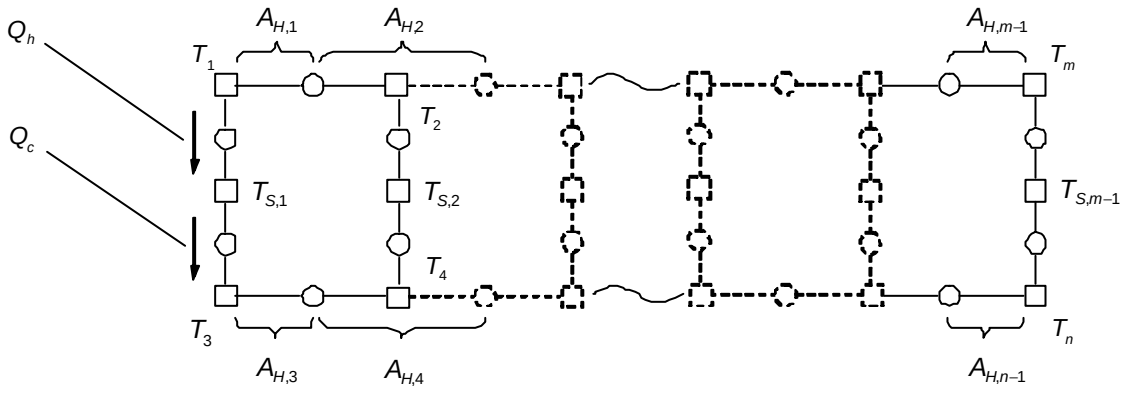


Figure 5.9: Notation for heat transfer calculations.

It should be noted that when the node-node configuration is used for heat exchangers, the temperature of a flow node at the entrance to the heat exchanger which has a convection element connected to it, must not be fixed. The reason for this is that heat is transferred from the node to the solid node by means of

convection. The temperature just after the node (for forward flow) should then be either less if energy was taken from the fluid or higher if energy was transferred to the fluid. If the temperature is fixed, the inlet temperature of the fluid is assigned this fixed value and therefore conservation of energy is not satisfied. The temperature should then be defined just before the node since there is heat transfer at the node with the convection element connected at the node and consequently a lower (or higher) temperature at the entrance to the pipe due to this heat transfer.

To eliminate this problem where the temperatures are defined at the inlets to the heat exchangers, a dummy element is introduced. This dummy element is a flow element just before the inlet to the heat exchanger. It has the same hydraulic diameter, and therefore the same velocity through the element, but the length of the element is very short to ensure that the pressure drop across the element is negligible.

This will ensure that the temperature variable is defined just before the node that has a convection element connected at it. Figure 5.10 shows how the dummy elements are used in the node-node connection of the convection elements in heat exchanger networks.

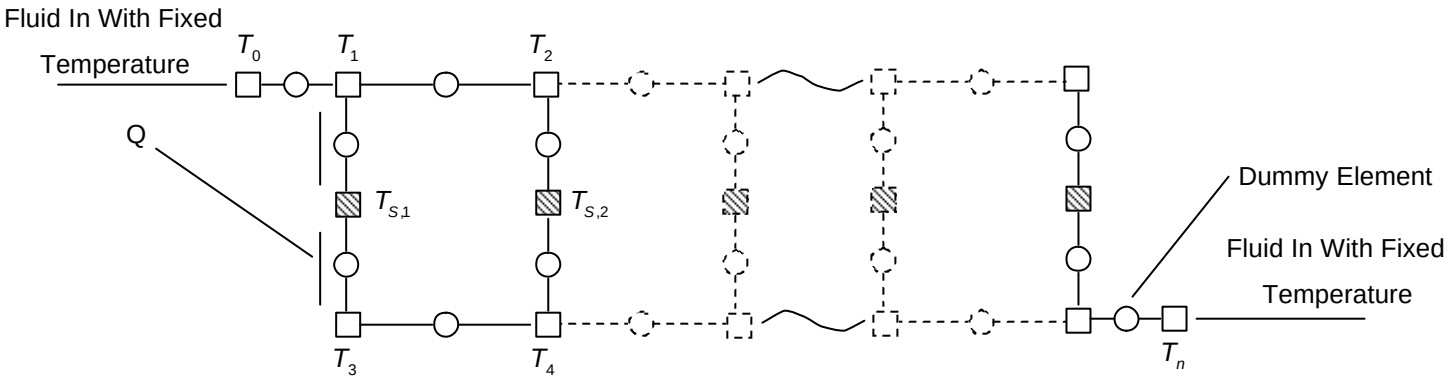


Figure 5.10: Schematic representation of the dummy elements used with the node-node connection.

These network topologies will now be implemented into EES to investigate the effect of the level of discretisation on the accuracy of the results.

5.4 Implementation of different network topologies into EES

Two different network topologies were presented in Section 5.3.2. It is necessary to investigate the effect of the network topology on the accuracy of the heat exchange results. The approach that will be followed here will be to use several levels of discretisation for each type of topology. These two schemes will then be compared to investigate the effect of the network topology on accuracy.

The two network topologies as well as the analytical effectiveness-NTU method will be implemented in EES. This implementation into EES will serve as a further independent benchmark to compare the results of Flownex and Xnet to. Since the approach used by EES to solve these schemes is very fast and simple to

programme, it eliminates the long process of assigning the individual data to elements as is the case for Flownex and Xnet. The time to determine the desired level of increments necessary will therefore be much faster in EES than in Flownex or even Xnet. The primitive element models will then only use a maximum number of 20 increments for the reasons stated above, whilst the composite RX element model will be discretised into 20 or more increments. This is done since the RX element is discretised in the solver and it is only necessary to enter the element data once. EES will be used to determine the maximum number of increments needed. To eliminate possible errors in the modelling of the flow equations (conservation of momentum and continuity) in EES, a fixed mass flow will be specified in order to calculate the heat transfer coefficients.

The analytical Effectiveness-NTU method for heat exchangers implemented in EES are described in detail in Chapter 2 as well as by Incropera and De Witt (1996), and will not be repeated here.

5.4.1 Physical heat exchanger model

The heat exchanger that will be used here, as illustrated in Figure 5.11, has the following specification.

Heat exchanger type and working fluid	Pipe-In-Pipe - Helium
Flow configuration	Parallel flow and Counter flow
Total heat transfer area, A_H	10 [m^2]
Area ratio = $\left(\frac{A_P}{A_S}\right)$	1.25 [-]
Metal heat capacity	71 [kJ/K]
Total length	10 [m]
Primary side hydraulic diameter	0.10 [m]
Secondary side hydraulic diameter	0.09 [m]

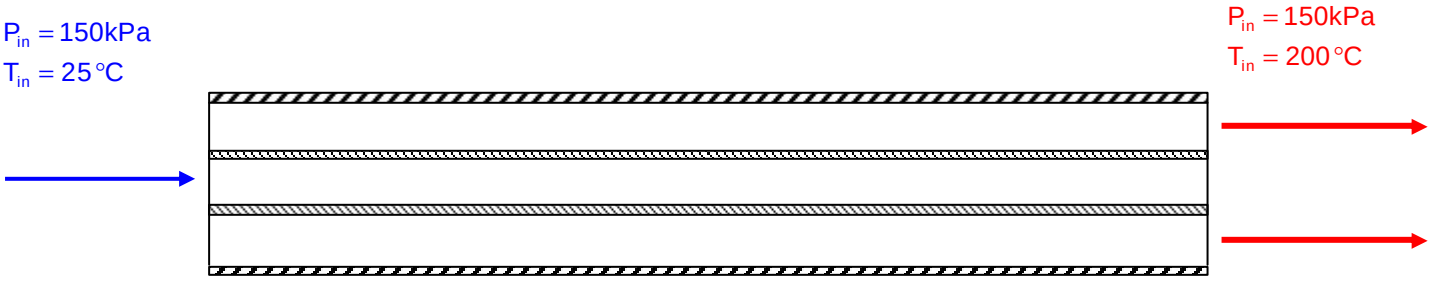


Figure 5.11: Schematic representation of the heat exchanger inputs for parallel flow configuration.

The heat exchanger will be modelled for 3 different mass flow ratios, namely the ratio of the mass flow of the hot stream to the mass flow of the cold stream. These ratios will represent a number of situations that might occur in industrial heat exchange applications. These ratios will be 0.10, 0.50 and 1.00.

The results that were obtained for the above model (parallel and counter flow configuration) with the different network topologies and mass flow ratios are presented and discussed below

5.4.2 Results and discussion

The different network topologies implemented in EES were compared to the primitive element models implemented in Flownex, and the composite RX element in Flownex. The reason why the network topologies in EES will not be compared to Xnet, is that a fixed mass flow rate boundary condition cannot be specified in Xnet, but only a fixed pressure boundary condition. The comparison with Xnet will not be meaningful at this point due to this fact. A steady-state and transient comparative study will nevertheless, be performed in Section 5.5 to show the results obtained in Xnet and Flownex for a given pressure drop.

Figure 5.12 to Figure 5.17 shows the results obtained from EES, Flownex's primitive element model, as well as the composite RX element in Flownex for the outlet total temperature against the number of increments. The results for the parallel flow heat exchanger show that whether the node-element or the node-node connection is used, more increments are necessary to reach the analytical effectiveness-NTU result as one increases the mass flow ratio. Even though more increments are necessary as the mass flow ratio is increased, the node-element connection produces very accurate results with only 5 increments, whereas the node-node connection requires a great amount of increments to reach an acceptable result. To qualitative compare the results, a percentage point error is defined as follows:

$$\%error = \frac{\left| y_{i\,Flownex} - y_{i\,Analytical} \right|}{y_{i\,Analytical}} \times 100 \qquad (5.6)$$

where

$y_{i\,Flownex}$ is the specific variable in Flownex such as the outlet total temperature and

$y_{i\,Analytical}$ is the specific variable of the analytical solution (e-NTU).

Table 5.1 shows the percentage point error from the analytical effectiveness-NTU solution for 5 increments for both network topologies.

$\dot{m}$ ***ratio***	**PERCENTAGE ERROR FOR 5 INCREMENTS**			
	Parallel flow HX		***Counter flow HX***	
	Node-Element	***Node-Node***	***Node-Element***	***Node-Node***
1.00	0.40	3.50	0.17	7.38
0.50	0.35	3.10	0.12	6.19
0.10	0.10	1.69	0.08	1.86

Table 5.1: Percentage error from the analytical solution for the different discretisation schemes for both configurations of the heat exchanger and 3 different mass flow ratios at 5 increments.

The results for the counter flow heat exchanger also show that whether the node-element or the node-node connection is used, more increments are necessary to reach the analytical Effectiveness–NTU result as one increases the mass flow ratio. Even though more increments are necessary as one increases the mass flow ratio, the node-element connection produces very accurate results with only 5 increments, whereas the node-node connection requires a large number of increments to reach an acceptable result. Table 5.2 shows the amount of increments necessary to achieve an error of less than 0.50% for the node-node connection. The higher the mass flow ratio the greater the number of increments required to achieve an acceptable error margin.

$\dot{m}$ ***ratio***	**NUMBER OF INCREMENTS**	
	Parallel flow HX	***Counter flow HX***
1.00	40	80
0.50	35	70
0.10	20	25

Table 5.2 Number of increments necessary to reach an error of less than 0.50% when a node-node connection is used for 3 different mass flow ratios.

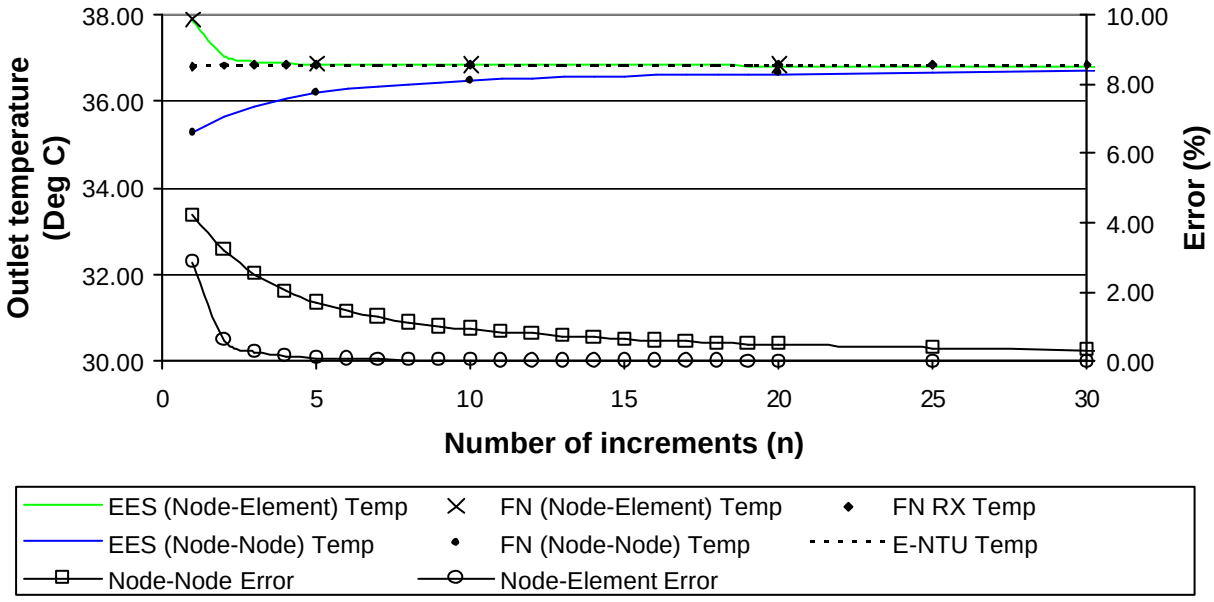


Figure 5.12: Discretisation scheme comparison results for a parallel flow heat exchanger with a mass flow ratio of 0.10.

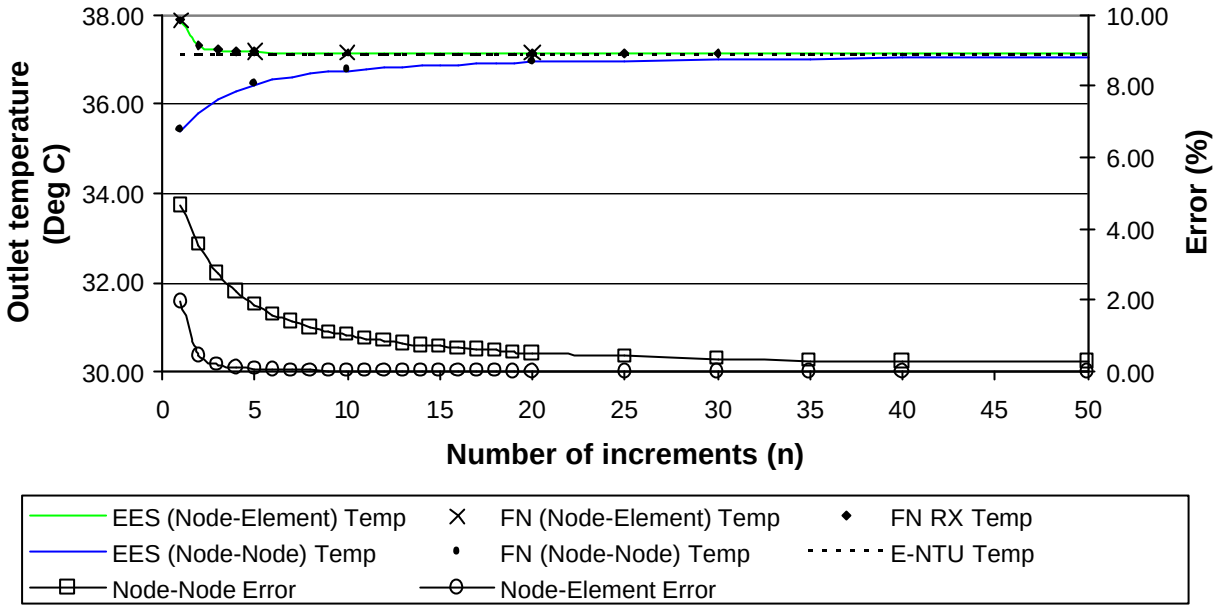


Figure 5.13: Discretisation scheme comparison results for a counter flow heat exchanger with a mass flow ratio of 0.10.

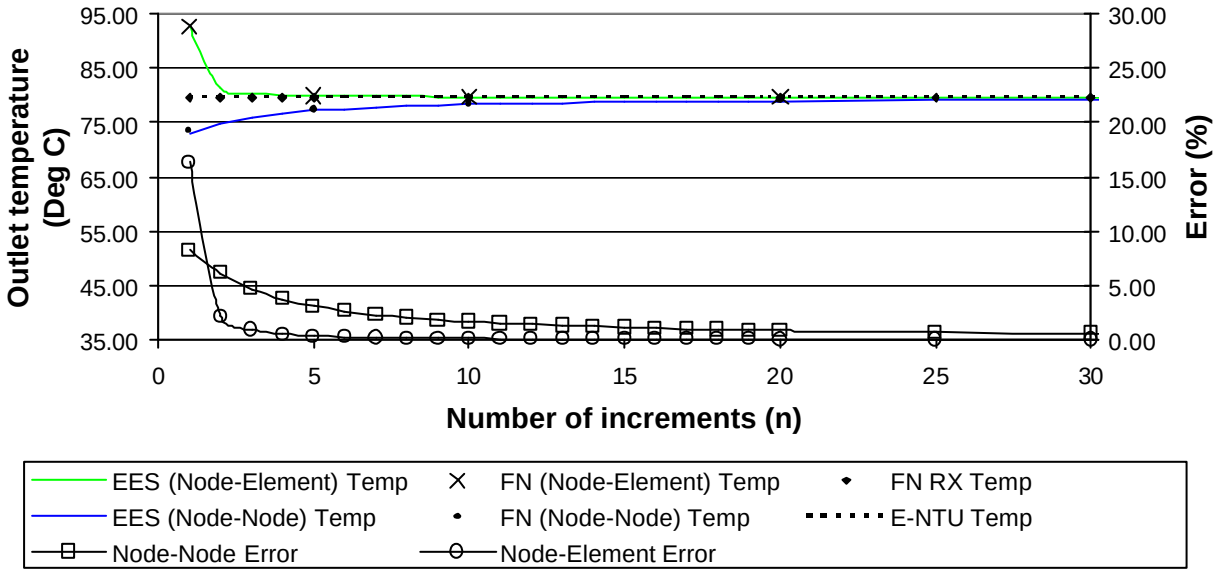


Figure 5.14: Discretisation scheme comparison results for a parallel flow heat exchanger with a mass flow ratio of 0.50.

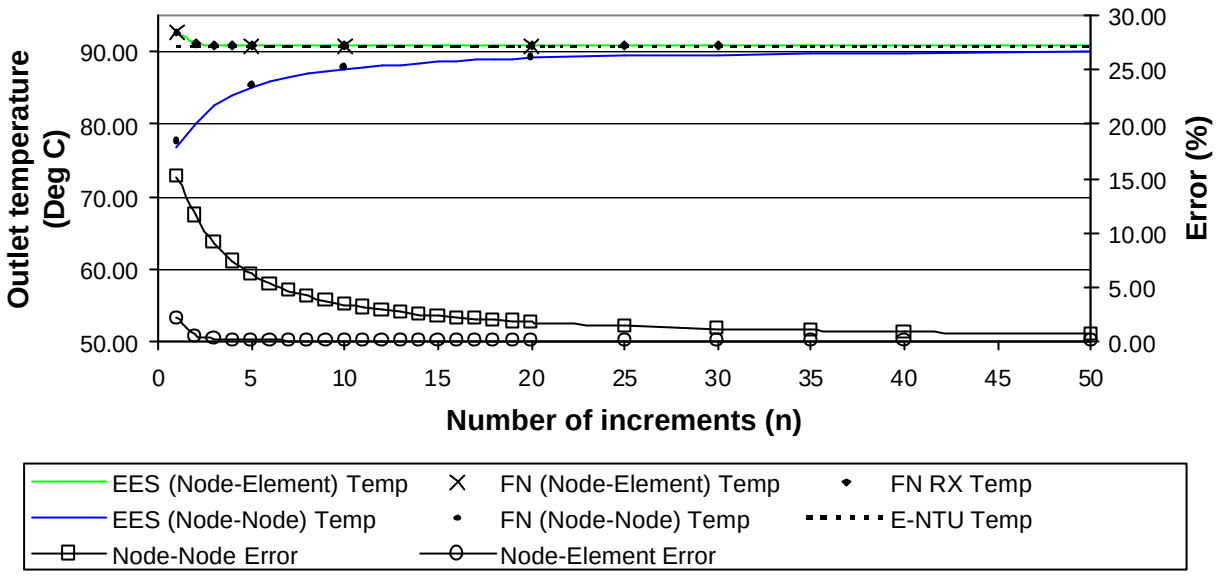


Figure 5.15: Discretisation scheme comparison results for a counter flow heat exchanger with a mass flow ratio of 0.50.

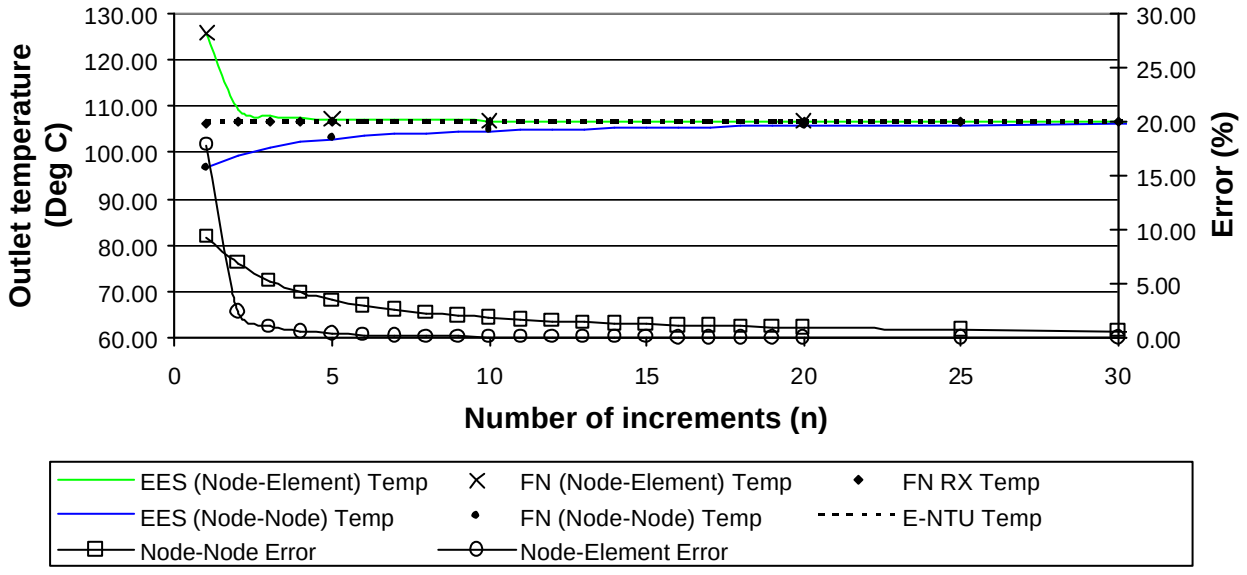


Figure 5.16: Discretisation scheme comparison results for a parallel flow heat exchanger with a mass flow ratio of 1.00.

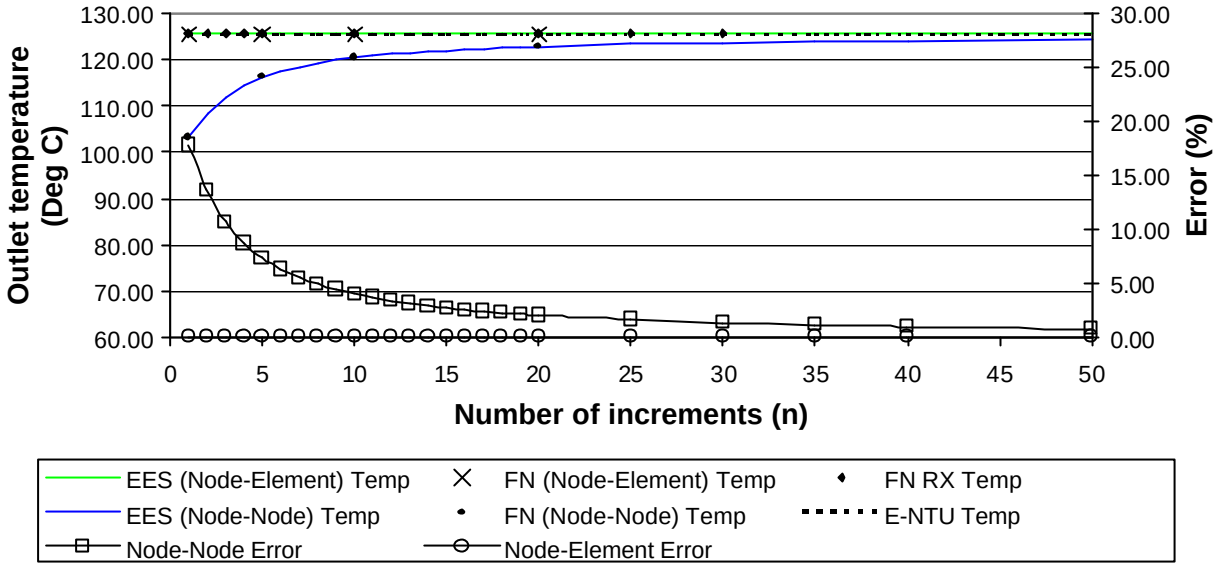


Figure 5.17: Discretisation scheme comparison results for a counter flow heat exchanger with a mass flow ratio of 1.00.

5.5 Comparative study

The purpose of this section is to compare Xnet's primitive element model, the Flownex primitive element model and the composite Flownex RX model for the case of an annular tube-in-tube heat exchanger for both steady-state and transient operating conditions. In the previous section the two different network topologies were implemented in EES and compared to Flownex. The EES model was not compared to Xnet since a fixed mass flow rate boundary condition could not be specified in Xnet, as was pointed out in Section 5.4. It should be noted that the node-element network topology will be used here to model the heat exchanger. The physical model that will be used has the same specification as the heat exchanger model in Section 5.4.1. This model will have a fixed pressure drop across both the hot and cold fluid streams of $0.20 kPa$. A complete definition of the boundary values is given in Appendix C.

The steady-state results for different levels of discretisations are presented below.

5.5.1 Steady-state results

Figure 5.18 to Figure 5.21 show the steady-state results for the parallel flow heat exchanger modeled in Flownex and Xnet for four different levels of discretisation. The graph shows that the temperature profile across the length of the heat exchanger is asymptotic, which is characteristic of a parallel flow heat exchanger. The primitive element models in Flownex and Xnet show good agreement with the results predicted by the RX element in Flownex.

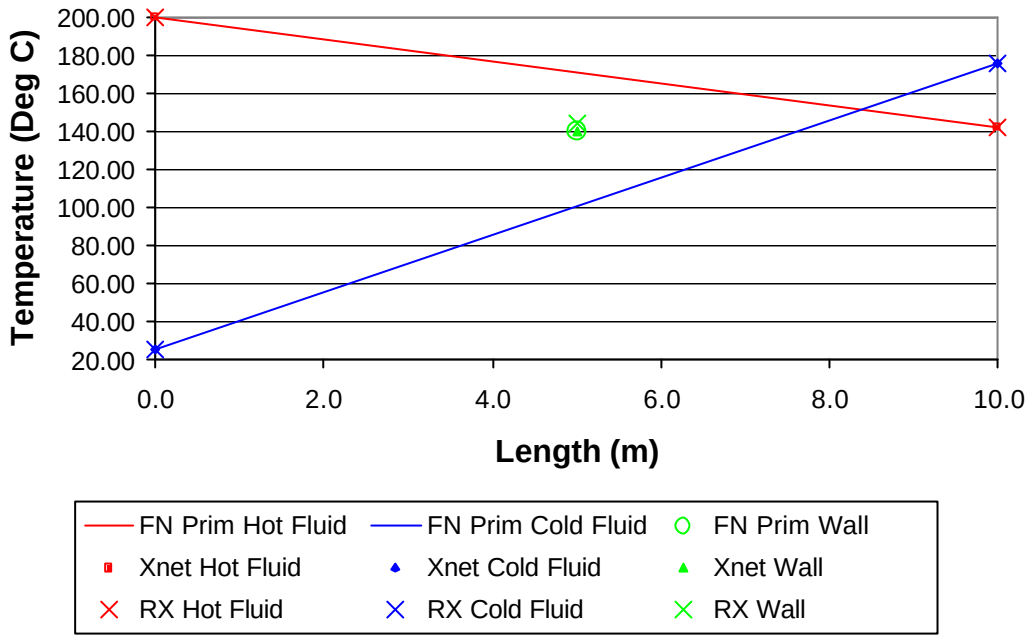


Figure 5.18: Steady-state results for a parallel flow heat exchanger with 1 increment.

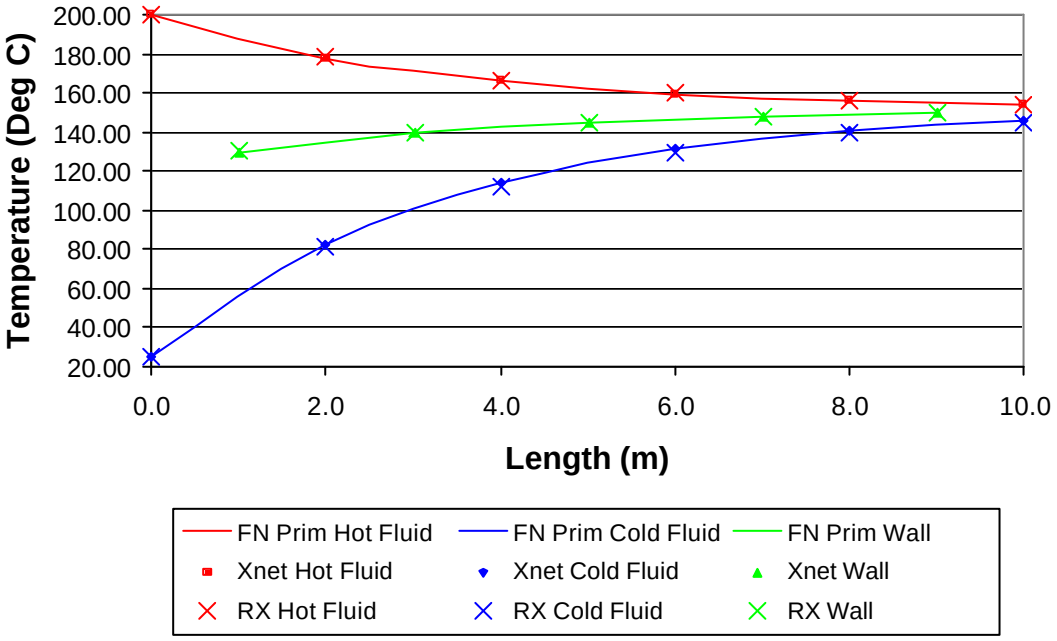


Figure 5.19: Steady-state results for a parallel flow heat exchanger with 5 increments.

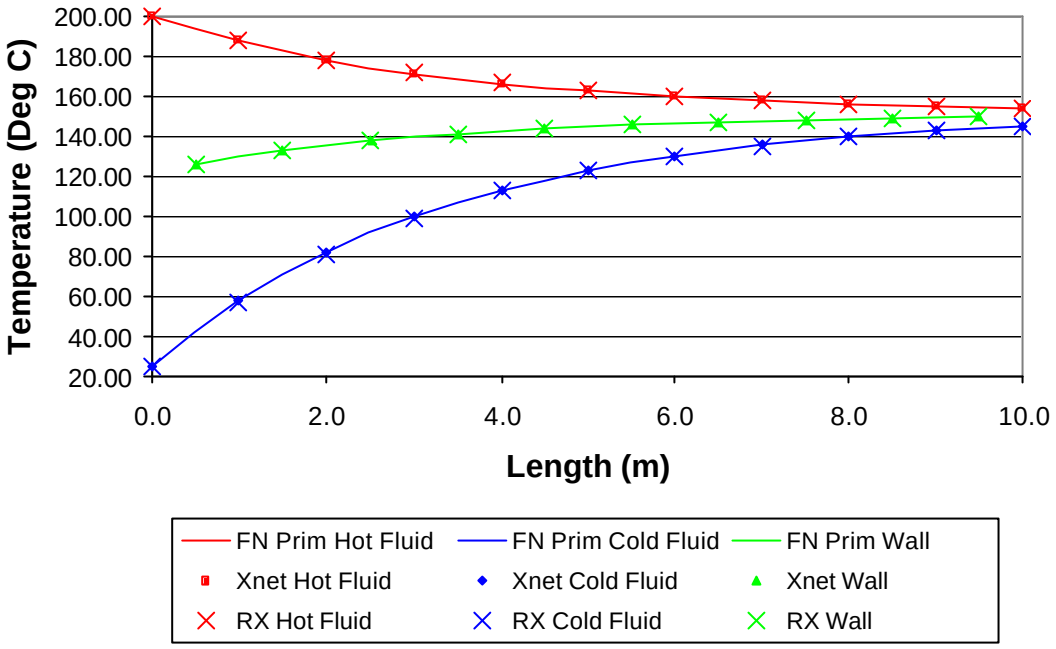


Figure 5.20: Steady-state results for a parallel flow heat exchanger with 10 increments.

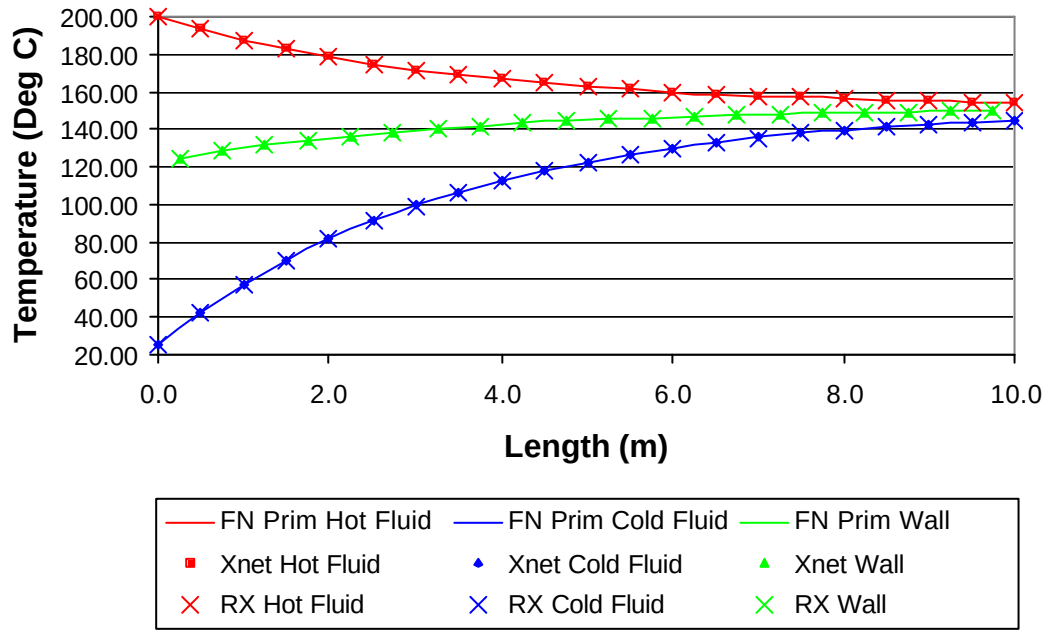


Figure 5.21: Steady-state results for a parallel flow heat exchanger with 20 increments.

Figure 5.22 to Figure 5.25 shows the steady-state results for a counter flow heat exchanger modeled in Flownex and Xnet for 4 different levels of discretisation. The graph shows that the temperature profile across the length of the heat exchanger is almost a linear profile, which is characteristic of a counter flow heat exchanger. When only 1 increment is used, the results for the primitive element models in Flownex and

Xnet show that there is no difference to the value predicted by the RX element in Flownex, as was the case of the parallel flow configuration. When more increments are used, the results of the primitive element models in Flownex and Xnet show good agreement with the results predicted by the RX element in Flownex.

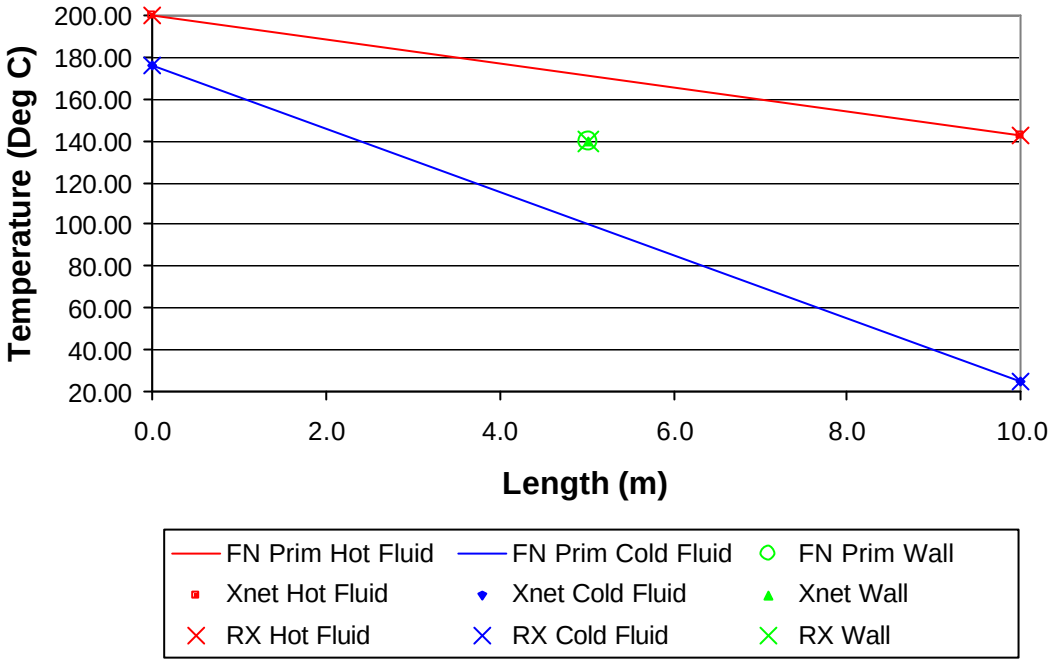


Figure 5.22: Steady-state results for a counter flow heat exchanger with 1 increment.

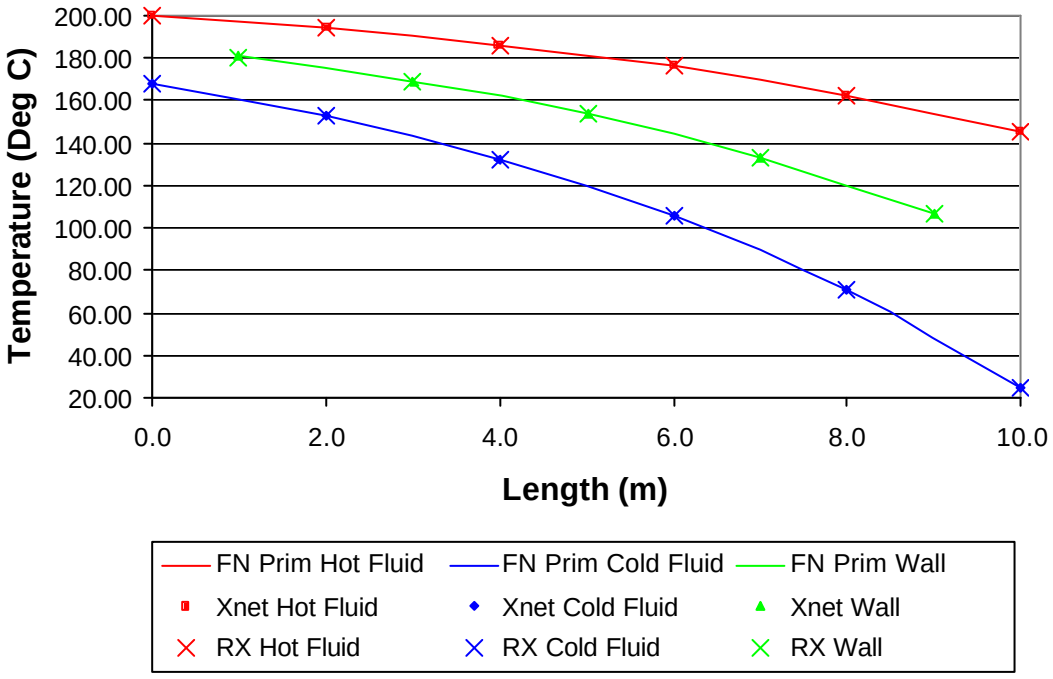


Figure 5.23: Steady-state results for a counter flow heat exchanger with 5 increments.

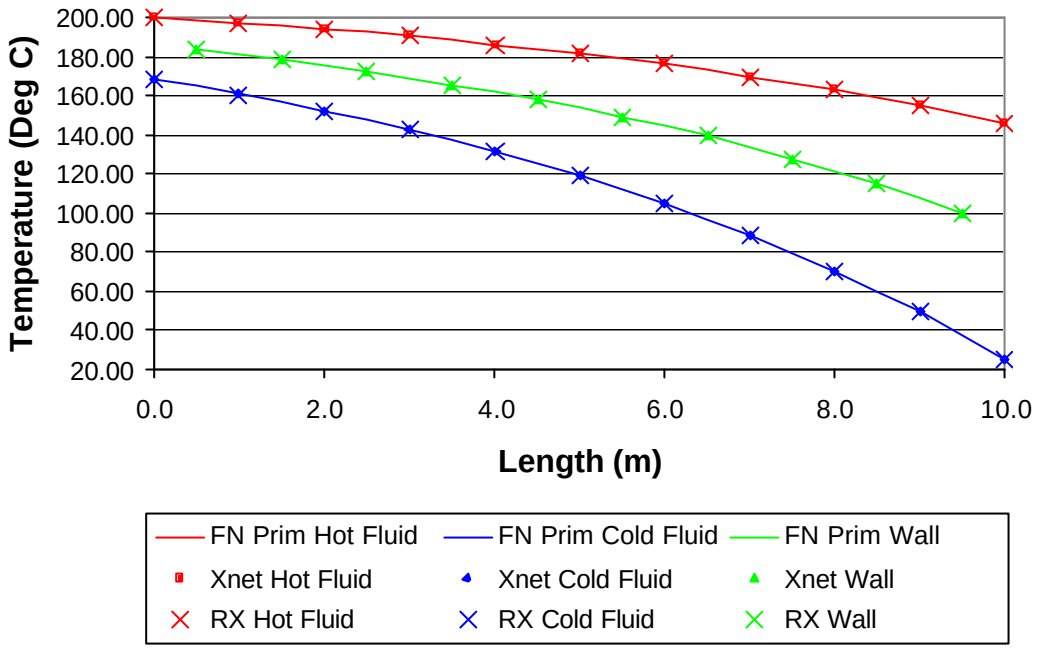


Figure 5.24: Steady-state results for a counter flow heat exchanger with 10 increments.

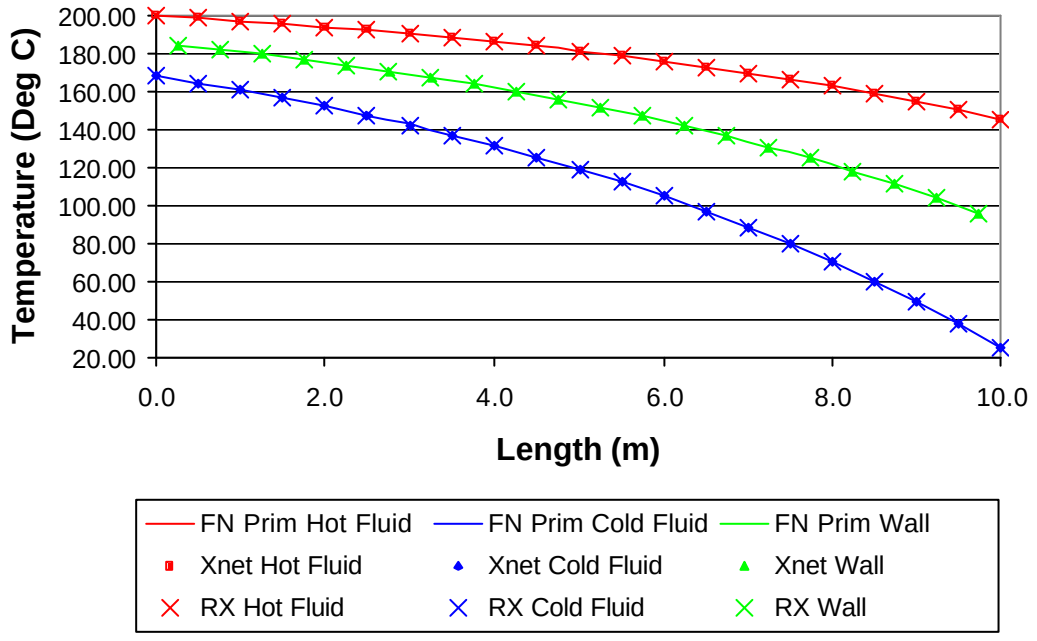


Figure 5.25: Steady-state results for a counter flow heat exchanger with 20 increments.

5.5.2 Transient results

For the transient simulations, the heat exchanger is subjected to a step increase in the hot inlet temperature of the heat exchanger. The inlet temperature is instantaneously increased from 200℃ to 300℃ from a steady-state condition. The outlet temperature is then plotted to see the effect of the step input of the hot fluid.

Figure 5.26 to Figure 5.29 show the transient results for a parallel flow heat exchanger modelled in Flownex and Xnet for 4 different levels of discretisation. With a step increase in the hot fluid's inlet temperature, there is a slight increase in the cold fluid's outlet temperature. The primitive element models in Flownex and Xnet show good agreement with the results predicted by the RX element in Flownex for the transient simulation.

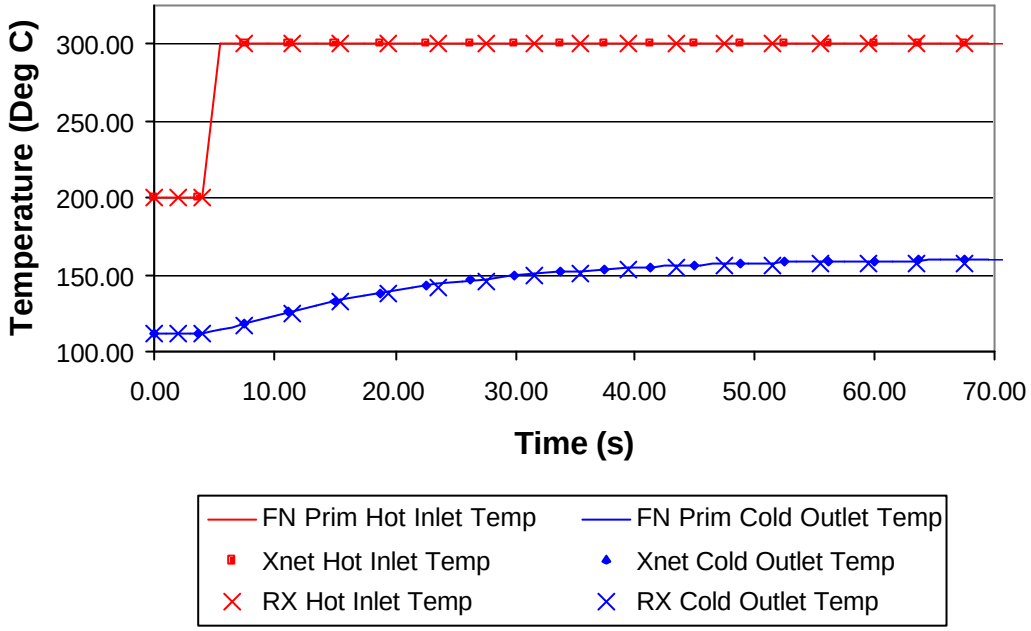


Figure 5.26: Transient results for a parallel flow heat exchanger with 1 increment.

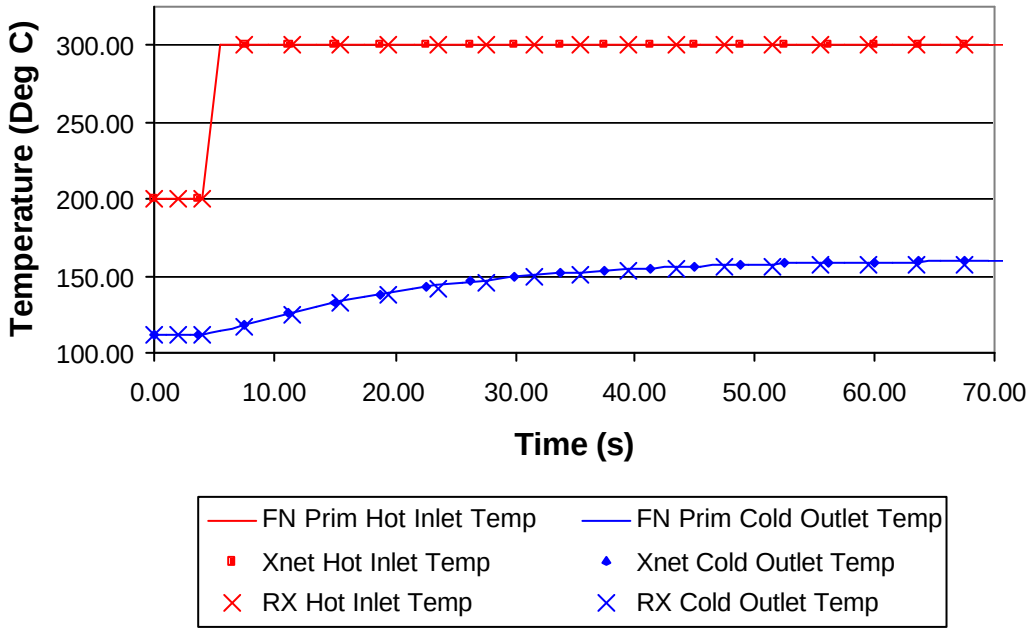


Figure 5.27: Transient results for a parallel flow heat exchanger with 5 increments.

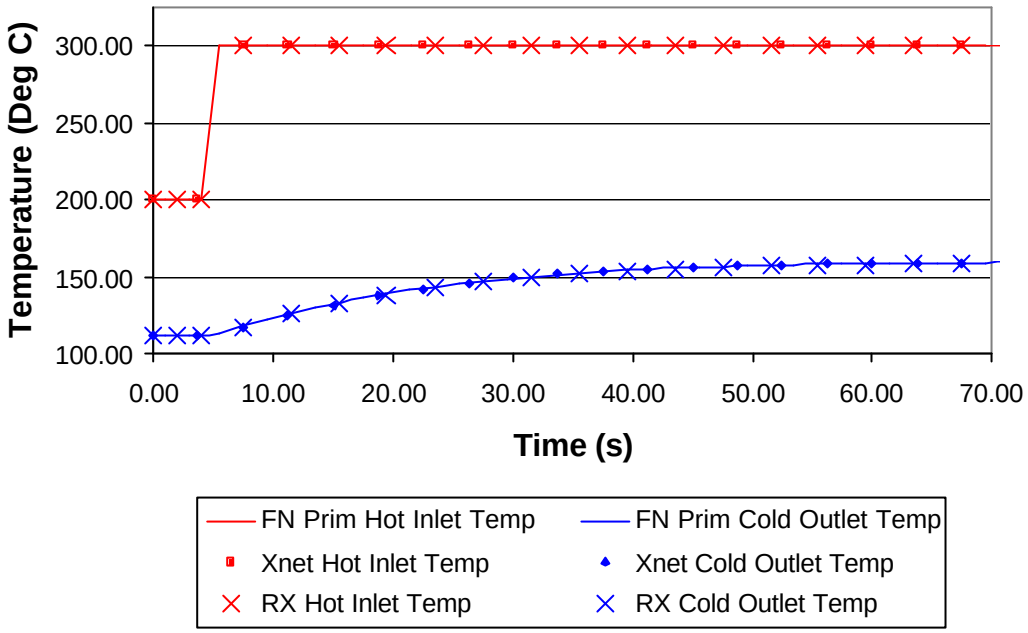


Figure 5.28: Transient results for a parallel flow heat exchanger with 10 increments.

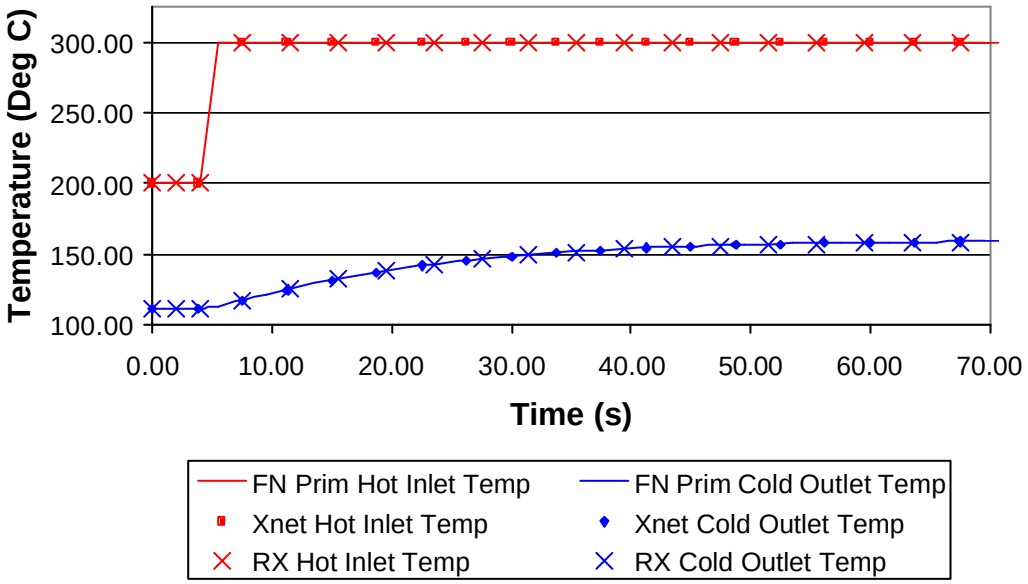


Figure 5.29: Transient results for a parallel flow heat exchanger with 20 increments.

Figure 5.30 to Figure 5.33 show the transient results for a counter flow heat exchanger modelled in Flownex and Xnet for 4 different levels of discretisation. With a step increase in the hot fluid's inlet temperature, there is a slight increase in the cold fluid's outlet temperature. The primitive element model in Flownex and Xnet shows good agreement with the results predicted by the RX element in Flownex for the transient simulation.

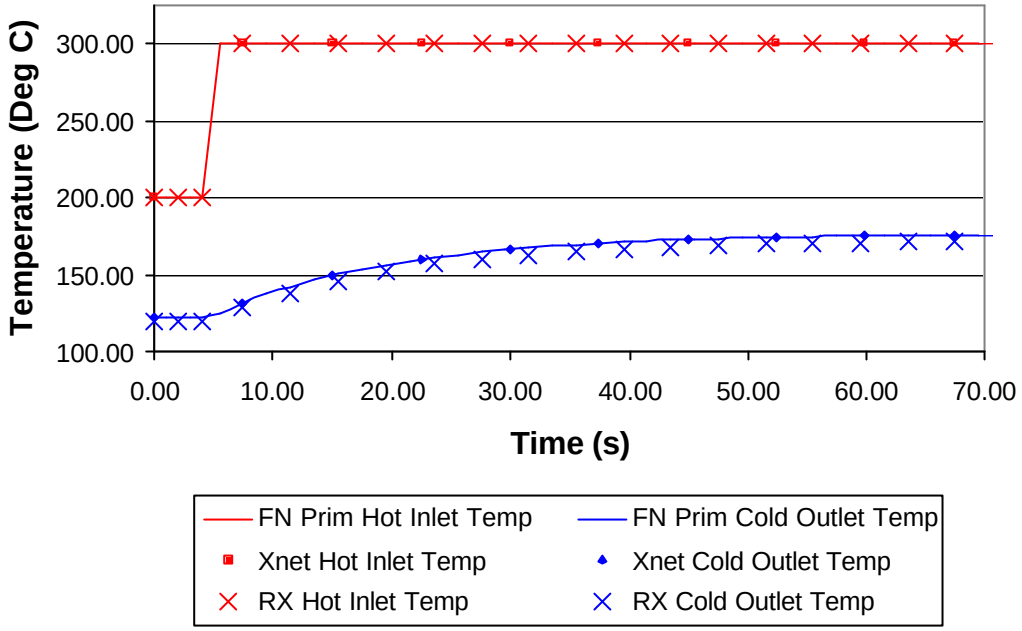


Figure 5.30 Transient results for a counter flow heat exchanger with 1 increment.

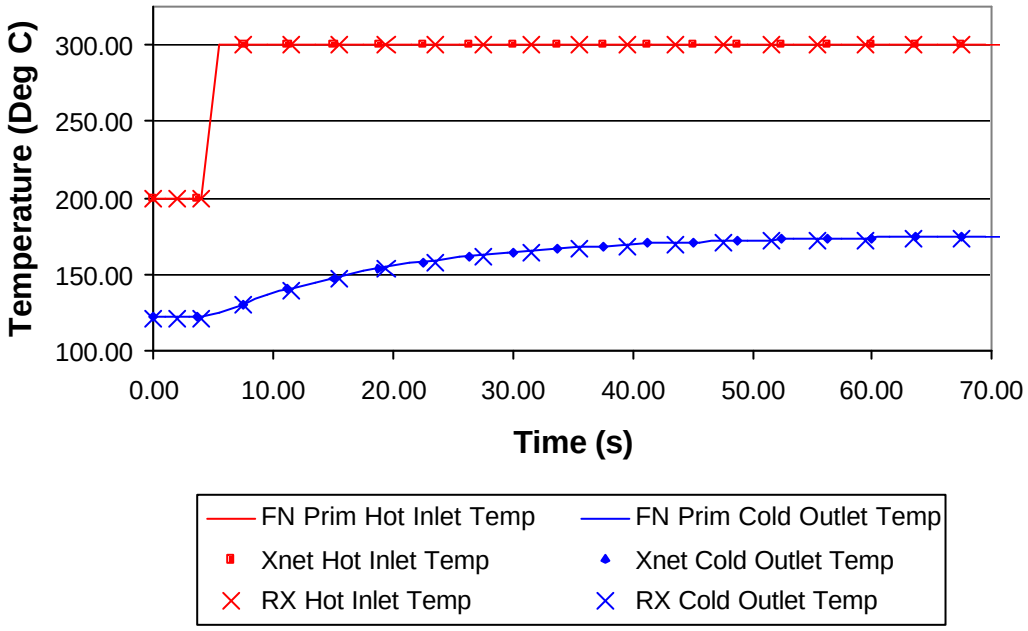


Figure 5.31: Transient results for a counter flow heat exchanger with 5 increments.

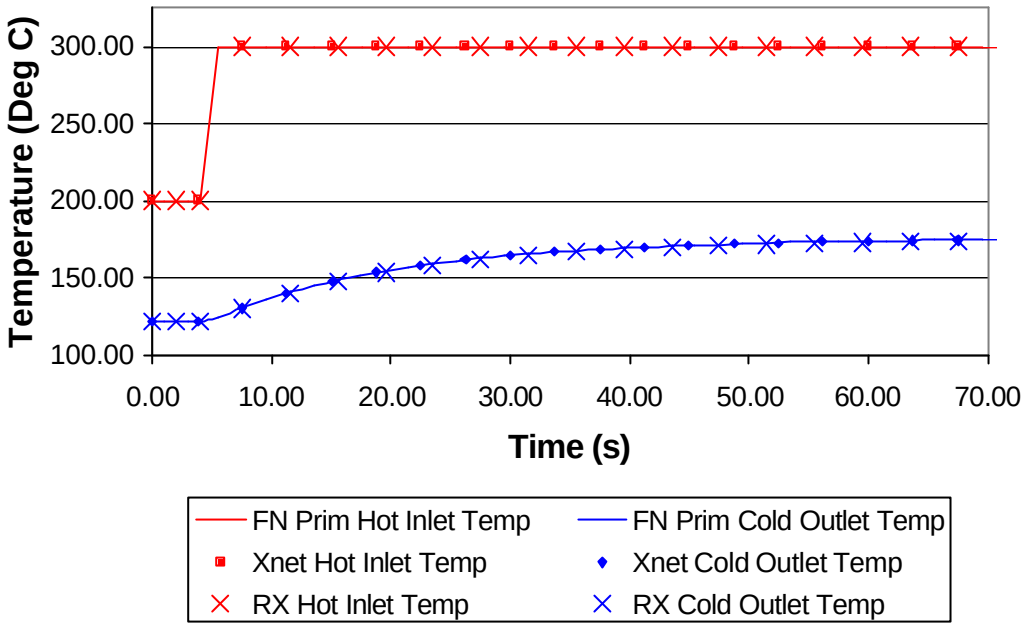


Figure 5.32: Transient results for a counter flow heat exchanger with 10 increments.

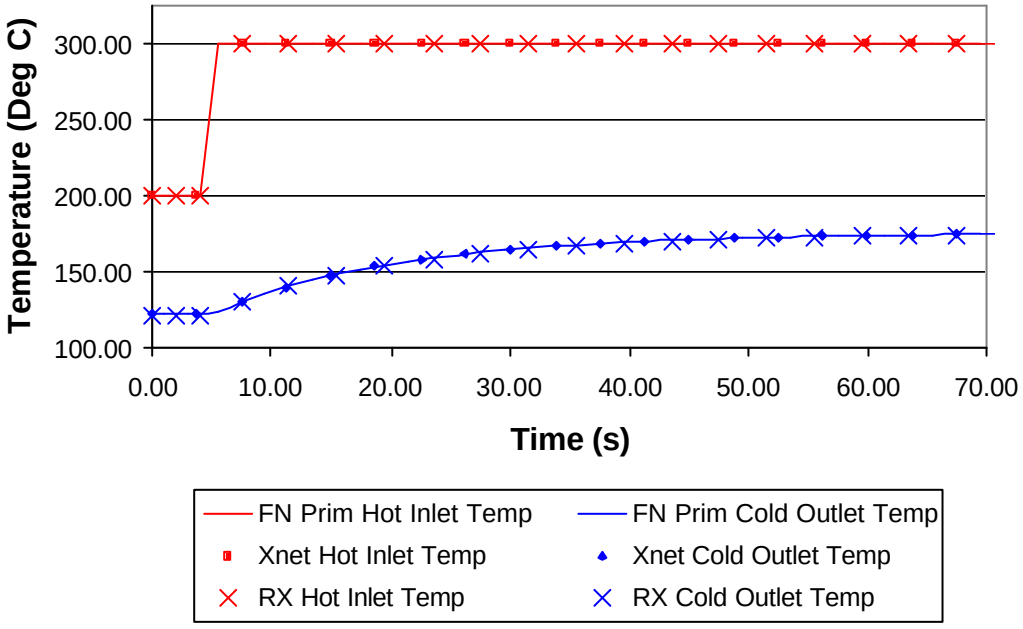


Figure 5.33: Transient results for a counter flow heat exchanger with 20 increments.

5.5.3 Conclusion

It is evident from the steady-state and transient results presented above that the primitive element networks modelled in Xnet show good agreement to the results from the primitive element models in Flownex and the composite RX element in Flownex. When 5 increments are used, both the parallel flow and counter flow models yield almost identical results to the RX element in Flownex. There was a reasonable difference between Xnet's solver and Flownex's solver for this heat exchanger with such a low thermal inertia. Xnet took quite a while to solve the problems that were more discretised.

5.6 Summary

In this chapter the most basic type of heat exchanger, namely the annular type tube-in-tube heat exchanger, was discussed. Two network topologies were also discussed in this chapter. The methodology and implementation of these topologies in Flownex and Xnet were discussed.

These two topologies were also implemented in EES to investigate the effect of the level of discretisation on the accuracy of the results for both topologies. These topologies were compared to a primitive element network in Flownex and the composite RX element in Flownex. It was shown that the best network topology for a heat exchanger cell is the node-element topology rather than a node-node topology.

Using a node-element topology to represent a discrete heat exchanger cell, a tube-in-tube heat exchanger was modelled for parallel flow and counter flow configuration. The results obtained in Xnet were compared to that of a primitive element model in Flownex, as well as the composite RX model in Flownex. The steady-state and the transient results in Xnet show good agreement with the composite RX element and the Flownex primitive element model, but Xnet took almost two hours in total to solve all the transient problems.

In the next chapter an introduction will be given on complex heat exchangers and their uses. It will be shown how these types of heat exchangers are modelled in Xnet and Flownex, and a shell-and-tube heat exchanger will then be used as a case study to investigate the transient behaviour of heat exchangers using a network approach.

CHAPTER 6 - MODELLING OF COMPLEX HEAT EXCHANGERS

6.1 Introduction

This chapter deals with the modelling of complex heat exchangers, such as multi-pass shell-and-tube heat exchangers where the network topology that is used to represent the heat exchanger deviates from a simple counter flow or parallel flow configuration. This deviation is a result of the fluid that moves in a cross flow manner over one or more finned tube bundles. The main focus of this chapter will be to investigate the applicability of the thermal-fluid network approach to the modelling of complex heat exchanger performance.

In the next section a brief description of complex heat exchangers will be presented to give the reader enough insight in the task at hand, namely to represent the heat exchanger as a discretised thermal-fluid network. The section thereafter will be dedicated to a discussion on the physical discretisation of multi-pass shell-and-tube heat exchangers. This section will also discuss the network connection topology that will be used and the reasoning for it. The composite STX element that is available in Flownex will be discussed briefly to highlight the main aspects of this element.

The rest of the chapter will be dedicated to an in-depth comparative study on steady-state and transient modelling of heat exchanger performance. This study will include the STX element in Flownex, and primitive element models in both Flownex and Xnet. The steady-state comparison will include the investigation of the influence of the level of discretisation on the accuracy of the results. The transient comparison will entail temperature transients, pressure transients, as well as combinations of temperature and pressure transients.

The chapter will be concluded with a discussion on the results, as well as a summary of the chapter.

6.2 Complex heat exchangers

The term “complex heat exchanger“ refers to the network topology of a discretised heat exchanger. If the network topology of the heat exchanger deviates from the parallel flow or counter flow topology, the heat exchanger is classified as a complex heat exchanger. The network topology is determined by the internal flow configuration of the heat exchanger. A typical complex heat exchanger is the shell-and-tube heat exchanger, which is widely found in almost every industry. This type of heat exchanger will be discussed in the following section to familiarise the reader with the task at hand to model these types of heat exchangers.

6.2.1 The shell-and-tube heat exchanger

By far the most common type of heat exchanger encountered in the chemical process industry is the shell-and-tube design. These are available in a variety of configurations with numerous construction features and with differing materials for specific applications (Kacac, *et al.*, 1980). A typical example is shown in Figure 6.1.

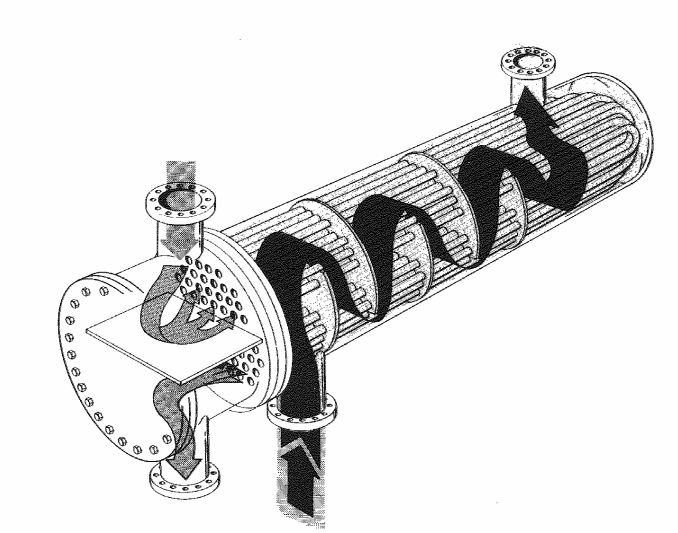


Figure 6.1: Typical shell-and-tube heat exchanger.

In this arrangement, the tube side fluid enters the front-end head through the flange at the upper extreme left of the unit. A pass partition, dividing the upper and lower half of the front end, serves to restrict the entering fluid, so that it may enter only the upper portion of the tubes running through the heat exchanger. After flowing through the length of these tubes, the fluid enters the rear end head, flows to the bottom, and there enters the bottom tubes. The fluid passes through these tubes back toward the front of the unit, thereby making a second pass through the heat exchanger. After exiting the tubes into the lower portion of the front-end head, the fluid exits through the flange at the bottom. The second fluid enters the shell side of the heat exchanger through the second flange at the lower left of the unit. From there it passes in cross-flow across the tubes, around the upper edge of the first baffle and returns toward the bottom of the shell, making a second pass across the outside surface of the tubes. The shell side fluid continues a serpentine flow across the tubes, flowing around the baffle cut-off at the top and lower sides of the unit until it has transversed the entire unit and exits the flange at the upper right.

The number of tube passes as well as shell passes are determined by the specific application of the heat exchanger, and it will not be discussed how it is calculated. It is, however, a very important factor from the network topology viewpoint. The number of tube passes and shell passes determine the specific network topology. This is also affected by the flow direction of the fluids. Figure 6.2 shows several possible configurations for shell-and-tube heat exchangers.

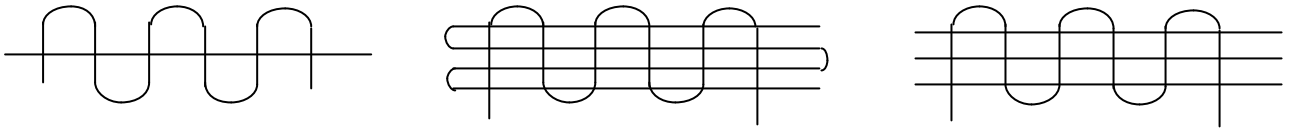


Figure 6.2: Possible internal configurations for shell-and-tube heat exchangers.

Another consideration which affects the network topology is whether the tube side flow is mixed or unmixed. Mixed flow or unmixed flow is determined by the design of the tube headers (Kacac, *et al.*, 1980) and is purely a physical design consideration. Figure 6.4 and Figure 6.5 schematically shows the difference between unmixed and mixed flow respectively. Unmixed flow is accomplished by using a continuous tube bundle with a 180° bend inside the header. The flow inside each tube stays unaffected in the sense that it does not mix with another fluid stream in another tube as the fluid moves through the header to complete another pass. With mixed flow, the tube bundle exits the pass inside the header, where mixing takes place before entering the following tube pass. With mixed flow it is therefore possible to mix fluids at different temperatures inside the headers. It is also possible for the fluid to enter a tube bundle(s) with more or even less tubes than the previous pass, with different flow characteristics and entrance losses.

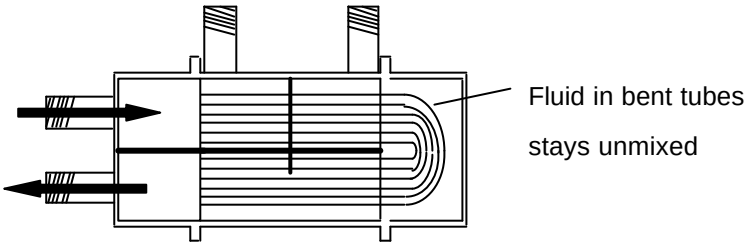


Figure 6.3: Schematic representation of an unmixed tube header.

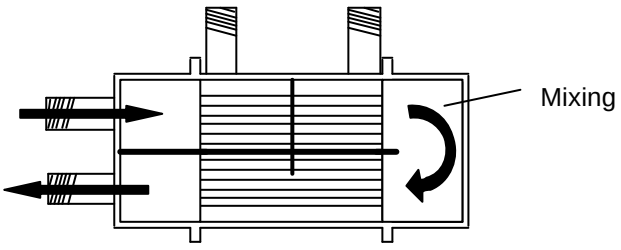


Figure 6.4: Schematic representation of a mixed tube header.

This also influences the network representation of the heat exchanger. With all the above considerations it is clear that the shell-and-tube heat exchanger is represented by a rather complex network topology. In the following section these aspects will be discussed from a network point of view with the physical discretisation of the heat exchanger.

6.3 Discretisation of complex heat exchangers

The approach followed to discretise a complex heat exchanger is to divide the heat exchanger into a two-dimensional array of smaller cross flow heat exchangers, whilst careful attention is given to the physical attributes discussed in the previous section. These smaller cross flow heat exchangers are then solved using known relationships for cross flow heat exchangers.

6.3.1 Discretisation of a shell-and-tube heat exchanger

A typical shell-and-tube heat exchanger is shown in Figure 6.5. Figure 6.5 represents the configuration for a shell-and-tube heat exchanger with 8 shell passes and single tube pass.

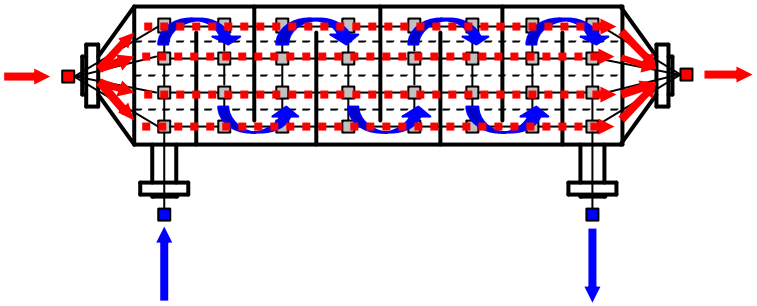


Figure 6.5: Schematic representation of a multipass shell-and-tube heat exchanger, divided into small heat exchangers.

At every intersection where the shell-side fluid passes over the tube bundles, the tube-side exchanges heat with the shell-side through the wall of separating metal. At each of these intersections a small cross flow heat exchanger exists. This is denoted by the grey squares inside the heat exchanger.

In Chapter 5 two network topologies for the connection of a convection element to a flow network were discussed in detail. These two topologies were a node-element connection and a node-node connection. In the case of parallel flow and counter flow heat exchangers it was possible to use both of these topologies to represent the heat exchanger network.

In the case of cross flow heat exchangers, the task of representing the one-dimensional network is a more complex process if the wrong network topology is used. Representing a cross flow heat exchanger with the node-node topology is a difficult task, and will not be discussed here. To represent a cross flow heat exchanger with the node-element topology is straightforward and is shown as a three-dimensional model in Figure 6.6. This figure shows a typical section of a cross flow heat exchanger. With this topology the discretised physical network could be easily and accurately described.

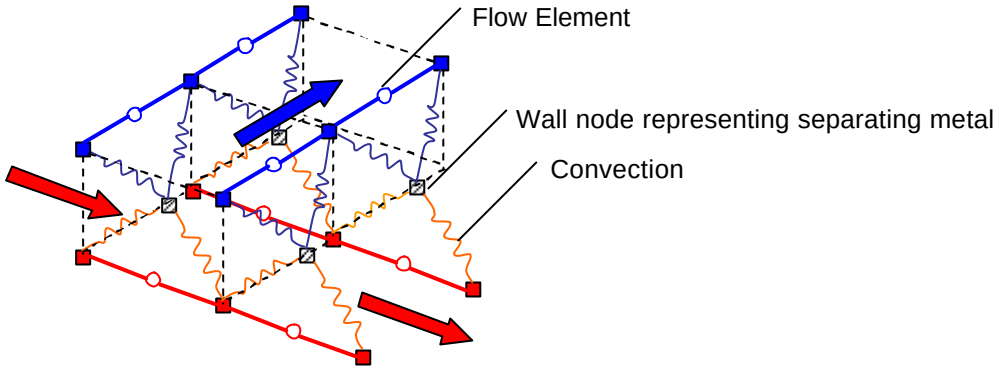


Figure 6.6: Three-dimensional representation of a typical cross flow heat exchanger with a node-element network topology.

For the discretised heat exchange network, the shell side and tube side flow will be considered as flow elements and the separating metal (finned tube wall), where the heat exchange takes place, will be regarded as a solid node. This heat transfer element will consist of two convection elements, between the fluids and the tube walls, and the finned tube wall will be regarded as a solid node. These small cross flow heat exchangers depicted in Figure 6.5 are schematically shown as a one-dimensional network in Figure 6.7.

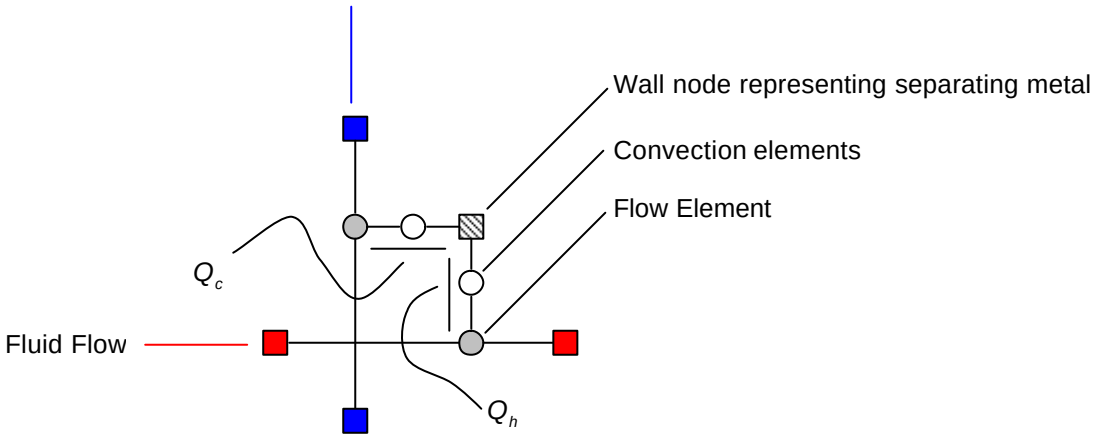


Figure 6.7: Schematic representation of a small cross flow heat exchange element.

Careful attention is required when the shell-and-tube heat exchanger is discretised into small cross flow heat exchangers. For different flow configurations, different models exist. The tube side and the shell side elements must be connected to each other through two convection elements with a solid node separating the convection elements.

Equations 5.1 and 5.2 in Section 5.3.2 calculate the heat transfer from the hot fluid to the solid node, and the solid node to the cold fluid. The heat transfer coefficients used in these equations are dependent on the geometry of the heat transfer surfaces. These heat transfer coefficients for various heat transfer geometries are calculated from heat transfer correlations. Curves that give these correlations (Figure 6.8) in terms of

the Colburn j factor as a function of the Reynolds number can be found in the literature (Kays and London, 1984) from experimental data, or from the manufacturer of the heat exchanger.

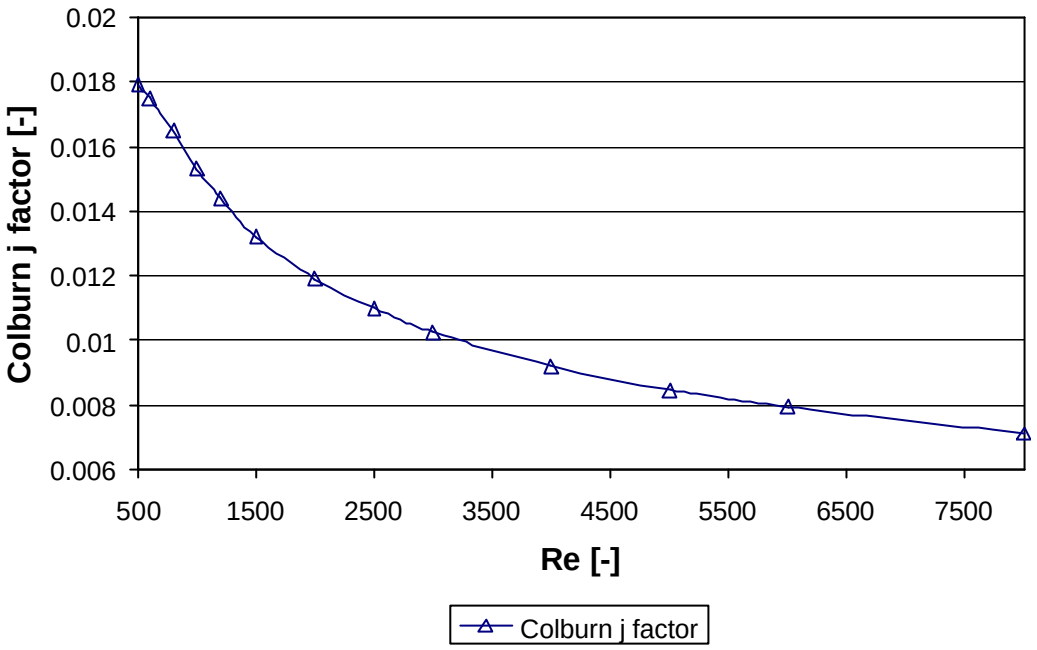


Figure 6.8: Typical heat exchanger chart showing the relation between the Reynolds number and the Colburn j factor.

The Colburn j factor, j_H, are related to the Stanton and Prandtl numbers as follows:

$$j_H = St\,\mathrm{Pr}^{2/3} \tag{6.1}$$

where Pr is the Prandtl value defined as:

$$\mathrm{Pr} = \frac{c_p m}{k} \tag{6.2}$$

with m the viscosity of the fluid. The heat transfer coefficient can now be calculated from:

$$h = j_H \left(\frac{r \mathrm{Vc_p}}{\mathrm{Pr}^{2/3}} \right) \tag{6.3}$$

If the heat transfer surface is calculated for a tube with a circular heat transfer surface (such as a tube), the heat transfer coefficient is calculated from Equation 5.3.

6.4 Composite shell-and-tube heat exchanger (STX) model in Flownex

The STX element used in Flownex is used to model any type of shell-and-tube heat exchanger. This model discretises the shell-and-tube heat exchanger into several small discrete cross flow heat exchangers. The user is able to specify the level of discretisation of the heat exchanger. The effect of this level of discretisation on the results will be investigated in the next section. The model is capable of modelling the four different types of configurations shown in Figure 6.9. These four configurations describe almost every possible configuration since the user can specify the number of tube passes, shell passes etc. It should be noted that other configurations exist, but will not be considered here. A complete description of the STX element can be found in the Flownex 6.4 User Manual (Coetzee *et al*., 2003).

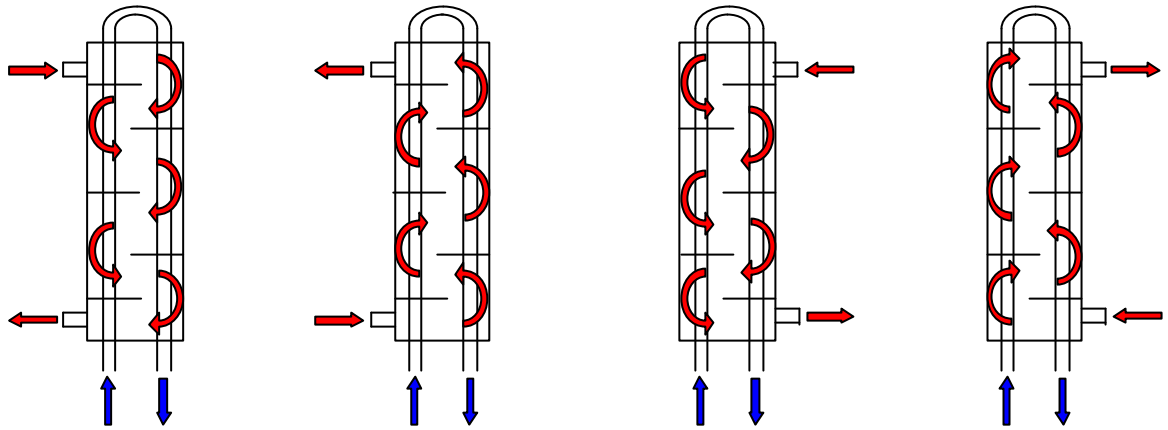


Figure 6.9: Different flow configurations for the composite STX model in Flownex.

6.5 Investigation of the level of discretisation

6.5.1 Introduction

It is possible to adjust the number of increments into which a tube bundle is discretised with the STX model in Flownex. Figure 6.10 shows a single tube bundle on the left-hand side that is not subdivided further. On the right-hand side, the same tube bundle is used, but divided into 4 equal sized smaller flow paths. The number of tubes of each parallel circuit will be the same and will add up to the same amount of tubes in the undivided tube bundle. The reason for this finer “discretisation” of the bundle is to yield more accurate results for complex configurations. This finer discretisation will also impact on the calculation time of the solution. It is therefore essential to investigate the effect of the level of discretisation on the accuracy of the results and the computational time.

Figure 6.10: Schematic representation of a single tube pass divided into 4 parallel circuits.

A shell-and-tube heat exchanger (specification given in Appendix D) with 4 shell passes and 2 and 4 tube passes were used and the number of parallel circuits was varied. This was done for 2 different mass flow ratios (1.0 and 0.5). This process can be schematically represented as follows:

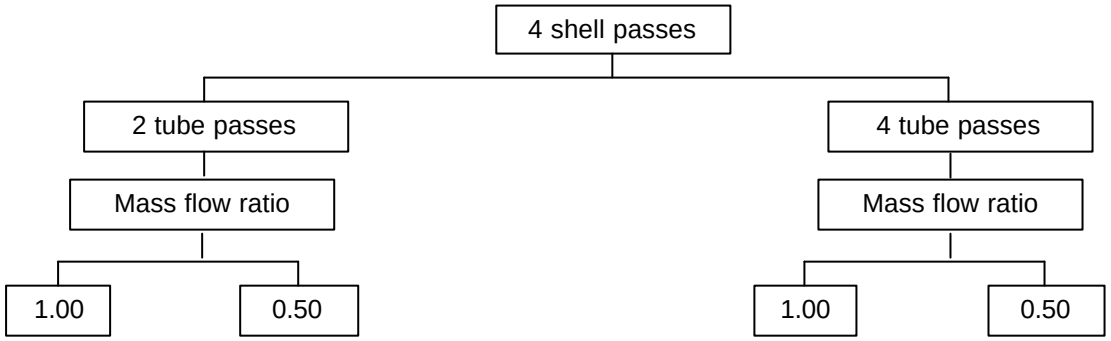


Figure 6.11: Schematic layout of the investigation into the level of discretisation.

A special version of the Flownex solver was used with a constant heat transfer coefficient. This value was set to 1000 [W/m^2 K] for both the hot and cold fluid sides. This investigation did not include primitive element models, because of the following reasons:

The question at hand here is to investigate the effect of the number of parallel circuits on the accuracy of some variable and not to compare Xnet to Flownex. This will be done in a comparative study in the following sections.

A mass flow is specified in Flownex, to acquire specified mass flow rates of 1.0 and 0.5. Xnet uses an explicit solution algorithm, which limits the user to specify a certain pressure drop across the element or network. This tends to make it difficult to use a certain mass flow ratio. Even though the mass flow values predicted by Xnet could be used in Flownex, as the network size is increased (2-4 tube passes etc.), this values differs due to the longer flow length.

The amount of parallel circuits used in the study varied between 1 and 15. To set up such a model with the primitive elements is a time-consuming process. For the above reasons, a meaningful study with the primitive element models are not possible.

The investigation done here on the STX studied the effect of the number of parallel circuits on the accuracy of certain variables for 4 shell passes with 2 and 4 tube passes respectively. This was done in turn for a mass flow ratio ($\dot{m}_h/\dot{m}_c$) of $1.0(0.8/0.8)$ and $0.50(0.4/0.8)$. The shell had an inlet temperature of 300 ºC and the tube inlet temperature was 25 ºC. The effect on the following variables were investigated:

- Shell outlet temperature (hot),
- Tube outlet temperature (cold),
- Total heat transfer,
- Total tube side pressure drop,
- Total shell side pressure drop.

To avoid confusion, only selected results will be discussed. All the results, however, are given in tabular form in the next section.

6.5.2 Results and discussion

Table 6.1 to Table 6.4 give all the results for the variables that were investigated. It is evident from these tables that an increase in the number of parallel circuits affect the variables. This affect can only be seen in the first and second decimal of most of the variables. The variable where the number of parallel circuits tends to have a greater affect is the pressure drop through the shell side. There is also an abnormal prediction of the pressure drop for a single circuit for the case of 2 tube passes. Since most of the results show the same trend, only the values in Table 6.3 are plotted and discussed after the tables.

$STX\ 4_{shellpass}, 2_{tubepass}$					
Par.Circuits	T_{shell} **[°C]**	T_{tube} **[°C]**	Q_{Total} **[kW]**	ΔP_{shell} **[kPa]**	ΔP_{tube} **[kPa]**
1	214.978	110.022	353.352	5.205	81.664
2	214.922	110.078	353.584	4.642	81.687
4	214.901	110.099	353.672	4.980	81.695
5	214.898	110.102	353.685	5.122	81.697
10	214.893	110.107	353.704	5.542	81.699
15	214.892	110.108	353.707	5.744	81.699

Table 6.1: Effect of number of parallel circuits on selected variables for 4 shell passes, 2 tube passes and a mass flow ratio of 1.0.

$STX\,4_{shellpass},2_{tubepass}$					
Par.Circuits	T_{shell} **[°C]**	T_{tube} **[°C]**	Q_{Total} **[kW]**	ΔP_{shell} **[kPa]**	ΔP_{tube} **[kPa]**
1	156.279	96.860	298.652	1.210	76.397
2	156.315	96.842	298.577	1.077	76.391
4	156.321	96.839	298.564	1.149	76.390
5	156.322	96.839	298.563	1.179	76.390
10	156.323	96.839	298.562	1.270	76.389
15	156.323	96.839	298.562	1.314	76.390

Table 6.2: Effect of number of parallel circuits on selected variables for 4 shell passes, 2 tube passes and a mass flow ratio of 0.5.

$STX\,4_{shellpass},4_{tubepass}$					
Par.Circuits	T_{shell} **[°C]**	T_{tube} **[°C]**	Q_{Total} **[kW]**	ΔP_{shell} **[kPa]**	ΔP_{tube} **[kPa]**
1	215.289	109.711	352.059	4.309	82.216
2	215.242	109.758	352.255	5.051	82.236
4	215.225	109.775	352.325	5.926	82.244
5	215.223	109.777	352.334	6.197	82.245
6	215.222	109.778	352.339	6.408	82.245
7	215.221	109.779	352.342	6.576	82.246
8	215.220	109.780	352.344	6.714	82.246
9	215.220	109.780	352.346	6.829	82.246

Table 6.3: Effect of number of parallel circuits on selected variables for 4 shell passes, 4 tube passes and a mass flow ratio of 1.0.

$STX\ 4_{shellpass}, 4_{tubepass}$					
Par.Circuits	T_{shell} **[°C]**	T_{tube} **[°C]**	Q_{Total} **[kW]**	ΔP_{shell} **[kPa]**	ΔP_{tube} **[kPa]**
1	156.672	96.664	297.836	0.994	76.847
2	156.669	96.665	297.842	1.155	76.848
4	156.667	96.667	297.847	1.347	76.848
5	156.666	96.667	297.848	1.406	76.848
6	156.666	96.667	297.848	1.452	76.848
7	156.666	96.667	297.849	1.488	76.848
8	156.666	96.667	297.849	1.517	76.848
9	156.666	96.667	297.849	1.542	76.848

Table 6.4: Effect of number of parallel circuits on selected variables for 4 shell passes, 4 tube passes and a mass flow ratio of 0.5.

Figure 6.12 to Figure 6.14 represent the data in Table 6.3. Figure 6.12 shows that the shell outlet temperature and the tube outlet temperature increase and decrease respectively as the number of parallel circuits are increased. This change in these values becomes smaller with an increase in the amount of parallel circuits to such a point where the difference is almost negligible and equal to the values predicted by the Effectiveness-NTU relation for shell and tube heat exchangers (Incropera and De Witt, 1996:600). The value predicted by the E-NTU method for the shell outlet temperature is 215.220 ºC and the tube outlet temperature 109.780 ºC.

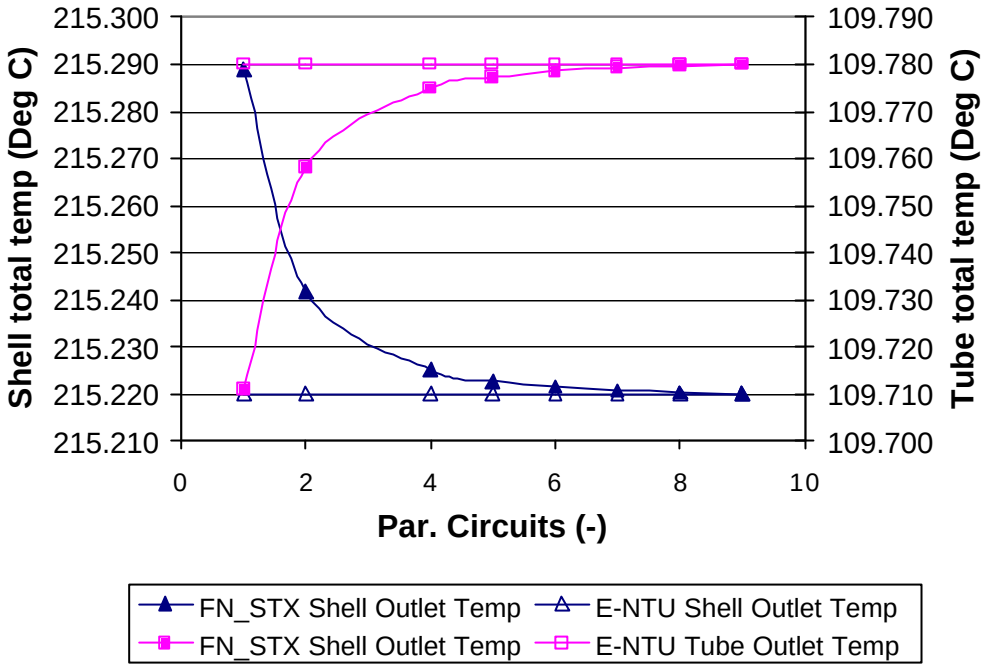


Figure 6.12: STX outlet temperatures versus number of parallel circuits. Data was taken from Table 6.3.

Figure 6.13 shows the effect of the number of parallel circuits on the pressure drop results for the shell side and the tube side of the heat exchanger. The tube side pressure drop shows the same trend, as the temperatures. The shell side, however, shows that the predicted value still rises after 9 parallel circuits. When the number of parallel circuits is increased, the tube bundle width decreases. The difference between the inlet and outlet flow areas decreases and a more accurate velocity over the bundle is predicted since the geometry is better estimated.

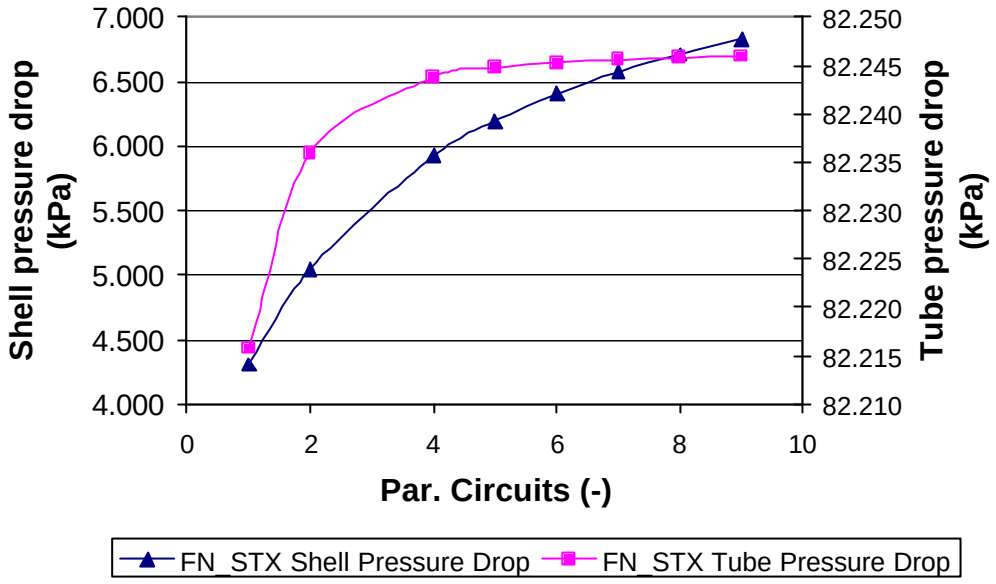


Figure 6.13: STX pressure drop versus number of parallel circuits. Data was taken from Table 6.3.

Figure 6.14 shows the total heat transfer rate as a function the number of parallel circuits. The value converges to the value predicted by the E-NTU method.

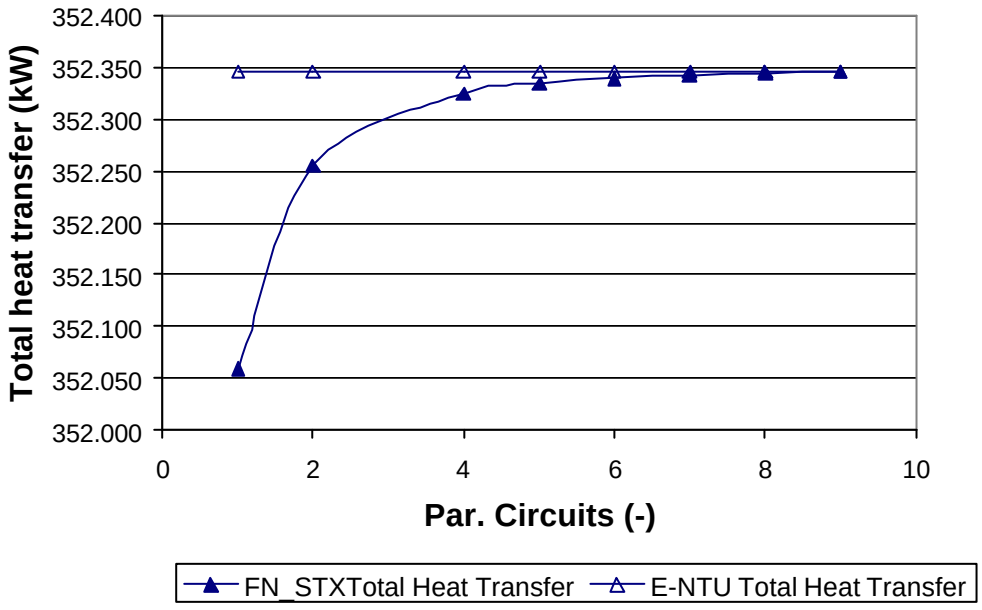


Figure 6.14: STX total heat transfer versus number of parallel circuits. Data was taken from Table 6.3.

The results presented above have shown that as the number of parallel circuits is increased the accuracy of the steady-state values of temperature and total heat transfer predicted by Flownex converges to the E-NTU prediction. The value of the total pressure drop for the tube also shows the same trend. The number of

parallel circuits required for this is ≥ 5. The only variable, which deviates from this trend, was the total pressure drop in the shell side. This is a possible problem in Flownex.

This investigation has shown that the number of parallel circuits for the composite STX model in Flownex has an effect on the accuracy of the results. This, however, was negligible for the total heat transfer, and outlet temperatures, which improved with only 0.07 ºC. The only real improvement was found on the pressure drop prediction of Flownex. This effect is to a great extent dependent on the physical size of the model, and this investigation should be performed before the heat exchanger is modelled as part of a larger network. It is also possible to use four or more parallel circuits to ensure a more accurate answer.

6.6 Comparative study

In this section an investigation will be done to compare Flownex's and Xnet's ability to model shell and tube heat exchangers. The same network topology is used in both codes, (see the node-element topology discussed in Section 6.3). This chapter highlighted the simplicity of representing the physical network with the node-element topology. The heat exchanger considered in this comparative study has 8 shell passes and 1 tube pass and will use helium as fluid in both the tube and shell sides. The heat exchanger will be modelled for all four flow configuration shown in Section 6.4. To ensure an accurate result for the predicted pressure drop through the shell side of the heat exchanger, a small investigation will be done to see the level of discretisation to use. A complete specification of the heat exchanger and the helium used is found in Appendix A. The steady-state and transient comparisons will be discussed below.

6.6.1 Steady-state comparison

The steady-state comparison between Xnet and Flownex's primitive element models and the composite STX element in Flownex will comprise of comparing the steady-state temperature and pressure distribution through the heat exchanger. For the temperature distribution, a two-dimensional plot will be given for the wall temperatures. Several nodes within the network will be compared. These values will also be given in tabular form. The respective node numbers are shown in Figure 6.15. The heat exchanger used in this study (shown below) consists of 8 shell passes and 1 tube pass.

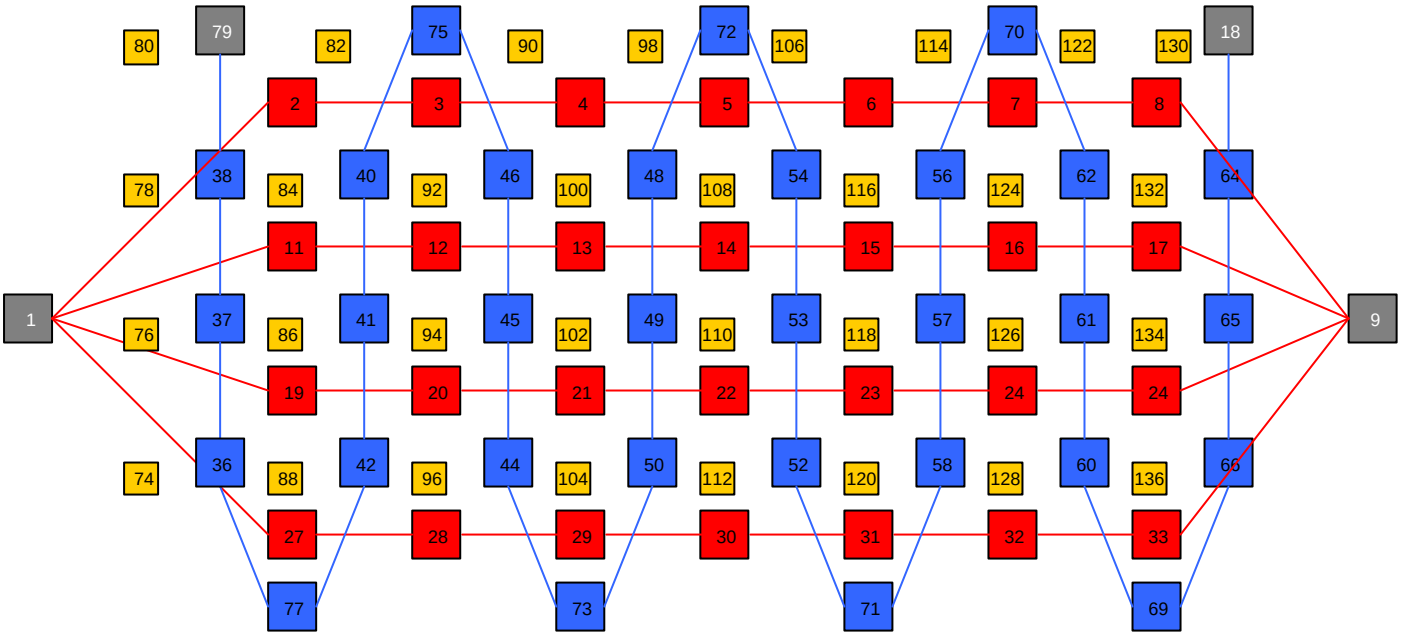


Figure 6.15: Schematic layout of flow and solid node positions.

It should be noted that the smaller squares denote the wall of separating metal and the larger squares the flow nodes. For clarity, the flow elements are not shown. The blue squares represent the shell side flow and the red squares the tube side flow. The grey squares denote nodes with fixed boundary conditions. The heat exchanger will be modelled for all the configurations shown in Figure 6.9. The steady-state boundary conditions specified for the heat exchanger for the comparative study are shown in Table 6.5. The inlet conditions specified here will also be used to investigate the level of discretisation needed for accurate results. It should be noted that for the investigation on the level of discretisation, a mass flow would be specified instead of a pressure drop, since the effect on pressure drop needs to be investigated.

STEADY-STATE BOUNDARY CONDITION SPECIFICATION		
Node 79	Shell Inlet Pressure [kPa]	150
Node 18	Shell Outlet Pressure [kPa]	140
Node 79	Shell Inlet Temperature [°C]	25
Node 1	Tube Inlet Pressure [kPa]	150
Node 9	Tube Outlet Pressure [kPa]	140
Node 1	Tube Inlet Temperature [°C]	300

Table 6.5: Steady-state boundary condition specification.

6.6.2 Level of discretisation for the comparative study

To determine the least amount of parallel circuits needed for an accurate result, a mass flow of 0.05 and 0.64 was specified for the STX on the shell side and tube side respectively. Figure 6.16 to Figure 6.18 show the results from the investigation. Figure 6.16 shows the effect of the number of parallel circuits on the outlet temperatures. It is clear that four parallel circuits would be sufficient to have an accurate result. Figure 6.17 shows the effect on the pressure drop for the shell and tube side. These graphs show that there is a strange shape for the pressure drop versus the number of parallel circuits. This graph shows that for 2 or less parallel circuits, the predicted pressure drop is differs greatly. Figure 6.18 show the effect on the total heat transfer for the heat exchanger. It is recommended that four or more parallel circuits be used in the study of steady-state and transient heat exchanger performance.

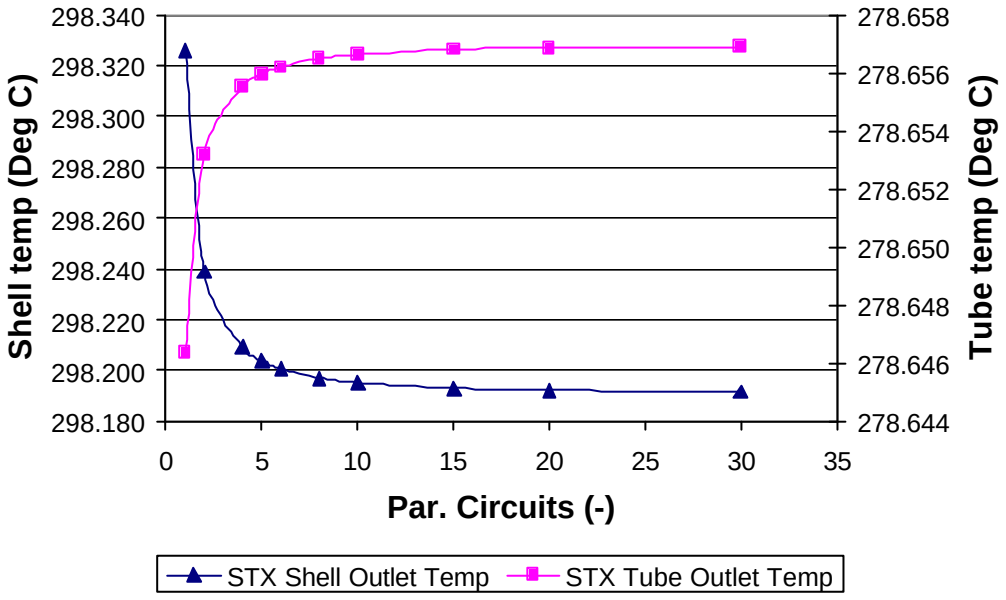


Figure 6.16: Outlet temperatures versus number of parallel circuits for the shell-and-tube heat exchanger in Appendix A.

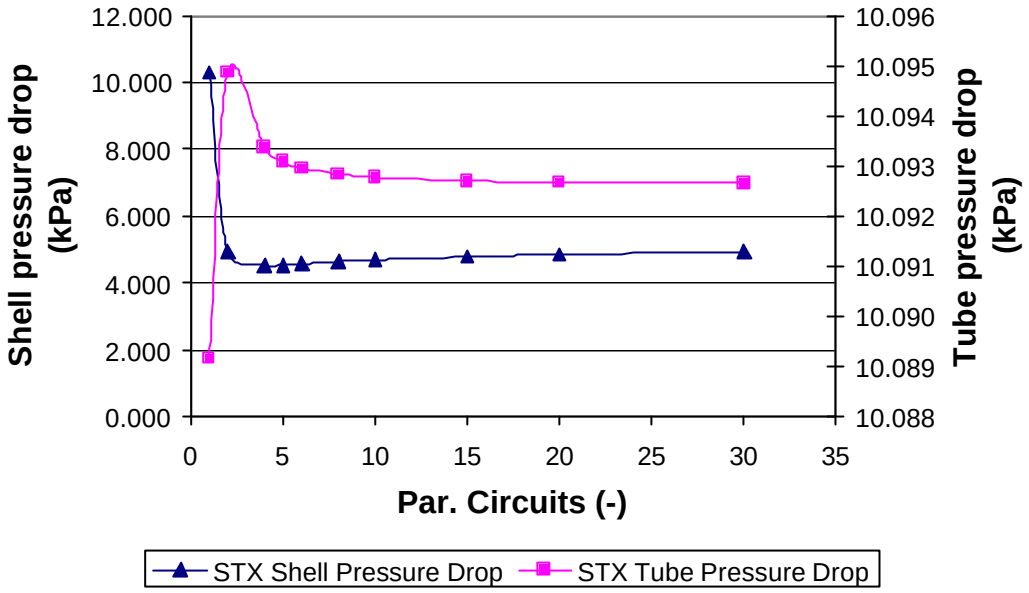


Figure 6.17: Pressure drop versus number of parallel circuits for the shell-and-tube heat exchanger in Appendix A.

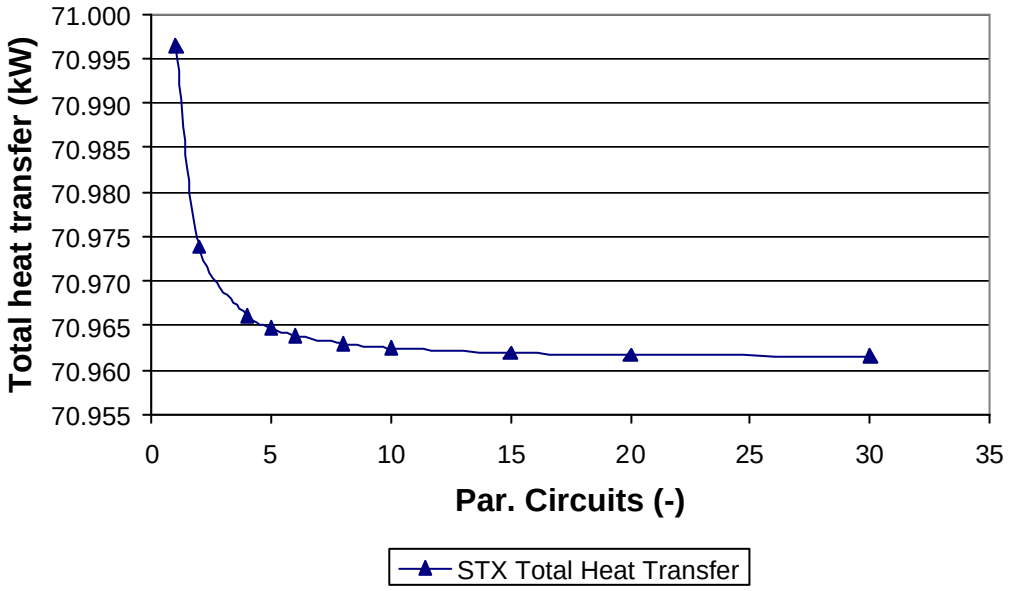


Figure 6.18: Total heat transfer versus number of parallel circuits for the shell-and-tube heat exchanger in Appendix A.

From the above results it is clear that at least four parallel circuits should be used to accurately model the shell-and-tube heat exchanger. The steady-state comparison between Flownex and Xnet will be presented in the following section.

6.6.3 Steady-state comparison results

The fluid temperatures at selected nodes within the network (Figure 6.15) are given in Table 6-6. This table show the difference between the steady-state results of the primitive models in Xnet and Flownex, as well as the composite STX model in Flownex. It is evident from the table that there is little difference in the results for the different solution procedures.

TEMPERATURE COMPARISON – Top left flow configuration			
Position	***Composite STX Model***	***Flownex Primitives***	***Xnet***
Tube Outlet (Node 9)	266.60 °C	269.55 °C	269.54 °C
Shell Outlet (Node 79)	298.36 °C	298.28 °C	298.31 °C
Internal Flow Node 22 (Tube Side)	299.94 °C	299.96 °C	299.96 °C
Internal Flow Node 49 (Shell Side)	296.26 °C	297.89 °C	297.89 °C
TEMPERATURE COMPARISON - Bottom left flow configuration			
Position	***Composite STX Model***	***Flownex Primitives***	***Xnet***
Tube Outlet (Node 9)	272.64 °C	272.65 °C	272.67 °C
Shell Outlet (Node 18)	269.55 °C	269.44 °C	269.51 °C
Internal Flow Node 22 (Tube Side)	272.80 °C	272.79 °C	272.83 °C
Internal Flow Node 49 (Shell Side)	274.35 °C	274.80 °C	274.85 °C

Table 6-6: Temperature comparison between Xnet, Flownex primitive element model, and the composite STX model in Flownex for different flow nodes and 2 different flow configurations.

Figure 6.19 to Figure 6.22 show the steady-state results for the fluid temperature profile through the heat exchanger as the fluids in both sides moves through the heat exchanger. The path for the shell side is given in Figure 6.15 as the blue path from node 18 to 79 and for the tube side the red path from node 1 to 9. Figure 6.19 and Figure 6.20 show the result for a top left flow configuration (Section 6.4). The two graphs show that there is a slight difference in the results between the primitive element models and the composite STX model. This difference occurs between the entrance of the fluid into the shell (node 18) and the 3rd shell pass (node 71). At the same point, the tube side flow shows a slight difference as well. As the shell fluid moves further through the shell, the difference becomes negligible.

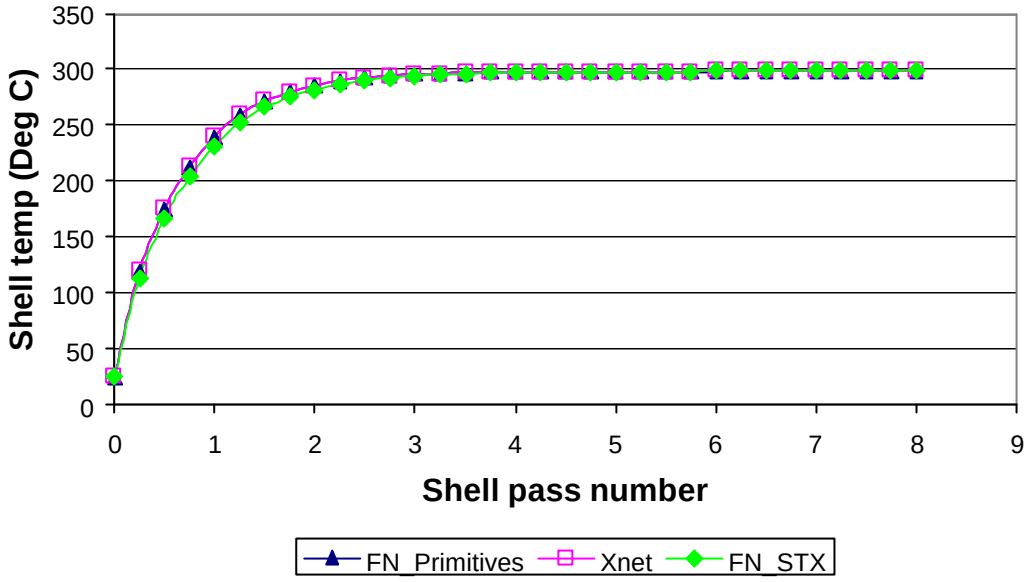


Figure 6.19: Steady-state fluid temperature comparison (top left flow configuration) for the shell side fluid as it moves through the heat exchanger.

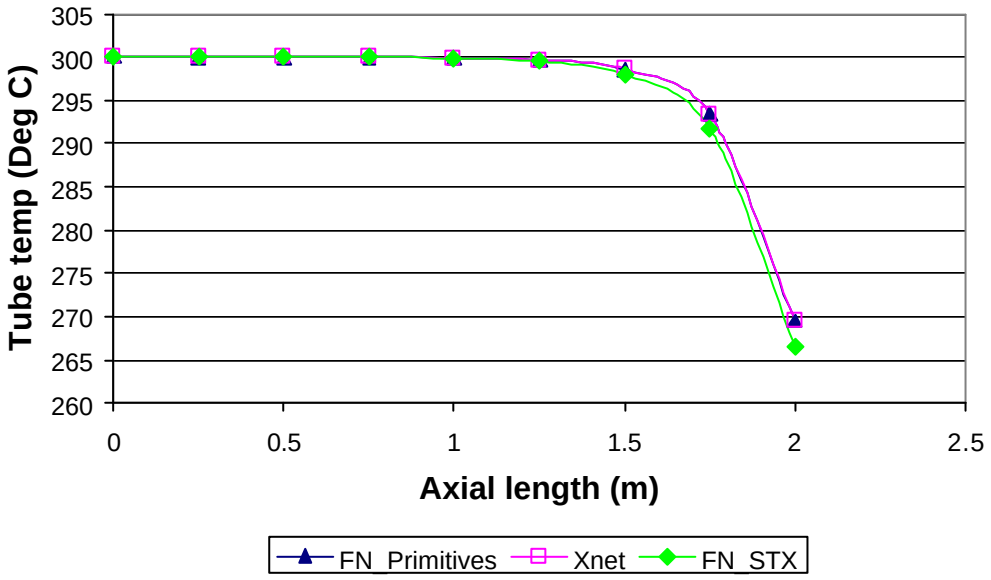


Figure 6.20: Steady-state fluid temperature comparison (top left flow configuration) for the tube side fluid as it moves through the heat exchanger.

Figure 6.21 and Figure 6.22 show the steady-state fluid temperature profile for a bottom left flow configuration. For this configuration, the results of the primitive element models and the composite STX model show good agreement.

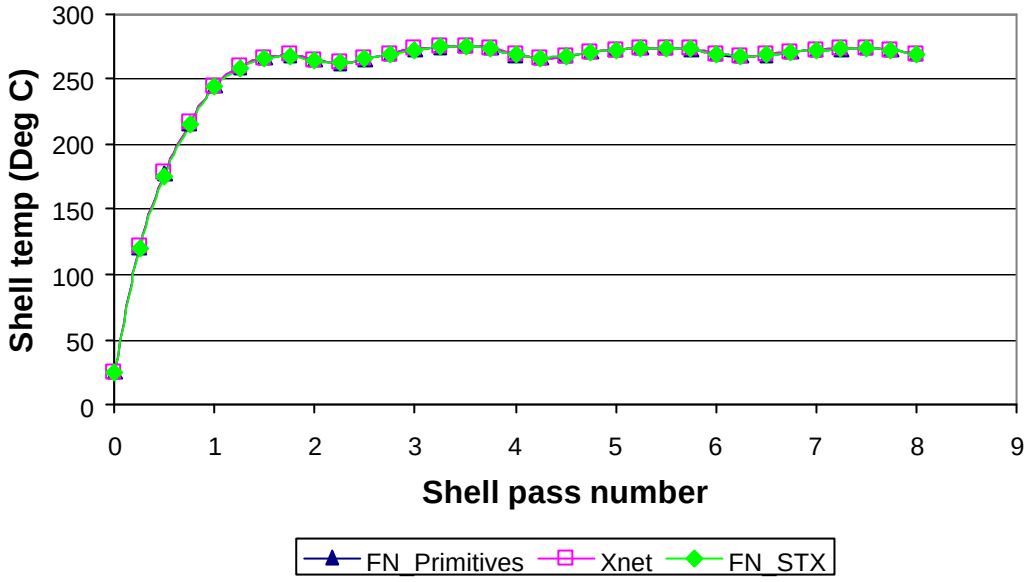


Figure 6.21: Steady-state fluid temperature comparison (bottom left flow configuration) for the shell side fluid as it moves through the heat exchanger.

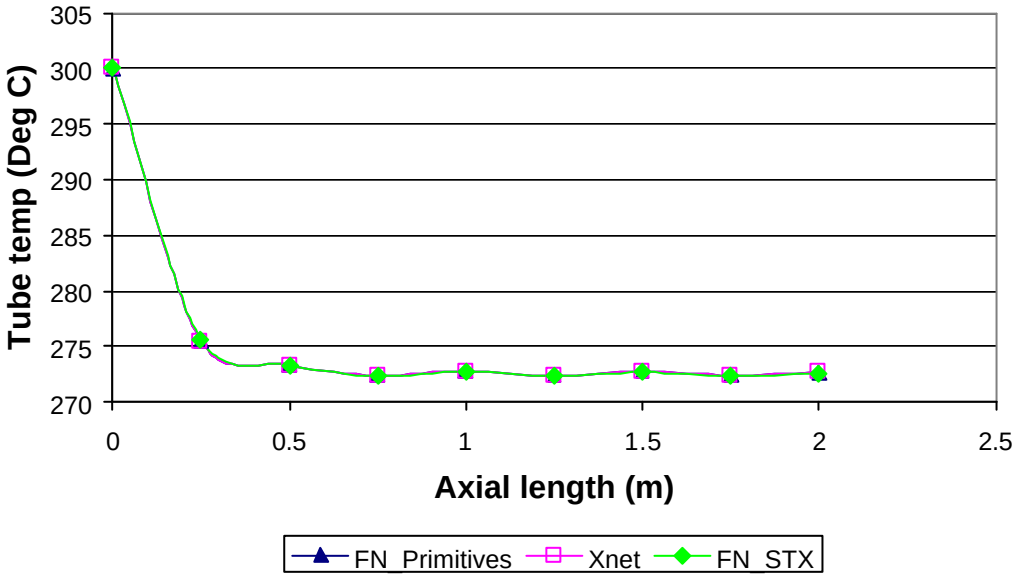
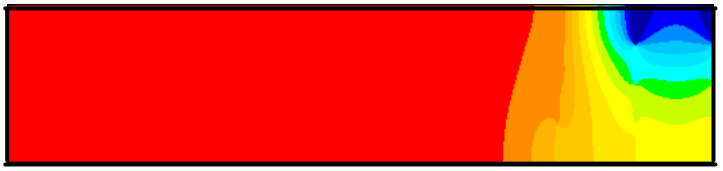
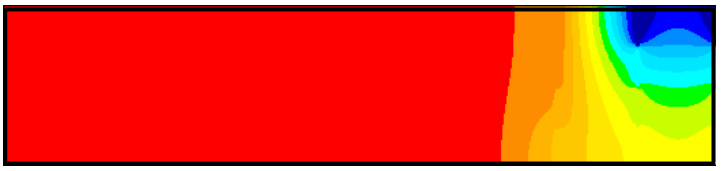


Figure 6.22: Steady-state fluid temperature comparison (bottom left flow configuration) for the tube side fluid as it moves through the heat exchanger.

Figure 6.23 to Figure 6.26 show the steady-state results obtained for the wall temperature distribution for the different flow configurations. The temperatures of the solid nodes where plotted with the CFD code Flo++. The figures shows that the model in Xnet shows good comparison to the primitive element model in Flownex as well as the composite STX model in Flownex.



Flownex Primitives



Xnet

STX

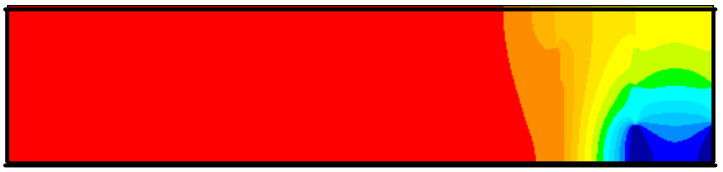
Figure 6.23: Steady-state wall temperature results for a top left flow configuration.

Flownex Primitives

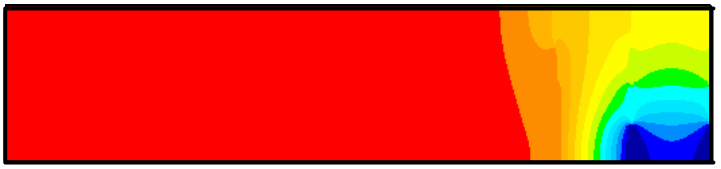
Xnet

STX

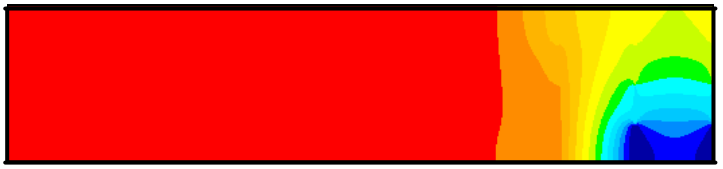
Figure 6.24: Steady-state wall temperature results for a bottom left flow configuration.



Flownex Primitives



Xnet



STX

Figure 6.25: Steady-state wall temperature results for a top right flow configuration.

Flownex Primitives

Xnet

STX

Figure 6.26: Steady-state wall temperature results for a bottom right flow configuration.

Figure 6.27 shows the tube side pressure drop comparison between the primitive element model in Flownex and Xnet and the composite STX element in Flownex for a top left flow configuration. This pressure drop is the mean pressure drop for the 4 parallel circuits of the heat exchanger. The result shows that there is a

decrease in pressure over the length of the tubes. There is greater pressure loss at the beginning and the end of the tubes. This increased pressure drop is a result of additional pressure losses that occur when the fluid enters and exits through the tube headers. The three methods show good agreement with each other.

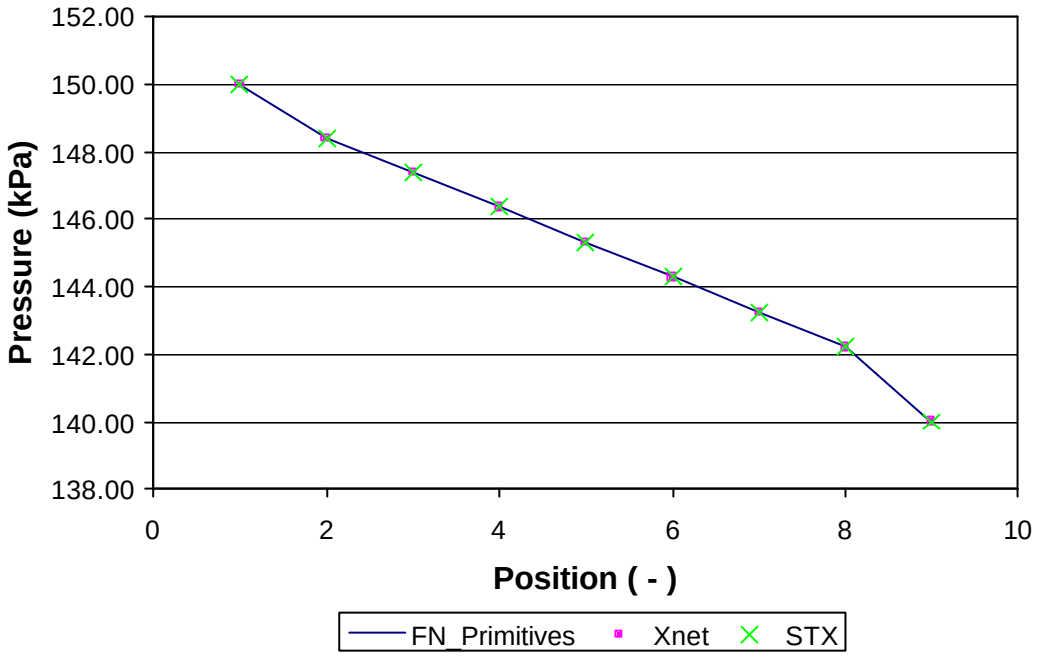


Figure 6.27: Tube side pressure drop comparison between Xnet and Flownex primitive models and the composite STX element in Flownex.

Figure 6.28 shows the shell side pressure drop comparison between the primitive element model in Flownex and Xnet and the composite STX element in Flownex for a top left configuration. The result shows that there is a decrease in pressure along the length of the shell passes. The figure shows that the pressure drop through each tube bank is less than the pressure drop at the entrance and exit of the tube bank. When the fluid enters a tube bank there is entrance losses due to the different area geometry (secondary losses), and the same at the exit of the tube bank. When the fluid moves through the tube bank, the only pressure loss that occurs is the loss as a result of friction. The three methods show good agreement.

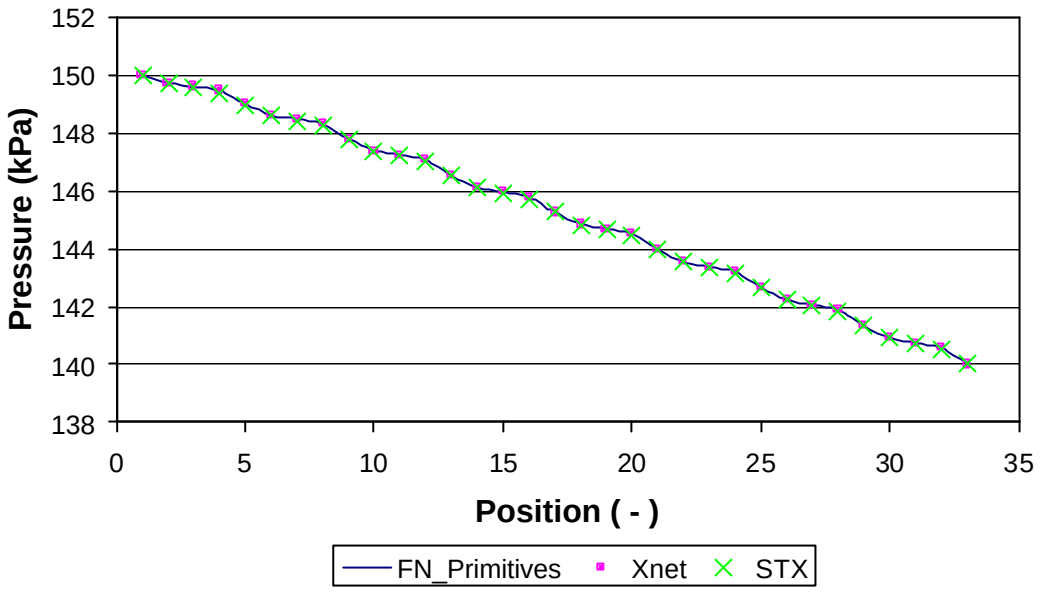


Figure 6.28: Shell side pressure drop comparison between Xnet and Flownex primitive models and the composite STX element in Flownex.

6.6.4 Transient comparison

The transient solution will comprise of simulating the heat exchanger starting at a steady-state condition at which certain step inputs will be applied. The simulation will initially use the steady-state inputs presented above to reach a steady-state. These inputs will then be used for the transient simulation from this steady-state condition. The transient input conditions for the transient solution from the steady-state condition will consist of temperature step input, followed by a combined pressure and temperature step input, followed by a step input in the inlet pressure of the cold fluid in the shell side of the heat exchanger. The transient comparison will be done for all the configurations shown in Figure 6.9. The transient inputs are as follow,

TRANSIENT SPECIFICATION			
Parameter	***Initial Value***	***Step Change To***	***Time***
Temperature Step Input (Tube Inlet) [°C]	300	400	0 (Steady-state)
Combined Pressure And Temperature Input			
Pressure (Shell Inlet) [kPa]	150	165	100 s
Temperature (Tube Inlet) [°C]	400	250	100 s
Pressure Step Input (Shell Inlet) [kPa]	165	150	150 s

Table 6.7: Transient parameter specification set for the transient solution from the steady state condition.

Transient comparison results

Figure 6.29 to Figure 6.32 show the results for the transient simulation of the shell-and-tube heat exchanger, which was subjected to the transient specification given in Table 6.7 for the four different configurations. The figures have been enlarged to show the results more clearly. The results show that the temperatures of the hot fluid side and the cold fluid side rise gradually from the steady-state condition from where a step input in the hot fluid side's temperature are given. Just before the steady-state condition is reached, a combined pressure and temperature step input is applied. A decrease in the temperature of the hot fluid side and an increase in the inlet pressure of the cold side inlet are applied and a gradual decrease in the hot and cold fluid sides temperatures are noticeable. The inlet pressure of the cold fluid side is again decreased and the temperature eventually reaches steady-state. It is evident that the heat exchanger reacts slowly to sudden temperature and pressure transients and shows the effect of thermal inertia. From the figures it is also noticeable that Xnet's prediction of the transient show good agreement with the transient prediction of the primitive element model in Flownex, as well as the composite STX model in Flownex.

For the problem at hand the results of a top left and top right configuration show similar hot and cold outlet temperature profiles. The same holds for a bottom left and bottom right configuration if a single tube pass is used.

The transient comparison between Flownex's solver and Xnet's solver showed that there was a large time difference in the time to solve the network. Flownex's solver took approximately 6 seconds with a time step, Δt , of 0.50. Xnet's solver took approximately 10 hours, with a Courant limited time step, Δt , of approximately 6.65e-05.

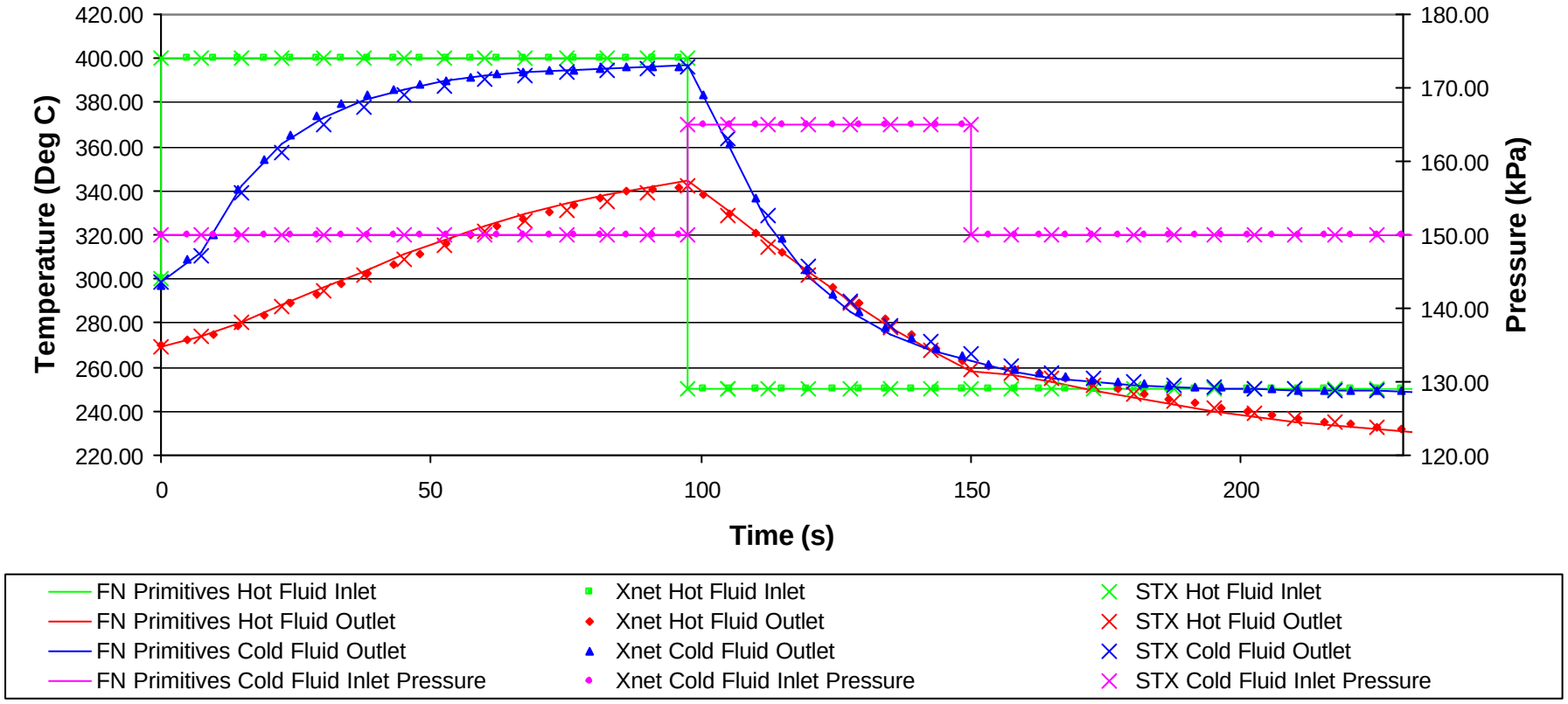


Figure 6.29: Transient simulation of the hot and cold outlet temperature of a shell-and-tube heat exchanger from the steady-state initial condition for a top left flow configuration.

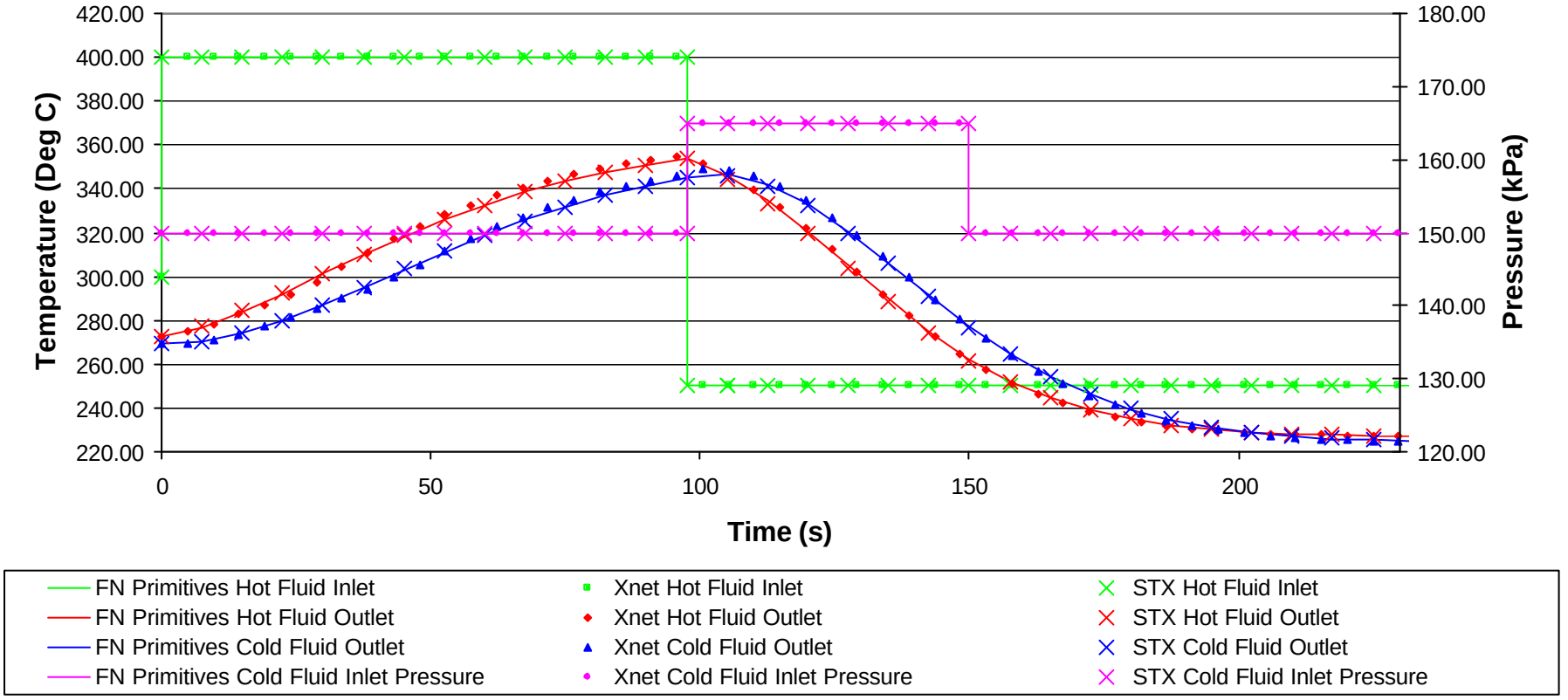


Figure 6.30: Transient simulation of the hot and cold outlet temperature of a shell-and-tube heat exchanger from the steady-state initial condition for a bottom left flow configuration.

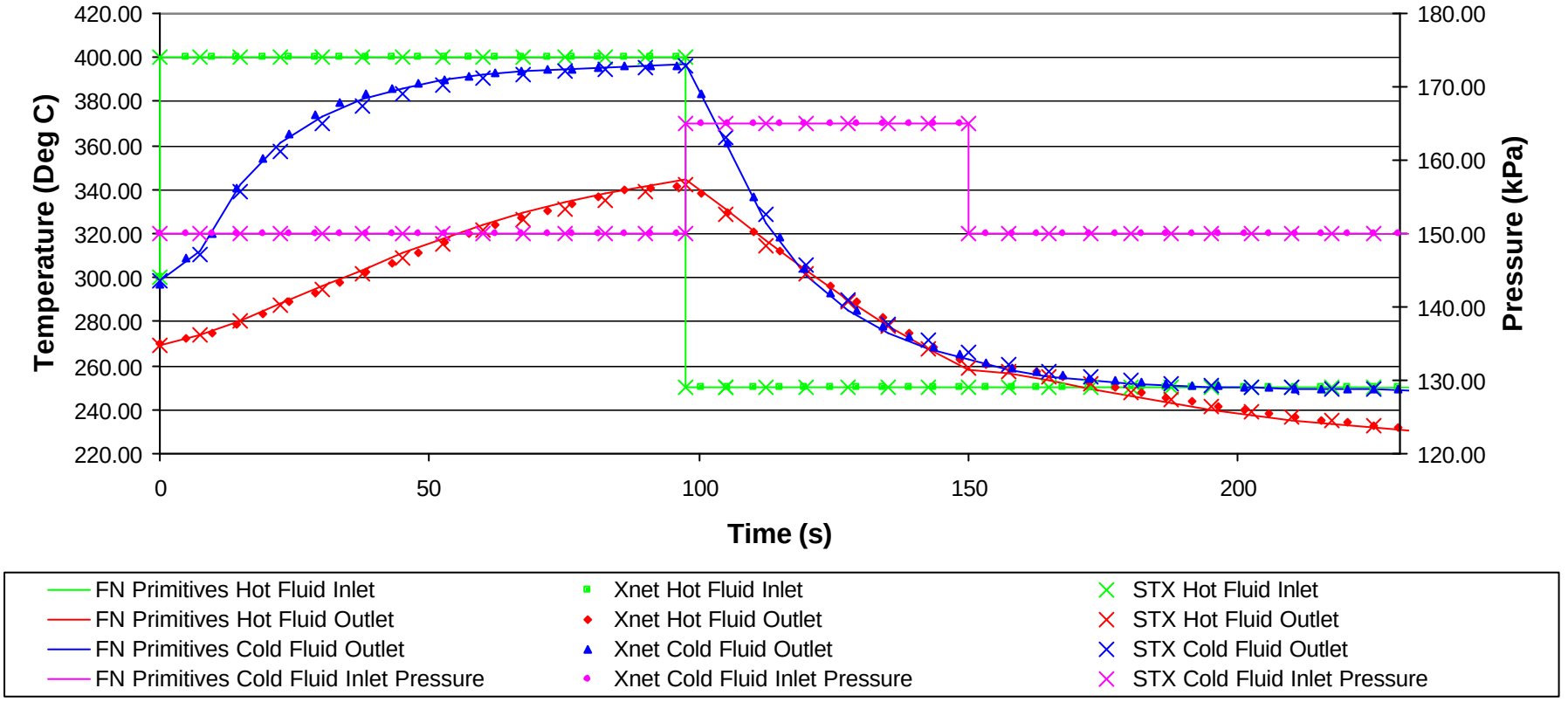


Figure 6.31: Transient simulation of the hot and cold outlet temperature of a shell-and-tube heat exchanger from the steady-state initial condition for a top right flow configuration.

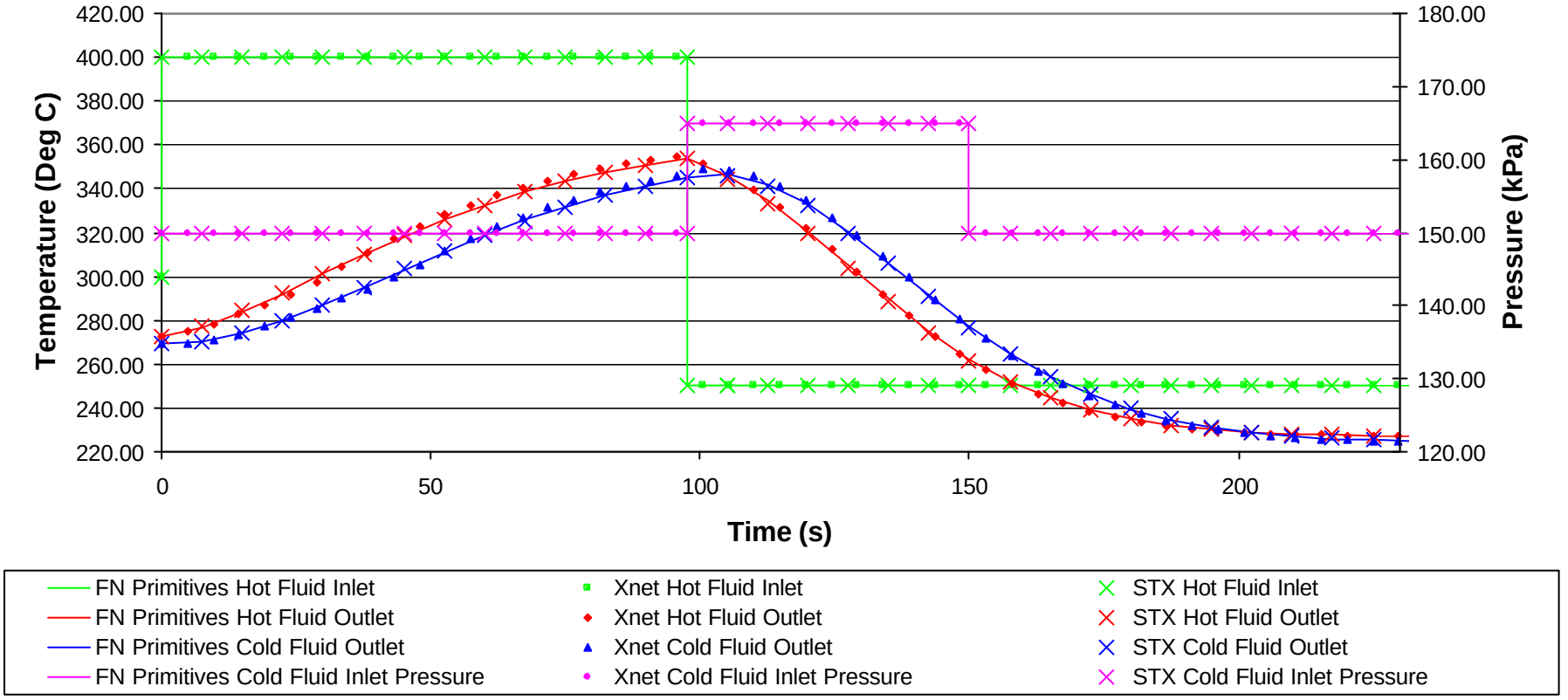


Figure 6.32: Transient simulation of the hot and cold outlet temperature of a shell-and-tube heat exchanger from the steady-state initial condition for a bottom right flow configuration.

6.6.5 Conclusion

The results obtained for the shell-and-tube heat exchanger modelled with the primitive elements of Xnet and Flownex shows good agreement with the composite STX element in Flownex. It is evident from Figure 6.29 that Flownex is capable of solving the combined effect of pressure and temperature transients accurately with the composite STX model. Flownex is therefore capable of solving large arbitrary heat exchange networks found in shell-and-tube heat exchangers with the composite STX model that involve heat exchange across a solid wall by means of convection from the hot fluid to the cold fluid with both pressure and temperature transients. This is done fast with its implicit solver. Xnet, which also produces very accurate results, can be used as a very good benchmark code.

6.7 Summary

In this chapter a complex heat exchanger, namely the shell-and-tube heat exchanger, was introduced and discussed. The method used to discretise the heat exchanger in Flownex and Xnet is also discussed. A comparative study was done to show the extent to which thermal-fluid network solvers are able to predict transient heat exchanger performance. The comparative study was done using the IPCM (Flownex) and RKTD method (Xnet), which used totally different solution algorithms. The study also included the composite STX element in Flownex, as well as the model built up with the primitive elements in Flownex as part of the validation of Flownex.

It is evident from the data that the steady-state and the transient result in Xnet show good agreement with the composite STX element and the Flownex primitive models.

In the next chapter a brief conclusion will be given of the capability of thermal-fluid network codes such as Xnet and Flownex to model transient heat exchangers by using a network approach, and recommendations for further work on the code will be made

CHAPTER 7 - CONCLUSION AND RECOMMENDATIONS

7.1 Introduction

The objective of this study was to investigate the extent to which thermal-fluid network codes are able to predict transient heat exchanger performance. Two different codes, namely the IPCM (Flownex) and RKTD (Xnet) methods, were used to perform this investigation. This study also contributed to the validation of Flownex's composite heat exchanger models. In this chapter a summary of the conclusions that resulted from this study will be presented and recommendations for further work will be made.

7.2 Summary

The modelling of transient fluid flow in complex heat exchange problems sometimes requires detailed analysis of the problem, which in turn requires a great amount of computational effort and man hours. To reduce the time required to set up and solve such a problem, different types of thermal-fluid software packages exist. Some of these software codes require lengthy processes to set up the model of the problem, which is similar to CFD software codes. Thermal-fluid network codes are examples of the type of codes that reduce the time needed to set up such a model, due to the fact that these models are one-dimensional.

The time and accuracy in which network codes solve the model are highly dependent on the numerical scheme that is used in the solver. In this study the difference between an implicit thermal-fluid network code Flownex and an explicit thermal-fluid network code Xnet have become evident through the modelling of transient heat exchanger performance.

7.2.1 Flow elements

Several flow elements that are applicable to heat exchangers and available in Xnet to build primitive element models of heat exchangers were compared to similar elements in Flownex. Xnet's results of these basic flow elements have shown good agreement with the results produced by Flownex. Even though Xnet produces dispersion at some discontinuities during fast transients the code is still able to accurately predict fluid flow phenomena in these flow elements.

7.2.2 Heat elements

Xnet has shown that it can predict accurate convective heat transfer. The results produced by Xnet have also shown good agreement with that of Flownex during several comparisons.

7.2.3 Heat exchanger discretisation topology

The two network topologies used in network modelling of thermal-fluid problems were investigated to see which of the two topologies produce the most accurate results for the least amount of increments. It was shown that the node-element connection topology produced the most accurate results for the least amount

of increments. These topologies also show good agreement with both the primitive model in Flownex of the simulated heat exchanger and the composite RX element.

7.2.4 Complex heat exchangers

The complex heat exchanger modelled with primitive elements in Xnet and Flownex proved to be a good example of a combination of fluid flow and heat transfer. The shell-and-tube heat exchanger had 68 flow elements, 68 heat elements and heat capacity of 170 [kJ/kg K]. A short study on the effect of discretisation on the accuracy of the results has shown that the heat exchanger must be discretised finer to improve the accuracy of the results. The improvement in the results comes with an increased computational time. This resulted in a time-consuming simulation during the transient modelling with Xnet. Even though the simulation required a great amount of computational time, Xnet produced a very accurate result and compared well to the primitive element model set up in Flownex and the composite STX element in Flownex. Even though the improvement in the accuracy of the results were very little for the temperature and total heat transfer, a more visible improvement was seen in the total pressure drop across the heat exchanger. This improvement justifies the increased computational time. Flownex simulated the model in very short time, whilst Xnet reached the required time step in the transient in roughly 10 hours.

7.3 Conclusion

The results that were produced in this study have shown that the IPCM and RKTD method used in the thermal-fluid network solvers Flownex and Xnet are more than adequate to solve complex arbitrary networks and are able to accurately predict transient heat exchanger performance. The methods available in Flownex and Xnet to represent and discretise thermal-fluid networks are adequate and simple to use. With this study it also became evident that Flownex's results for transient heat exchanger performance have shown good agreement with that predicted by Xnet. This shows that the composite models in Flownex are correctly implemented. A more detailed study is, however, needed to validate other aspects of the composite heat exchanger models, such as loss factors etc.

In the next section a few recommendations for future research will be made with regard to the thermal-fluid network code Xnet.

7.4 Recommendations for future research

Certain further developments that would improve the functionality of the thermal-fluid network code Xnet, are described below:

- It is recommended that a convergence criterion for transient solutions be defined in Xnet. A comparison between Xnet and Flownex reveals that large heat capacity values in Xnet require that the steady-state convergence criteria need to be set very strict to give good steady-state results. If

the transient solution starts form this point, the computational time is increased and a great amount of processor memory is required to complete the transient solution.

- Implementation of a faster solution algorithm could make a possible improvement in the solution time required by Xnet to solve complex networks.

- Xnet contains only the basic elements (primitive elements) required for most thermal-fluid problems. It is recommended that composite models be built into Xnet (similar to Flownex) that discretises the model to reduce the effort to set up the network file required by Xnet and eliminate the possibility of human error. These models must also be refined to include empiric-correlations and experimental data. This data can make a definite dfference in the accuracy of the results in comparison to experimental results. An example of this is the flow over the tube bundles in a shell-and-tube heat exchanger which is assumed in this study to be 100% cross flow. In most cases where the flow goes across the baffle plates, the fluid moves at a different angle across the tubes.

- Experimental research needs to be conducted to further validate the results produced by Xnet, as well as Flownex.

- Since transient solutions may be lengthy, the time needed to model steady-state models will make Xnet the more attractive choice if simulation time-accuracy is important.

REFERENCES

ALVARADO, F.L. LASSETER R.H. SANCHEZ, J. 1983. Testing of Trapezoidal Integration with Damping for the Solution of Power Transient Problems, IEEE Transactions on Power Apparatus and Systems, Vol. PAS-102, pp. 3783-3790.

BOTHA, F. 2000. An Explicit Method For The Analysis of Transient Compressible Flow In Pipe Networks, Potchefstroom: PU for CHE (Dessertation – M.Ing.), Appendix A, pp. 1-11, 2000.

COETZEE, R.V. VAN DER MERWE, J. AND VAN RAVENSWAAY, J.P. 2003 Flownex version 6.4 user manual. M-Tech Industrial, Potchefstroom.

CORREA, D.J. MARCHETTI, J.L. 1987. Dynamic Simulation of Shell-and-Tube Heat Exchangers, *Heat Transfer Eng.,* vol. 8, no.1 pp. 50-59.

CULLIMORE & RING TECHNOLOGIES. 2002. SINDA/FLUINT [Web:] http://www.crtech.com/sindaCompare.html [Date of access: 10 April 2003].

ERGUN, S. 1952. Chem. Eng. Progress, Vol. 48, p.89.

F-CHART SOFTWARE. 2003. EES - Engineering Equation Solver. [Web:] http://www.fchart.com/. [Date of access: 1 September 2003].

FERZIGER, J.H. PERIC, M. 2002. Computational Methods for Fluid Dynamics. New York: Springer, 408p.

FLETCHER, C. A. J. 1991. Computational Techniques for Fluid Dynamics, Springer-Verlag

FORGIERI, R. PAPA, G. 1978. Dynamic Response of Counter Current Heat Exchangers to Temperature Disturbances and Step Flow Variations, *Proc. of the Sixth Int. Heat Transfer Conf., Toronto,* vol. 8, HX-18.

GREYVENSTEIN, G.P. VAN RAVENSWAAY, J.P. ROUSSEAU, P.G. 2002. Dynamic modelling of heat, mass and momentum transfer in the pebble bed modular reactor. *1st International conference on Heat Transfer, Fluid Mechanics, and Thermodynamics.*

GREYVENSTEIN, G.P. 2001. An implicit method for the analysis of transient flows in pipe networks. *International journal for numerical methods in engineering*, (53):1127-1143, Jan. 29.

HARLOW, F.H. WELSH, J. E. 1965, Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface, The Physics of Fluids, Vol. 8, No. 12, pp. 2182-2189.

HOLMAN, J.P. 1992. Heat Transfer. Seventh Edition. London: McGraw-Hill Book Company. 137p.

INCROPERA, F.P. AND DE WITT, D.P. 1996. Fundamentals of heat and mass transfer. New York: School of Mechanical Engineering Purdue University. 886p.

INNOVATIVE SYSTEMS SOFTWARE. 1999. RELAP5 [Web:] http://www.relap.com/general/relap5.html [Date of access: 10 April 2003].

KAKAC, S. BERGLES, A.E. MAYINGER, F. 1980. Heat Exchangers: Thermal-Hydraulic Fundamentals And Design. London: McGraw-Hill Company. pp.9-20.

KAYS, W.M. LONDON, A.L. 1984. Compact heat exchangers, 3rd ed., McGraw-Hill, New York.

KREYSZIG, E.1993. Advanced Engineering Mathematics. Seventh Edition. New York: Wiley & Sons, 1271p.

LACHI, M. EL WAKIL, N. PADET, J. 1997 The Time Constant of Double Pipe and One Pass Shell-and-Tube Heat Exchangers in the Case of Varying Fluid Flow Rates, *Int. Journal of Heat and Mass Transfer,* vol. 40, no 9, pp. 2067–2079.

McFARLAND, R.E. 1997 Stability of Discrete Integration Algorithms for a Real Time, Second Order Systems, NASA, Ames Research Center.

MASUKUCHI, M. 1960 Dynamic Response and Control of Multipass Heat Exchangers, *Trans. ASME,* vol. 82, pp.51-82.

MCCABE, M.L. SMITH, J.C. HARRIOTT, P. 1993. *Unit Operations of Chemical Engineering*, 5th Ed., McGraw-Hill, New York.

MECKLENBURGH, J.C. Hartland, S. 1975 The Theory of Backmixing. Wiley, London.

OLIVIER, J.C. 2003. XNet 2.00 Theory and user manual. M-Tech Industrial, Potchefstroom.

PAYNTER, H.M. TAKAHASHI, Y. 1983. A New Method of Evaluating Dynamic Response of Counterflow and Parallel-Flow Heat Exchangers, *Trans. ASME,* vol. 78, pp.749-758.

ROACH, P. 1972. Computational Fluid Dynamics. Hermosa Publishers. Albuquerque, NM

ROPPO, M.N. GANIC, E.N. 1983. Time Dependant Heat Exchanger Modelling, *Heat Transfer Eng*., vol 4, no 2, pp 42 – 46.

ROUSSEAU, P.G. 2002. Advanced Thermal-Fluid Systems Course Notes. (Course presented at the PU for CHE from 21-27 October 2002.) Potchefstroom. 50p. (Unpublished).

SONNTAG, R.E. BORNAKKE, C. VAN WYLEN, G.J. 1998. Fundamentals Of Thermodynamics. New York: Wiley & Sons, 781p.

SHAMES, I.H. 1992. Mechanics of Fluids. New York: McGraw-Hill, 857p.

STAINTHORP, F.P. AXON, A.G. 1965. The Dynamic Behaviour of a Multipass Steam Heated Heat Exchanger – 1. Response to Steam Temperature and Steam Flow Perturbations, *Chem. Eng. Sci.,* vol. 20, pp.107-119.

TAN, K.S. SPINNER, I.H. 1984. Numerical Methods of Solution for Continuous Counter Current Processes in the Nonsteady State, *AIChE J*., vol. 30, no. 5, pp. 770-786.

VAN DER MERWE, J.J. 2003. Simulation of heat and mass transfer in the high temperature test reactor: PU for CHE (Dessertation – M.Ing.) 152 p.

VAN RAVENSWAAY, J.P. 1998. An implicit method for transient pipe network analysis. Potchefstroom: PU for CHE (Dessertation – M.Ing.) 131 p.

WILLIAMS, T.J.MORRIS, H.J. 1961 A Survey of the Literature on Heat Exchange Dynamic and Control, *Chem. Eng. Prog. Symp. Ser.,* vol. 57, no 36, pp.20-33.

WINTERBONE, D.E. PEARSON, R.J. 2000. Theory of engine manifold design. London : Professional Engineering Publishing Limited. 476p.

WYLIE, E.B. STREETER, V.L. 1993. Fluid Transients in Systems. Englewood Cliffs, N.J.: Prentice Hall. 463p.

APPENDICES

APPENDIX A - SHELL-AND-TUBE HEAT EXCHANGER SPECIFICATION

This appendix contains the heat exchanger specification for the heat exchanger used in Chapter 6. This appendix also includes a fluid and material property specification that was used throughout the study.

A.1 HEAT EXCHANGER SPECIFICATION

Type		Conventional E-Type Shell-And-Tube Heat Exchanger
Shell side variables		
Number Of Gas Passes [ ]		8
Shell Diameter [m]		0.3870
Shell Length [m]		1.90
Void Fraction [ ]	0.41995	
Sigma [ ]		0.20025
Kin [ ]		0.50
T1 [m]		0.33780
Z1 [m]		0.02460
Tube side variables		
Number Of Shell Passes [ ]		1
Number Of Parallel Circuits [ ]	4	
Number Of Pipes Per Pass [ ]	448	
Pipe Inside Diameter [m]		0.01022
Length Per Pass [m]		2. 0
Inside Roughness [μm]	10	
K_i [ ]		0.50
K_o [ ]		1.0
Common variables		
Heat Capacity [kJ/kg K]	170.20	
Gas Heat Area [m^2]		34.50
Area Ratio [ ]		1.243
Metal Coefficient [W/m^2K]		10000

A.2 FLUID PROPERTY SPECIFICATION

The following fluid properties were specified for Xnet and Flownex.

Fluid Type & Name	Compressible Helium
Thermal Conductivity, k, [W/m K]	0.150
Gas Constant, R, [J/kg K]	2078.0
Viscosity, μ, [kg/m.s]	2.0×10^{-5}
Specific Heat @ Constant Pressure, c_p, [J/kg K]	5195.0

A.3 MATERIAL PROPERTY SPECIFICATION

The following material properties were specified for Xnet (Material 5) and Flownex (User Defined).

Material Name	Copper (Composition Unknown)
Thermal Conductivity, k, [W/m K]	252
Density, ?, [kg/m^3]	8933.0
Specific Heat @ Constant Pressure, c_p, [J/kg K]	482

APPENDIX B - XNET INPUTFILE FOR SHELL-AND-TUBE HEAT EXCHANGER

```
11//version number
GENERAL DATA
Pinit Tinit Tamb
[kPa] [°C] [°C]
150 25 25
Rgas cp mu thermconduct
[J/kg.K] [J/kg.K] [kg/m.s] [W/m.K]
2078 5195 2.0E-05 0.15
isothermal fact concrit
0 0.8 1.0E-08
SS[0]/Transient[1] StartFromZero[0]/StartSS[1] Steps
1 1 3600000
"TROutput[0=element,1=node]"
1
//////////////////////////////////////////////////////
EVENTS
0 100
Node# P[0]/T[1] || heat[0]/T[1] Flag[0/1] a0 a1 "[P=kPa,T=°C]"
1 1 1 400 0
0
100 150
Node# P[0]/T[1] || heat[0]/T[1] Flag[0/1] a0 a1 "[P=kPa,T=°C]"
1 1 1 250 0
35 0 1 165 0
0
150 1.0E20
Node# P[0]/T[1] || heat[0]/T[1] Flag[0/1] a0 a1 "[P=kPa,T=°C]"
35 0 1 150 0
0
-1//Delimeter
////////////////////// Flow elements ////////////////////////////////
SUBDIVIDEDPIPE
# N1 N2 Ni L[m] D[m] e[µm] kf kb kif kib kof kob hflag q[kW]/twall[°C]
0
PIPEDW
# N1 N2 L[m] Di[m] Do[m] e[µm] Number kf kb k1f k1b k2f k2b
1 1 2 0.25 0 0.01022 10 112 0 0 0.5 1 0 0
2 2 3 0.25 0 0.01022 10 112 0 0 0 0 0 0
3 3 4 0.25 0 0.01022 10 112 0 0 0 0 0 0
4 4 5 0.25 0 0.01022 10 112 0 0 0 0 0 0
5 5 6 0.25 0 0.01022 10 112 0 0 0 0 0 0
6 6 7 0.25 0 0.01022 10 112 0 0 0 0 0 0
7 7 8 0.25 0 0.01022 10 112 0 0 0 0 0 0
10 1 11 0.25 0 0.01022 10 112 0 0 0.5 1 0 0
11 11 12 0.25 0 0.01022 10 112 0 0 0 0 0 0
12 12 13 0.25 0 0.01022 10 112 0 0 0 0 0 0
```

13	13	14	0.25	0	0.01022	10	112	0	0	0	0	0	0
14	14	15	0.25	0	0.01022	10	112	0	0	0	0	0	0
15	15	16	0.25	0	0.01022	10	112	0	0	0	0	0	0
16	16	17	0.25	0	0.01022	10	112	0	0	0	0	0	0
19	1	19	0.25	0	0.01022	10	112	0	0	0.5	1	0	0
20	19	20	0.25	0	0.01022	10	112	0	0	0	0	0	0
21	20	21	0.25	0	0.01022	10	112	0	0	0	0	0	0
22	21	22	0.25	0	0.01022	10	112	0	0	0	0	0	0
23	22	23	0.25	0	0.01022	10	112	0	0	0	0	0	0
24	23	24	0.25	0	0.01022	10	112	0	0	0	0	0	0
25	24	25	0.25	0	0.01022	10	112	0	0	0	0	0	0
28	1	27	0.25	0	0.01022	10	112	0	0	0.5	1	0	0
29	27	28	0.25	0	0.01022	10	112	0	0	0	0	0	0
30	28	29	0.25	0	0.01022	10	112	0	0	0	0	0	0
31	29	30	0.25	0	0.01022	10	112	0	0	0	0	0	0
32	30	31	0.25	0	0.01022	10	112	0	0	0	0	0	0
33	31	32	0.25	0	0.01022	10	112	0	0	0	0	0	0
34	32	33	0.25	0	0.01022	10	112	0	0	0	0	0	0
8	8	9	0.25	0	0.01022	10	112	0	0	0	0	1	0.1
17	17	9	0.25	0	0.01022	10	112	0	0	0	0	1	0.1
26	25	9	0.25	0	0.01022	10	112	0	0	0	0	1	0.1
35	33	9	0.25	0	0.01022	10	112	0	0	0	0	1	0.1

0

PIPEDG

#	N1	N2	L[m]	A1[m²]	C1[m]	A2[m²]	C2[m]	e[µm]	kf	kb	k1f	k1b	k2f	k2b

0

PIPEHX

#	N1	N2	L[m]	Dh[m]	Vol[m³]	A1[m²]	Sig1	A2[m²]	Sig2	k1f	k1b	k2f	k2b	Curvenum	Fric
37	35	36	0.08445	0.004912	0.0024	0.04485	0.20025	0.0827	0.20025	0.5	1	0	0	1	0
38	36	37	0.08445	0.004912	0.00315	0.0827	0.20025	0.091912	0.20025	0	0	0	0	1	0
39	37	38	0.08445	0.004912	0.00315	0.091912	0.20025	0.0827	0.20025	0	0	0	0	1	0
40	38	39	0.08445	0.004912	0.0024	0.0827	0.20025	0.04485	0.20025	0	0	1	0.1	1	0
41	39	40	0.08445	0.004912	0.0024	0.04485	0.20025	0.0827	0.20025	0.5	1	0	0	1	0
42	40	41	0.08445	0.004912	0.00315	0.0827	0.20025	0.091912	0.20025	0	0	0	0	1	0
43	41	42	0.08445	0.004912	0.00315	0.091912	0.20025	0.0827	0.20025	0	0	0	0	1	0
44	42	43	0.08445	0.004912	0.0024	0.0827	0.20025	0.04485	0.20025	0	0	1	0.1	1	0
45	43	44	0.08445	0.004912	0.0024	0.04485	0.20025	0.0827	0.20025	0.5	1	0	0	1	0
46	44	45	0.08445	0.004912	0.00315	0.0827	0.20025	0.091912	0.20025	0	0	0	0	1	0
47	45	46	0.08445	0.004912	0.00315	0.091912	0.20025	0.0827	0.20025	0	0	0	0	1	0
48	46	47	0.08445	0.004912	0.0024	0.0827	0.20025	0.04485	0.20025	0	0	1	0.1	1	0
49	47	48	0.08445	0.004912	0.0024	0.04485	0.20025	0.0827	0.20025	0.5	1	0	0	1	0
50	48	49	0.08445	0.004912	0.00315	0.0827	0.20025	0.091912	0.20025	0	0	0	0	1	0
51	49	50	0.08445	0.004912	0.00315	0.091912	0.20025	0.0827	0.20025	0	0	0	0	1	0
52	50	51	0.08445	0.004912	0.0024	0.0827	0.20025	0.04485	0.20025	0	0	1	0.1	1	0
53	51	52	0.08445	0.004912	0.0024	0.04485	0.20025	0.0827	0.20025	0.5	1	0	0	1	0
54	52	53	0.08445	0.004912	0.00315	0.0827	0.20025	0.091912	0.20025	0	0	0	0	1	0
55	53	54	0.08445	0.004912	0.00315	0.091912	0.20025	0.0827	0.20025	0	0	0	0	1	0
56	54	55	0.08445	0.004912	0.0024	0.0827	0.20025	0.04485	0.20025	0	0	1	0.1	1	0
57	55	56	0.08445	0.004912	0.0024	0.04485	0.20025	0.0827	0.20025	0.5	1	0	0	1	0
58	56	57	0.08445	0.004912	0.00315	0.0827	0.20025	0.091912	0.20025	0	0	0	0	1	0
59	57	58	0.08445	0.004912	0.00315	0.091912	0.20025	0.0827	0.20025	0	0	0	0	1	0
60	58	59	0.08445	0.004912	0.0024	0.0827	0.20025	0.04485	0.20025	0	0	1	0.1	1	0

61	59	60	0.08445	0.004912	0.0024	0.04485		0.20025	0.0827		0.20025	0.5	1	0	0	1	0
62	60	61	0.08445	0.004912	0.00315	0.0827		0.20025	0.091912		0.20025	0	0	0	0	1	0
63	61	62	0.08445	0.004912	0.00315	0.091912		0.20025	0.0827		0.20025	0	0	0	0	1	0
64	62	63	0.08445	0.004912	0.0024	0.0827		0.20025	0.04485		0.20025	0	0	1	0.1	1	0
65	63	64	0.08445	0.004912	0.0024	0.04485		0.20025	0.0827		0.20025	0.5	1	0	0	1	0
66	64	65	0.08445	0.004912	0.00315	0.0827		0.20025	0.091912		0.20025	0	0	0	0	1	0
67	65	66	0.08445	0.004912	0.00315	0.091912		0.20025	0.0827		0.20025	0	0	0	0	1	0
68	66	67	0.08445	0.004912	0.0024	0.0827		0.20025	0.04485		0.20025	0	0	1	0.1	1	0
0																	
PIPEPB																	
#	N1	N2	L[m]	Dh[m]	A1[m²]	A2[m²]	Voidfrac	Direct	k1f	k1b	k2f	k2b	Option				
0																	
//////////////////////////	Heat	elements	//////////////////////////////////														
CHT																	
#	N1	N2	L[m]	A1[m²]	sig1	A2[m²]	sig2	H[m]	Material#	MassOption							
0																	
CHTCROSS																	
#	N1	N2	Config	"(0=parallel,"		1=counter)											
0																	
CONVECTION																	
#	Num1	Type1	Num2	Type2	A[m²]	Config	Curvenum	Nu_lam	C	m	nh	nc					
73	28	1	74	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
74	19	1	76	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
75	10	1	78	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
76	1	1	80	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
77	2	1	82	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
78	11	1	84	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
79	20	1	86	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
80	29	1	88	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
81	3	1	90	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
82	12	1	92	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
83	21	1	94	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
84	30	1	96	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
85	4	1	98	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
86	13	1	100	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
87	22	1	102	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
88	31	1	104	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
89	5	1	106	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
90	14	1	108	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
91	23	1	110	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
92	32	1	112	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
93	6	1	114	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
94	15	1	116	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
95	24	1	118	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
96	33	1	120	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
97	7	1	122	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
98	16	1	124	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
99	25	1	126	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
100	34	1	128	0	0.8676	0	0	4.36	0.023	0.8	0.4	0.3					
109	37	1	74	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3					
110	38	1	76	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3					
111	39	1	78	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3					

112	40	1	80	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
113	41	1	82	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
114	42	1	84	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
115	43	1	86	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
116	44	1	88	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
117	45	1	96	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
118	46	1	94	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
119	47	1	92	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
120	48	1	90	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
121	49	1	98	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
122	50	1	100	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
123	51	1	102	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
124	52	1	104	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
125	53	1	112	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
126	54	1	110	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
127	55	1	108	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
128	56	1	106	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
129	57	1	114	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
130	58	1	116	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
131	59	1	118	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
132	60	1	120	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
133	61	1	128	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
134	62	1	126	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
135	63	1	124	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
136	64	1	122	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
137	65	1	130	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
138	66	1	132	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
139	67	1	134	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
140	68	1	136	0	1.07813	0	1	4.36	0.023	0.8	0.4	0.3
9	8	1	130	0	0.867595	0	0	4.36	0.023	0.8	0.4	0.3
18	17	1	132	0	0.867595	0	0	4.36	0.023	0.8	0.4	0.3
27	26	1	134	0	0.867595	0	0	4.36	0.023	0.8	0.4	0.3
36	35	1	136	0	0.867595	0	0	4.36	0.023	0.8	0.4	0.3

0
RADIATION

#	N1	N2	F12	A1[m²]	A2[m²]	Eps1	Eps2

0
////////////////////////// Nodes //////////////////////////////////////
FLOW_NODES

#	Pflag	Tflag	P	T	Vol
35	1	1	150	25	0.00063
67	1	0	140	25	0.00063
39	0	0	140	25	0.00126
43	0	0	140	25	0.00126
47	0	0	140	25	0.00126
51	0	0	140	25	0.00126
55	0	0	140	25	0.00126
59	0	0	140	25	0.00126
63	0	0	140	25	0.00126
1	1	1	150	300	0
9	1	0	140	25	0

0

SOLID_NODES

#	Tflag	T[°C]	heat	m	Material#
74	0	25	0	11.0347	5
76	0	25	0	11.0347	5
78	0	25	0	11.0347	5
80	0	25	0	11.0347	5
82	0	25	0	11.0347	5
84	0	25	0	11.0347	5
86	0	25	0	11.0347	5
88	0	25	0	11.0347	5
96	0	25	0	11.0347	5
94	0	25	0	11.0347	5
92	0	25	0	11.0347	5
90	0	25	0	11.0347	5
98	0	25	0	11.0347	5
100	0	25	0	11.0347	5
102	0	25	0	11.0347	5
104	0	25	0	11.0347	5
106	0	25	0	11.0347	5
108	0	25	0	11.0347	5
110	0	25	0	11.0347	5
112	0	25	0	11.0347	5
114	0	25	0	11.0347	5
116	0	25	0	11.0347	5
118	0	25	0	11.0347	5
120	0	25	0	11.0347	5
128	0	25	0	11.0347	5
126	0	25	0	11.0347	5
124	0	25	0	11.0347	5
122	0	25	0	11.0347	5
130	0	25	0	11.0347	5
132	0	25	0	11.0347	5
134	0	25	0	11.0347	5
136	0	25	0	11.0347	5

0

////////////////////////////////// Monitor node number //////////////////////////////////

(User number of monitor node)

1

////////////////////////// HX characteristic curves //////////////////////////

#	Filename
1	Intercooler.hxc

APPENDIX C - PROBLEM DEFINITION (CD-ROM)

This appendix contains a detailed definition of all the validation and/or comparison models that were used in this study. The problems (models) will be indexed similar to the chapter index to avoid confusion. The problems defined here are found on the CD-Rom (Appendix C – Problem Definition) included in this thesis. The problem definitions include the network files of Flownex and Xnet. Below is a brief summary (index) of the problems contained on the CD-Rom.

CHAPTER 4 – Thermal-Fluid Network Solver

Section 4.4.1 Comparison Of Fluid Flow Elements

Section 4.4.1.1 PipeDW – Flow Element

Section 4.4.1.2 PipeDG – Flow Element

Section 4.4.1.3 PipeHX – Flow Element

Section 4.4.1.4 PipePB – Flow Element

Section 4.4.2 Comparison Of Heat Transfer (Convective)

Section 4.4.2.1 Test Case 1 – Steady-State

Section 4.4.2.1 Test Case 1 – Transient

Section 4.4.2.1 Test Case 2

Section 4.4.2.1 Test Case 3 – Steady-State

Section 4.4.2.1 Test Case 3 – Transient

Section 4.4.2.1 Test Case 4

Section 4.4.1 Combinations Of Fluid Flow And Heat Transfer

Section 4.4.3 Sudden Valve Closure At The End Of A Pipe

Section 4.4.3 Pressure Equalisation In Tanks

CHAPTER 5 – Modelling Of Parallelflow And Counterflow Heat Exchangers

Section 5.4 Discretisation Topologies

EES Node-Element Topology Code

EES Node-Node Topology Code

EES Analytical E-NTU Code

Flownex Node-Element Topology Networks

Flownex Node-Node Topology Networks

Flownex RX Element Network

Section 5.5 Comparative Study

Section 5.5.1 Steady-State Comparison

Section 5.5.2 Transient Comparison

CHAPTER 6 – Modelling Of Complex Heat Exchangers

Section 6.5 Comparative Study

Section 6.5.1 Steady-State Comparison

Section 6.5.2 Transient Comparison

APPENDIX D - DISCRETISATION STUDY HEAT EXCHANGER SPECIFICATION

This appendix contains the heat exchanger specification used for the discretisation study in Chapter 6.

C.1 DISCRETISATION HEAT EXCHANGER SPECIFICATION

Type		Conventional E-Type Shell-And-Tube Heat Exchanger – Config 0
Shell side variables		
Number Of Gas Passes [ ]		4
Shell Diameter [m]		0.5
Shell Length [m]		2.00
Void Fraction [ ]	0.7	
Sigma [ ]		0.4
Kin [ ]		0.00
2 tube bundles:		
T1 [m]		0.20
Z1 [m]		0.0333
T2 [m]		0.20
Z2 [m]		0.0333
4 tube bundles:		
T1 [m]		0.10
Z1 [m]		0.02
T2 [m]		0.10
Z2 [m]		0.02
T3 [m]		0.10
Z3 [m]		0.02
T4 [m]		0.10
Z4 [m]		0.02
Tube side variables		
Number Of Shell Passes [ ]		2 & 4
Number Of Parallel Circuits [ ]		1,2,3,4…
Number Of Pipes Per Pass [ ]		200
Pipe Inside Diameter [m]		0.010
Length Per Pass [m]		2. 2

Inside Roughness [μm]	30
K_i [ ]	0.0
K_o [ ]	1.0

Common variables

Heat Capacity [kJ/kg K]	100.0
Gas Heat Area [m^2]	20.0
Area Ratio [ ]	10.0