AN ABSTRACT OF THE DISSERTATION OF

Brian J. Daniels for the degree of Doctor of Philosophy in Mechanical Engineering

presented on September 10, 2008.

Title: A Study of Adiabatic and Diabatic Flow Boiling in Parallel Microchannels and Fractal-like Branching Microchannels

Abstract approved:

_____

James A. Liburdy

A one-dimensional numerical model has been developed to study diabatic and adiabatic flow boiling in microchannels. This model accounts for developing flow effects and variable property effects. The model uses correlations for void fraction and two-phase multipliers found in the literature. The model has been used to study the performance differences between diabatic flow in single-phase and two-phase flow in fractal-like branching channels. It has also been used to examine the performance differences between diabatic two-phase flow boiling in these same fractal-like branching channels and straight parallel channels with identical wall surface areas. The model also was used to study pressure drop and exit quality differences for adiabatic two-phase flow boiling

between fractal-like branching channels, with different geometries than the diabatic cases, and parallel channels again with identical wall surface areas. Results from experiments examining flow boiling in fractal-like branching channels are also reported. These results include channel pressure drop and void fraction for mass flow rates ranging from 100 to 225 g/min and inlet subcooling levels of 0 to 5 °C. The fractal-like branching network studied had four branching levels, a length ratio of 0.7071, a channel width ratio of 0.7071, a channel height of 150 um, a total channel length of 18 mm and a terminal channel width of 100 μm. The channel pressure drops varied from 20 kPa to 90 kPa. These results were also compared to results from the 1-D model, and the model showed good agreement with the pressure drop results. The agreement with the void fraction results was not as good driven primarily by noise in the experimental measurement.

A Study of Adiabatic and Diabatic Flow Boiling in Parallel Microchannels and Fractal-like Branching Microchannels

by
Brian J. Daniels

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosopy

Presented September 10, 2008

Commencement June 2009

Doctor of Philosophy dissertation of Brian J. Daniels presented on September 10, 2008

APPROVED:

Major Professor, representing Mechanical Engineering

Head of the School of Mechanical, Industrial, and Manufacturing Engineering

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

Brian J. Daniels, Author

ACKNOWLEDGEMENTS

I would like to thank my wife, Patricia, and family for their patience, understanding, and the sacrifices they have made which enabled me to devote the time, and energy necessary to complete this dissertation. I would also like to thank my major professor, Jim Liburdy, and my entire committee for their encouragement and guidance through this process.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF TABLES

NOMENCLATURE

$ALLP$	liquid pixel count from all liquid image	P_v	Under pressure of vaporization
A_w	Wall Area	ΔP	Pressure drop
C_{LM}	Phase Interaction Parameter	$\dot{q}''$	Wall heat flux
D	Diameter	Re	Reynolds number
D_h	Hydraulic Diameter	T	Temperature
f_f	Fanning friction factor	T_{sub}	Inlet Subcooling
G	Mass flux through channel	u	Velocity
h	Convective film coefficient	v	Specific volume
h_c	Channel height	VF	Void fraction (experimental)
$\hbar$	Enthalpy	V	Velocity
k	Branching levels	x	Quality
L	Length	X^2	Lockhart – Martinelli parameter
L_k	Length of kth channel	$(1-y)$	mass fraction of superheated fluid
L_{tot}	Total length of all branching channels	w	Quality
LP	Liquid pixel count from data image	w_t	Width of terminal branching channel
$\dot{m}$	Mass flow rate through channel	w_k	Width of channel in kth branching level
P	Pressure	z	Streamwise coordinate
P_{sat}	Saturation pressure		

NOMENCLATURE (Continued)

α	Void fraction (analytical)	γ	Channel length ratio
β	Homogenous void fraction	λ	Friction factor
β_k	Channel width ratio	λ_{app}	Apparent friction factor for developing flow
ζ	Dimensionless length	μ	Viscosity
ε	Performance parameter – ratio of heat dissipated to pumping power	ϕ^2	Two-phase multiplier
		ρ	Density

Subscripts

a	Acceleration	*l*	Liquid basis
b	Bulk	*lo*	Liquid only basis
exit	Evaluated at channel exit	*tp*	Two-phase
f	Friction	*v*	Vapor basis
inlet	Evaluated at channel inlet	*w*	wall

A STUDY OF ADIABATIC AND DIABATIC FLOW BOILING IN PARALLEL MICROCHANNELS AND FRACTAL-LIKE BRANCHING MICROCHANNELS

Introduction

Problem Statement

Microchannels have been shown to have much greater heat transfer rates than either miniscale or macroscale channels. This advantage however comes at the cost of increased pressure gradients. In macroscale flows, large length to diameter (L/D) ratios (100 or greater) are common, and since the pressure gradients are small they do not generate large pressure drops. Because of these small pressure drops, the fluid properties such as viscosity, enthalpy of vaporization, saturation temperature, etc, do not change significantly from the entrance to the exit and the assumption of constant fluid properties is justified. However in microscale flows, with their large pressure gradients, the fluid properties can change significantly from the entrance to the exit, therefore it is important to include these effects into a microscale pressure drop model. Also due to these increased pressure gradients, flow boiling in microchannels can have significant phase change due to the pressure drop alone (flashing). The current 1-D models developed for macroscale flows assume constant properties, and therefore include no effect due to flashing. To accurately predict pressure drop and heat adsorption capabilities in two-phase micro-channel heat transfer, it is necessary first to quantify this flashing effect in microscale two-phase flow, and second to include these effects into a predictive model for micro-scale two-phase flow boiling. One of the goals of this work is to quantify the flashing effect by experimentally studying adiabatic flow boiling in micro-channels.

Another goal of this work is to develop a predictive 1-D model which included these flashing effects.

Because of their small scale, the flow through microchannels tends to be laminar. Laminar flows have longer developing lengths than the turbulent flows commonly found in macro-scale flows. Current 1-D models developed for microscale flow boiling assume laminar flow, but also assume constant friction factors associated with fully developed flows. The large pressure gradients of micro-channel flows make small L/D ratios attractive to minimize the total pressure drop. This in combination with the small Reynolds number make it important to include developing flow effects into the predictive 1-D model for microscale flow boiling. Inclusion of developing flow effects is another goal of this work.

The ability to predict the pressure drop due to flow boiling in microchannels is critically important to the ability to design systems which make use of these devices. As an example, a portable heat pump used to provide cooling for personnel in Hazardous Material suits could make use of microchannel flow boiling in both the evaporator and desorber components. It is necessary to be able to predict the pressure drop through these devices in order to asses the system performance and operating pressures, size pumps, and accomplish detailed designs of the components. In addition to knowing the pressure drop, it is also important to be able to quantify how much of the phase change is due to heat addition.

In this work, the development of a predictive 1-D model for pressure drop, quality, and void fraction in adiabatic and diabatic flow in parallel as well as fractal like-

branching channels are discussed. This model will be used to compare single phase flows to two phase flows in fractal-like branching channel heat sinks. It will also be used to compare diabatic flows in parallel channel heat sinks to fractal-like branching channel heat sinks. The model will also be used to see the effects of geometry variations in these fractal-like branching channel flow networks.

Adiabatic flows will also be examined with this model. Particularly the comparison of parallel channel flows to fractal-like channel flows as well as the effect of particular fractal-like branching channel parameters such as length ratio, and branching levels.

An experimental study of adiabatic flow in a specific fractal-like geometry is also described. The relative size of the frictional and acceleration components of the pressure drop will be discussed. Void fraction measurements obtained by imaging the two-phase flow will also be presented. The global pressure drop and void fraction measurements will be compared to the 1-D model predictions.

Thesis goals

Some heat sink designs such as the branching channel networks developed by Pence have relatively small length to diameter ratios, and they have shown that inclusions of these effects are critical for accurate predictions with single phase flows. A 1-D model has been developed to predict the pressure drop in the fractal-like branching channel models. This model uses correlations for the void fraction and two-phase multiplier developed for mini and microscale channels. It also includes a variable friction factor to account for developing flow effects at the beginning of each branching level. As

validation, the model is compared to for macroscale, miniscale and microscale data found in the literature. The model will also be used to compare the performance of single phase flow and two phase flow through the fractal-like branching channels. It will also be used to compare the performance of a parallel channel heat sink to a fractal-like channel heat sink. Experimental results for adiabatic flow boiling in a fractal-like heat sink will also be reported and compared to the model results.

Literature Review

Flow boiling in microchannels

Peng and Wang [1] were among the first investigators of flow boiling in microchannels. They performed tests on rectangular channels with cross section dimensions 600 by 700 μm, and 60 mm long with mass fluxes between 1500 and 4000 kg/m^2-s, and wall heat fluxes between 2 and 100 W/cm^2. They observed that the nucleate boiling heat transfer is intensified in microchannels as compared to channels with diameters on the order of 10 mm. They made no report of pressure drop across the channels. Bowers and Mudawar [2-4] were also among the early investigators of flow boiling in microchannels. Using R-113 as the working fluid they compared the performance of a microchannel heat sink with 510 μm channels and an L/D ratio of 50 to a minichannel heat sink with 2.54 mm channels and an L/D ratio of 10. The microchannel mass flux ranged from 95 to 47 kg/m^2-s and the wall heat flux ranged from 3-33 W/cm^2. They found that the microchannel heat sink could dissipate higher heat fluxes than the minichannel heat sink for the same mass flow rate of cooling fluid. The microchannel heat sink had a much higher pressure drop than the minichannel heat sink, Due to this larger pressure drop, the temperature of the two-phase mixture along the channel, and therefore the fluid properties themselves, were not constant. They also observed that the fluid velocity and Mach number were much higher for the microchannel than the minichannel.

Other investigators of flow boiling in microchannels include Ravigururajan and co-workers [5-8] who looked at flow boiling in parallel channels and a network of

crisscrossing rectangular channels with a height of 1000 μm and a width of 270 μm. Using water they examined mass fluxes between 28 and 342 kg/m^2-s and wall heat fluxes between 5 and 60 W/cm^2. They found that the heat transfer coefficients are high for microchannels and that the heat transfer coefficient and pressure drop were smaller for the crisscrossing channels than the parallel channels. The group lead by Kenny, Goodson, and Santiago [9-11] at Stanford have also been looking at flow boiling as a way to cool VLSI chips and have had success in experimentally and numerically characterizing the performance of these heat sinks with channel sizes on the order of 25-50 μm. They examined the mass flux range of 80-375 kg/m^2-s and the wall heat flux range of 6-24 W/cm^2. Jiang et al also investigated boiling curves in microchannels and found the behavior different that that found in macroscale channels in that the typical boiling plateau (increasing heat flux for relatively constant device temperature) is not present for microchannels. The advantage of high heat transfer coefficient for flow boiling in microchannels is the same as for single-phase flows. However, the disadvantage of larger pressure gradients is also the same as for single-phase flows.

Fractal branching networks

One method which has been used to reduce pressure drop in micro-channel flows is the use of fractal-like branching channels developed by Pence [12]. These channels mimic the distribution network patterns found in nature. Examples of these networks with asymmetric and symmetric branching are shown in Fig 1. The width and length of the channels in adjacent levels are related through fixed ratios per the following definitions:

$$\beta = \frac{w_{k+1}}{w_k} \tag{1}$$

$$\gamma = \frac{L_{k+1}}{L_k} \tag{2}$$

where β is the width ratio, w_k is the width of the k^{th} level channel (the $k = 0$ level originates at the inlet plenum), γ is the length ratio, and L_k is the length of the k^{th} level channel. The total length, L_{tot} of the channels in the network is obtained from the length of the final branching level, L_m through the following

$$L_{tot} = L_m \sum_{i=0}^{m} \frac{1}{\gamma^i} \tag{3}$$

Pence [12] has shown that for single phase flows the fractal-like branching channels have smaller pressure drops for identical flow rates than straight rectangular channels with the same wall surface area.

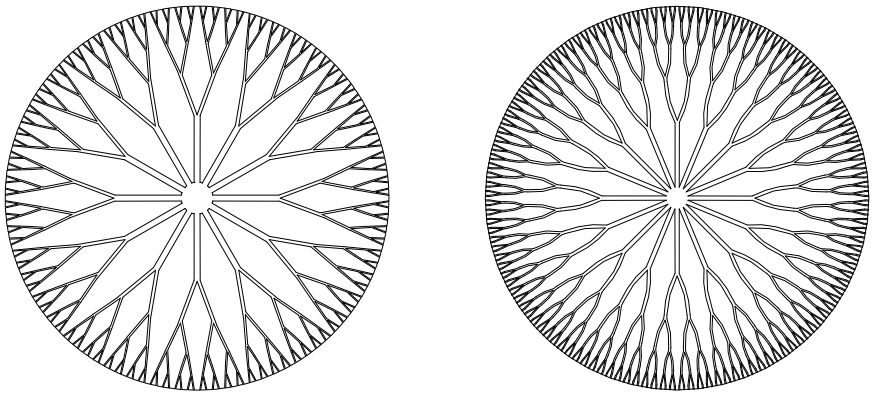


Figure 1. Schematic layout of fractal-like channel networks in a circular heat sink. Showing a) asymmetric branching and b) symmetric branching

Two-Phase Pressure Drop Modeling

Accurate predictions of the pressure drop due to flow boiling in microscale channels is critical for the selection of channel construction materials, manufacturing process, as well as sizing of other system components, such as pumps, fittings and tubing materials. The two important parameters used in predicting pressure drop in flow boiling are the void fraction and the two-phase multiplier. These parameters are used in 1-D models for the calculation of the acceleration and friction component of pressure drop, respectively. One-dimensional models fall into two basic categories: homogenous and separated flow models. Homogenous models assume that the liquid and vapor phase travel at the same velocity. Separated flow models assume the liquid and vapor phase travel at different velocities. For most one-dimensional models the frictional component of the two-phase pressure drop can be expressed as the product of a two-phase multiplier, ϕ^2 and a single phase pressure drop. Three definitions follow

$$\phi_v^2 = \frac{\left.\frac{dP}{dz}\right|_{tp}}{\left.\frac{dP}{dz}\right|_{vapor}} \tag{4}$$

$$\phi_l^2 = \frac{\left.\frac{dP}{dz}\right|_{tp}}{\left.\frac{dP}{dz}\right|_{liquid}} \tag{5}$$

$$\phi_{lo}^2 = \frac{\left.\frac{dP}{dz}\right|_{tp}}{\left.\frac{dP}{dz}\right|_{lo}} \tag{6}$$

where ϕ_v^2, ϕ_l^2, and ϕ_{lo}^2 are the gas phase, liquid phase and liquid only two-phase multipliers. The gas phase multiplier, ϕ_v^2, uses the flow rate of the gas phase flowing through the channel as the basis for the single phase pressure gradient. The liquid basis multiplier, ϕ_l^2, uses the flow rate of the liquid phase as the basis for the single phase gradient. The liquid only basis multiplier, ϕ_{lo}^2, assumes the entire flow rate as a liquid for the basis for the single phase pressure gradient. This last form is the easiest to use since this flow rate will not change along the channel where the gas and liquid phase flows do change along the channel.

Macroscale Channels

Macroscale channels are defined as having characteristic dimensions larger than 3.0 mm. In the simplest of the homogenous flow models, the two-phase multiplier can be expressed in terms of a constant two-phase Darcy friction factor, λ. A value between 0.0116-0.0132 has been suggested by Collier [13] for macroscale low pressure flashing flows of steam-water mixtures. Another has been developed by Stanley [14] for microchannel flows. Other homogenous flow models utilize a two-phase multiplier which is a function of the two-phase mixture specific volume and two-phase viscosity models. Viscosity models developed by Dukler et al. [15], McAdams et al [16]., Cicchitti et al. [17] or Lin et al. [18] are the most commonly used and are documented in the texts by Collier [13] and Wallis [19] and Chisholm [20].

Most separated flow models are based on the Lockhart-Martinelli parameter, X^2, defined as the ratio of the pressure drop from the liquid phase to the pressure drop due from the vapor phase flowing in the pipe.

$$X^2 = \frac{\left.\frac{dP}{dz}\right|_{liquid}}{\left.\frac{dP}{dz}\right|_{vapor}} \tag{7}$$

Lockhart and Martinelli [21] plotted curves for the liquid phase and gas phase two-phase multipliers, ϕ_l^2 and ϕ_v^2, respectively, as functions of X^2 for all the combinations of laminar and turbulent gas and liquid phases. No correlation is provided for the relationship between the two phase multiplier and X^2, which limits it's usefulness since the two-phase multiplier must be integrated to determine the pressure drop. Martinelli and Nelson [22] improved on the Lockhart and Martinelli work in two ways, first by providing plots of the liquid only two-phase multiplier which is simpler to use, and by providing plots of the integral of the two-phase multiplier as a function of the exit quality and pressure. Tabular values were also provided, but again a functional relationship between the exit quality, pressure and the two phase multiplier is not provided. Chisholm, [23,24] and, Chisholm and Laird [25] provided an analytic expression for the curves provided in Lockhart and Martinelli [21]. This analytic expression has the following form.

$$\phi_l^2 = 1 + \frac{C_{LM}}{X} + \frac{1}{X^2} \tag{8}$$

where the value for C_{LM} depends on whether the liquid and vapor phases are in the laminar or turbulent regimes. The Lockhart-Martinelli method and the Chisholm-Laird method are well documented in the texts by Collier [31], Wallace [19] and Chisholm [20].

The assumptions inherent in all these models are that the flow is fully developed, (i.e. the value of the friction factor is constant), and that the pressure drop is small enough that it does not affect the fluid properties (e.g. liquid viscosity, specific volumes, saturation temperature, heat of vaporization, are all constant.).

Miniscale Channels

Miniscale channels are defined as having hydraulic diameters between 0.8 and 3.0 mm. Models for the void fraction and two-phase multiplier have been developed for miniscale channels. Two researchers have developed theoretical predictions of void fractions for this scale, Armand [26] and Zivi [27]. Armand's correlation [26] is based on the homogenous void fraction, while Zivi's [27] is based on the quality and density of each phase. Experimental investigations by Zhao and Bi [28] show that for air-water flow through vertical triangular channels with hydraulic diameters on the order of 1mm, the Armand correlation matches the measured data well. Triplet et al [29] looked at void fractions of gas/liquid flows through circular and triangular channels on the order of 1 - 1.5 mm. Comparing the results to several macroscale correlations they found reasonable agreement. Comparing the pressure drop, they found the homogenous mode worked well for the bubbly and slug flow regimes. Bao et al [30] performed experiments with air/water flows in circular channels ranging in size from 0.74 – 3 mm. They compared their measured void fraction against several macroscale correlations and found the Lockhart-Martinelli correlation matched the data well. They also compared the pressure drop to several macroscale correlations and found the Chisholm method to match the data with the smallest error.

Several investigators have looked at the effect of very small channel sizes on the frictional pressure drop, and have found that the classic Lockhart-Martinelli and Chisholm separated flow modeling methods can be adjusted by modifying the phase interaction parameter. Lee and Lee [31] studied air-water flows through channels with hydraulic diameters ranging from 0.4 mm to 4 mm. The test section which followed a long flow settling section had L/D ratios ranging from 50-500. They proposed a correlation for the phase interaction parameter which includes effects due to surface tension, liquid viscosity, inertia, body forces, and hydraulic diameter. Mishima and Hibiki [32] investigated air-water flows in circular tubes with diameters ranging from 1-4 mm. In their study the test section also followed a long settling section and had L/D ratios of 150-400. They developed a correlation for the phase interaction parameter which is only a function of the channel diameter.

Several Researchers have looked at flow boiling in miniscale channels and compared these to available correlations. Shuai et al. [33] examined rectangular channels 2mm wide, 330 mm long with hydraulic diameters of 0.8 and 2.7 mm, with mass fluxes between 100 and 700 kg/m^2-s and wall heat fluxes between .75 and 11 W/cm^2. They studied the heat transfer rate and pressure drop. They compared their pressure drop results to the correlation developed by Chisholm and Laird [25] and found they agreed within $\pm$ 30%. Kaminaga et al. [34] studied both flow boiling and gas/liquid two-phase flows in 1.45 mm diameter tubes 100 mm long. They compared their results against the models of Chisholm, Mishima & Hibiki, and a homogenous model. They found all three lacking, and developed their own model by adding a coefficient to the $1/X^2$ term in the

Chisholm and Laird correlation. Currently all the studies proposing correlations for the two-phase multiplier have been performed on channels where entrance effects have been eliminated, and therefore are based on the same assumptions as their macroscale forbearers.

Adiabatic Capillary Flow Boiling

Adiabatic flow in capillary tubes has been utilized in refrigeration systems for many years to control the mass flow rate of refrigerant through the evaporator. These capillary tubes generally have inside diameters on the order of 1mm and lengths on the order of 1 m. The flow in these tubes generally starts as single phase subcooled liquid then as the pressure drops below a critical level some of the liquid flashes and two-phase flow ensues. Whitesel [35,36] investigated adiabatic flow of refrigerants through capillary tubes and developed empirically based formulas to predict the mass flow rate in capillary tubes given the geometry and the inlet and exit pressure. There was still significant variation between predicted and measured flow rates. The difference ranged from -10% to 14%, indicating that for identical inlet and exit conditions the measured mass flow rate varied. Mikol [37] and Mikol and Dudley [38] found that the flow does not phase change immediately when the static pressure drops below the saturation pressure, but that it remains in liquid phase until a vaporization pressure is reached. Non-equilibrium phase change then occurs for a small distance and the pressure, temperature and quality asymptotically reach equilibrium conditions.

The region of the capillary where the pressure is below the saturation pressure is defined as the metastable region. Recently, Meyer and Dunn [39] studied the behavior of

this metastable region as a function of the inlet liquid subcooling. They found that for the same subcooling, the mass flow rate obtained depended on whether this subcooling was approached from above or below with the approach from below resulting in a lower flow rate. The difference in flow rate was largest for small levels of inlet subcooling. Bittle et al. [40] confirmed the results of Meyer and Dunn, and showed that by providing nucleation sites in the form of either a small diameter wire or small holes in the tube wall the variation in mass flow rate was significantly reduced.

Chen et al. [41] applied nucleation theory to adiabatic flow in capillary tubes and developed an expression for the amount of under pressure required for vaporization to begin. Experimental data for R-12 were used to develop a correlation for the heterogeneous nucleation factor. When this is substituted into the formula developed from the nucleation theory, it results in an expression for the under pressure of vaporization based on fluid properties, tube geometry, and flow conditions. Although this correlation has been developed specifically for R-12, it has been applied in flow models of other refrigerants [42], and the flow models have shown good agreement between predicted and measured flow rates. The Chen et al. correlation has also been used as the basis for a diabatic under pressure correlation developed by Chen and Lin [41] for R-134a. Chen and Lin only observed a metastable region below a critical heat transfer rate. They examined a specific configuration of a capillary expansion tube attached to a compressor suction tube. The usefulness of this study is limited to this specific application and the method used to characterize the heat transfer rate between the tubes (the ratio of the mass flux in the capillary tube to the mass flux in the suction tube).

A model for the subpressure of vaporization for water has also been developed by Lackme [43] particularly for the blow down of high pressure subcooled liquid through cracks in pipe walls. Hardy and Mali [44] introduce a correlation developed by Seynhaeve for lower pressure flows. A model for the rate of change of the amount of superheated liquid water has also been developed by Feburie et al. [45]. Wong and Ooi [46] compared various homogenous two-phase viscosity models against data from other adiabatic capillary researchers and found that the Dukler viscosity model best matches the data.

Prediction models for single and two-phase flow through capillary tubes have been developed by several researchers [42, 47, 48] using various means of simulating the metastable region. However all have used homogenous flow models with either the Dukler [15], McAdams [16], or Cicchitti [17] two-phase viscosity models, and have shown good agreement with experimental data.

Microscale Channels

Microscale channels are defined as having characteristic dimensions below 0.8 mm. An investigation by Serizawa et al [49] for air-water and steam water flow in circular tubes with diameters ranging from 20-100 μm found that the Armand correlation matches the trend of the data well. The experimental work by Kawaji and co-workers [50-54] looked at the pressure drop, flow patterns, and void fraction for nitrogen-water adiabatic flows in channels with diameters ranging from 50 μm to 530 μm. They also examined circular and square channel geometries with diameters on the order of 100 μm. From these investigations, they have concluded that the Armand type correlation holds

for channels sizes down to channel diameters of about 250 μm [50]. For smaller channels, diameters of 100 μm and less, they have developed a correlation based on the homogenous void fraction with two fit parameters that matches the measured void fraction data well [51-55]. The fit parameters are not affected by the channel geometry (circular or square), they are however effected by the channel diameter [50]. They also observe that there is a significant effect due to viscosity at high superficial gas velocities [56] Kawaji and co-workers [50-54] show that the correlations developed by Lee and Lee [31] and Mishima and Hibiki [32] also work well for gas-liquid flows in microscale channels. Qu and Mudawar [57-59] investigated flow boiling in rectangular channels for mass fluxes between 135 to 400 kg/m^2-s and wall heat fluxes between 5 and 50 W/cm^2 with a hydraulic diameter of 348 μm and a L/D ratio of about 1300 also show that the correlations by Lee and Lee and Mishima and Hibiki match the measured data well. To better match the experimental data Qu and Mudawar developed a modification of the Mishima and Hibiki correlation to include the effect of mass flux through the channel. Lee and Mudawar [60] examined flow boiling of R-134a in identical channels as Qu and Mudawar, and compare several two-phase viscosity models and several separated flow models to their measured pressure drops. They find that the Ciccihitti two-phase viscosity model has the lowers mean error but does not represent the trend of the data. The macro scale models have lower mean errors that expected. They also develop a completely new correlation for the phase interaction parameter which shows good agreement with the R-134a data set as well as the water dataset of Qu and Mudawar. Revellin and Thome [61] studied two-phase flow of R-134a and R245fa

through circular microtubes for flow rates which spanned the laminar and transition regimes. The compared their results with the available correlations in the literature and found the agreement in the transition regime lacking so they developed friction factor correlations based on the two phase Reynolds number for each tube diameter studied.

For adiabatic flows, it has been shown that vaporization occurs at a pressure lower than the saturation pressure, P_{sat}. This vaporization pressure, or under pressure of vaporization, P_v, is due to the minimum energy required to form a vapor bubble. Lackme [62] developed the following correlation to predict the vaporization pressure

$$P_v = kP_{sat} \tag{9}$$

where, *k* has a value between 0.96 and 0.98, 0.97 will be used for this study. Once vaporization begins, it occurs under non-equilibrium conditions and asymptotically reaches equilibrium conditions. The rate of change of super heated liquid has been described by Febure et al.[63]. They defined a new variable to track the mass fraction of saturated liquid, *y*, such that the mass fraction of super heated fluid is expressed by *(1-y)*. The rate of change of the mass fraction of saturated fluid is described by the following differential equation

$$\frac{dy}{dz} = \frac{K}{D_h}(1-y)\left(\frac{P_0 - P_{sat}}{P_C - P_{sat}}\right)^{\frac{1}{4}} \tag{10}$$

where *K* is a constant equal to 0.08, and P_C is the critical pressure. Although this effect has been documented, it will not be incorporated into the 1-D model which is developed.

Numerical Model

Introduction

A one dimensional numerical model to predict pressure, quality, void fraction and bulk fluid temperature along a microchannel network in two-phase flow has been developed. The model uses 1-D void fraction and either two-phase viscosities or two-phase multiplier correlations available in the literature, as well as the incorporating effects due to variable fluid properties, and developing flow. In this chapter the theory behind the model is explained as well as the implementation of that theory into the numerical code. The model is validated against tabular data available in the literature. Model results for three different comparisons are also discussed. These comparisons are: single and two-phase diabatic flows through fractal-like branching channels; diabatic two-phase flows through parallel and fractal-like channels; and adiabatic two-phase flows through parallel and fractal like channels

Theory

As stated in the introduction, there are well established one-dimensional modeling methods for the prediction of flow boiling pressure drop in macroscale channels. The model that has been developed to predict the pressure drop in mini- and microscale channels has expanded on these macroscale methods. Following the macroscale methodology, the pressure gradient is written as the sum of the gradient due to acceleration of the fluid due to the phase change and the gradient due to frictional effects. Integration of the pressure gradient results in the pressure drop being the sum of the pressure drop due to acceleration and the pressure drop due to frictional effects.

The pressure drop resulting from the acceleration of the fluid due to the phase change, ΔP_a, constant area channel along the direction of the flow is derived from.

$$\frac{dP_a}{dz} = -\rho u \frac{du}{dz} \tag{11}$$

When this is integrated from point 1 to 2 along the flow under two phase flow conditions results in:

$$-\Delta P_a = G^2 \left(\left. \frac{w^2 v_v}{\alpha} \right|_{z=z_2} - \left. \frac{w^2 v_v}{\alpha} \right|_{z=z_1} + \left. \frac{(1-w)^2 v_l}{(1-\alpha)} \right|_{z=z_2} - \left. \frac{(1-w)^2 v_l}{(1-\alpha)} \right|_{z=z_1} \right) \tag{12}$$

where, G is the mass flux through the channel, x is the vapor quality, v_v and v_l are the vapor and liquid phase specific volumes respectively and α is the void fraction. The only parameter in this equation that is not either a thermodynamic property or can be calculated directly from the flow rate and channel geometry is the void fraction. The void fraction is partially a function of the thermodynamic properties, but it is also dependent on the two-phase flow regime (e.g. homogenous, separated, slug, ring, annular, etc.). Therefore in order to accurately model the acceleration pressure drop, the void fraction must be accurately modeled. If the two-phase flow is considered homogenous, then the void fraction is only a function of the thermodynamic properties, and it is represented by the symbol, β. If the two-phase flow regime is considered separated, the void fraction is determined from correlations based on either the homogenous void fraction, a flow based parameter such as the Lockhart-Martinelli parameter, or directly from the thermodynamic properties. Table I shows some of the void fraction correlations for macroscale flows, and some of those for mini- and microscale flows.

Table I. Void Fraction correlations

Flow Regime	Source	Formula	Eqn.
Homogenous	Collier [13]	$\beta = \frac{x v_g}{\bar{v}}$	(13)
Separated	Lockhart-Martinelli [21]	$\alpha = 1 - \frac{1}{\sqrt{1 + \frac{20}{X} + \frac{1}{X^2}}}$	(14)
	Armand [26]	$\alpha = 0.833\beta$	(15)
	Zivi et al. [27]	$\alpha = \frac{1}{1 + \left(\frac{1-x}{x}\right)\left(\frac{v_l}{v_v}\right)^{\frac{2}{3}}}$	(16)
	Chung et al. [51]	$\alpha = \frac{0.03\sqrt{\beta}}{1 - 0.97\sqrt{\beta}}$	(17)

The frictional pressure drop gradient is evaluated by multiplying the pressure gradient due to a single phase flow by a two-phase multiplier, ϕ^2, this expression is typically written either on a liquid or liquid only basis.

$$\left.\frac{dP}{dz}\right|_{tp} = \phi_l^{\,2} \left.\frac{dP}{dz}\right|_{l} = \phi_{lo}^{\,2} \left.\frac{dP}{dz}\right|_{lo} \tag{18}$$

Typically for homogenous flows, correlations for the two phase multiplier are given for the liquid only basis. For the liquid only basis, the two phase mass flow rate is used with the liquid density and viscosity to calculate the pressure gradient. For separated flows correlations for the two phase multiplier are given either for the liquid or liquid only basis. For the liquid basis, the mass flow rate of the liquid phase is used with the liquid density and viscosity to calculate the pressure gradient. The liquid only two phase multiplier is more convenient for flow boiling applications since the flow rate of this stream does not change along a closed conduit, where the flow rates of the liquid and vapor phases do change due to the phase change process. The liquid only two-phase

multiplier and the liquid two-phase multiplier can be obtained from one another. Assuming the friction factor can be expressed in the Blasisus form

$$\lambda = \frac{C}{\mathrm{Re}^n} \tag{19}$$

where for laminar flow the constant $C = 64$, and the exponent on the Reynolds number $n = 1$, and for turbulent flows with Re > 2000, $C = 0.375$ and $n = 0.25$. Assuming both phases are either laminar or turbulent and the Blasius relation describes the flows well the relation between the two phase multiplier can be expressed as

$$\phi_{lo}^2 = \phi_l^2 \left(1 - x\right)^{2-n} \tag{20}$$

where n is the exponent on the Reynolds number in the friction factor. If both phases of the flow are not of the same flow regime (i.e. turbulent or laminar), the relationship between the two phase multipliers includes the liquid phase Reynolds number, the vapor phase Reynolds number or both.

Integrating the frictional pressure gradient for two phase flow using the liquid only formulation and the typical macroscale assumption of fully developed flow and constant fluid properties results in:

$$\Delta P_f = \frac{\lambda_{lo} G^2 v_l}{2 D_h} \int_{z_1}^{z_2} \phi_{lo}^2 dz \tag{21}$$

where λ_{lo} is the friction factor calculated using the liquid only basis, and D_h is the hydraulic diameter of the channel. The only parameter in this equation which is not dependent on the flow rate, fluid properties, channel geometry or a combination of these is the two phase multiplier. Just as for the void fraction, the evaluation of the two-phase

multiplier depends on the flow regime. If the flow is homogenous and laminar the two-phase multiplier reduces to

$$\phi_{lo}^2 = \frac{\mu_{tp} v_{tp}}{\mu_l v_l} \tag{22}$$

where, μ_{tp} is the two phase viscosity and v_{tp} is the specific volume of the two-phase mixture and calculated using the quality according to standard thermodynamic relations. Several researchers have developed definitions for the two phase viscosity based on the liquid and vapor phase viscosities and the vapor quality. Table II shows several of these definitions for the two-phase viscosity.

Table II. Two-Phase viscosity correlations

Source	Formula	Eqn.
Lin et al. [18]	$\mu_{TP} = \frac{\mu_l \mu_v}{\mu_v + x^{1.4}(\mu_l - \mu_v)}$	(23)
McAdams et al. [16]	$\frac{1}{\mu_{TP}} = \frac{x}{\mu_v} + \frac{1+x}{\mu_l}$	(24)
Chicchitti et al. [17]	$\mu_{TP} = x\mu_v + (1-x)\mu_l$	(25)

For separated flows correlations are used to calculate the two-phase multiplier based on the channel geometry and parameters of the flow. All the methods used in this work are based on the Lockhart-Marninelli parameter, X^2, and the Chisholm-Laird formulation for the two phase multiplier. The Chisholm-Laird formulation for the two phase multiplier is (equation is repeated from literature review for clarity)

$$\phi_l^2 = 1 + \frac{C_{LM}}{X} + \frac{1}{X^2} \tag{8}$$

where C_{LM} is the phase interaction parameter. The Lockhart-Martinelli parameter is expressed as a function of the quality and fluid parameters

$$X^2 = \left(\frac{1-x}{x}\right)^{n-2}\left(\frac{\mu_l}{\mu_v}\right)\left(\frac{v_l}{v_v}\right) \tag{26}$$

Table III. Phase interaction parameter correlations

Source	Formula	Eqn.
Mishima & Hibiki [32]	$C_{LM} = 21\left(1 - e^{-0.319 D_h}\right)$	(27)

Lee & Lee [31]	$C_{LM} = A\varphi^q \psi^r \,\mathrm{Re}_{lo}^s$
	$\varphi = \frac{\mu_l^2}{\rho_l \sigma D_h}; \quad \psi = \frac{\mu_l j}{\sigma}$
	Laminar-Laminar flow:
	A = 6.933; q = -1.317;
	r = 0.719; s = 0.557;
	Turbulent-Turbulent Flow:
	A = 0.408; q = 0;
	r = 0; s = 0.451;
		(28)

Qu & Mudawar [57]	$C_{LM} = 21\left(1 - e^{-0.319 D_h}\right)(0.00418G + 0.0613)$	(29)

where *n* again is the exponent from the Blasius equation, Eqn (17), and liquid and vapor phase flows both have to be laminar or both have to be turbulent. For macroscale flows, C_{LM} is a constant determined by the liquid and gas phase Reynolds numbers. For example if both the liquid and vapor phase flows are laminar, $C_{LM} = 21$. Correlations for the phase interaction parameter for mini and microscale flows have been developed by several researchers; some of the correlations for mini or micro-scale applications are shown in Table III.

In microchannels, since the channel sizes are small, the flow tends to be laminar. As a result of this the non-dimensional developing lengths are longer than for typical

macroscale flows which tend to be turbulent. Microchannels can also have small length to diameter ratios such as the fractal like branching channels. Due to these reasons, the length of the channel in the developing flow region for microscale channels can be significant compared to the total length of the channel. Also in microscale flows pressure gradients tend to be higher than macro scale flows because of this, property values such as saturation temperature and vapor density can vary significantly along the flow direction. Including developing flow effects and variable properties in the integration of the pressure gradient results in the friction factor, λ, the two-phase multiplier, ϕ^2, and liquid specific volume being functions of the axial coordinate, resulting in the following formula

$$\Delta P_f = \frac{G^2}{2D_h}\int \phi_{lo}^2 \lambda_{lo} v_l dz \tag{30}$$

In order to calculate the integral, local values of the friction factor in the developing flow region must be obtained. The author was unable to find an expression for this in the literature. So the apparent friction factor from Shah [62] which is an average friction factor from the beginning of a channel to some point L along the channel was modified to provide a local friction factor. Shah and London's expression is shown below

$$\lambda_{app} = \frac{1}{L}\int_0^L \lambda dz = \frac{1}{\mathrm{Re}}\left(\frac{13.76}{\sqrt{\zeta}} + \frac{\lambda_0 \mathrm{Re} + \frac{k_\infty}{\zeta} - \frac{13.76}{\sqrt{\zeta}}}{1+\frac{C_0}{\zeta}}\right) \tag{31}$$

where the parameter ζ is a non-dimensional length, and λ_0, k_∞, and C_0 are constants based on the channel geometry. Starting with the definition

$$L\lambda_{app} = \int_0^L \lambda dz \tag{32}$$

and differentiating both sides with respect to L

$$\frac{d}{dL}\left(L\lambda_{app}\right) = \frac{d}{dL}\int_0^L \lambda dz \tag{33}$$

results in an expression for the local friction factor evaluated at L as a function of the apparent friction factor and the derivative of the apparent friction factor evaluated at L

$$\lambda\big|_L = \lambda_{app}\big|_L + L\frac{d\lambda_{app}}{dz}\bigg|_L \tag{34}$$

For the diabatic case, to evaluate the evolution of the vapor quality along the channel, the enthalpy of the flow as it travels along the channel is tracked using an energy balance. The energy balance for the fluid flowing in the channel between points 1 and 2 along the flow is written in terms of the mass flux through the channel as

$$\dot{q}''A_w = GA_f\left(\hbar_2 - \hbar_1\right) \tag{35}$$

where, $\dot{q}''$ is the channel wall heat flux, A_f is the flow area of the channel, A_w is the wall area, and $\hbar$ is the enthalpy. In the two-phase flow region the quality is obtained from the enthalpy from the following thermodynamic relation

$$\hbar = (1-x)\hbar_l + x\hbar_v \tag{36}$$

where $\hbar_l$ and $\hbar_v$ are the saturated liquid and vapor enthalpies evaluated at the local fluid pressure. The local channel pressure is compared to the saturation pressure at the local fluid temperature to determine if the flow is single phase or two phase flow. If the

pressure is above the saturation pressure, then the flow is still single phase liquid flow. Adiabatic flows are simply a special case of diabatic flows where the enthalpy does not change in the flow direction. The transition from single phase to two phase flow is determined for the adiabatic case as the diabatic case. The quality is also determined in an identical way.

For diabatic flows, the channel wall temperature can be predicted from an energy balance across the channel wall

$$\dot{q}'' = h\left(T_w - T_b\right) \tag{37}$$

where h is the convective film coefficient and T_w and T_b are the wall and bulk fluid temperatures respectively. the correlation by Kandlikar [63] is used for the film coefficient.

Implementation

Code has been developed to predict pressure drop for three different channel geometries, straight circular channels, straight rectangular channels, and fractal-like branching channels. To implement the formulas shown in the previous section, the channels were discretized such that there are an integer number of elements along their length. In the case of the fractal-like branching channels, each level is divided into an integer number of elements. Therefore for the fractal-like branching channels, an element does not cross levels, and the branching is assumed to occur instantly between elements. Obtaining an integer number of elements is achieved by selecting the nominal step size, dividing the channel length, or branching level length for the case of fractal

channels, by this step size, then rounding this result to the nearest integer and dividing it back into the channel length to obtain the actual step size.

The inputs required to run the code are, the channel geometry, flow conditions and boundary conditions. For circular channels the geometry inputs required are, the diameter, D, and the length, L. For rectangular channels, the channel height, h, channel width, w, and length, L, are all the geometry parameters needed. For the case of fractal-like branching channels, the channel-to-channel length ratio, γ, the channel-to-channel width ratio, β, the number of branching levels, k, the channel height, h, terminal channel width, w_k, and the total length of the flow path, L, are required. The flow conditions required are the same for all the channel geometries. These are the mass flow rate entering the channel, and whether the flow is adiabatic or diabatic. If the flow is diabatic, the channel wall heat flux is also needed. The boundary conditions are also common for all channel geometries. The required parameters are, the outlet pressure, and since the formulas are evaluated in the flow direction and in order to propagate them in this direction an initial inlet pressure is required to start the calculations, since this value is not know a priori, an initial guess must be provided. The inlet temperature or inlet subcooling must also be provided as boundary conditions to fully define the problem. From these inputs, the code predicts the pressure, vapor quality, void fraction, and the bulk fluid temperature and along the channel.

Based on the initial inlet pressure guess, the code propagates along the flow direction calculating pressures, qualities, void fraction and temperatures to the end of the channels where the exit pressure is calculated. It then compares this exit pressure to the

specified exit pressure, and if they do not agree within the convergence criteria, the code adjusts the inlet pressure guess and repeats the process until the specified exit pressure is reached within the convergence criteria. Once the exit pressure convergence criteria is satisfied, the channel wall temperatures are calculated for diabatic flows.

The code was set up in a modular fashion where the main program calls various sub programs to perform specific tasks. Figure 2 shows a flow chart of the code. The code starts out loading input data such as flow rate, and exit pressure, and model set-up parameters such as channel geometry, and two-phase multiplier model to use. It then sets-up the channel geometry arrays, flow parameter arrays, and then starts to iterate on the channel pressures, qualities, void fractions and bulk fluid temperatures. The code then generates output files and saves the data generated.

All fluid properties were evaluated using correlations based solely on temperature. In the two phase region, the liquid and vapor phase are assumed to be saturated and in equilibrium. As a result, the Pressure and Temperature are no longer independent properties. In the single phase region the liquid transport properties are either solely functions of temperature, or are such weak functions of pressure that the pressure effect can be ignored. Also, the assumption is made that the thermodynamic properties for the subcooled liquid can be evaluated based on the saturated liquid at the same temperature. Correlations from Irvine et al. [64] were used to evaluate the liquid and vapor saturated enthalpy, and specific volumes. The bulk fluid temperatures in the single phase region were evaluated by inverting the correlation for the saturated liquid

enthalpy to solve for temperature. In the two phase region, the bulk fluid temperature was evaluated based on the local pressure using the correlation between pressure and

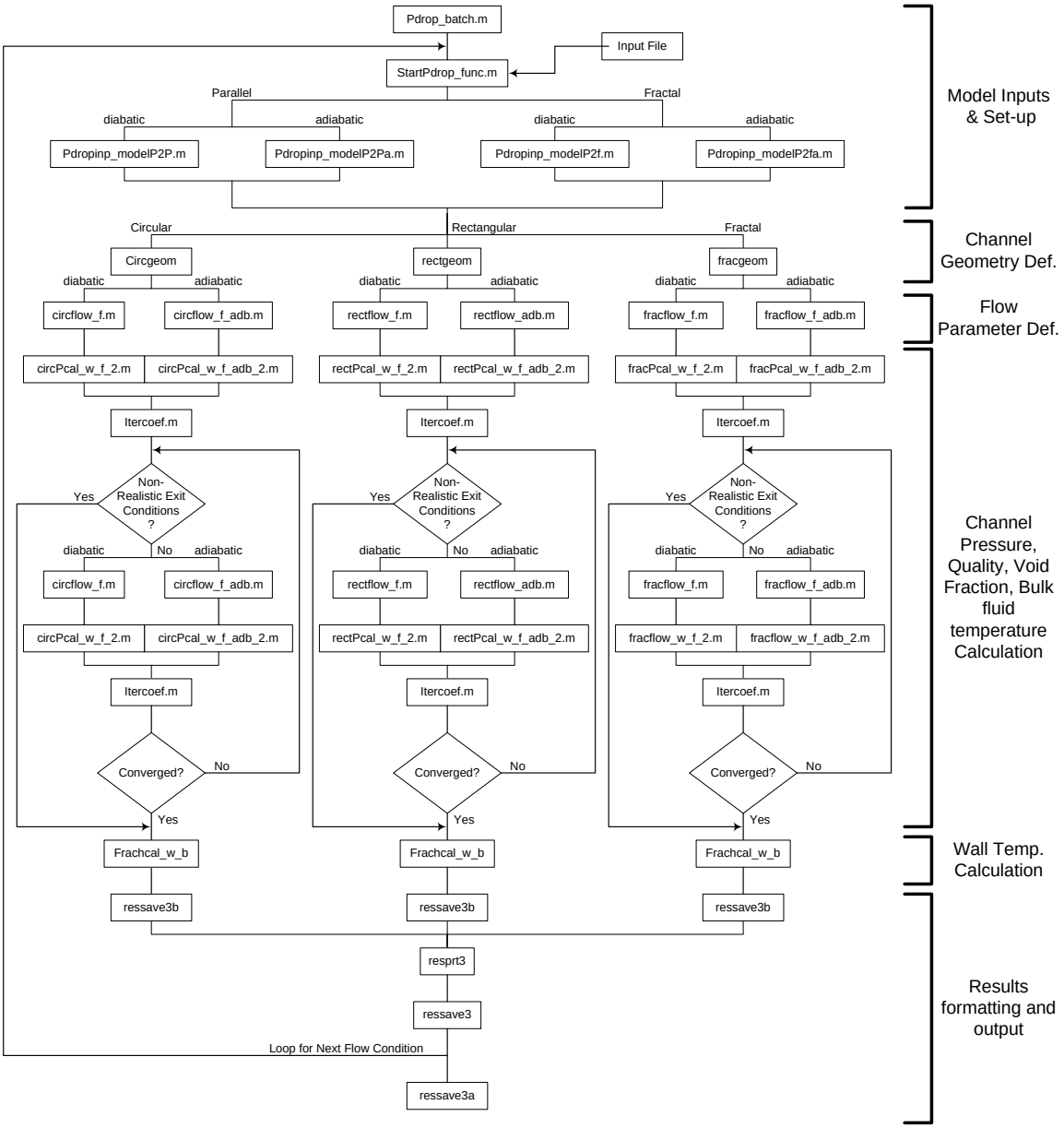


Figure 2. Flow chart of two phase flow model

saturation temperature provided by Irvine et al. [64]. The viscosity, thermal conductivity, and Prandlt number were calculated base on curve fits of data found in Moran and Shapiro [65], Rogers and Mayhew [66] and Cengel[67].

The acceleration and frictional pressure drops are calculated for each element sequentially along the channel, updating the local pressure, temperature, quality and void fraction at the end of each element. The acceleration pressure drop is only evaluated in the two phase region since the change in liquid specific volume is negligibly small in the single phase region. The acceleration pressure drop formula, Eqn. (10), is exact since it results directly from the integration of the acceleration gradient. The frictional pressure drop is evaluated in both the single phase and two-phase regions. For elements in the single phase region the two-phase multiplier is set to 1, and the integral across this element reduces to the liquid phase pressure drop. For elements in the two phase region, the two phase multiplier is evaluated per the appropriate formula resulting in the product of the two phase multiplier and the liquid only pressure drop. In evaluating the frictional pressure drop the integral must be approximated. A trapezoidal integration approximation is used to evaluate the integral. The nominal error of this method is on the order of Δz^2.

In the fractal like branching channels, each channel splits into two smaller channels; however the total flow area of the two smaller channels is larger than the upstream channel. This allows the flow to decelerate which causes a pressure recovery. In the code, the transition occurs instantaneously between two elements, however in the real channels, the transition occurs over a finite length which adds frictional effects

offsetting the pressure recovery. Neither of these effects are currently accounted for in the model.

Validation

Validation of the model occurred in three steps. The first step was to validate the pressure drop calculation methodology with single phase test cases. The model results are compared to analytic results based on methods described in Shah [62] and White [68]. The test cases examined are shown in Table IV, along with the analytical predictions of the pressure drop in kPa. The results of this comparison are also shown in Fig. 3 where the model predicted pressure drop is plotted as a function of the analytically predicted pressure drop. As can be seen the model results match the analytic results extremely well, there is less than 0.5% difference between the model predictions and the analytic predictions.

Table IV. Single-phase validation test cases and analytically predicted pressure drops.

hc [mm]	wc [mm]	G $\left[\frac{kg}{m^2 s}\right]$	Pressure Drop [kPa] L/D = 25	L/D = 50	L/D = 100	L/D = 200
0.5	2.5	50	0.0387	0.0763	0.1514	0.3017
		100	0.0797	0.1550	0.3052	0.6057
		250	0.2160	0.4042	0.7802	1.532
		500	0.4860	0.8640	1.617	3.121
		1000	1.777	1.943	3.456	6.468
1	2	50	0.0207	0.0398	0.0781	0.1546
		100	0.0444	0.0828	0.1594	0.3125
		250	0.1327	0.2296	0.4217	0.8048
		500	0.3315	0.5310	0.9182	1.687
		1000	0.8758	1.326	2.126	3.673
1.5	1.5	50	0.0173	0.0329	0.0641	0.1264
		100	0.0370	0.0693	0.1316	0.2563
		250	0.1186	0.1981	0.3548	0.6670
		500	0.3053	0.4742	0.7922	1.419
		1000	0.8120	1.221	1.897	3.169

The second step was to validate the two-phase multiplier calculation. This was accomplished by comparing the two-phase multiplier results obtained from the model to those found in the literature. This was accomplished for both the homogenous multiplier using McAdams two phase viscosity correlation, and Chisholm's separated two phase macro-scale two-phase multiplier correlation.

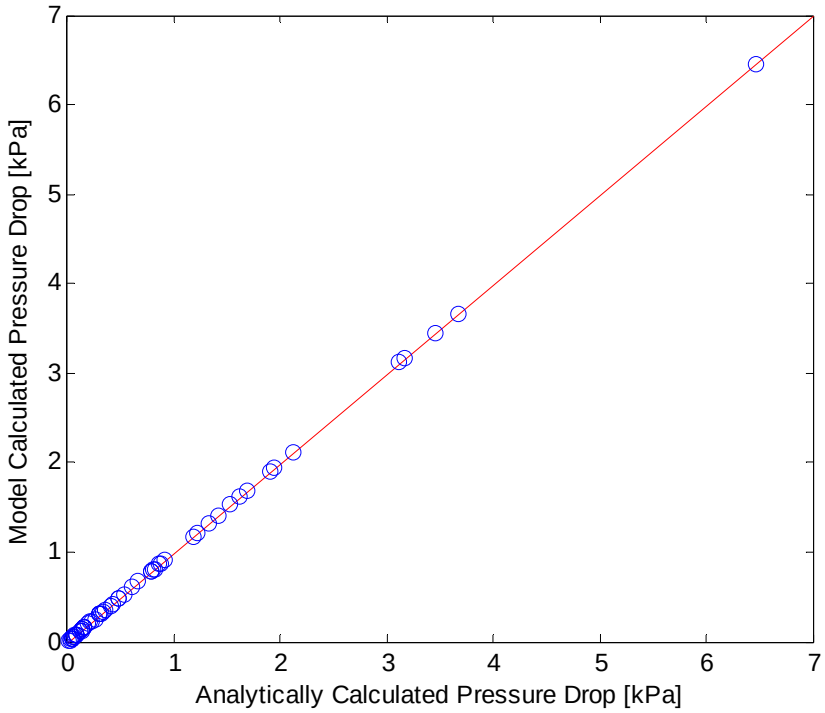


Figure 3. Model predicted single phase pressure drop plotted against the analytic prediction for the pressure drop.

Tabular values are available for the homogenous two phase multiplier [13] these along with the model calculated results and the percent difference are shown in Table V for two different pressures. Figure 4 also shows these results graphically. The percent difference for these two cases never exceed 1.5%, with most values being ~1% or less.

Since tabular values for Chisholm's correlation of the separated flow two-phase multiplier are not available, the results are compared graphically. Figure 5 shows the

model predicted two phase multiplier again compared against the Lockhart-Martinelli data obtained from Table 7-2 in Chisholm [20]. The data is presented in a format similar to Fig. 7-4 from Chisholm [20] which compares the Lockhart-Martinelli two-phase multiplier to two-phase multipliers calculated form real macroscopic flow. The nature of the model calculated two-phase multiplier in Fig. 5 and the nature of the best fit line through the data in Fig. 7-4 from Chisholm [20] are nearly identical indicating excellent agreement.

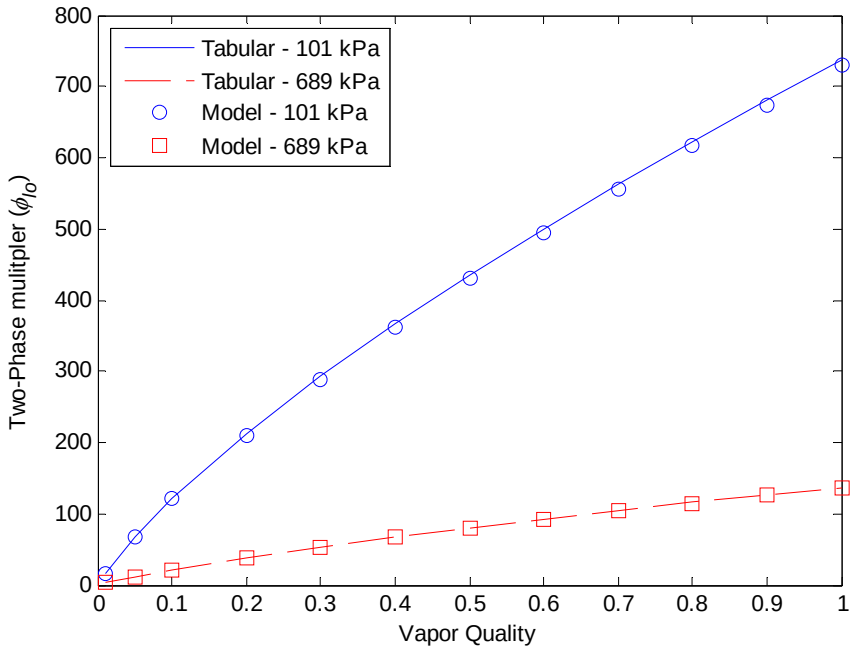


Figure 4. Comparison of Tabulated homogenous multiplier to the model calculated homogenous two phase multiplier. McAdams two phase viscosity was used. Tabular values from Collier [13].

Table V. Comparison of calculated and published homogenous two phase multipliers at two pressures

Quality	P = 101 kPa			P = 686 kPa		
	$\phi_{Collier}$	$\phi_{Calculated}$	% Diff	$\phi_{Collier}$	$\phi_{Calculated}$	% Diff
0.01	16.21	16.18	0.18	3.40	3.40	-0.05
0.05	67.6	67.3	0.50	12.18	12.10	0.69
0.1	121.2	120.3	0.75	21.8	21.6	0.76
0.2	212.2	210.4	0.85	38.7	38.3	1.10
0.3	292.8	289.4	1.16	53.5	53.0	0.88
0.4	366	362	1.12	67.3	66.6	1.03
0.5	435	430	1.17	80.2	79.4	1.04
0.6	500	495	1.10	92.4	91.5	0.98
0.7	563	556	1.17	104.2	103.1	1.04
0.8	623	616	1.10	115.7	114.3	1.18
0.9	682	674	1.18	127.0	125.2	1.42
1.0	738	730	1.05	137.4	135.8	1.20

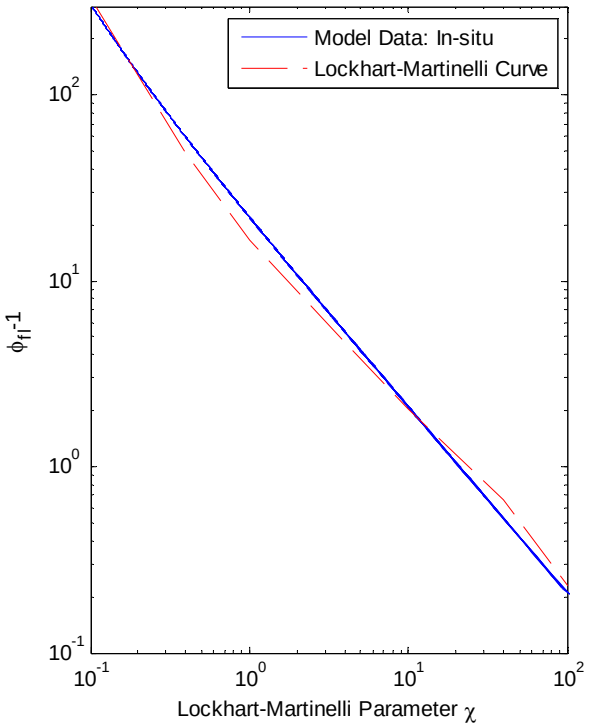


Figure 5. Comparison of published data on Chisholm's separated two-phase multiplier to model calculation of Chisholm's two-phase multiplier.

The last step was to compare the pressure drop predicted by the model to measured pressure drop data reported in the literature. The data of Qu and Mudawar [57] is used for this comparison. Figure 6 show these results. The measured pressure drops and model predicted pressure drops agree within 10% using the both the Mishima and Hibiki [32] and Qu and Mudawar[57] phase interaction parameter correlations. The pressure drop using the Qu and Mudawar [57] correlation for the phase interaction parameter agrees with the measured data better than the pressure drop using the Mishima and Hibiki [32] correlation for the phase interaction parameter, this is to be expected since the Qu and Mudawar [57] correlation is derived based on this data.

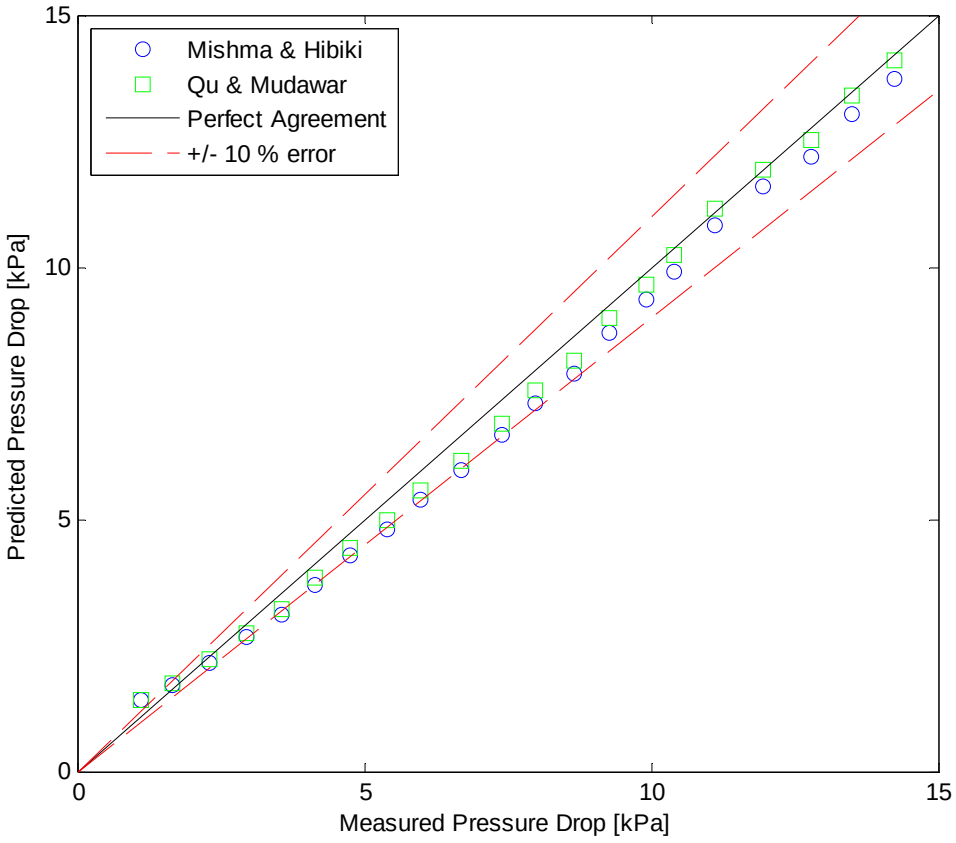


Figure 6. Comparison pressure drop data as reported by Qu & Mudawar [57] to model predicted pressure drop. Model used separated flow correlations developed by both Mishima and Hibiki [32], and Qu and Mudawar [57]

Grid Refinement

A grid refinement study for flow boiling in the fractal like branching channels was also performed. Two geometries were used for this study. Both had $k = 4$, $w_t = 100\mu m$, $L_{tot} = 18mm$, $h_c = 150\mu m$, $\beta_k = 0.7071$, and $N_0 = 16$, one geometry had $\gamma = 0.7071$, and the other had $\gamma = 1.4142$. Table VI shows the inlet mass flow rates, wall heat flux and inlet subcooling conditions used in the grid refinement study for both fractal-like branching channel geometries. Nominal grid sizes of 500, 250, 100, 50, 25, 10 and 5 μm were used in this study. Figure 7 shows the percent change in pressure drop as the nominal grid size is reduced for both the $\gamma = 0.7071$ and $\gamma = 1.4142$ cases. This plot shows that as the grid size is reduced the magnitude of the percent change decreases indicating the code is converging. The figure also shows that for nominal grid sizes below 25 μm, the percent change is less than 0.5%. Figure 8 shows the percent change in exit quality as the nominal grid size is reduced for both the $\gamma = 0.7071$ and $\gamma = 1.4142$ cases. This figure shows the same trends as Fig. 7, however the magnitude of the percent change is smaller, for nominal grid sizes below 25 μm, the percent change in exit quality is below 0.25%. Based on these results a nominal grid size of 10 μm will be used for all future work.

Table VI. Inlet mass flow rate, wall heat flux, and inlet subcooling conditions for grid refinement study of fractal-like branching channels

$\dot{m}_{in}$ $\left[\frac{\text{g}}{\text{min}}\right]$	q''_{wall} $\left[\frac{\text{W}}{\text{cm}^2}\right]$	ΔT_{sub} $[^{\circ}C]$
25	15	0
25	30	0
50	15	0
50	30	0
100	15	0
100	30	0

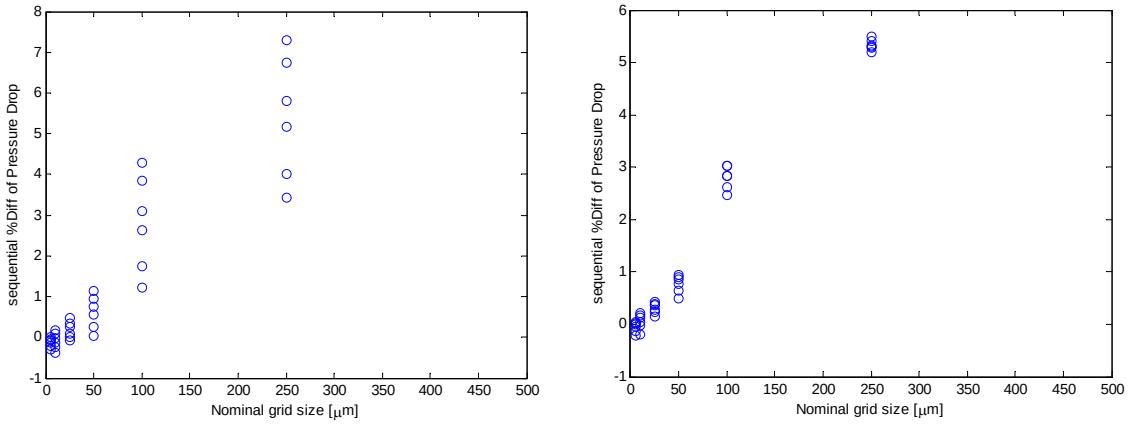


Figure 7. Sequential percent difference of pressure drops for grid refinement study. Examined fractal-like branching channels with a) $\gamma = 0.7071$, and b) $\gamma = 1.4142$.

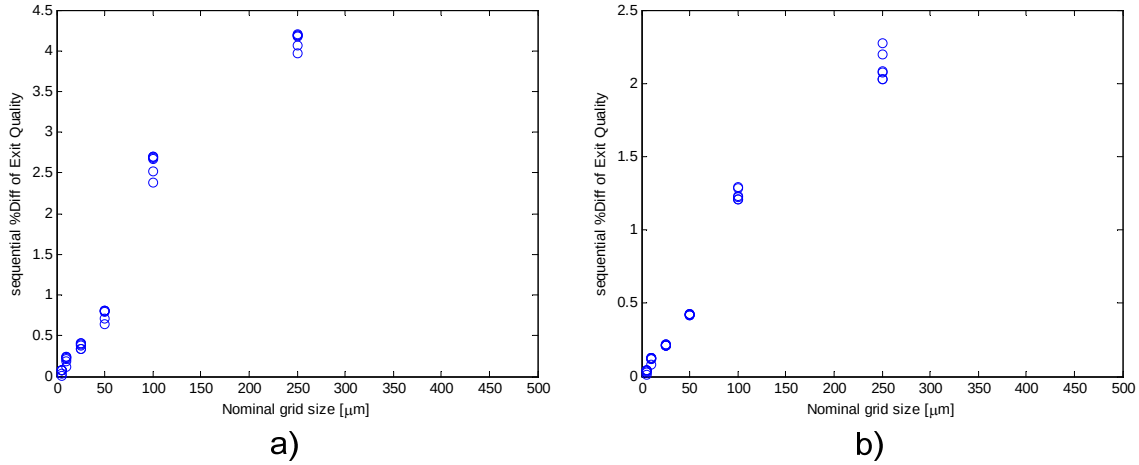


Figure 8. Sequential percent difference of exit quality for grid refinement study. Examined fractal-like branching channels with a) $\gamma = 0.7071$, b) $\gamma = 1.4142$.

Results

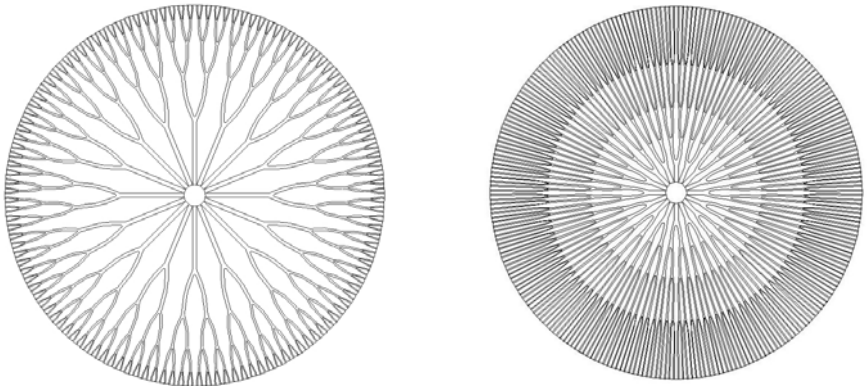
The modeling results are divided into three sections. The first section compares single phase diabatic flows to two-phase diabatic flows through fractal-like branching channels. The second section takes the diabatic two-phase flow results through the fractal-like branching channels, and compares them to results from two-phase diabatic flow through straight parallel channels. The third section compares adiabatic flow through fractal-like branching channels to adiabatic flow through parallel channels.

The ranges of fractal geometries studied in all three sections are similar in that they all examine branching channel length ratios of 0.7017 and 1.4142. They all also examine fractal-like branching channels with 4, 5 and 6 branching levels. All the fractal-like branching channels studied also have the same terminal channel width of 100 μm, and a total channel length of 18 mm. The channel heights studied in the first and second sections were 250, 500 and 750 μm. The channel heights studied in the third section were 100, 150 and 200 μm. The channel heights of the parallel channels studied in the second and third sections match those of the fractal channels in those sections.

Single and Two-Phase Flow Comparison in Fractal-Like Channels

Single and two-phase diabatic flows for fractal like branching channels with a terminal channel widths of 100 μm and total lengths of 18 mm, 4, 5 and 6 branching levels, channel length ratios of 0.7017 and 1.4142, and channel heights of 250, 500 and 750 μm were studied, see Table VII for a complete description of all the geometries examined. The two-phase multiplier model by Qu and Mudawar [57] and the void fraction model of Zivi [27] were used for this analysis. The branching channels are laid

out radially in a circular heat sink as shown in Fig. 9, the number of inlet channels are determined by 1) a minimum channel spacing requirement of center to center spacing of twice the channel width, both within the heat sink and at the periphery, and 2) a maximum ratio between the radius of the inlet plenum and the disk diameter of 6%. All of the fractal-like geometries studied here meet these requirements.



a) b)

Figure 9. Schematic layout of fractal-like channel networks showing effect of channel length ratios. Ratios shown are a) 0.7071, and b) 1.4142

Wall heat fluxes of 5 – 40 W/cm^2 in 5 W/cm^2 increments were applied to both the fractal-like and parallel channel geometries. Mass flow rates of 25, 50, 75 and 100 g/min were examined for the two-phase cases, and mass flow rates of 25, 50, 75, 100, 200, 300, 400 and 500 g/min were examined for the single phase cases. For the two-phase cases, the flow entered the heat sink with zero sub-cooling, and for the single phase cases the flow entered the heat sink at 20 °C. For the single phase to two-phase flow comparison, the single phase mass flow rates were varied over a broader range so a fairer comparison of all the performance criteria could be made.

Table VII. Fractal geometries used for diabatic single phase to diabatic two-phase flow studies

Geometry #	β_k	γ	Branching Levels	Channel Height	Terminal channel Width	Total Channel Length	Number of Inlets
F-1	0.7071	0.7071	4	250	100	18	16
F-2	0.7071	0.7071	4	500	100	18	16
F-3	0.7071	0.7071	4	750	100	18	16
F-4	0.7071	0.7071	5	250	100	18	12
F-5	0.7071	0.7071	5	500	100	18	12
F-6	0.7071	0.7071	5	750	100	18	12
F-7	0.7071	0.7071	6	250	100	18	8
F-8	0.7071	0.7071	6	500	100	18	8
F-9	0.7071	0.7071	6	750	100	18	8
F-10	0.7071	1.4142	4	250	100	18	16
F-11	0.7071	1.4142	4	500	100	18	16
F-12	0.7071	1.4142	4	750	100	18	16
F-13	0.7071	1.4142	5	250	100	18	8
F-14	0.7071	1.4142	5	500	100	18	8
F-15	0.7071	1.4142	5	750	100	18	8
F-16	0.7071	1.4142	6	250	100	18	4
F-17	0.7071	1.4142	6	500	100	18	4
F-18	0.7071	1.4142	6	750	100	18	4

Figure 10 shows the pressure drop across the branching channel network as a function of wall heat flux and mass flow rate for both single phase and two-phase flow for $k = 4$ & 6 and $\gamma = 0.7071$ & 1.4142. As expected, the single phase pressure drop is smaller than the two- phase pressure drop at the same mass flow rate for the entire heat flux range tested. Additionally, the single phase pressure drop decreases with increasing

heat flux due to decreasing viscosity with increasing average fluid temperature. This is a direct result of the constant inlet temperature. The two-phase pressure drop increases with heat flux due to the increase in exit vapor quality which increases the acceleration pressure drop as well as increasing the frictional pressure drop due to increasing the two-phase multiplier. This increase in exit quality with wall heat flux is a direct result of the zero sub-cooling inlet condition. This difference of slope with respect to the wall heat flux results in the pressure drop for the 400 & 500 g/min single phase flow conditions at the highest heat flux to be comparable to the two-phase pressure drop at 100 g/min. It is also important to note that an order of magnitude change in the flow rate between the single phase and two-phase cases does not correspond to an order of magnitude change in pressure drop, and that the trend with increasing heat flux is to reduce the pressure drop difference between the single and two-phase cases. The slope of the two-phase pressure drop with heat flux is steeper at larger mass flow due to the higher fluid velocities in the channels, this effect is more pronounced for the $k = 4$ and $\gamma = 0.7071$ channels.

The pumping power, shown in Fig. 11, shows trends for single phase and two-phase flows which are similar to that of the pressure drop. The relative magnitudes of each are different due to the larger range of the single phase flow rates. The single phase values of pumping power are comparable to those of the two-phase values with mass flow rates ⅓ to ½ as large.

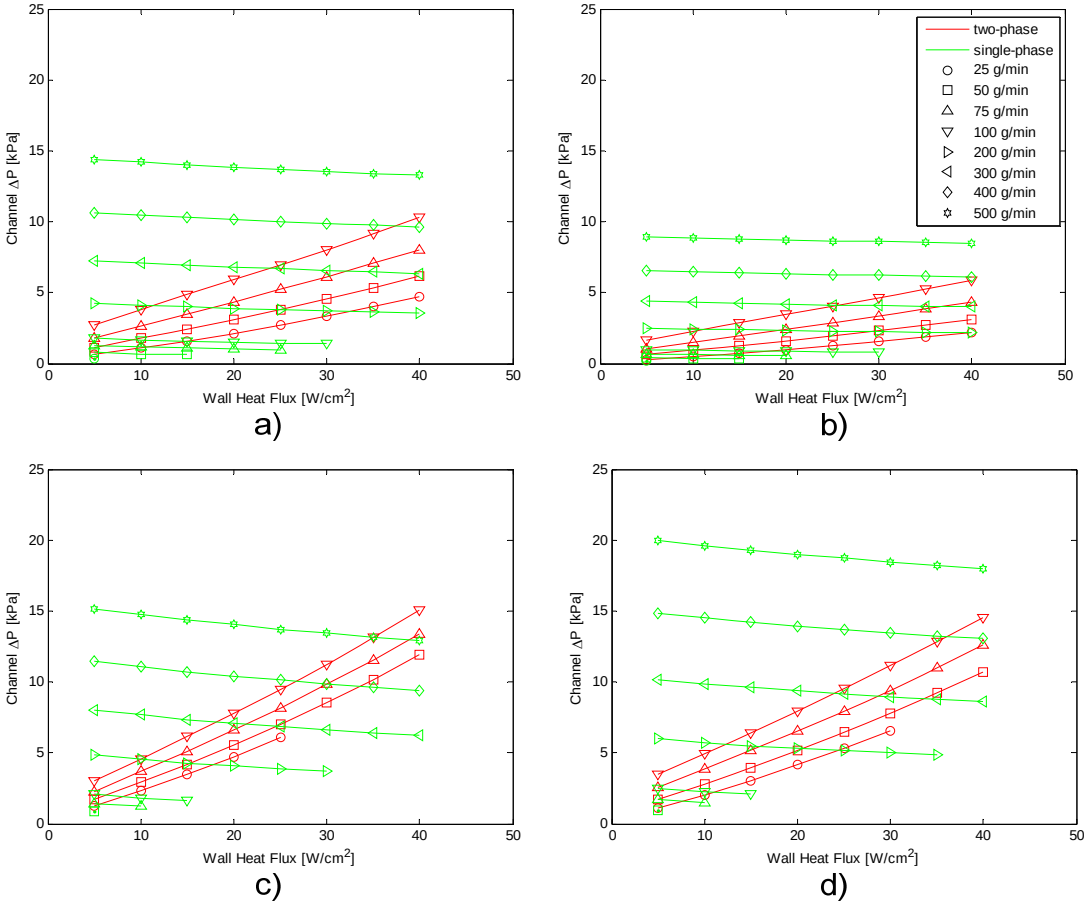


Figure 10. Channel pressure drop for diabatic single and two-phase flow in fractal-like branching channels as a function of wall heat flux and mass flow rate. Data for a channel height of 500 μm a) k=4, γ=0.7071, b) k=6, γ=0.7071, c) k=4, γ=1.4142, d) k=6, γ=1.4142.

A performance parameter, ε, defined as another means to asses the performance of the heat sinks:

$$\varepsilon = \frac{\dot{q}'' A_{w,Total}}{\dot{m} v_l \Delta P} \tag{38}$$

where $\dot{m}$ is the mass flow rate, and $A_{w,Total}$ is the total channel wall surface area in the heat sink. A plot of ε as a function of wall heat flux and mass flow rate is shown in Fig. 12, again for k = 4 & 6 and γ = 0.7071 & 1.4142. For the single phase conditions the value of

ε increases monotonically with heat flux although the slope is decreasing with heat flux. For the two-phase conditions ε initially increases then begins to decrease with heat

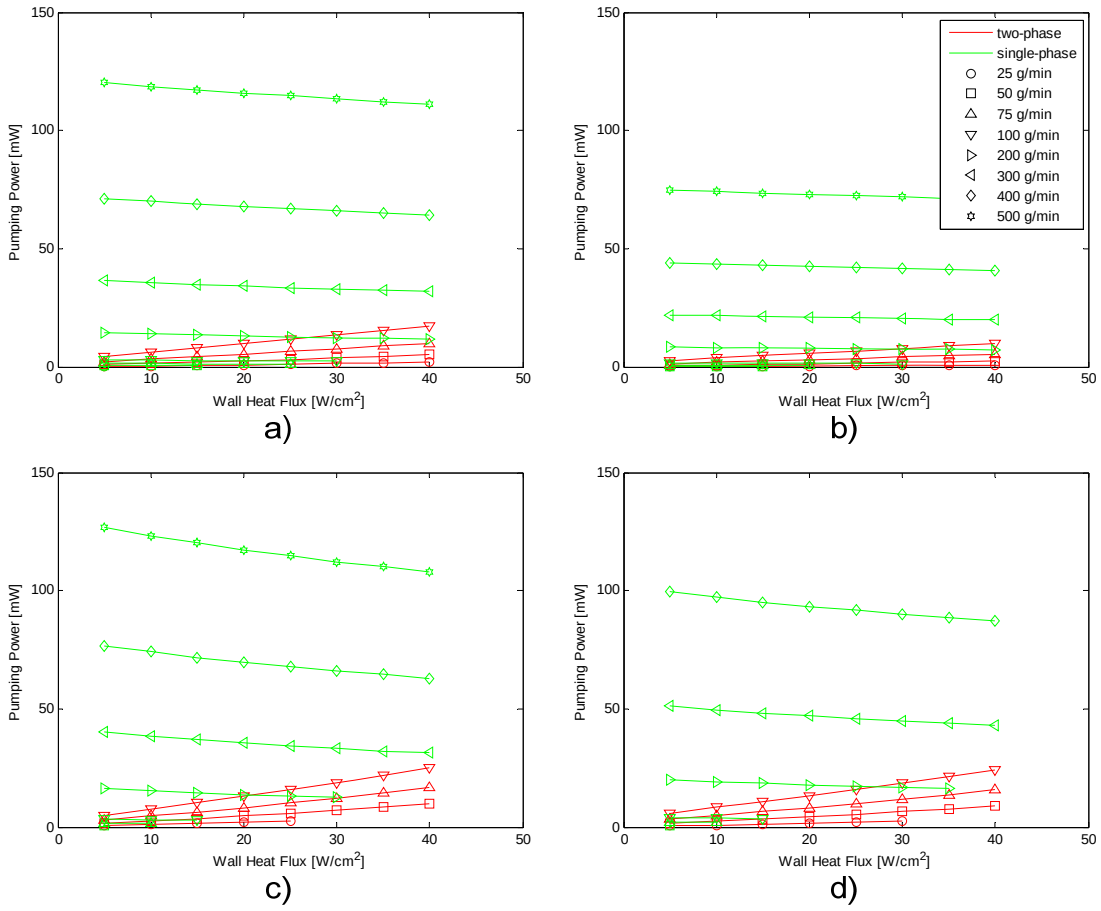


Figure 11. Pumping power for diabatic single and two-phase flow in fractal-like branching channels as a function of wall heat flux and mass flow rate. Data for a channel height of 500 μm and a) k=4, γ=0.7071, b) k=6, γ=0.7071, c) k=4, γ=1.4142, d) k=6, γ=1.4142.

flux indicating an optimal flux for each mass flow rate. The single phase flow either does not have an optimum or the optimum is outside the range of heat fluxes examined. The value of the single phase performance parameter is always larger than the two-phase performance parameter for identical flow rates and heat flux due to the smaller pressure drop of the single phase flow. The performance parameter also decreases with increasing mass flow for both the single and two phase flows due to the increasing pumping power

with mass flow. However, the single-phase flows decrease faster at higher heat fluxes than the two-phase flows, this is due to the decreasing pressure drop with heat flux and the smaller rate of change of pressure drop with mass flow for the single phase flows. The extent of the region where it is possible to have only single-phase flow is also identifiable in these figures, indicating that there is an upper limit on the performance parameter for single phase flow. The two-phase flow is also limited by an exit quality of one, however this limitation was not encountered in the cases examined.

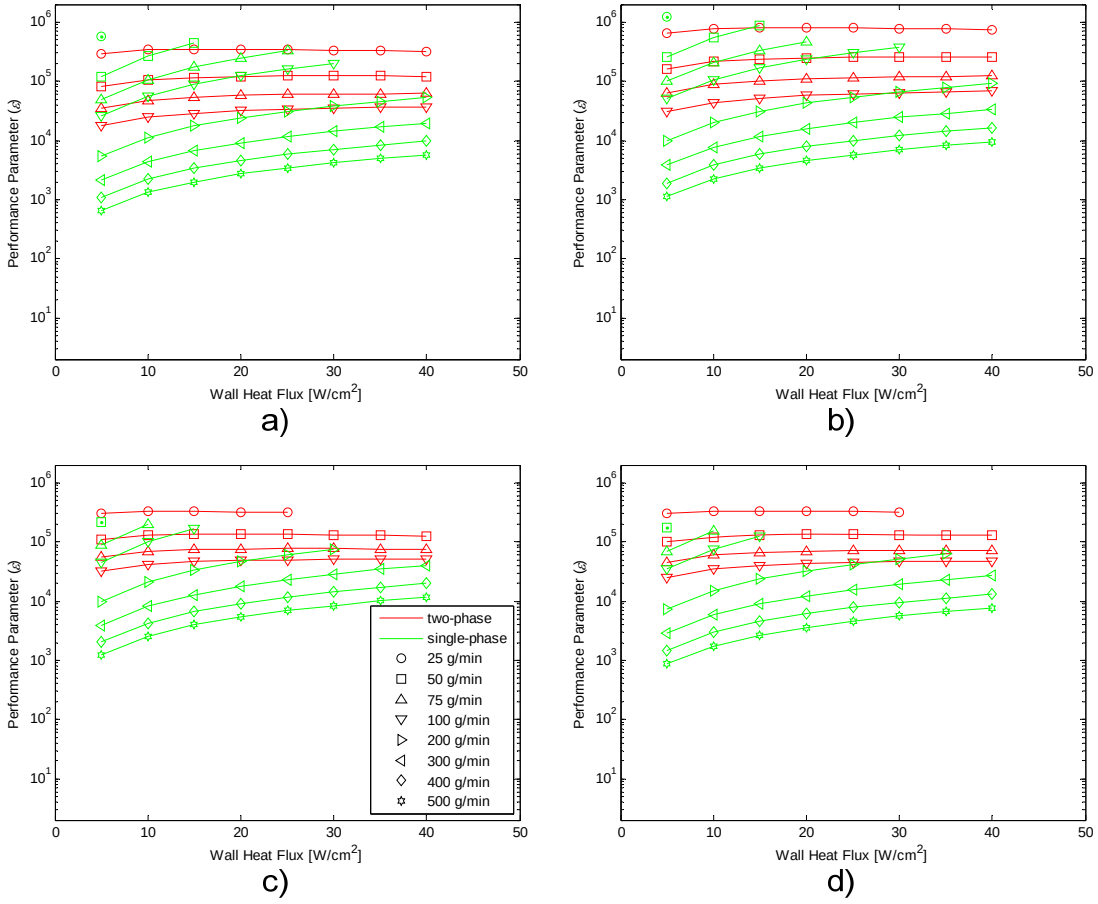


Figure 12. Performance parameter for diabatic single and two phase flow in fractal-like branching channels as a function of wall heat flux and mass flow rate. Data for a channel height of 500 μm and a) k=4, γ=0.7071, b) k=6, γ=0.7071, c) k=4, γ=1.4142, d) k=6, γ=1.4142.

The maximum wall temperature and temperature change across a heat sink are also critical parameters in the design of heat sinks. One of the major contributions to the maximum wall temperature and temperature change across the heat sink is the maximum fluid temperature and change in fluid temperature across the heat sink. In electronics cooling, the maximum fluid temperature is important, but the temperature uniformity across the heat sink is also critical. For single phase flow, the maximum fluid temperature is strongly dependent on the inlet temperature which can be selected appropriately given that the temperature change across the heat sink is constant. For two phase flow in heat sinks, the upper limit on the fluid temperature is completely dependent on the saturation temperature which can be controlled by appropriate selection of the cooling medium. Because of the ability to select these parameters, this paper will only examine the temperature change experienced by the cooling fluid as it passes through the heat sink. Figure 13 shows a plot of the cooling fluid temperature change across the heat sink as a function of wall heat flux and mass flow rate for k = 4and γ = 0.7071, and h_c = 250 μm. The other length ratios and numbers of branching levels follow similar trends. The single phase cases show that the temperature change across the heat sink is positive, and the magnitude increases with heat flux but decreases with increasing mass flow, as would be expected. The two-phase cases show that the temperature change across the heat sink is negative, and increases with increasing heat flux and mass flow rate, this is driven by the fact that the fluid temperature is always at the saturated liquid value, and this is a function of pressure, therefore at higher inlet pressures the temperature change will be greater. It is important to notice that the temperature change for the single-phase

cases at the highest mass flow, approximately 15 °C, is not much greater than the temperature change for the two-phase flow cases at these same heat fluxes, approximately 10 °C.

If minimizing the mass flow through the heat sink for a given heat dissipation, is the desired goal, then two-phase flow would be the method of choice. However if minimizing the pressure drop, pumping power or maximizing the performance parameter

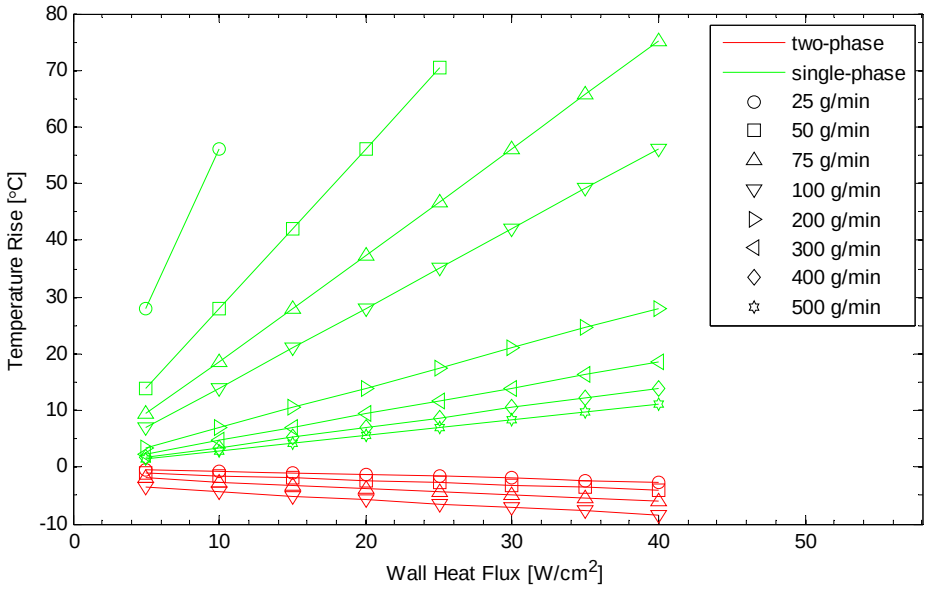


Figure 13. Temperature change across the heat sink for diabatic single and two-phase flow in the fractal-like branching channels. Data for a channel height of 250 μm, 4 branching levels and a length ratio of 0.7071.

are the criteria, then single phase flow can be the better choice as long as the temperature change across the channel does not have to be minimized. If the temperature change across the channel is critical, then the data indicates that two-phase flow is better suited. Although the data also indicates that there is a flow rate beyond which the temperature change for the single phase flow will be less than the two phase flow. it is also clear from this data, that the pressure drop of the two-phase flow at this flow rate will be greater than

the single phase flow. All this indicates that there is a region of flow rates and heat fluxes which will have smaller pressure drops and pumping powers with two-phase flows, and a region of mass flows and heat fluxes which are better suited to single phase flow.

Diabatic Parallel and Fractal-Like Two-Phase Flow Comparison

In this section the two-phase fractal-like branching channel results from the previous section are compared to two-phase parallel channel flow results. Table VIII describes the parallel channel geometries added in this study, Table VII describes the fractal geometries examined. The parallel channels studied had the same overall length, channel heights and width as the terminal fractal-like branching channel (100 μm). The number of channels was varied to match the total surface area of the fractal-like branching channels. The mass flow rates and wall heat flux conditions were identical between the parallel and branching channel geometries. The two phase multiplier model by Qu & Mudawar[57] was used in conjunction with the void fraction model by Zivi [27].

Figure 14 shows the pressure drop between the fractal-like channel heat sink for $k = 4$ & 6 and $\gamma = .7071$ and 1.4142 and the parallel channel heat sink with equivalent wall surface area and exit channel geometry as a function of mass flow rate and wall heat flux. The pressure drops for the straight channels are significantly higher than the branching channels, with the relative difference the straight-parallel and branching channels with $\gamma = 0.7071$ cases being largest. This is due to the combination of two effects, the first is the fact there are much fewer parallel channels in the equivalent heat

sink for the $\gamma = 0.7071$ cases than the $\gamma = 1.4142$ cases. This causes the velocity in the heat sinks equivalent to the branching channels with $\gamma = 0.7071$ to be higher than the ones

Table VIII. Parallel channel geometries used for diabatic parallel channel two phase flow to diabatic fractal-like branching channel two-phase flow studies

Geometry #	Channel Length	Channel Width	Channel Height	Number of Channels	Fractal Geometry with Identical Wall Area
P-1	18	100	250	78	F-1
P-2	18	100	500	72	F-2
P-3	18	100	750	70	F-3
P-4	18	100	250	87	F-4
P-5	18	100	500	79	F-5
P-6	18	100	750	76	F-6
P-7	18	100	250	87	F-7
P-8	18	100	500	77	F-8
P-9	18	100	750	73	F-9
P-10	18	100	250	151	F-10
P-11	18	100	500	146	F-11
P-12	18	100	750	144	F-12
P-13	18	100	250	143	F-13
P-14	18	100	500	138	F-14
P-15	18	100	750	137	F-15
P-16	18	100	250	137	F-16
P-17	18	100	500	133	F-17
P-18	18	100	750	131	F-18

equivalent to the $\gamma = 1.4142$ branching channels which drives the pressure drop up for the higher velocity cases as can be seen in the figure. The second is the small width channels are shorter for the $\gamma = 0.7071$ branching channels than the $\gamma = 1.4142$ channels, this leads to lower pressure drops for the $\gamma = .7071$ channels due to the flow spending less time in

the small high gradient channels. Increasing the number of branching levels also decreases the pressure drop in the fractal channels for the same reasons that the smaller length ratio decreases the pressure drop, since the total length of the channels is the same between the two branching levels, the lengths of the smallest channels are smaller for the higher number of branching levels, therefore the contribution of these high gradient channels is smaller causing the total pressure drop to be smaller.

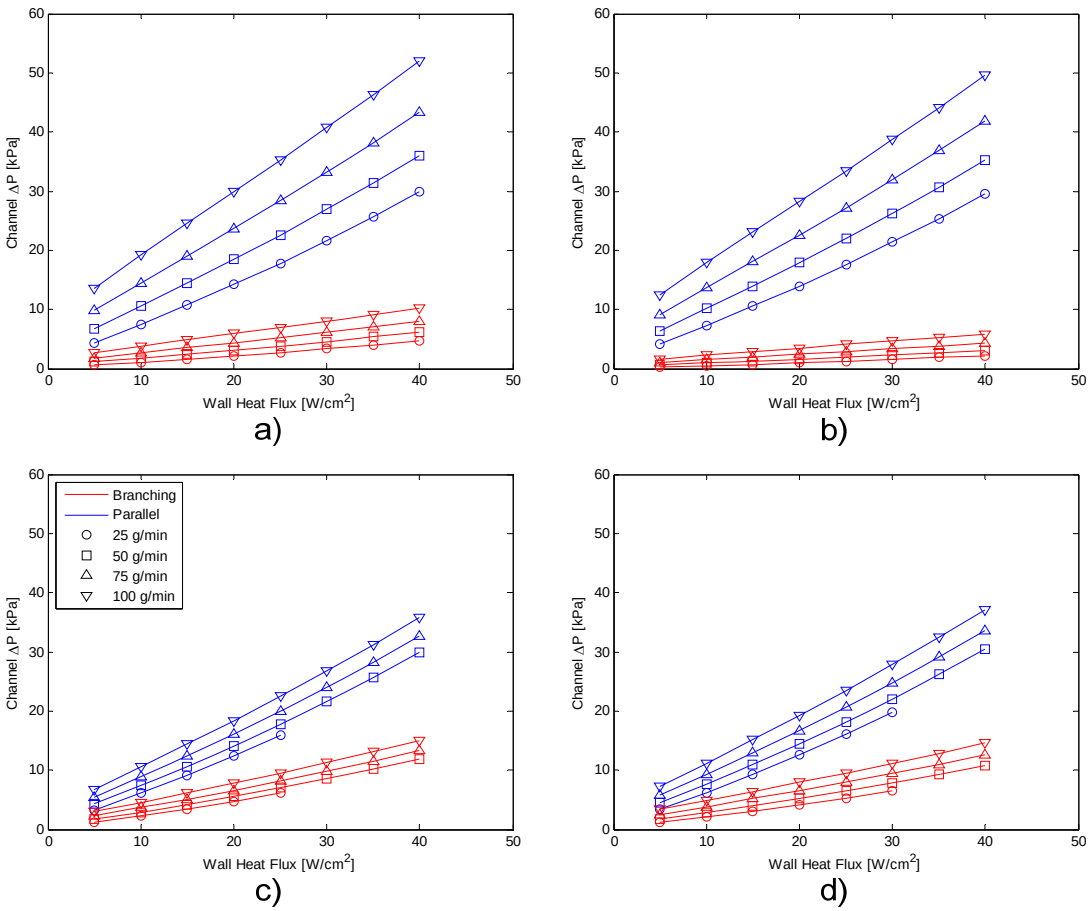


Figure 14. Pressure drop for diabatic two phase flow in parallel and fractal-like branching channels with matching wall areas as a function of wall heat flux and mass flow rate. Data for a channel height of 500 μm a) k=4, γ=0.7071, b) k=6, γ=0.7071, c) k=4, γ=1.4142, d) k=6, γ=1.4142.

The performance parameter, ε, is plotted as a function of mass flow rate and wall heat flux for branching channels with $k = 4$ & 6 and $\gamma = 0.7071$ & 1.4142, along with the equivalent parallel channel heat sinks in Fig. 15. Because there is such a large difference in the pressure drops between the parallel and branching channels, and the heat dissipation is identical between the two types of channels, the value of ε is larger for the fractal channels than the parallel channels. This difference for the $\gamma = 0.7071$ cases is as much as an order of magnitude for the same flow rate and heat flux condition. The trends of both the parallel and branching channel are similar, indicating that the physics of the pressure drop are similar between the two types of channels. Comparing the branching channel behaviors shows that increasing the number of branching levels for the smaller length ratio increases the value of the performance parameter, this is driven by the reduction in pressure drop. However at the larger length ratio, changing the number of branching levels is not as significant. With $k = 6$, the smaller length ratio increased the performance parameter, again due to the significant shift in the pressure drop. With $k = 4$, changing the length ratio has a smaller effect.

As was done between the single phase and two-phase cases, the temperature change along the channel for the branching channel with $k = 4$, $\gamma = 0.7071$, and $h_c = 500$ μm and the equivalent parallel channel is plotted as a function of wall heat flux and mass flow rate in Fig. 16. Both the parallel and branching channel temperature changes are driven entirely by pressure change across the channel, so it is no surprise that the parallel channel temperature change is larger than the branching channel and that the trends are identical to the pressure drop only inverted.

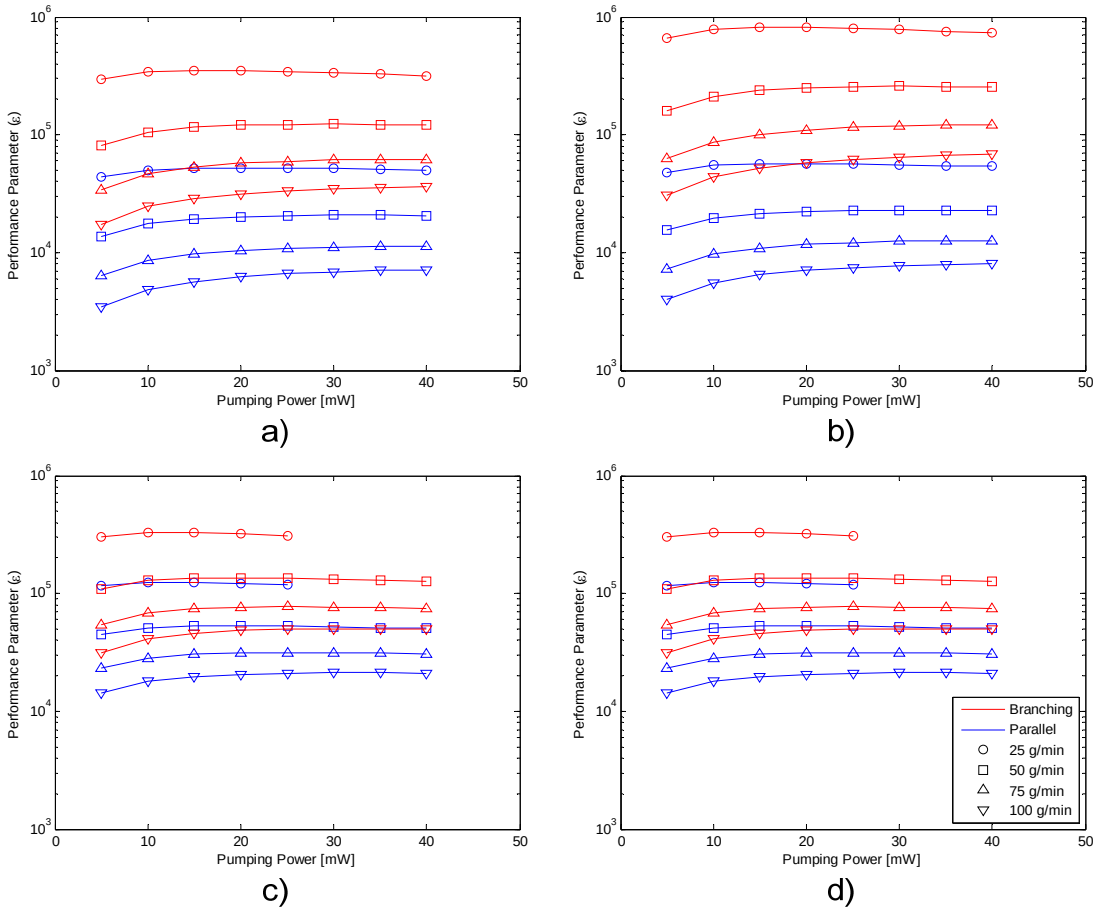


Figure 15. Pressure drop for diabatic two phase flow in parallel and fractal-like branching channels with matching wall areas as a function of wall heat flux and mass flow rate. Data for a channel height of 500 μm a) k=4, γ=0.7071, b) k=6, γ=0.7071, c) k=4, γ=1.4142, d) k=6, γ=1.4142.

Based on all the criteria examined pressure drop, ε, and temperature change, the branching channels out perform the parallel channels. The impact of changing the number of branching levels and length ratio is less clear, under some conditions increasing the number of branching levels tends to improve the performance, while for others the impact is neutral. The smaller length ratio has higher performance particularly for 6 branching levels, but the effect is smaller for 4 branching levels.

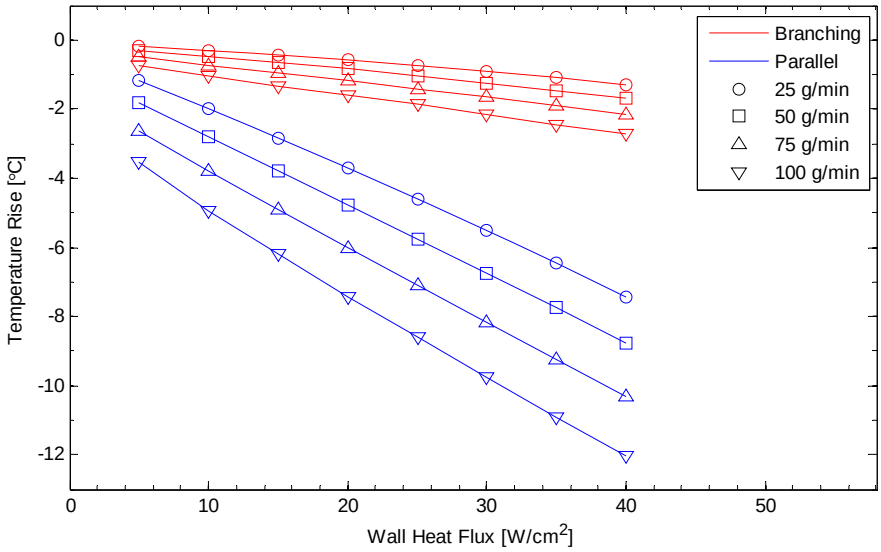


Figure 16. Temperature change across the heat sink for diabatic two phase flow in parallel and fractal-like branching channels with matching wall areas as a function of wall heat flux and mass flow rate. Data for a channel height of 500 μm, $k = 4$, and $\gamma = 0.7071$.

Adiabatic Parallel and Fractal-Like Two-phase Flow Comparison:

Adiabatic two-phase flow through fractal like branching channels are compared to adiabatic two-phase flow through parallel channels with the same width as the branching channel terminal width (100 μm), the same channel heights as the branching channels and identical wall surface areas. A summary of the channel geometries studied are shown in Table IX. The two-phase multiplier of Qu & Mudawar [57] was used along with the void fraction model of Zivi [27]. The pressure drop and vapor quality results are compared between the fractal-like flow network against the parallel channel network.

The case of four branching levels, $\gamma = 0.7$ and exit channel width of 100 μm is shown in Fig. 17 for a flow rate of 15 g/min and sub cooling of 1 °C. The pressure gradient increases once phase change is initiated. The re-developing flow after each

branching is evident for the fractal case. However, as seen for level one, the onset of phase change somewhat hides the re-developing flow characteristics. The initiation of

Table IX. Fractal and Parallel geometries used for adiabatic two-phase flow studies

Fractal Geometries L_{tot} = 18mm; w_t=100μm; β_k =0.7071					Parallel Geometries L_{tot} = 18mm; w_t=100μm		
Geometry #	γ	Branching Levels	Channel Height [μm]	Number of inlets	Geometry #	Channel height [μm]	Number of Inlets
Adb-F-1	0.7071	4	100	1	Adb-P-1	100	5.5
Adb-F-2	0.7071	4	150	1	Adb-P-2	150	5.2
Adb-F-3	0.7071	4	250	1	Adb-P-3	250	4.8
Adb-F-4	0.7071	5	100	1	Adb-P-4	100	8.5
Adb-F-5	0.7071	5	150	1	Adb-P-5	150	7.9
Adb-F-6	0.7071	5	250	1	Adb-P-6	250	7.3
Adb-F-7	0.7071	6	100	1	Adb-P-7	100	13
Adb-F-8	0.7071	6	150	1	Adb-P-8	150	12
Adb-F-9	0.7071	6	250	1	Adb-P-9	250	10.9
Adb-F-10	1.4142	4	100	1	Adb-P-10	100	9.8
Adb-F-11	1.4142	4	150	1	Adb-P-11	150	9.6
Adb-F-12	1.4142	4	250	1	Adb-P-12	250	9.4
Adb-F-13	1.4142	5	100	1	Adb-P-13	100	18.8
Adb-F-14	1.4142	5	150	1	Adb-P-14	150	18.3
Adb-F-15	1.4142	5	250	1	Adb-P-15	250	17.8
Adb-F-16	1.4142	6	100	1	Adb-P-16	100	36.3
Adb-F-17	1.4142	6	150	1	Adb-P-17	150	35.4
Adb-F-18	1.4142	6	250	1	Adb-P-18	250	34.4

phase change for the fractal flow network occurs approximately 1 mm further downstream compared with the parallel channel. In this case, the exit quality for the parallel channel flow is more than two times greater than for the fractal case. The general

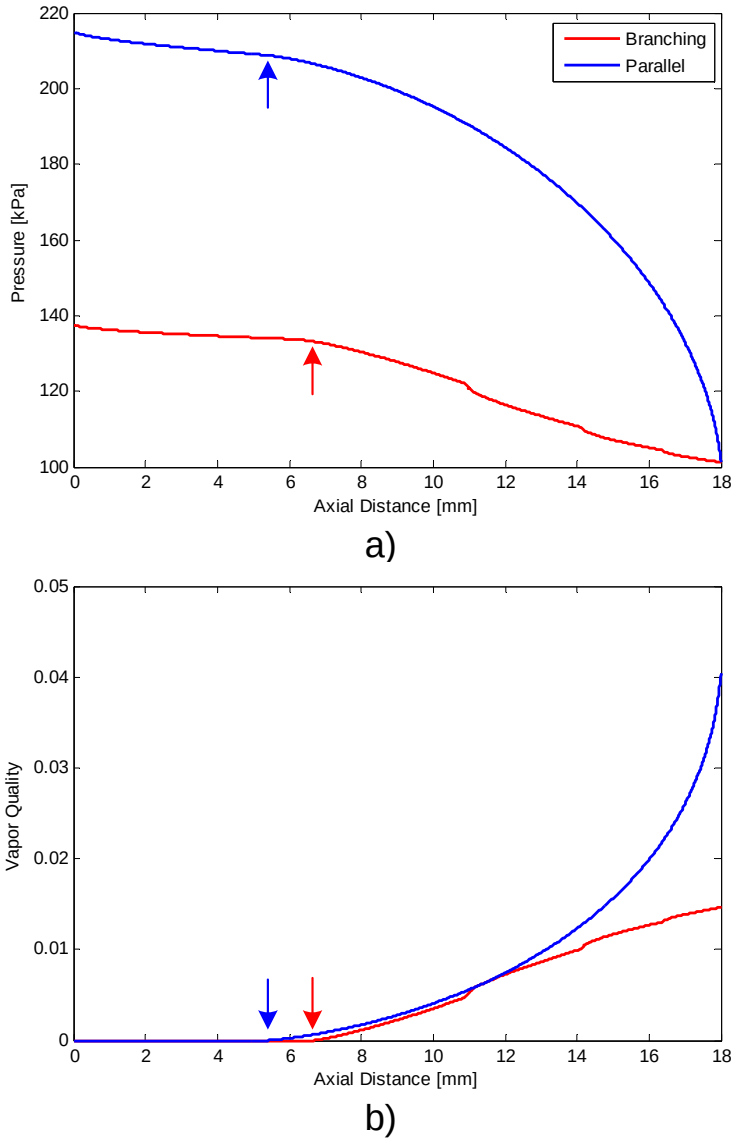


Figure 17. Pressure drop and vapor quality as a function of axial distance along microchannel for parallel and fractal like branching examples. Arrows indicate point of initial phase change

trend of quality variation along the flow shows that the fractal flow network yields a more linear variation compared to the more exponential trend of the parallel channel flow. The parallel channel system is a much stiffer system than the fractal system where small changes in mass flow could lead to large changes in exit quality and pressure drop.

This indicates that the parallel system will have more flow stability issues than the fractal system.

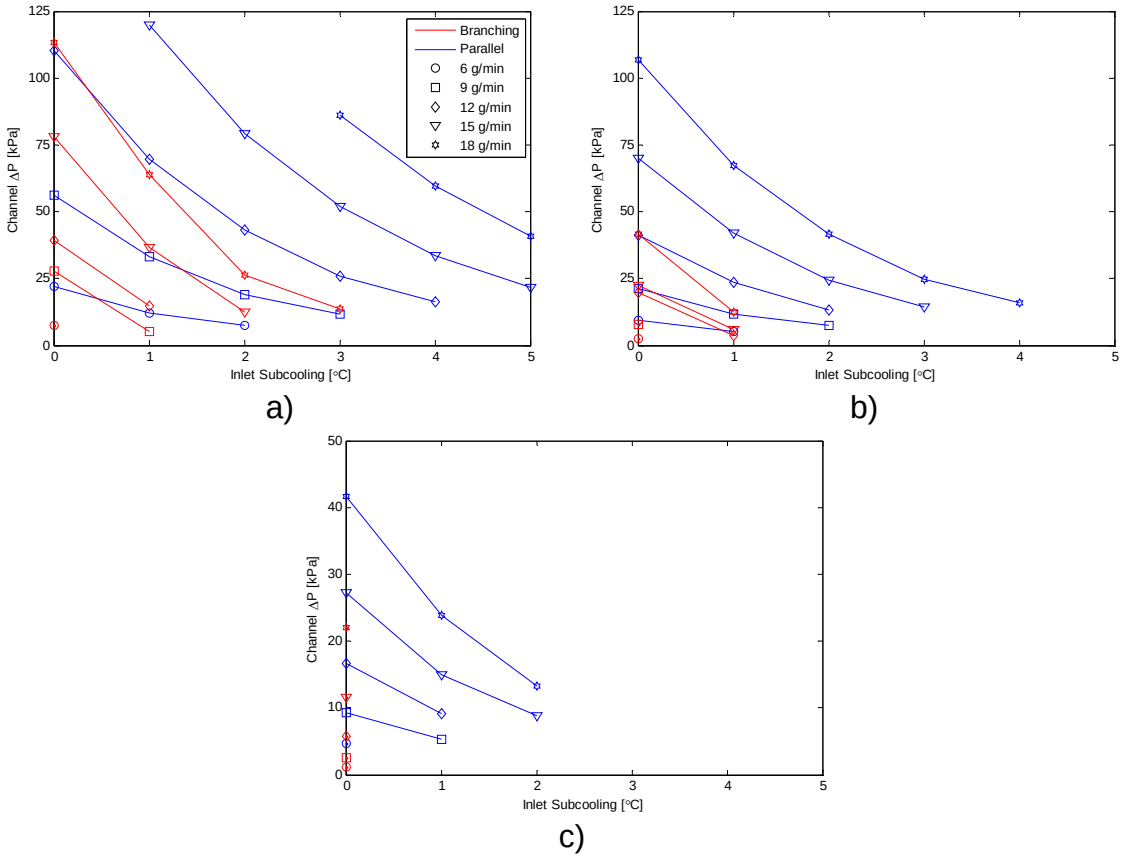


Figure 18. Channel pressure drop for adiabatic two-phase flow boiling as a function of sub-cooling and mass flow rate for fractal and parallel channels with equivalent wall surface area. Data for fractal channel geometries with $\gamma = .7071$, $h_c = 250$ µm, $w_t = 100$ µm, and a) $k = 4$, b) $k = 5$, c) $k = 6$.

For $\gamma = 0.7071$, $h_c = 250$ µm, and $w_t = 100$ as, the pressure drop versus inlet sub-cooling for a range of flow rates from 6 to 18 gm/min for total branching levels of 4, 5, and 6 are given in Fig. 18. Note that results are only presented for cases where phase change occurs. The fractal-like network yields a pressure drop less than half as large as the corresponding parallel channel flow. The rate of decrease of pressure drop with increasing sub-cooling is greater for the fractal-like channels as compared to the straight-

parallel network, whereas the rate of increase of pressure drop with increasing flow rate is somewhat higher for the fractal case.

The exit quality is shown in Fig. 19 for the three branching levels for the smaller length ratio cases of, $\gamma = 0.7071$. The parallel channel cases have significantly higher exit qualities, consistent with the higher pressure drops. The rate of decrease of exit quality with increasing sub-cooling is greater for the fractal flow network. This may be indicative of less vapor expansion in the fractal channel network due to the larger cross-sectional flow area.

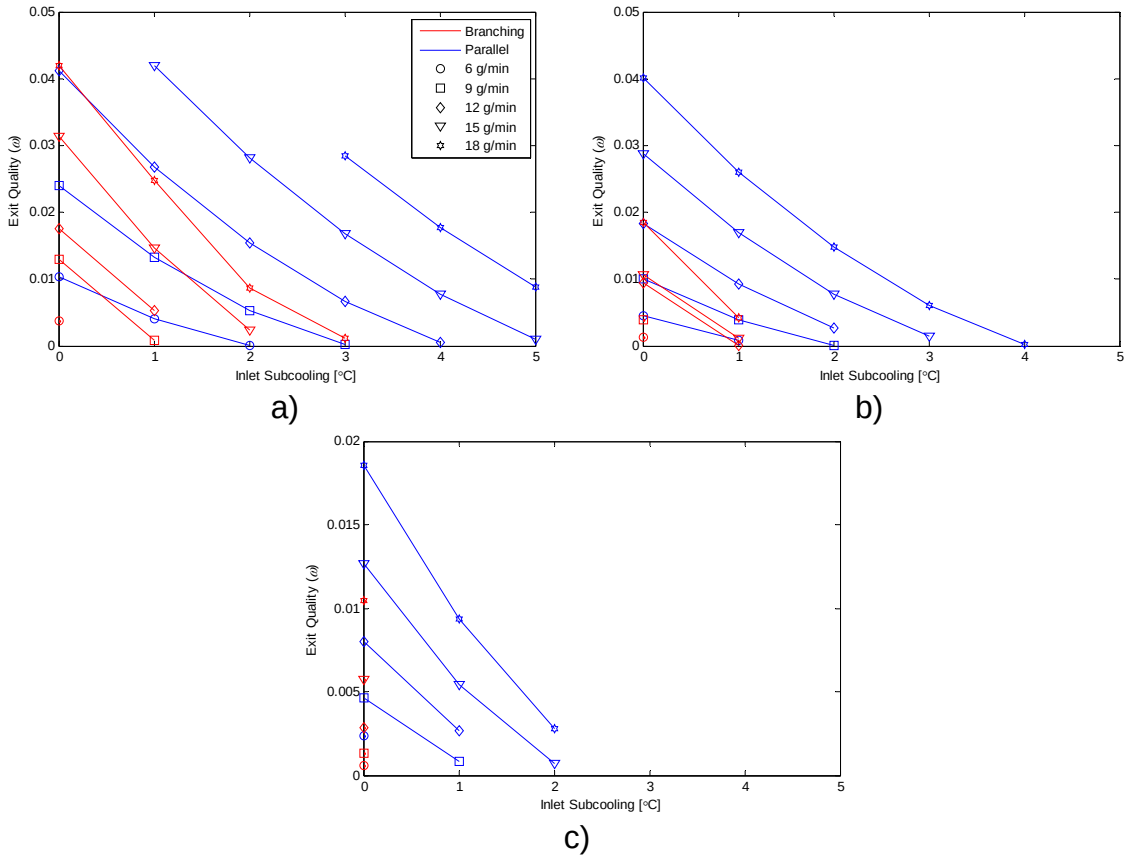


Figure 19. Channel exit quality for adiabatic two-phase flow boiling as a function of sub-cooling and mass flow rate for fractal and parallel channels with equivalent wall surface area. Data for fractal channel geometries with $\gamma = .7071$, $h_c = 250$ µm, $w_t = 100$ µm, and a) $k = 4$, b) $k = 5$, c) $k = 6$.

Pressure drop for the case of $\gamma = 1.4142$ is shown in Fig. 20 for the smaller channel height of $h_c = 150$ μm. For this larger length ratio geometry, the parallel channel network has a lower pressure drop. This is because the parallel channel network has a larger total exit flow area in order to maintain the same total channel surface area. This results in lower velocities for the parallel channel network. Note that all of the pressure drops merge for high inlet sub-cooling conditions. This is because the number of channels in the parallel geometry, about 10, is close to the number of exit channels in the fractal geometry, 16. These last channels for this geometry configuration are also the longest channels which means they will dominate the pressure drop, particularly at the higher subcooling levels where two-phase flow is only occurring in these higher branching levels. Since the pressure drop is driven primarily by the pressure drop in the two phase region, the fractal channels will begin to behave much more like the parallel channels for the larger length ratio and higher subcoolings.

Figure 21 shows the exit quality for the same flow conditions as Fig. 20. The exit quality shows the same trends as the pressure drop because the same factors are influencing these trends as were influencing the pressure drop trends. That is the flow is dominated by the end channels. The fractal flow and the parallel flow are similar because there are a similar number of exit channels for both configurations.

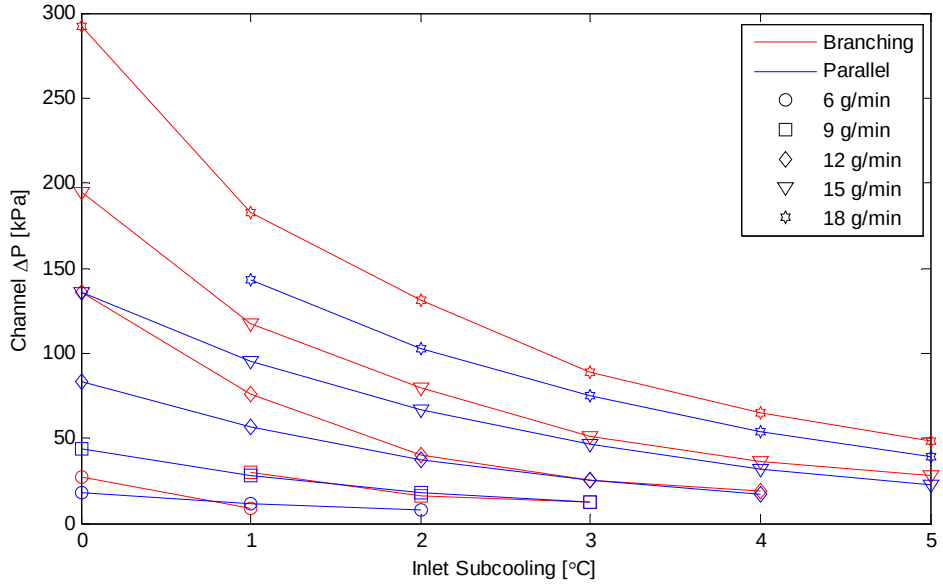


Figure 20. Channel pressure drop for adiabatic flow boiling as a function of inlet sub-cooling and mass flow rate for fractal and parallel channels with equivalent wall surface areas and large branching ratio. Data for a fractal channel with $\gamma = 1.4142$, $h_c = 150$ µm, $w_t = 100$ µm, and $k = 4$.

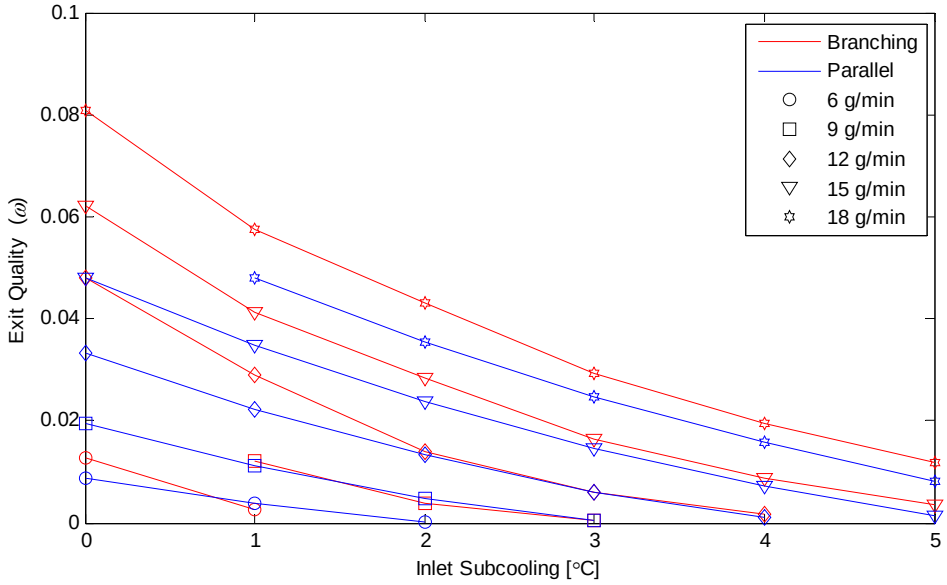


Figure 21. Exit quality for adiabatic flow boiling as a function of inlet sub-cooling and mass flow rate for fractal and parallel channels with equivalent wall surface areas and large branching ratio. Data for a fractal channel with $\gamma = 1.4142$, $h_c = 150$ µm, $w_t = 100$ µm, and $k = 4$.

A direct comparison of the smaller and larger length ratio cases is shown in Figures 22 and 23 for three channel heights, h_c = 100, 150, and 250 µm. The smaller length ratio case has a higher pressure drop and exit quality due to the fact that a greater portion of the total length is in the smaller number of larger channels which have significantly smaller channel cross-sectional areas. This increases the local velocity and consequently the pressure drop. The rate of decrease of pressure drop with increasing sub-cooling is larger for the small length ratio case. As the mass flow rate increases, the absolute differences are less.

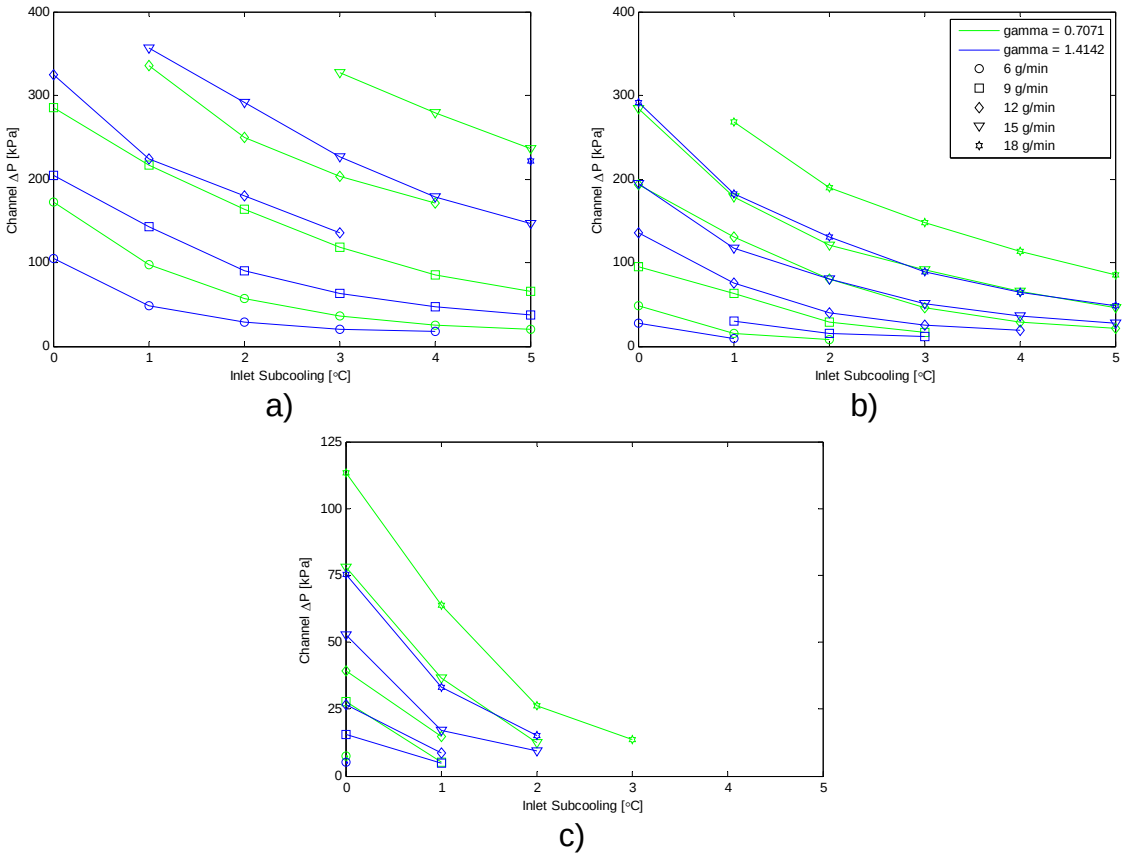


Figure 22. Pressure drop for adiabatic flow boiling as a function of inlet sub-cooling and mass flow rate for fractal branching channels of various channel heights. Data for channels with k = 4, w_t = 100 µm, two length ratios γ = .7071 and γ = 1.4142, and a) h_c = 100 µm, b) h_c = 150 µm, c) h_c = 250 µm.

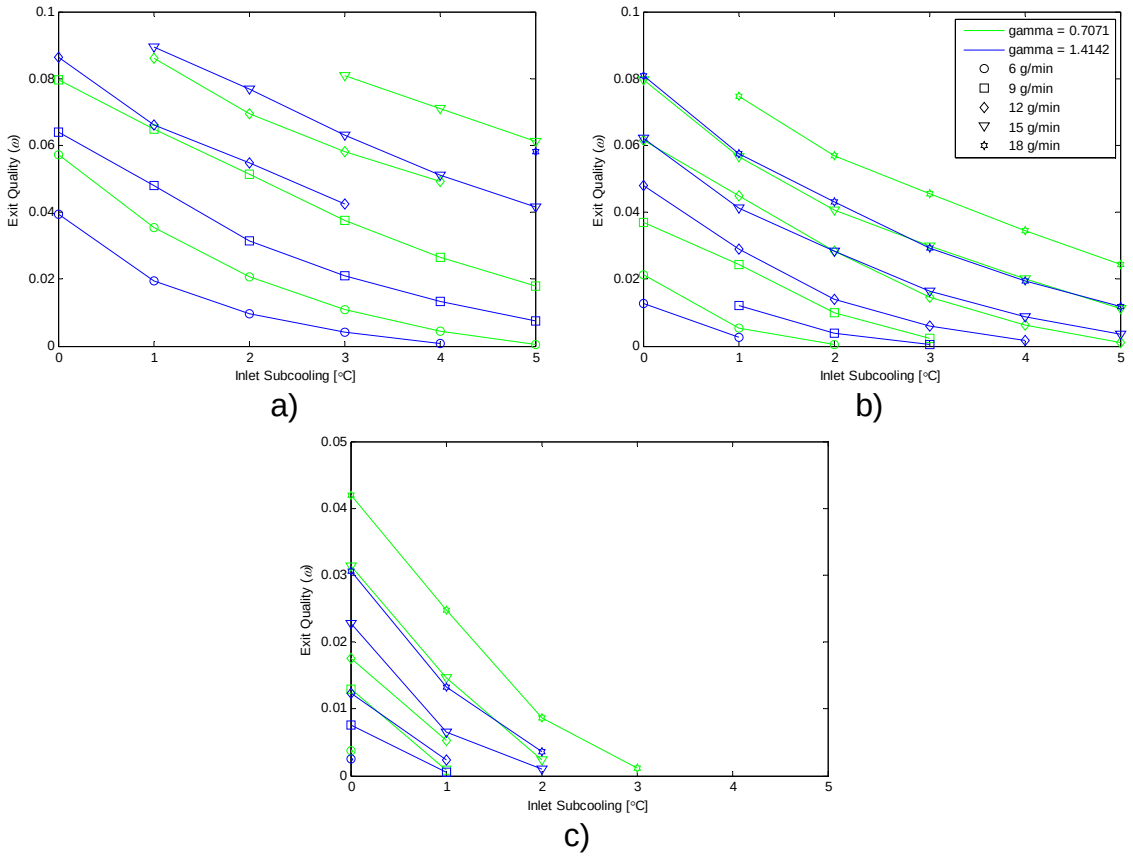


Figure 23. Exit quality for adiabatic flow boiling as a function of inlet sub-cooling and mass flow rate for fractal branching channels of various channel heights. Data for channels with $k = 4$, $w_t = 100$ µm, two length ratios $\gamma = .7071$ and $\gamma = 1.4142$, and a) $h_c = 100$ µm, b) $h_c = 150$ µm, c) $h_c = 250$ µm

Experimental Work

Experimental Set-up

Adiabatic flow boiling experiments were performed using a fractal-like branching channel heat sink. Pressure drop, inlet temperature, and mass flow rate data of the two phase flow through the fractal like branching channels were recorded. Images of the two phase flow within the branching channel network were also captured. A schematic of the flow loop used for the experimental work is shown in Fig. 24. Water with a small amount of Rhodamine 6G Chloride dye was the working fluid for this study, and was maintained at a constant temperature of 90 °C in the reservoir. The reservoir was a small capacity residential water heater with a 2 kW heating element (Rheem 81VD6S). The thermostat circuit on the water heater was bypassed due to maximum water temperature limitations of the circuit. A PID controller (Omega CNi3244-DC) and solid state relay (Omega SSR330DC50) were instead used with a thermocouple immersed in the reservoir to control the heating element and maintain a constant reservoir temperature. Fluid was pumped from the reservoir by a gear pump (Tuthill D-series) driven by a DC motor. Between the reservoir and pump the water passed through a 10 μm filter (Shelco FOS-784). The flow rate through the test loop was controlled two ways. First, the DC voltage supplied to the DC motor was supplied by a variable voltage supply. The motor would spin faster at higher voltages. The maximum voltage which could be supplied to the motor was 24V. Second the position of the needle valve downstream of the filter was adjusted. Just downstream of the needle valve the flow rate was measured with a high accuracy coriolis type mass flow meter (Micromotion CMF010). After the flow exited

the mass flow meter it passed through a coil immersed in a constant temperature bath (Hart Scientific 7320) filled with Paratherm, a non-toxic heat transfer fluid, to raise the working liquid temperature to the temperature required at the fractal inlet for the desired test condition. The flow then passed through a three way valve used direct the flow either through the fractal test device or bypassing the test device.

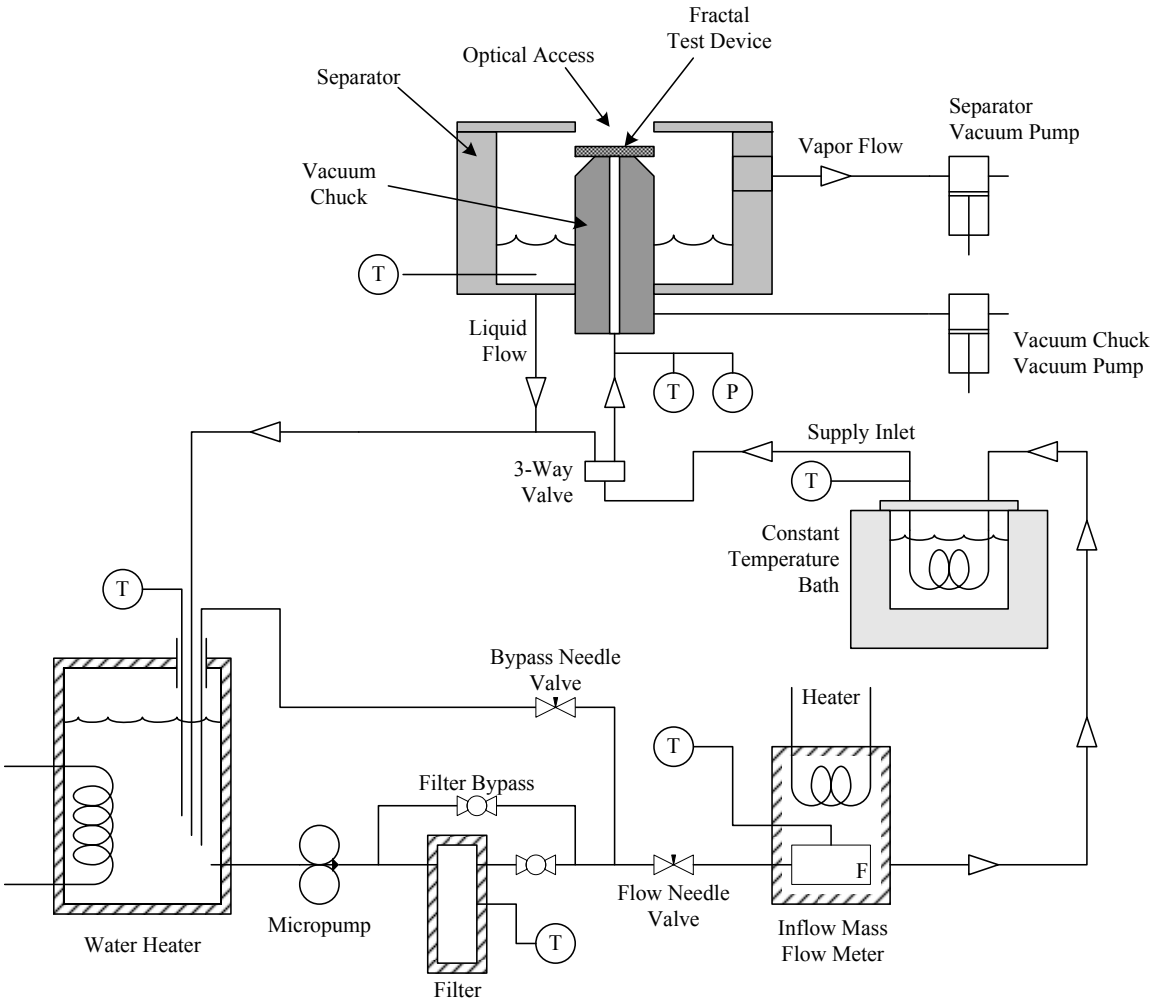


Figure 24. Schematic of flow loop used for adiabatic flow boiling studies.

As the flow passed through the vacuum chuck and then the fractal channel test device, where the largest pressure drop occurred, phase change occurred. The two-phase

mixture exited the fractal channel test device into the separator exit plenum. The vapor phase was drawn from the separator exit plenum through several ports in the exit plenum. The ports were covered with hydrophobic Teflon membranes to prevent any liquid phase from leaving with the vapor. The flow through these ports was generated by a vacuum pump. The flow generated by the pump was larger than the maximum vapor generated by any of the test conditions. This was so air would be drawn through the gap between the fractal and vapor exit plenum so the vapor or liquid mist did not collect on the camera lens and obscure the view of the fractal channels. This however was not sufficient by itself to prevent liquid from collecting on the top of the fractal. So for the flow rates above 175 g/min a conical deflector was fabricated from photocopier transparencies to deflect the liquid droplets from collecting on the top surface of the fractal. The liquid phase was collected in the exit plenum and gravity drained through ports at the bottom of the plenum.

A vacuum chuck was used to hold the fractal test device. This allowed all the fixturing to be below the fractal and leave the fractal channels unobstructed from viewing from above. A schematic cross section of the vacuum chuck assembly is shown in Fig. 25. The heated liquid entered the vacuum chuck at the bottom, passed through a glass tube to insulate it, then through the top half of the vacuum chuck and into the fractal test device. The inlet and periphery of the test device were sealed with O-rings. The pressure of the annular gap between the test device and the vacuum chuck between the o-rings was drawn down by a vacuum pump connected to the large chamber within the chuck. This chamber was connected with the annular gap via small holes around the circumference.

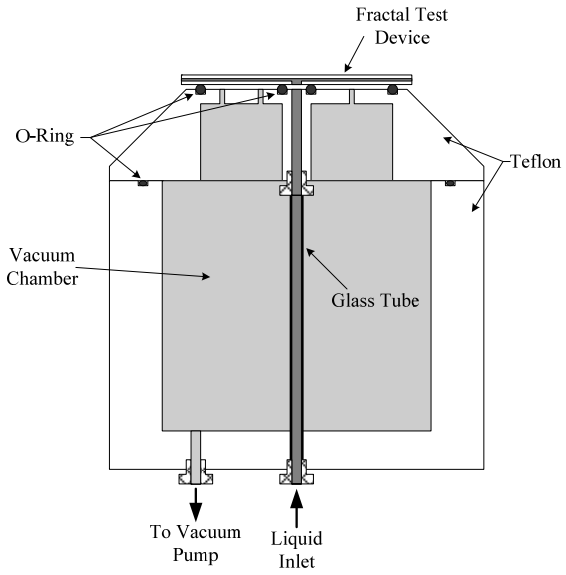


Figure 25. Schematic of vacuum chuck used to hold down fractal device

The pressure and temperature of the liquid were measured as it entered the vacuum chuck. A 0-175 kPa gauge pressure transducer (Cole Parmer 68074-08) with an uncertainty of $\pm$0.36 kPa and a response time of 5 ms was used to measure the inlet pressure. The temperatures of the inlet liquid flow and separator exit liquid flow were measured with RTD's (Therm-X D-SP-4TT-A18). A Kiethly data acquisition device was used to measure the resistance of the RTD's. A calibration curve was then used to convert the resistances to temperatures with an accuracy of $\pm$0.33 °C. Several other thermocouples were installed at the points indicated in the schematic shown in Fig. 24. These thermocouples were used only to monitor system warm-up and to determine steady-state system operation. Similarly the heaters on the mass flow meters were used only during system warm-up, and were off during all measurement conditions. The mass flow meter had an accuracy of $\pm$0.11% of the measured flow rate.

The sampling rate of the data acquisition from each instrument was different. The pressure transducer and the thermocouples were sampled at 200 Hz through the data

acquisition board on the PC. The thermistors were sampled at 1 Hz using the Kiethley device. The mass flow meter output a square wave, the frequency of which changed with the mass flow rate. This signal was measured using the digital counter in the data acquisition board to count the number of times the square wave rose through the trigger voltage. This count was then divided by the 5 second counting interval to obtain the average frequency over the 5 second data acquisition interval. This resulted in a frequency sampling rate of 0.2 Hz.

After the initial testing at the low flow rates, it was discovered that the spray and surface tension effects at the periphery of the disk caused liquid to collect on the top of the fractal, not only at the perimeter of the disk but over the entire disk surface. Because this liquid contained the Rhodamine dye, it also fluoresced when exposed to the laser light. Because it was not confined it tended to pool. The depths of these pools were much larger than the channel dimensions and therefore the fluorescence intensity from these pooled areas was stronger than the fluorescence from the channels. Therefore the pools washed out the information from the channels. To prevent the pools from influencing the data, they were periodically sopped up with an absorbent cloth. This worked fairly well at the moderate flow rates, however at the higher flow rates, it was impossible wipe the liquid often enough to capture any images. To prevent liquid from pooling on the fractal device at the highest flow rates, a conical deflector was fitted to the top of the fractal disk. This deflector worked to prevent the liquid from collecting on the top surface of the disk, however it did cause other problems which will be discussed in the data analysis section.

The fractal channel test device was based on the fractal network designs originated by Pence [1] as described in the introduction. The terminal channel width was 100 μm, the length, γ, and width ratios, β_λ, were both 0.7071, the total channel length was 18 mm, there were four branching levels, and 16 inlet channels. Each branching was symmetric, see Fig. 26 a) for a plan view of the flow network. The channels were Deep Reactive Ion Etched (DRIE) in silicon to a depth of 150 μm. The top walls of the channels were made by anodically bonding a Pyrex glass disk to the Silicon wafer. The inlet hole was cut using laser machining. Figure 26 b) and c) show a schematic cross sections of the network.

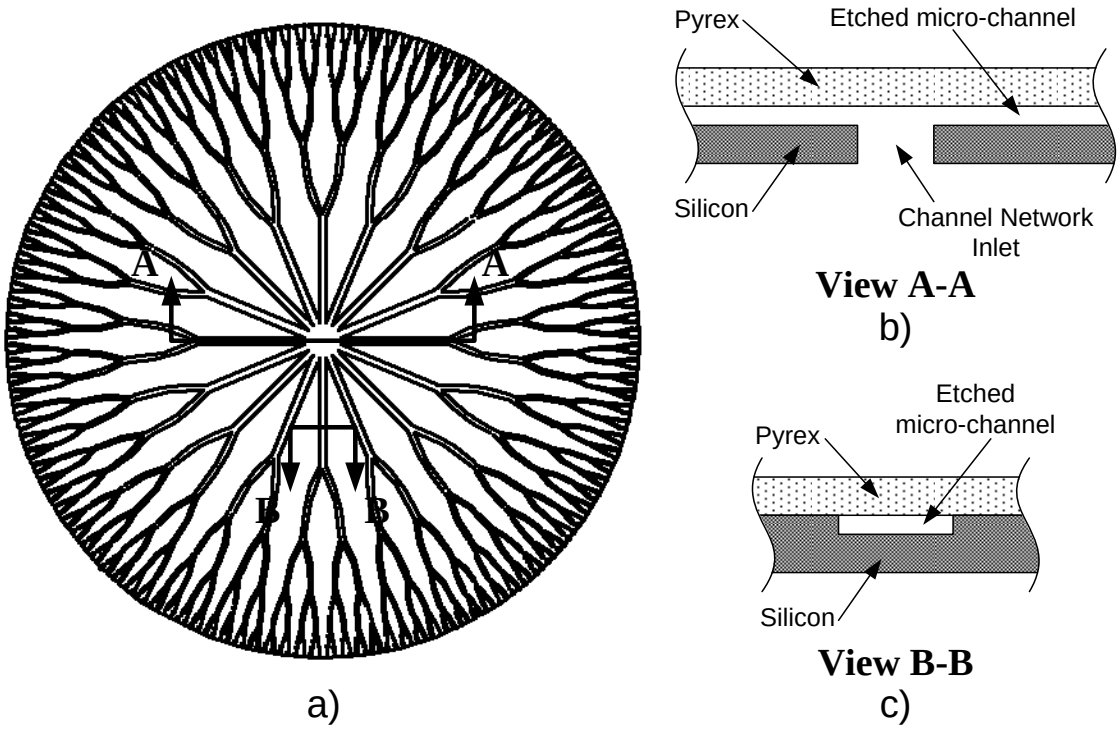


Figure 26. Schematic and cross section view of Fractal-like branching channel device. Showing a) Plan View of channels, b) section view showing inlet, and c) section view showing 0^{th} level channel

Images of the two-phase flow in the microchannels were recorded using a high resolution cooled CCD camera. This camera was used due to its low light sensitivity and high signal to noise ratio in low light conditions. The working liquid containing the fluorescent dye Rhodamine 6G chloride when excited by a light source of a wavelength of about 530 nm, emits light at a wavelength of 550 nm. The illumination source was a continuous wave argon Ion laser which lazes at a wavelength of 528.7 nm. The laser light was passed through an optical chopper to improve imaging. The camera was fitted with a high pass filter with a cutoff wavelength of 540 nm such that only the fluorescent light was transmitted and no reflected laser light passed through to the CCD array. This resulted in images where the everything but the liquid phase was low intensity, and only the liquid regions were of high intensity.

Test Plan

Pressure drop temperature and flow rate data as well as image data were taken for the combinations of flow rate and inlet subcooling shown in Table X. The mass flow rates ranged from 100 to 225 g/min in 25 g/min increments, and the inlet subcooling ranged from 0 to 3 °C in 0.5 °C increments. The test conditions in the upper right corner were eliminated by either single phase flow or an exit quality below 0.005. The maximum flow rate was limited by the maximum allowable inlet pressure of the vacuum chuck which was 212 kPa. The test conditions in the lower left corner were eliminated by the maximum heat transfer rate obtained in the constant temperature bath.

Table X. Adiabatic flow boiling Test Plan conditions

Inlet Mass Flow [g/min]	Inlet Subcooling [°C] 0	0.5	1	1.5	2	2.5	3.0
100	X						
125	X	X					
150	X	X	X	X			
175	X	X	X	X			
200	X	X	X	X	X		
225				X	X	X	X

The tests were performed in the order of increasing flow rate, then within a flow rate, in the order of increasing subcooling (decreasing inlet temperature). The increasing subcooling order was used to minimize the impact of the meta-stable region, and to improve the repeatability of the tests should data collection for the day be stopped before completing all the subcooling levels within a flow rate, and.

Test Procedure

The mass flow rate through the loop was adjusted using either the needle valve or the DC power supply or both to attain the target mass flow rate for the test condition. The temperature of the constant temperature bath was also adjusted to obtain the desired inlet subcooling. The inlet temperature target was determined using plots of the temperature vs. pressure for various levels of subcooling. See Fig. 27 for an example. These plots were created in Matlab using the saturation temperature formulas from Irvine [64] used in the 1-D model. The bath temperature and flow rate would have to be adjusted iteratively until steady state was reached. The system was determined to be at steady state when the fractal inlet temperature as well as the other system temperatures had changed less than 0.2 °C over a 5 minute period. Also, the mass flow rate could not have changed more than 5 g/min over the same 5 minute period. Once steady state was

achieved for the desired flow rate and subcooling level, the data acquisition program was triggered to begin recording the global pressure, temperature, and flow rate data. This data was recorded for 60 minutes in 5 second collection intervals. Once the global data collection was started, the image data collection began. A series of 500 images were captured of the two phase flow through the fractal branching channels at random times over a one hour period. The sampling rate was irregular because the image capture trigger was performed manually. Once the data collection was completed for a particular subcooling and flow rate, the next flow condition was set-up. As the system reached steady state for the new test condition, the 500 images obtained were converted from the camera system's proprietary format to TIFF files and both the proprietary format and the TIFF images were burned to Compact Disk for permanent storage.

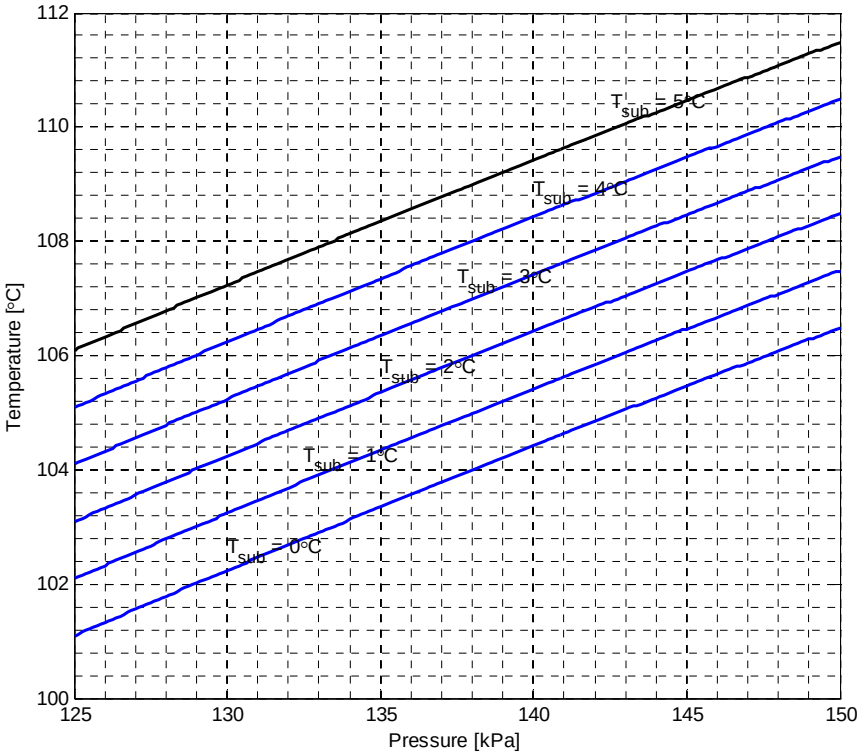


Figure 27. Example plot of inlet temperature targets as a function of pressure.

A documented start-up and shut-down procedure was used to ensure repeatable results and avoid potential equipment damage. The start-up procedure consisted of plugging in all the equipment, ensuring all heaters were initially set to minimum power levels, and starting the data acquisition program. The fractal was then mounted on the vacuum chuck and the valves in the flow loop were set to by pass the fractal and ensure a minimum pressure drop through the loop. The pump was then started and the flow rate was set to a moderate value. Once flow was established, the mass flow meter heaters, reservoir heater, and constant temperature bath were turned on. The data acquisition program was then monitored to determine when the temperatures stabilized. While the flow loop was stabilizing, the Micro-max camera was turned on and the cooled CCD array was allowed to stabilize at -15 °C. The image acquisition set-up was also verified at this time. Once the temperatures stabilized with the mass flow meter heaters on, these heaters were turned off and the temperatures were again allowed to stabilize. Once all the temperatures stabilized again, the bypass valve was positioned to direct the flow to the fractal test device. The loop was then be set to the desired flow condition and allowed to stabilize.

After all the testing had been completed for the day, the shut-down procedure was accomplished. This procedure consisted of first turning off the laser, then configuring the flow loop valves so the fractal test channel was bypassed, and the pressure drop through the loop was minimal. The constant temperature bath set-point was reduced to below 100 °C and allowed to stabilize there. The reservoir heater was turned off. The flow rate was again set to a moderate rate. While the loop cooled, the Micro-Max camera was turned

off, and the images from the last test condition were converted to TIFF format and the image files were transferred to CD. The global data files for the day were also transferred to a thumb drive. Once the fractal had cooled enough to handle, it was removed from the vacuum chuck and placed in the storage container. Once the temperatures throughout the loop dropped below 100 °C the pump was turned off. Then it was ensured that all equipment was turned off and unplugged.

The data acquisition program used to collect pressure, temperature and mass flow rate data from the flow loop was written in Labview, and was capable of simultaneously displaying and recording flow loop parameters. A screen shot of the data acquisition program is shown in Fig 28. All the collected data was displayed in this window. Starting in the upper left corner, the inlet and outlet mass flows and measured densities were displayed. Just to the left of these, the pressure drop across the fractal is displayed both digitally and as a dial indicator. In the upper right corner, the temperatures measured by the thermocouples installed throughout the loop were displayed. Both their current values and the values measured five minutes previously. The purpose of this was to aid the operator in determining steady state operation. In the lower left corner was a plot of the thermocouple data this was also displayed to aid the operator in determining steady state operation as well as giving them the ability to monitor the progress of the data collection. In the lower right corner of the window were the program controls. The switch allowed the operator to either monitor the flow loop with out recording data or monitor the loop and record data. The first position, monitoring only, was intended to be used during warm-up, cool-down and transitions from one test condition and another.

The second position, monitor and record, was intended to be used to record data while on condition and also be able to monitor the loop to ensure everything was working as expected and that the operating condition did not shift during the test.

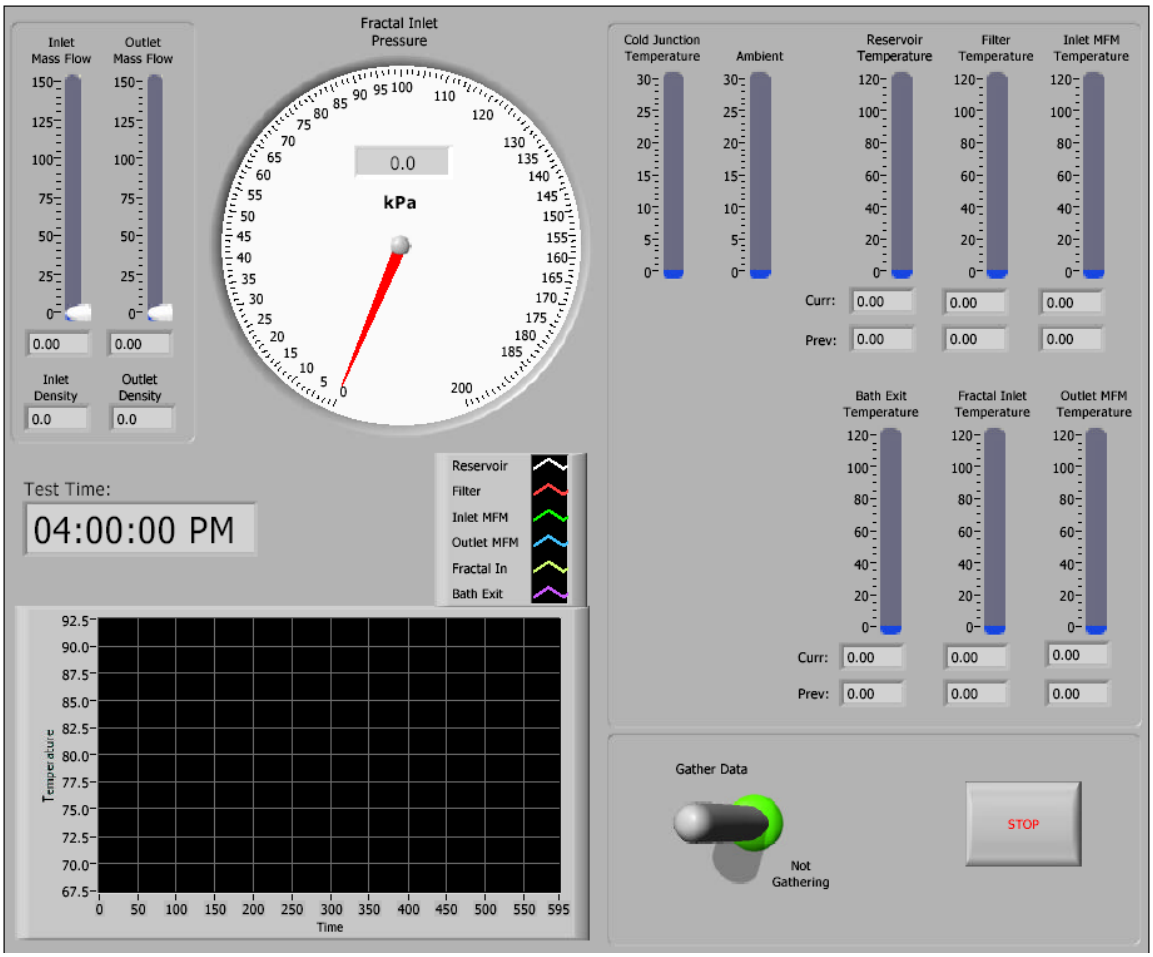


Figure 28. Screen shot of data acquisition program graphical user interface

The mean and standard deviation of each 5 second data interval was saved to the data file. the data was saved in the units as read by the data acquisition card, i.e. frequencies, voltages, resistances, or temperatures depending on the instrument. Each new data point was appended to the data file while in the monitor and record mode. Although data was collected for 5 second intervals, these intervals repeated with a period

of approximately8 seconds. The reason for the approximately 3 seconds of dead time between each data interval was a combination of the time required to process and convert the collected date for display in engineering units, and the time required for the PC to communicate with serial devices particularly the device being used to measure the resistance of the RTD's.

Data Analysis

Three values, exit quality, inlet subcooling and channel pressure drop were calculated from the data collected from flow loop instrumentation. The void fraction was calculated by analyzing the images captured at each flow condition.

Exit Quality

The exit quality was calculated based on three assumptions. The first was no heat loss from the liquid as it traveled along the channels. The second was the liquid and vapor phases were in equilibrium at the exit of the channels. And the third was the enthalpy of the liquid entering was the same as that of saturated liquid at the same temperature. Based on these assumptions, the inlet enthalpy was calculated from the inlet temperature then the exit quality, *w*, was calculated from

$$\hbar_{inlet} = \hbar_{exit} = (1-w)\hbar_{l,exit} + w\hbar_{v,exit} \tag{39}$$

Where $\hbar_{l,exit}$ and $\hbar_{v,exit}$ were calculated based on the saturation temperature calculated from the measured atmospheric pressure. Isolating and solving for the exit quality results in

$$w = \frac{\left(\hbar_{inlet} - \hbar_{l,exit}\right)}{\left(\hbar_{v,exit} - \hbar_{l,exit}\right)} \tag{40}$$

Inlet Subcooling

The inlet subcooling was calculated by subtracting the measured inlet temperature from the saturation temperature based on the inlet pressure.

$$T_{sub} = T_{sat}\big|_{P_{in}} - T_{in} \tag{41}$$

The saturation temperature is calculated directly from the measured inlet pressure. The correlation used to perform this calculation was obtained from Irvine [64]

Channel Pressure Drop

The channel pressure drop measurement was actually measured just at the inlet to the vacuum chuck. As a result this pressure drop must be corrected to account for the losses between where the pressure was actually measured and the beginning of the fractal channels. Figure 29 shows a schematic of the flow passage between the measurement point and the channel inlets. Table XI shows the values for the dimensions called out in Fig. 29.

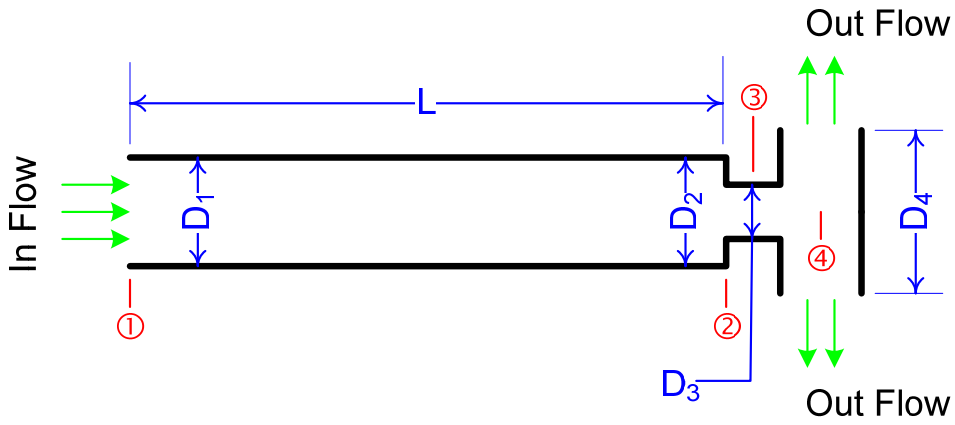


Figure 29. Schematic of passage between pressure measurement location and fractal channel inlets

Table XI. Inlet dimension values

Dimension	Value
D_1	2.03 mm
D_2	2.03 mm
D_3	1.75 mm
D_4	2.2 mm
L	69.6 mm

This passage has been broken down into three regions for which pressure drops are calculated. The first region, from point 1 to point 2, is a simple constant diameter tube, where the pressure drop is calculated using the standard formula shown below:

$$\Delta P_{1-2} = 2 f_f \frac{L}{D} \rho V_1^2 \tag{42}$$

Where f_f is the fanning friction factor, L is the length of the tube, D is the diameter of the tube, ρ is the density of the fluid and V_1 is the velocity of the fluid through the tube. Since the flow in the tube is turbulent for all flow cases, and the Reynolds number is less than 10,000 for all cases the Blasius formula for the fanning friction factor was used:

$$f_f = \frac{0.079}{\mathrm{Re}^{1/4}} \tag{43}$$

where Re is the Reynolds number.

The flow in the second region, from point 2 to point 3, is a sudden contraction, and the pressure drop is calculated from the following:

$$\Delta P_{2-3} = \tfrac{1}{2} \rho V_3^2 \left(1 - \left(\frac{D_3}{D_2} \right) + K_{2-3} \right) \tag{44}$$

where $K_{2\text{-}3}$ is the loss coefficient and it is calculated per the following formula:

$$K_{2-3} = \tfrac{1}{2}\left(1 - \frac{D_3}{D_2}\right) \tag{45}$$

The third region from point 3 to point 4 is a sudden expansion, and the pressure drop is calculated from

$$\Delta P_{3-4} = \tfrac{1}{2}\rho V_3^2 \left(\left(\frac{D_3}{D_4}\right)^2 - 1 + \left(1 - \frac{D_3}{D_4}\right)^2 \right) \tag{46}$$

The total inlet pressure correction is then calculated by summing pressure drop from the three regions:

$$\Delta P_{IC} = \Delta P_{1-2} + \Delta P_{2-3} + \Delta P_{3-4} \tag{47}$$

Figure 30 shows a plot of the inlet pressure correction as a function of mass flow rate. These corrections result in less than a 1% shift in the measured value.

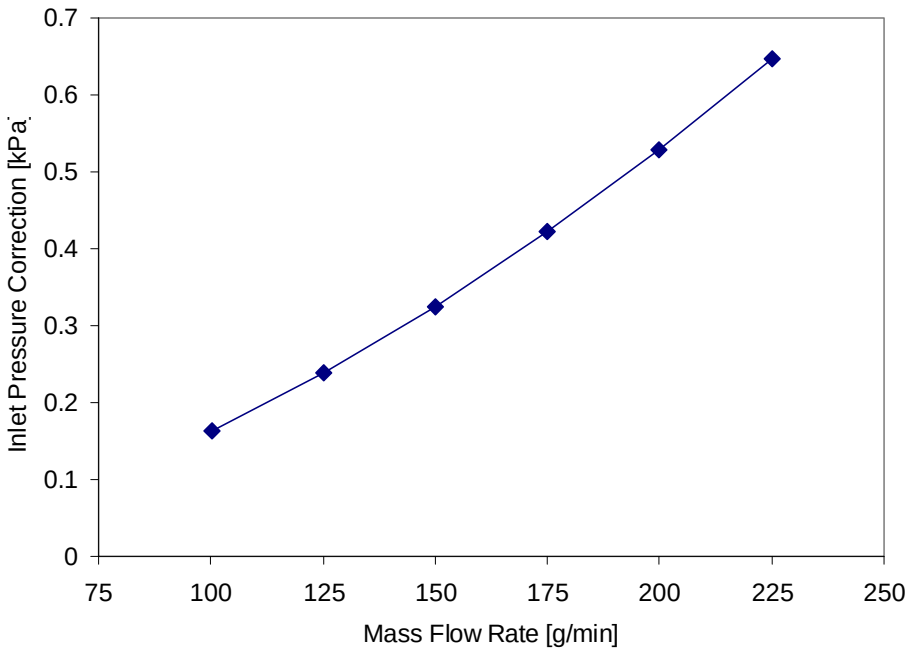


Figure 30. Inlet pressure correction as a function of mass flow rate.

The actual pressure drop across the fractal channel then is calculated by subtracting the calculated inlet pressure drop correction from the measured pressure drop.

$$\Delta P = \Delta P_C - \Delta P_{IC} \tag{48}$$

The inlet pressure drop correction is nearly negligible, but has been accounted for in the reported results.

The inlet pressure at the channel inlet which is needed for the saturation temperature is calculated by adding the atmospheric pressure, P_{atm}, to the channel pressure drop.

$$P_{in} = \Delta P + P_{atm} \tag{49}$$

Void Fraction

The void fraction was measured optically using high resolution digital images of the two phase flow in the fractal channels. The liquid phase is differentiated from the vapor phase by use of a fluorescent dye which when excited by a laser emits light at a different wavelength than the laser light. The reflected laser light is then filter out, allowing only the light from the fluorescing liquid through to the CCD array. The resulting image is one where the liquid regions are light colored and the vapor regions are dark colored. A sample image is shown below in Fig 31.

These images are converted to black and white by selecting a threshold intensity value above which the pixel is assumed to be liquid and below which the pixel is assumed to be vapor. The binarized sample image is shown below in Fig 32.

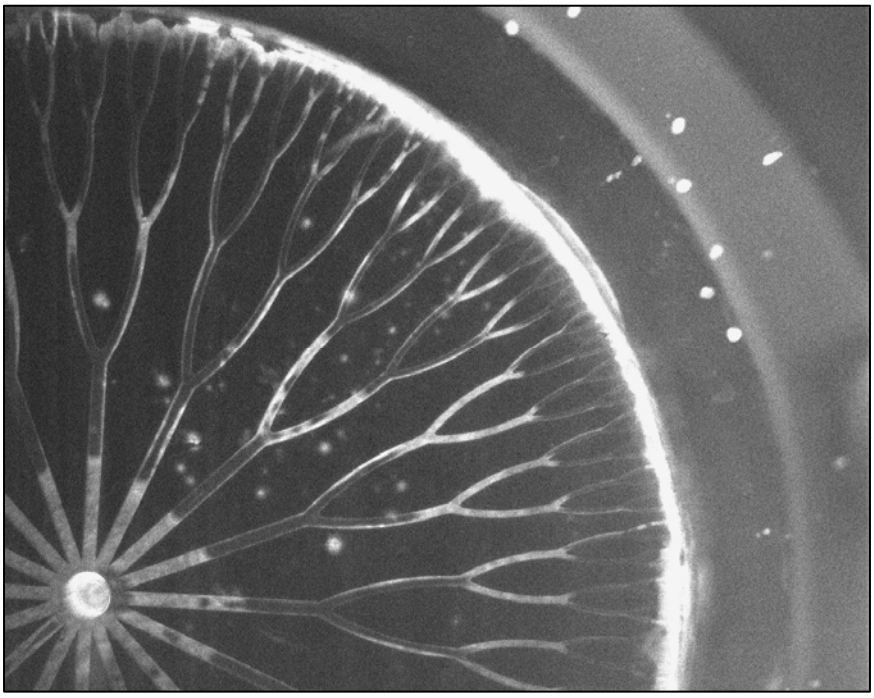


Figure 31. Sample image of two phase flow in fractal-like branching channel network. Showing liquid phase as light intensity areas and vapor phase as dark intensity areas

All liquid images are generated by logically or-ing all the data images together. An example all liquid image is shown in Fig 33. From this all liquid image regions of the branching channel network corresponding to one segment are identified. Regions are defined for every channel of each branching level. The bifurcations are excluded from the regions. An example of the identified data regions are shown in Fig. 34. Liquid (white) pixels are counted within these regions and void fractions are calculated for the channels sections within these regions based on these pixel counts.

Figure 32. Binarized sample image showing liquid regions as white and vapor and non flow regions as black.

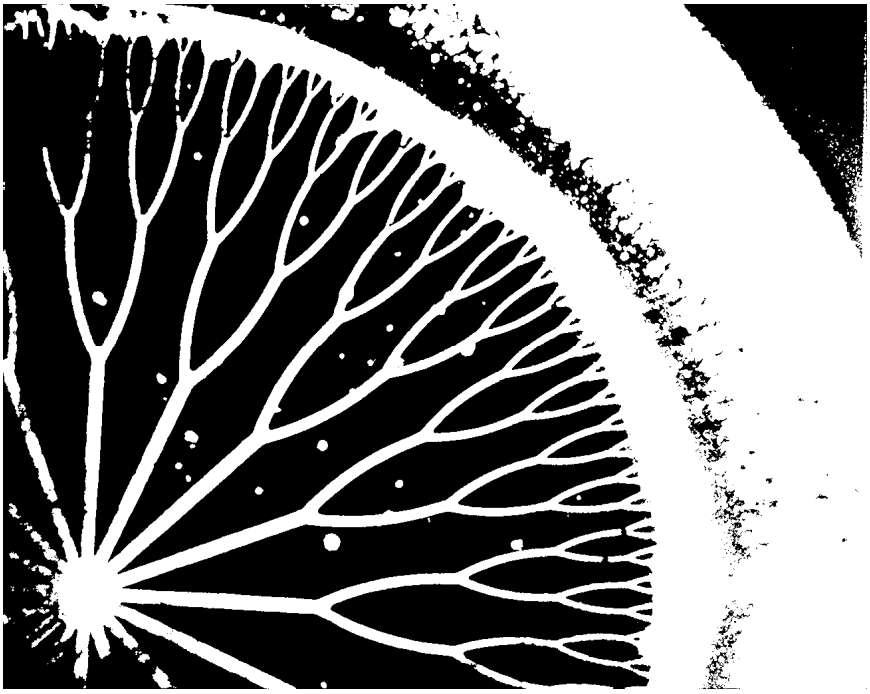


Figure 33. Sample all liquid image built by or-ing all data images together.

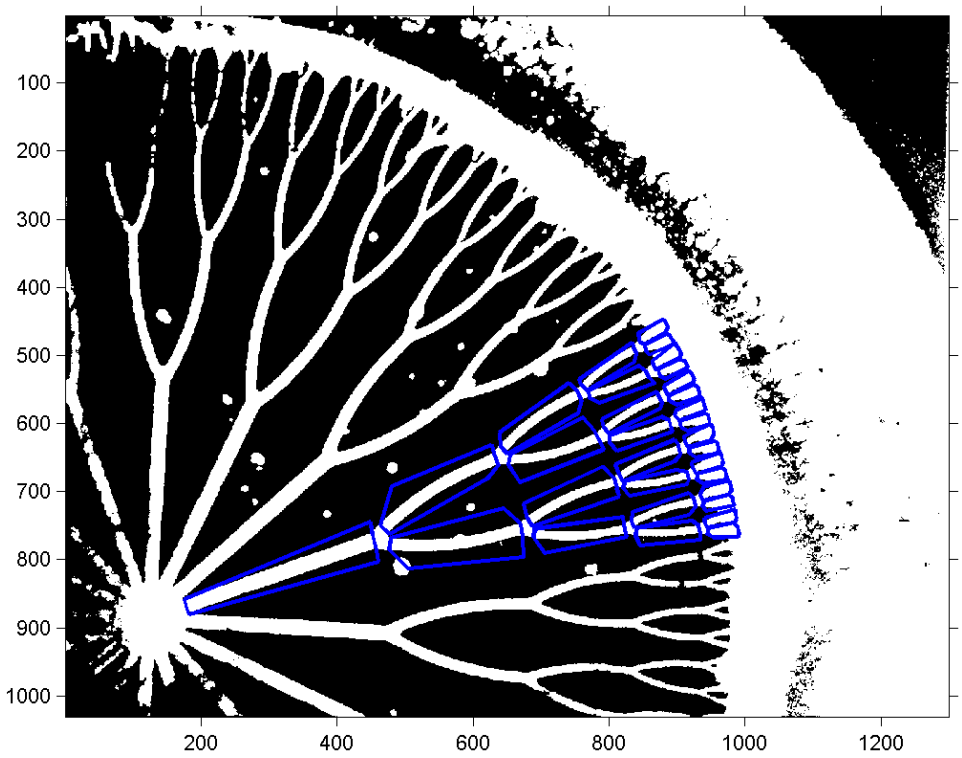


Figure 34. All liquid image from above with interrogation regions shown

The void fraction for the i[th] region of the j[th] data image, $VF_{i,j}$, is calculated by counting the liquid (white) pixels in the i[th] region in the data image, $LP_{i,j}$, then dividing this value by the liquid pixel count for the i[th] region of the all liquid image, $ALLP_i$, for that test condition, then subtracting the quotient from one.

$$VF_{i,j} = 1 - \frac{LP_{i,j}}{ALLP_i} \tag{50}$$

Data Problems

There were several problems encountered with the data collected that were only noticed once the data analysis began. Most of the problems were issues with the data images, but there was one issue with the global pressure drop data. All the issues were

either overcome or corrected for. The following sections describe how each issue was rectified.

Pressure drop shift

In examining the measured pressure drop data it was noticed that the pressure drop data had a larger shift between the 175 g/min flow rate and the 200 g/min flow rate than between the other flow rates, see Fig 35. The slopes of all the data sets however do seem to be similar. Assuming the slopes of the data for each mass flow rate were the same, the intercept of each data set was calculated. This intercept data is shown in tabular form in Table XII, and graphical form in Fig. 36. Notice that the intercepts the lowest four flow rates follow a parabolic trend. Also notice that the two highest flow rates seem to be shifted upward by a constant value. The two red data points are these same two data points shifted down by a constant value. The constant correction value was determined by minimizing the total deviation of these data points from the parabolic curve fit through the first four points. Resulting in a correction factor value of 10.9 kPa.

Table XII. Tabular intercept data for each flow rate assuming equal slopes

Mass Flow Rate	Intercept of data set
100	-0.368
125	-0.120
150	2.517
175	6.318
200	24.224
225	29.643

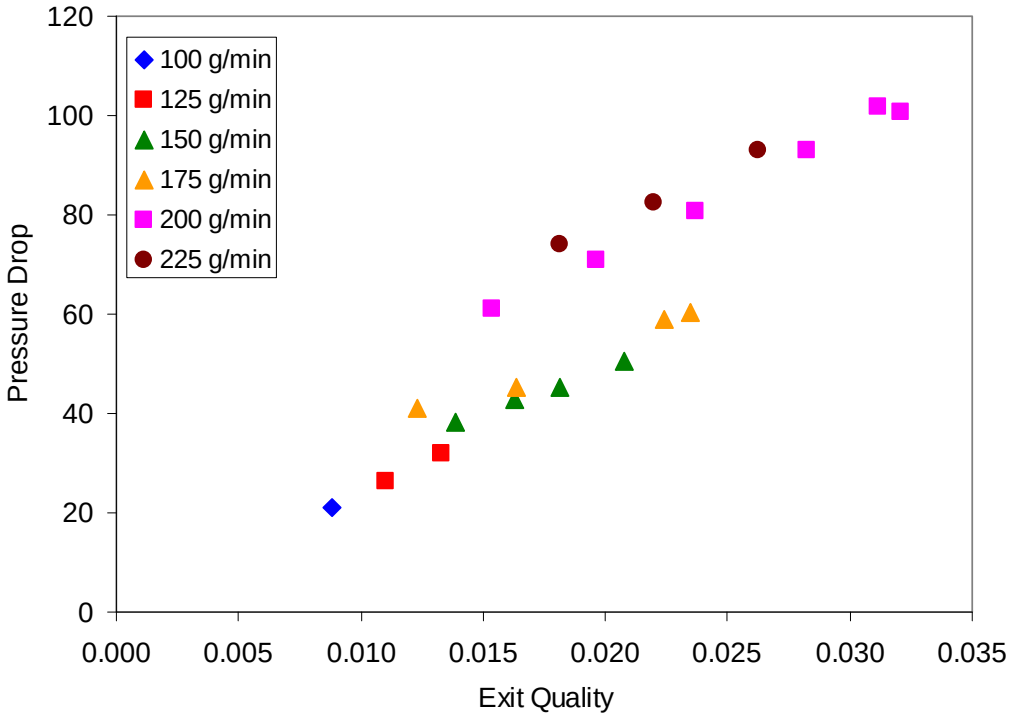


Figure 35. Pressure drop data as a function of exit quality as measured minus inlet pressure drop correction

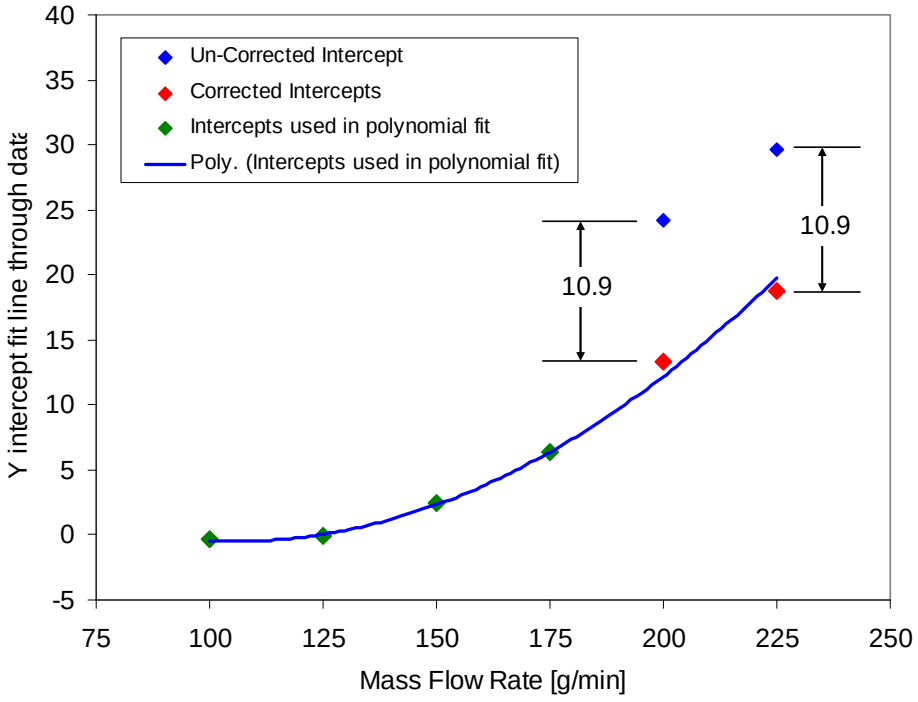


Figure 36. Plot of y-intercept of data for each mass flow rate. Plot shows as measured intercepts and corrected intercepts for 200 and 225 g/min flow rates.

Image Defects

There were several image defects which showed up in the data images and had an effect on the calculation of the void fraction in the interrogation regions shown in Fig 34. Some of these defects were corrected for, and some required the void fractions calculated from those images to be removed from the data set. In some image data sets, those from conditions 63, 64, 65, and 66, the images were of such poor quality due to combinations of all the effects detailed below, that none of the images could be processed to provide valid void fractions. For these data sets, only the data obtained from the flow loop was reduced.

Streaky Image Defects

As was mentioned in a previous section, a conical deflector was fitted to the top of the fractal disk to prevent liquid collection on the top surface at the highest flow rates. Although this method was successful in preventing the liquid collection on the top of the disk, it did collect on the deflector surface. The droplets that collected were large in comparison to the fractal channels and therefore fluoresced much brighter that the channels due to the larger volume of dye in them. These very bright droplets, when they became large tended to saturate the CCD array and caused streaking on the CCD image, see Fig. 37 a). It was not realized at the time of data collection, but the intensity of these streaks was larger than the intensity of the liquid phase in the non-streaked areas. When the image was binarized, these streaks caused large white stripes in the black and white images, see Fig. 37 b).

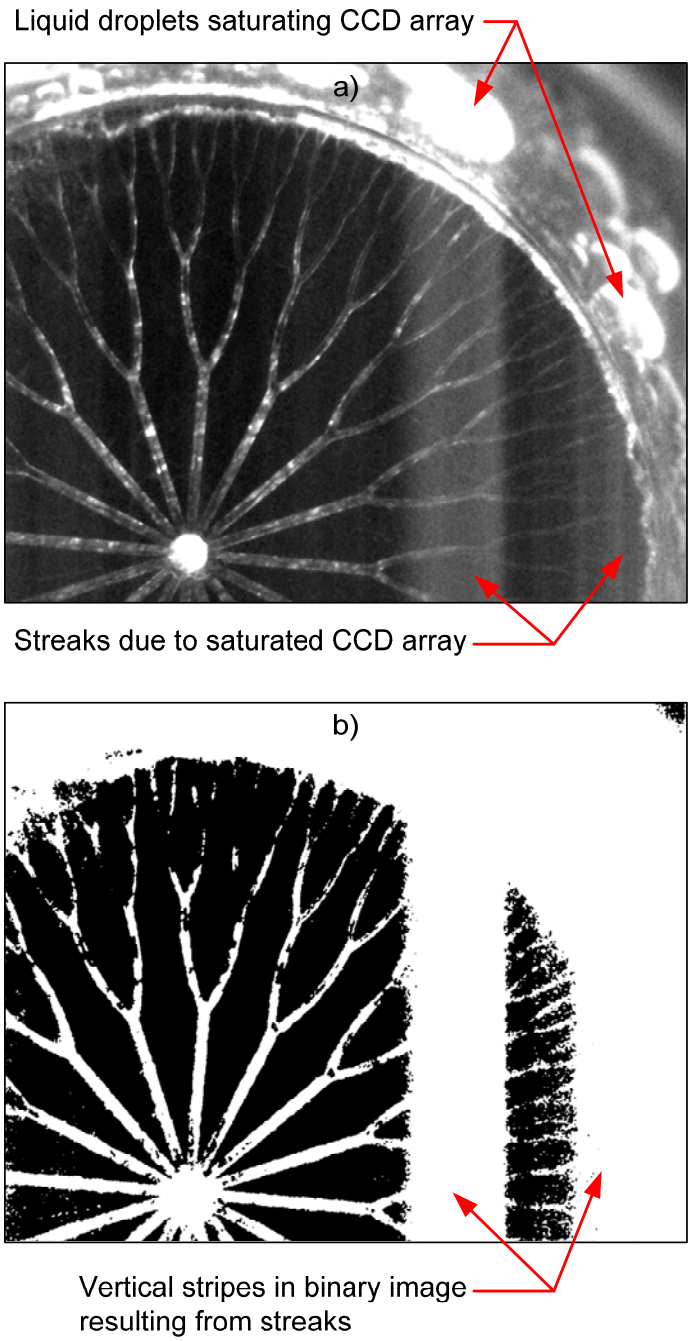


Figure 37. Image showing streaking due to CCD saturation at droplets indicated. Showing a) gray scale image, and b) resulting binary image

Only the region to the right of the inlet port needed to be corrected to have the streaks removed since the interrogation regions for the void fraction were only in this region of the image. In order to normalize the grey scale images and remove the streaks

three intensity profile paths were selected in areas that are not nominally non-channel regions, see Fig 38 a). Plots of the intensity as a function of horizontal pixel number are shown in Fig. 38 b) as the thin red, green and blue lines. All three paths show identical trends with random noise superimposed. The intensity values for each horizontal pixel value are averaged from the three paths. Since the three paths do not all begin and end at the same horizontal position, in the areas near the beginning and end of paths the average may be based on a sample of one or two rather than all three of paths. This average is then smoothed by passing applying a three cell rolling average filter to the data set. This smoothed average is also shown in Fig. 38 b) as the thick black line. The correction array was generated by subtracting 20 from the smoothed average intensity, if the smoothed average intensity value was less than 20 at a particular pixel location, then a correction value of 0 was assigned for that pixel location. The correction array is also plotted in Fig. 38 b) as the thick blue line.

The offset value of was 20 selected because the background intensity near the center of the images was approximately 20. This provided a smooth transition with the uncorrected image and a relatively flat background intensity over the corrected region.

The correction was applied to the grey scale image by subtracting the correction value for a horizontal pixel location from the intensity values of all the pixels at that horizontal pixel location. The resulting corrected image with the vertical smearing removed is shown in Fig. 39 a), and the corresponding binary image showing the liquid and vapor phases throughout the image is shown in Fig. 39 b). This technique was applied to all the images which exhibited the vertical smearing, conditions 51, 52, 53, 54.

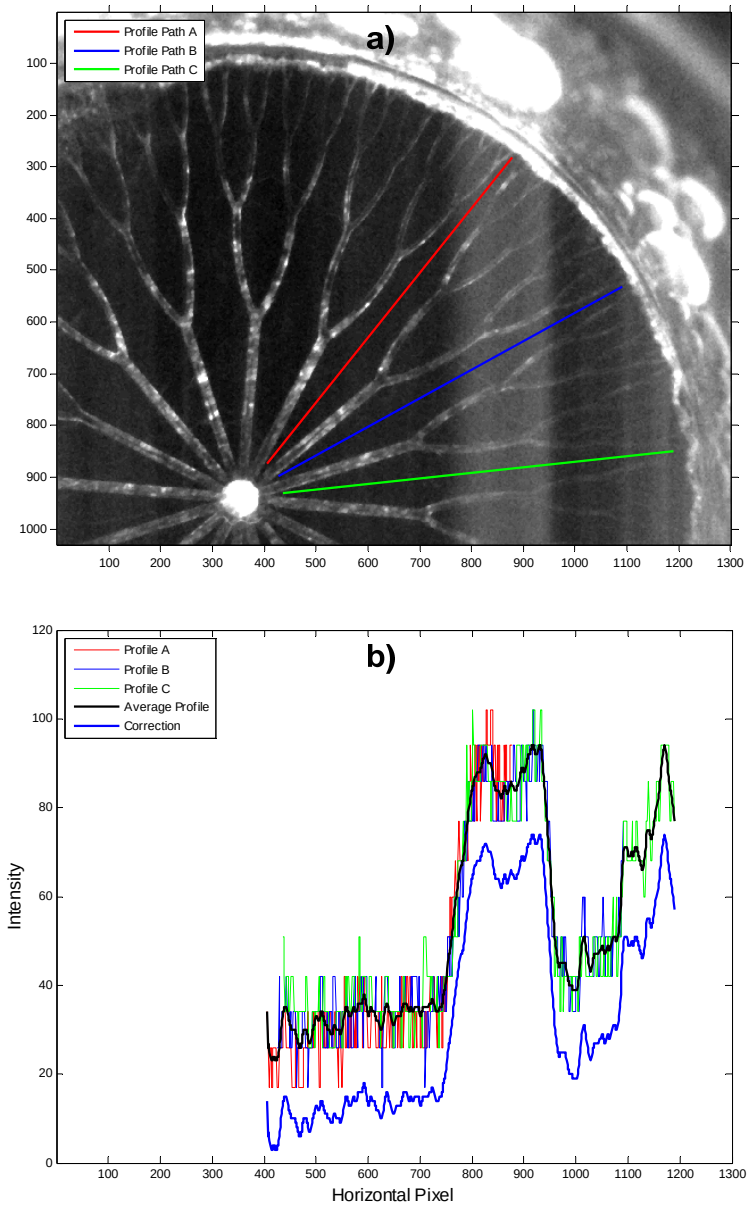


Figure 38. Intensity profile explanation. Showing a) intensity profile path locations on image, and b) plots of intensity profiles

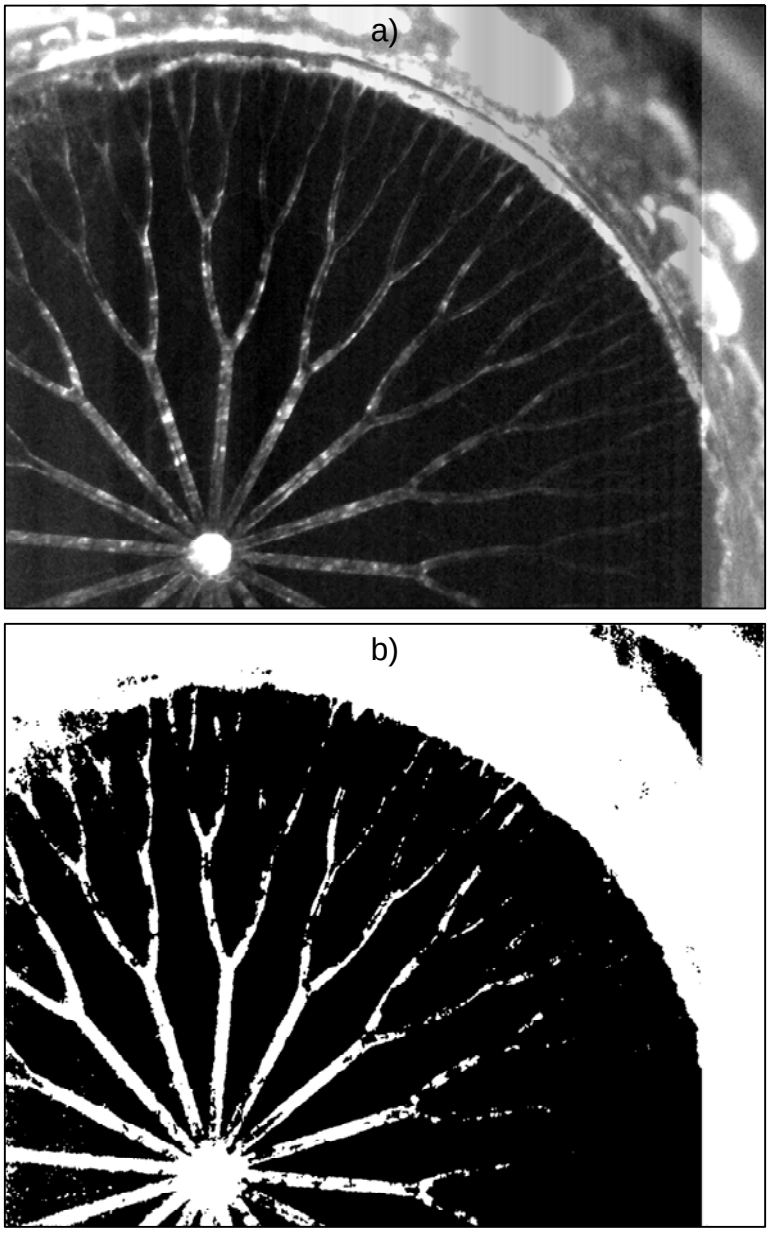


Figure 39. Example of image intensity profile correction. Showing a) corrected gray scale image and b) resulting binarized image

Blotchy Image Defects

The other type of image defect found in the data images are light blotches over the data collection regions. This defect is not as prevalent as the streaked image defect; however is it present in all test cases, however no test condition has more than about 100

images that fall into this defect category. This is not a significant number of images given that the average void fraction is well established by about the 100^{th} image in the data set, as will be shown later. There are two primary causes of these blotches. The first cause is liquid pools which collected on the top of the fractal due to over spray from the liquid/vapor mixture exiting the fractal. The second is an artifact of the streaky image defects when the streaks do not continue down the entire image. Since the blotches are bright areas with intensities above the threshold level they become white regions on the binary images and have the effect of shifting increasing the liquid pixel count above the count from the all liquid images. Since the void fraction is calculated by essentially first calculating the liquid fraction then subtracting this from one to get the void fraction, if the liquid pixel count for that region of the data image is greater than the all liquid pixel count for that same region, the liquid fraction then is greater than one and the void fraction is not valid. In most cases when this defect occurs the ratio of the liquid pixels, *LP*, to the all liquid pixels, *ALLP*, is much greater than one resulting in a negative void fraction. Figure 40 shows an example set of images when the blotch is caused by a liquid pool, and Fig. 41 shows an example of images when the blotch is caused by an artifact of the streaky image defect. Since there is no way to correct these images, the data from these images were removed from the data set.

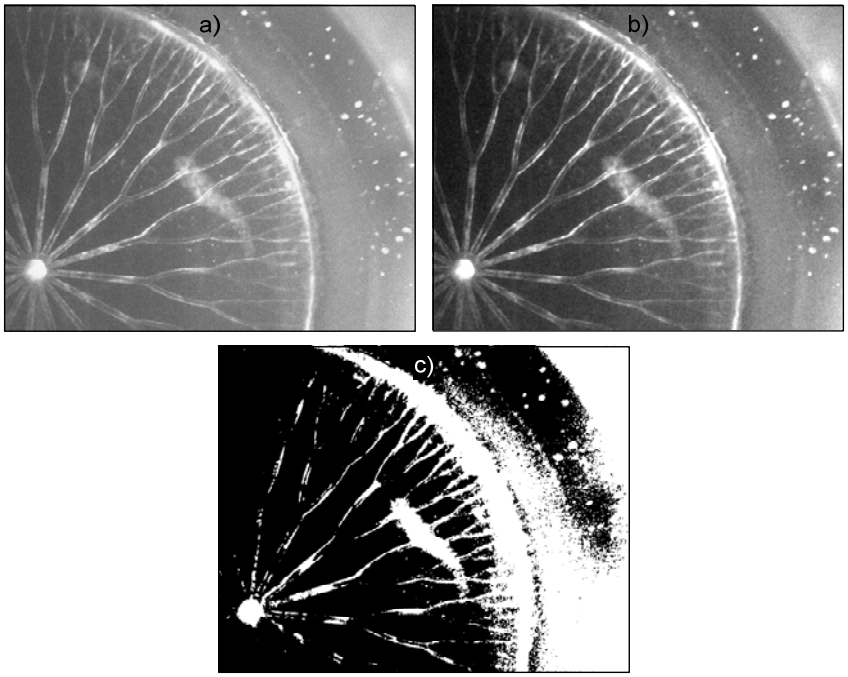


Figure 40. Example blotchy image, image 225 from flow condition 30, caused by liquid pool collected on top of fractal device. Showing a) raw image, b) intensity adjusted image, and c) black and white image.

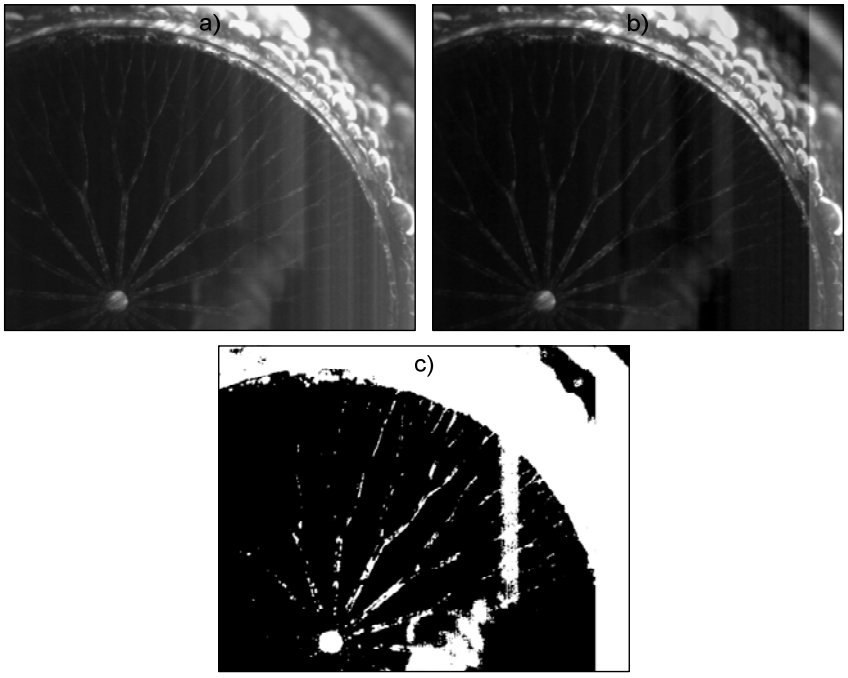


Figure 41. Example blotchy image, image 244 from flow condition 50, caused by an irregularity in a streaky image defect. Showing a) raw image b) intensity adjusted image including profile correction and c) black and white image.

Effect of Image Defects on reported Void Fraction

When there are a number of images with image defects in a sequence, such as from a liquid pool, they can have an effect on the cumulative void fraction average, $\bar{x}_c$, which is calculated according to the following formula

$$\bar{x}_{c\,i} = \frac{1}{i}\sum_{j=1}^{i} VF_j ; i = 1,2,3,...n \tag{51}$$

An extreme example from flow condition 20 is shown in Fig. 42, where for some reason a sequence of four images are significantly brighter over the entire image. These images are bright enough that the binarized images are also nearly entirely white. This causes the void fractions for the branching channel regions to be negative, Fig. 42 b). This also causes a shift in the cumulative average void fraction, Fig. 42 c). The cumulative average takes a while to recover, and in the plot shown, it still has not recovered even though an additional thirty images have been added to the average. Removal of the four offending image from the data set completely alleviates the problem as shown in Fig. 42 d)

Figure 43 shows the effect a single image can have on the cumulative void fraction average. In this case image #218 from flow condition 51 has an artifact from the streaky image defect, sequence of images shown in part a). The effect on the void fraction is apparent in the plot shown in part b), only channel B is significantly affected. The cumulative void fraction average also shows the effect of the negative void fraction with a step change for channel B at image #218. The cumulative average, part c), recovers to the value before the image defect, but not until after image #250. Part d) of the figure shows the cumulative void fraction average when the bad images are removed from the data set.

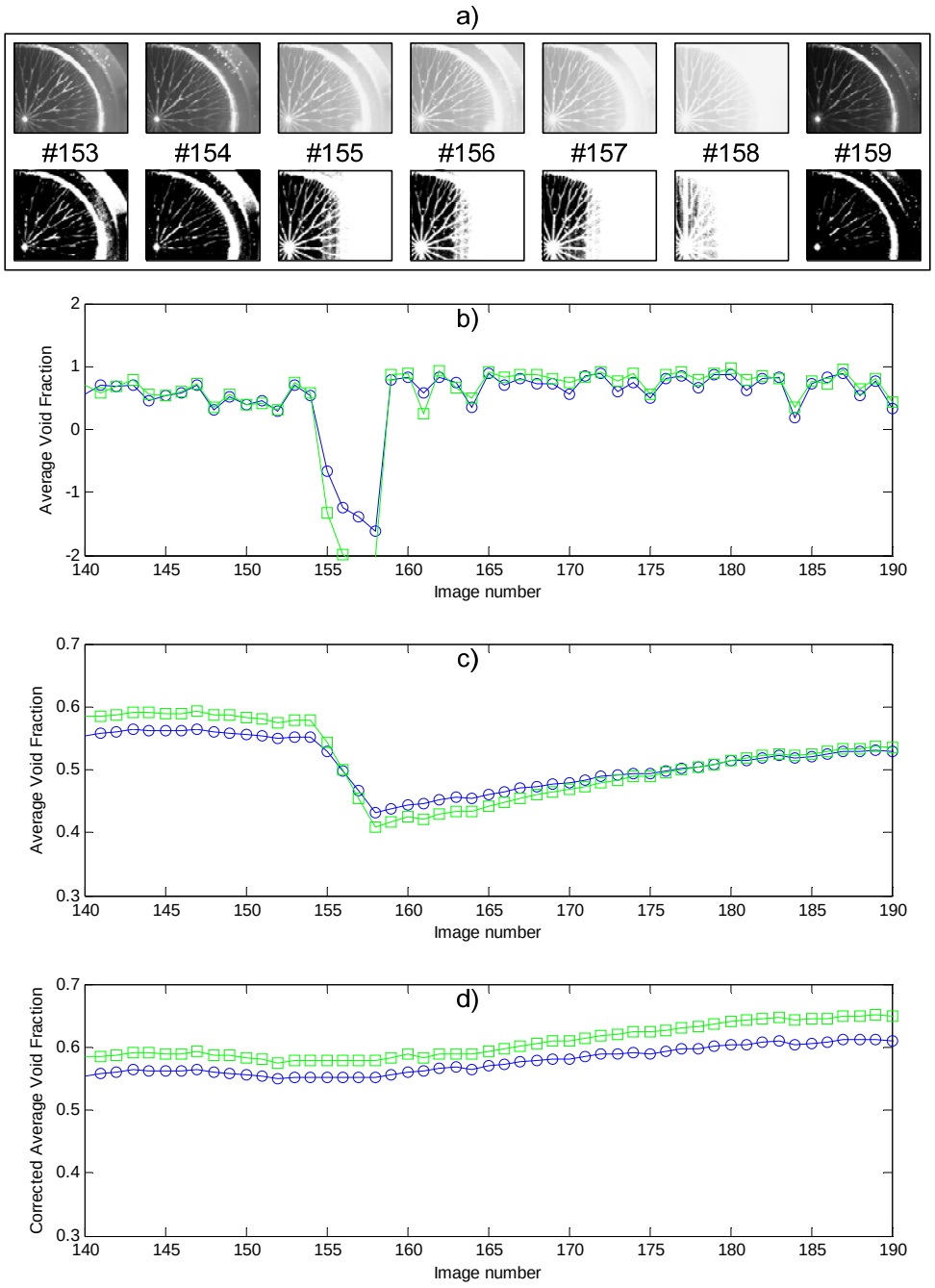


Figure 42. Effect of a series of image defects on the void fraction and the rolling average of the void fraction. Showing a) image sequence, image numbers 153 through 159 from flow condition 20, b) void fraction in 1st branching level as a function of image number, c) cumulative void fraction average as a function of image number, and d) corrected cumulative void fraction as a function of image number.

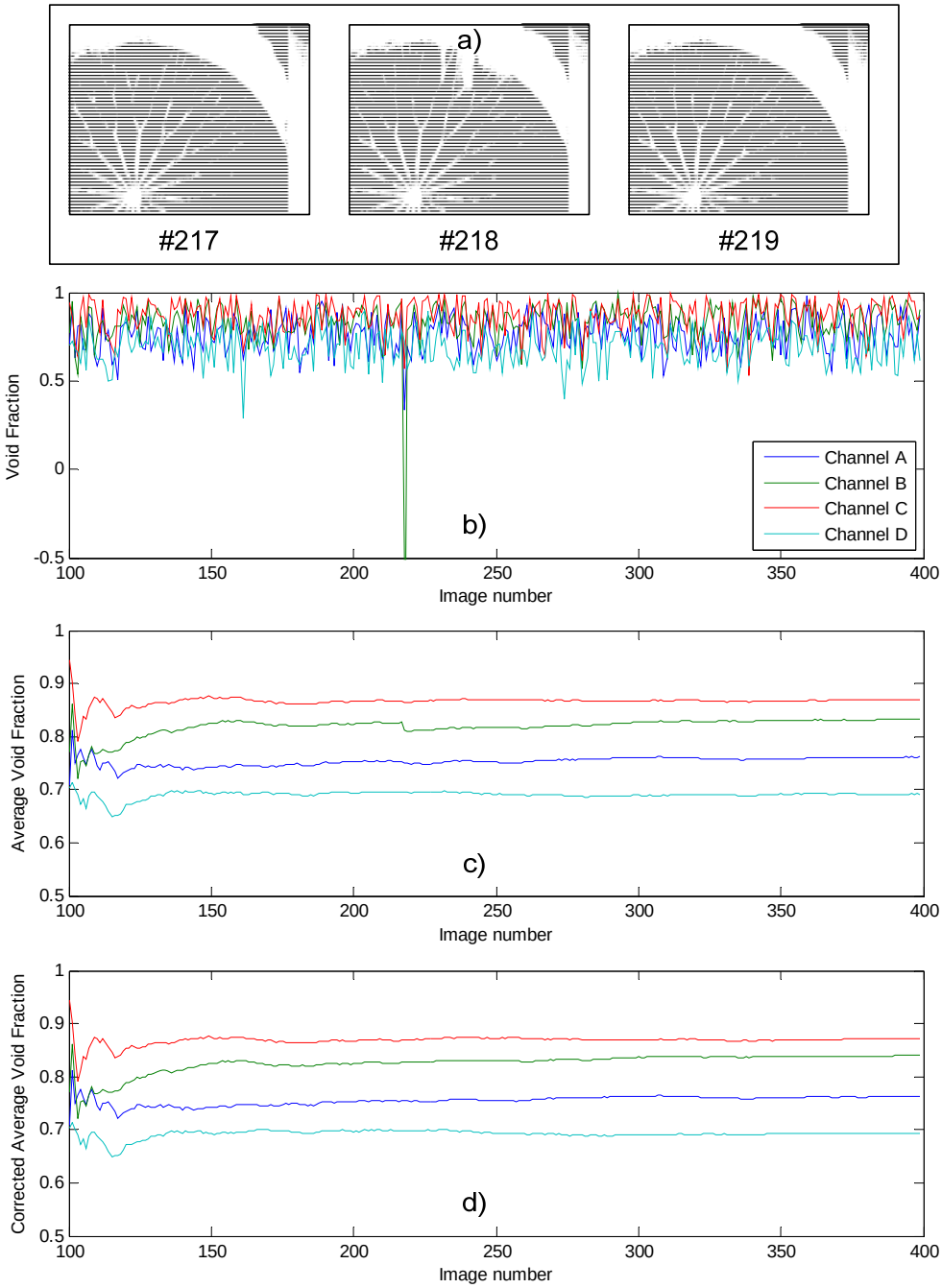


Figure 43. Effect of a single blotchy image defect on the void fraction and the rolling average of the void fraction. a) shows image sequence, images 217 through 219 from flow condition 51, b) void fraction in 2nd branching level as a function of image number, c) cumulative void fraction average, d) cumulative void fraction average after data from images with defects have been removed from data set.

One other item to observe from this figure is how quickly the cumulative void fraction average stabilizes to the steady state value. The image numbers begin at 100, so by about the 50th data point, image 150, the cumulative average has pretty much reached it's steady state value, and certainly by the 100th data point, image 200, there are not any more significant changes occurring.

Uncertainty Analysis

The Klien & McKlintock method was used to determine the uncertainty in the results derived from the measured values. The process is first explained, then used to obtain the uncertainty of a representative example shown below, and the calculations for the remainder of the calculated values are included in Appendix B

The total uncertainty is a combination of the bias and precision uncertainties per the following formula:

$$U_{T_{sub}} = \sqrt{B_{T_{sub}}{}^2 + \left(t_{\upsilon,c} P_{T_{sub}}\right)^2} \tag{51}$$

where $B_{T_{sub}}$ is the bias uncertainty, $P_{T_{sub}}$ is the precision uncertainty and $t_{\upsilon,c}$ is the student's t distribution parameter to account for a finite sample size. The two parameters in the student's t distribution parameter are υ which represents the degrees of freedom in the data set, and *c* which represents the confidence level, which for this study is 95%.

The bias and precision uncertainty of the calculated values are determined in similar ways from the base bias and precision uncertainties of the directly measured values. The base bias uncertainties of the directly measured values are determined from either the standard error of a calibration curve fit, instrument specifications, or for measurements with no calibration curve, the bias uncertainty is estimated as half the

smallest division of the measurement device. The precision errors are determined from statistical analysis of the repeated measurements at each condition. For those measurements which do not have repeated measurements, the total uncertainty of the measurement is assumed to be contained in the bias uncertainty alone.

The bias and precision uncertainties are propagated from the base measurement uncertainties. Formulas for some simple cases can be developed. The general case as well as two common cases, addition and products of powers, are shown in Table XII**.**

Table XIII. Uncertainty propagation formulas

Case	Result Formula	Uncertainty Formula	
General	$R = f(x, y, z)$	$U_R = \sqrt{\left(\frac{\partial R}{\partial x} U_x\right)^2 + \left(\frac{\partial R}{\partial y} U_y\right)^2 + \left(\frac{\partial R}{\partial z} U_z\right)^2}$	(52)
Addition	$R = ax + by$	$\frac{U_R}{R} = \sqrt{\frac{(aU_x)^2 + (bU_y)^2}{(ax + by)^2}}$	(53)
Product of Powers	$R = x^a y^b z^c$	$\frac{U_R}{R} = \sqrt{\left(a\frac{U_x}{x}\right)^2 + \left(b\frac{U_y}{y}\right)^2 + \left(c\frac{U_z}{z}\right)^2}$	(54)

Where x, y, and z represent the measured variables of interest, a,b and c represent constants, R is the calculated result and Ux, Uy, and Uz represent either the bias or precision uncertainties of the variables of interest.

Exit Quality Example

These uncertainty formulas, are applied to the formulas used to calculate the exit quality which are propagated back to the base measured values of temperature pressure, geometry, and flow rate. How the bias and precision uncertainties propagate back to these raw measured values is shown in tree form in Fig. 44.

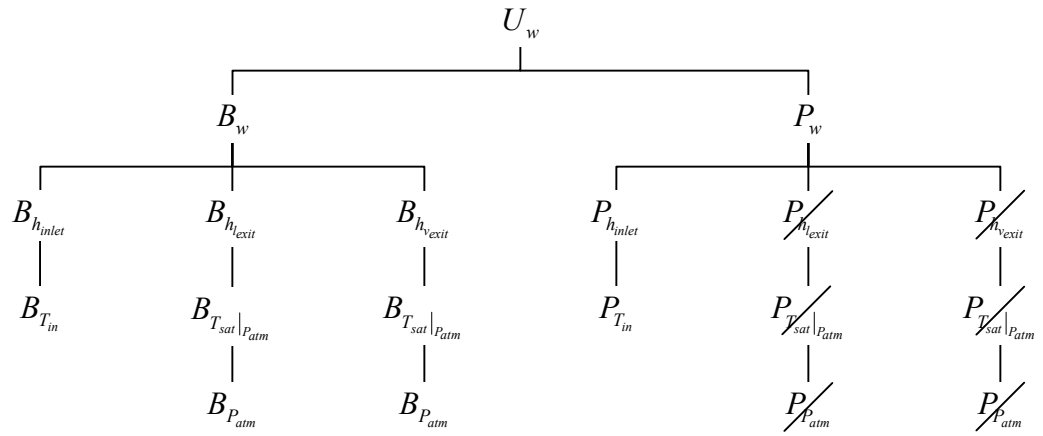


Figure 44. Tree diagram for uncertainty calculation for exit quality.

Starting with the bias uncertainty and applying the general case of the uncertainty formula to the exit quality formula results in the following expression for the bias error in the exit quality.

$$B_w = \sqrt{\left(\frac{1}{h_{v,exit} - h_{l,exit}} B_{h_{inlet}}\right)^2 + \left(\frac{h_{inlet} - h_{v,exit}}{\left(h_{v,exit} - h_{l,exit}\right)^2} B_{h_{l,exit}}\right)^2 + \left(-\frac{h_{inlet} - h_{l,exit}}{\left(h_{v,exit} - h_{l,exit}\right)^2} B_{h_{l,exit}}\right)^2} \quad (55)$$

where, $B_{h_{inlet}}$ is the bias uncertainty of the inlet enthalpy, $B_{h_{l,exit}}$ is the bias uncertainty of the enthalpy of the liquid exiting the fractal device, and $B_{h_{l,exit}}$ is the bias uncertainty of the enthalpy of the vapor exiting the fractal device. Since the inlet enthalpy is determined based on temperature using a complex formula evaluated using a Matlab function, the general formula is again used to determine the uncertainty of the inlet enthalpy based on the inlet temperature, however the partial derivative is estimated using the following expression based on the definition of the derivative

$$\frac{\partial h_{inlet}}{\partial T_{in}} = \frac{h_{inlet}\big|_{T_{in}+B_{T_{in}}} - h_{inlet}\big|_{T_{in}}}{B_{T_{in}}} \quad (56)$$

where $B_{T_{in}}$ is the bias error in the inlet temperature, when this expression is substituted into the formula for the bias error of the inlet enthalpy, the following expression results

$$B_{h_{iinlet}} = h_{inlet}\Big|_{T_{in}+B_{T_{in}}} - h_{inlet}\Big|_{T_{in}} \tag{57}$$

The exit liquid and vapor enthalpies formulas are based solely on temperature are also determined by complex formulas again, evaluated using Matlab functions, the temperature is determined by a complex formula based solely on the atmospheric pressure, and the enthalpies are mathematically based solely on the atmospheric pressure. Based on the inlet enthalpy development, the bias errors for the exit liquid and vapor enthalpies are calculated using the following expressions.

$$B_{h_{l,exit}} = h_{l,exit}\Big|_{Patm+B_{Patm}} - h_{v,exit}\Big|_{Patm} \tag{58}$$

$$B_{h_{v,exit}} = h_{v,exit}\Big|_{Patm+B_{Patm}} - h_{v,exit}\Big|_{Patm} \tag{59}$$

The precision errors are calculated in an identical way as the bias errors, however some variables are eliminated because there are no repeated measurements performed for those variables. For example, the atmospheric pressure was only measured once during each experimental condition, therefore it is not a repeated measurement and has no statistics associated with it. Since the precision errors are based on the statistics associated with the repeated measurements, there can be no precision error for single measurements. Therefore the precision errors associated with those variables are eliminated from the precision formulas. This results in the following expression for the precision error for the exit quality.

$$P_w = \frac{1}{h_{v,exit} - h_{l,exit}} P_{h_{inlet}} \tag{60}$$

where the $P_{h_{inlet}}$ is the precision error for the inlet enthalpy, the only variable in the exit quality expression based on a repeated measurement. The precision error for the inlet enthalpy is calculations in an identical manor as the bias error, this time using the precision error for the inlet temperature

$$P_{h_{iinlet}} = h_{inlet}\Big|_{T_{in}+P_{T_{in}}} - h_{inlet}\Big|_{T_{in}} \tag{61}$$

The bias and precision errors are combined using Eqn. (50) where the degrees of freedom are the same as the degrees of freedom for the inlet temperature measurement.

Table XIV. Maximum and Minimum uncertainty values for each calculated value at each flow rate:

Measured Value	Maximum Uncertainty	Test Condition for Maximum	Minimum Uncertainty	Test Condition for Minimum
Exit Quality	0.0027	10	0.0014	21
Subcooling	0.336 °C	10	0.332 °C	50
Channel inlet Pressure	0.376 kPa	42	0.373 kPa	10
Channel DP	0.362 kPa	42	0.360 kPa	10
Inlet Pressure Drop	0.041 kPa	42	0.006 kPa	10

Table XIV shows the minimum and maximum uncertainty values from the experimental values for the calculated values shown above. Most are well within acceptable levels especially compared to the magnitude of the calculated values. For example, the channel pressure drop uncertainty never exceeds more than 2% of the measured value. The uncertainty of the inlet pressure drop can reach 20% of the

calculated inlet pressure drop, however when compared to the measured pressure drop it is quite small, well less than 1%. The subcooling uncertainty is much higher, but this is driven by the uncertainty in the inlet temperature from the RTD, which is of the order of 0.3 °C. The uncertainty in the exit quality also reaches a maximum of about 30% of the measured value for the lowest qualities measured, but most of the measurements are at about 10% of the measured value.

Results

Pressure drop and void fraction data for two phase flow through the fractal device is presented as functions of inlet subcooling, mass flow rate and exit quality. The fractal device consisted of fractal like branching channels with 4 branching levels, with length and width ratios equal to 0.7071, a total channel length of 18 mm, a terminal channel width of 100 μm, a channel depth of 150 μm, and 16 0^{th} level channels off the inlet plenum. The experimental results are also compared to model results using a wide range of void fraction and friction models with the identical channel geometry and flow conditions as the experimental results in order to select the correlations which best match the experimental results.

The pressure drop has been plotted as a function of the inlet subcooling and mass flow rate in Fig. 45. The trend shows that at a constant flow rate, pressure drop decreases with increasing subcooling. This is caused by decreased exit quality as a result of the decreasing inlet energy content of the liquid with increasing subcooling. The larger the exit quality the larger the exit void fraction, and the larger the acceleration component of the pressure drop will be. The other trend observed is that at any given level of

subcooling, the pressure drop increases with increasing mass flow rate. This is expected given the dependence of both the acceleration and friction components of the pressure drop on the mass flow rate.

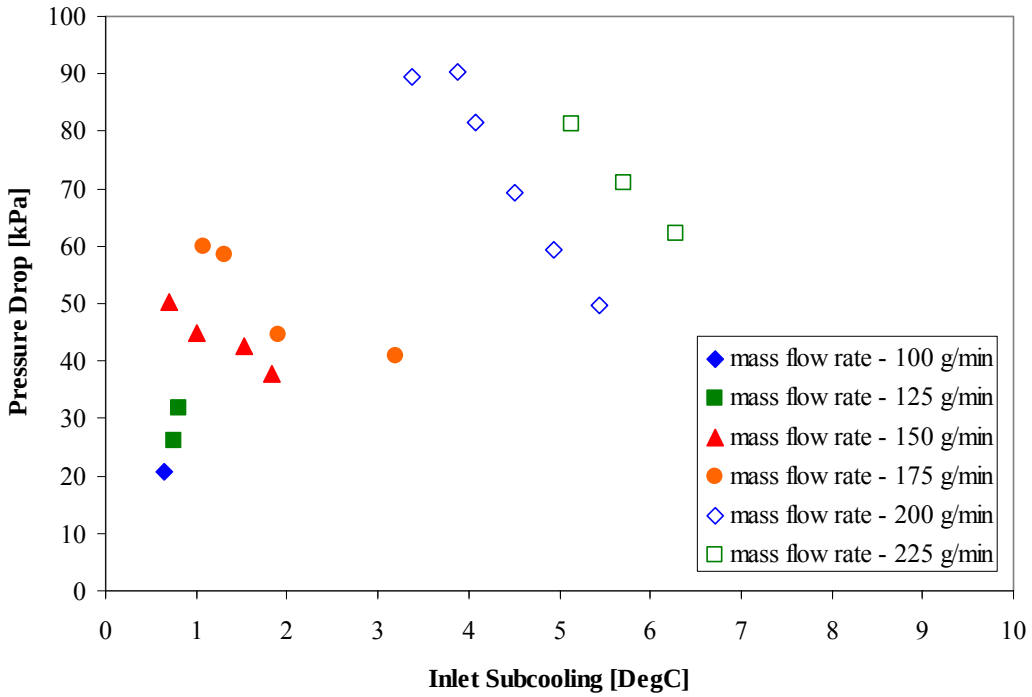


Figure 45. Experimentally measured pressure drop as a function of inlet sub-cooling and mass flow rate for adiabatic flow boiling through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

Figure 46 shows the same pressure drop data as Fig. 45, however, it is plotted as a function of exit quality and mass flow rate. This plot shows that the pressure drop increases linearly with exit quality. This is to be expected based on the linear dependence on the void fraction in the acceleration component of the pressure drop. Closer inspection of the data shows that for each mass flow rate the trend of the pressure drop is a linear increase with exit quality. However if a line is fit through the data for each mass flow rate, and the intercepts of these lines are plotted as a function of mass flow rate, the

curve is parabolic, as shown in the data analysis section.. Given the second order dependence of both the acceleration and friction components of the pressure drop on the square of the mass flow rate this result is also expected. What is interesting is the relative strength of each of these effects on the pressure drop data. The linear dependence on exit quality is much more obvious in the data indicating that the acceleration component of the pressure drop is the dominant of the two factors. Also because this is linear with exit quality the dominant term in the acceleration pressure drop is the velocity change due to the expansion of the fluid from the inlet to the exit, mass flow dependency is small.

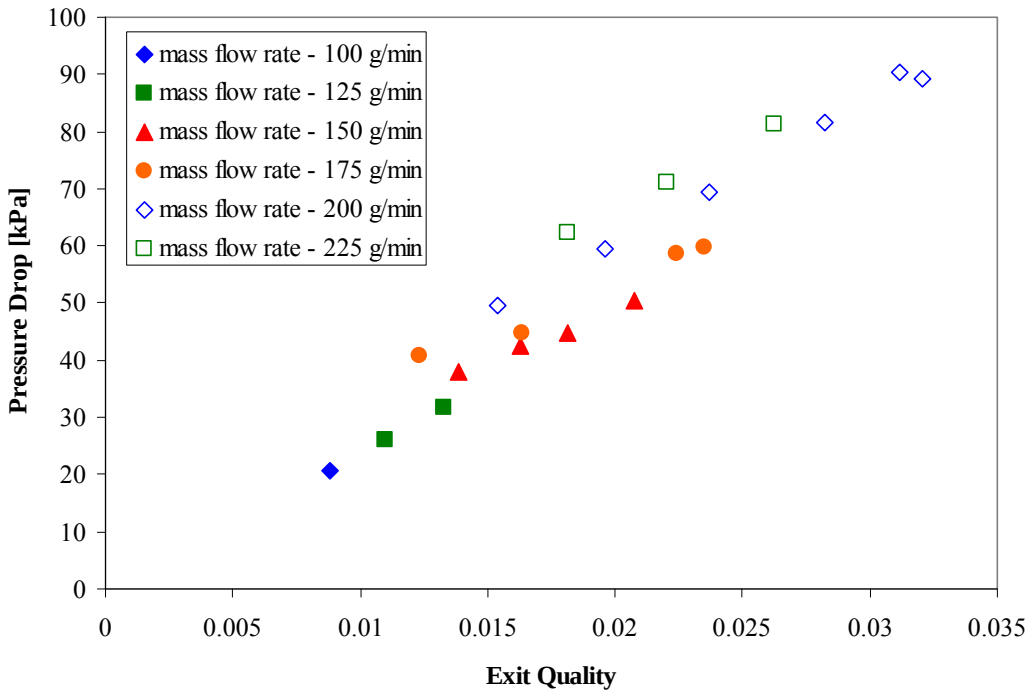


Figure 46. Experimentally measured Pressure drop as a function of exit quality and mass flow rate for adiabatic flow boiling through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

The model described earlier was run for the mass flow rates and inlet subcoolings of the test conditions plotted in Figs. 45 and 46. Figures 47 and 48 show comparisons of

the pressure drop results from the model to the experimentally measured pressure drops. Figure 47 shows the comparison of the experimental results to model results based on the homogenous flow conditions, including the homogenous void fraction shown in Table I and the two expressions for the two-phase viscosity shown in Table II. In general, the model agreed reasonably well with the experimental results, the average deviation for either homogenous model was about 12%, and the maximum deviation was about 20%. The maximum and average deviations for all the homogenous models are shown in Table XIV. In general the models tended to over predict the actual pressure drop. The homogenous based models did tend to have convergence and stability problems for some of the experimental conditions. This resulted in some experimental conditions not having model results to compare to.

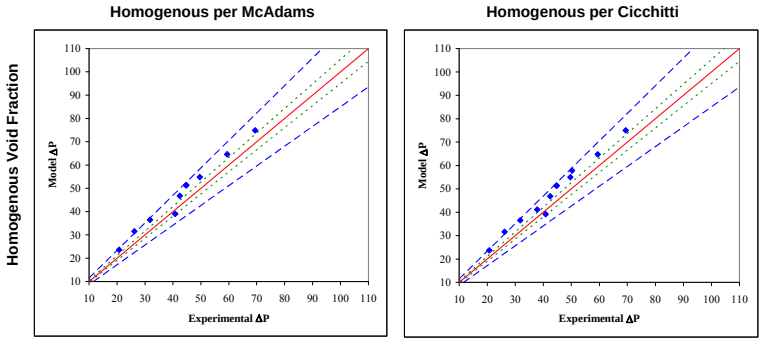


Figure 47. Comparison of experimentally measured pressure drop and model predicted pressure drops using homogenous flow models through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

Table XV. Average and maximum absolute deviations of model predictions from experimental measurements expressed as a percent for homogenous flow models

	Two-phase Multiplier Model	
	McAdams	Cicchitti
Homogenous Void Fraction	Avg. Dev. 12.0% Max Dev. 20.3%	Avg. Dev. 12.1% Max Dev. 20.8%

Figure 48 shows the comparison of the experimental results to model results based on the separated flow models, including the void fraction models shown in Table I and the separated two phase multiplier models shown in Table III. Again the model results show good agreement with the experimental results, with no combination of void fraction and multiplier model having a maximum deviation greater than about 25%, and all the combinations had average deviations below 15%. See Table XV for a complete accounting of the maximum and average deviations combinations. Again there were some convergence and stability issues, for some model combinations. The problems were isolated to the cases in which the void fraction model by Armaand [26] was used.

Some of the combinations of void fraction and multiplier models had excellent agreement with the experimental results with average deviations well less than 5%. The two best combinations used the two phase multiplier correlation developed by Qu and Mudawar [57], and void fraction correlations provided by Chung et al [50-55] and Zivi [27] respectively. Both of these combinations had average deviations of approximately 3.5%.

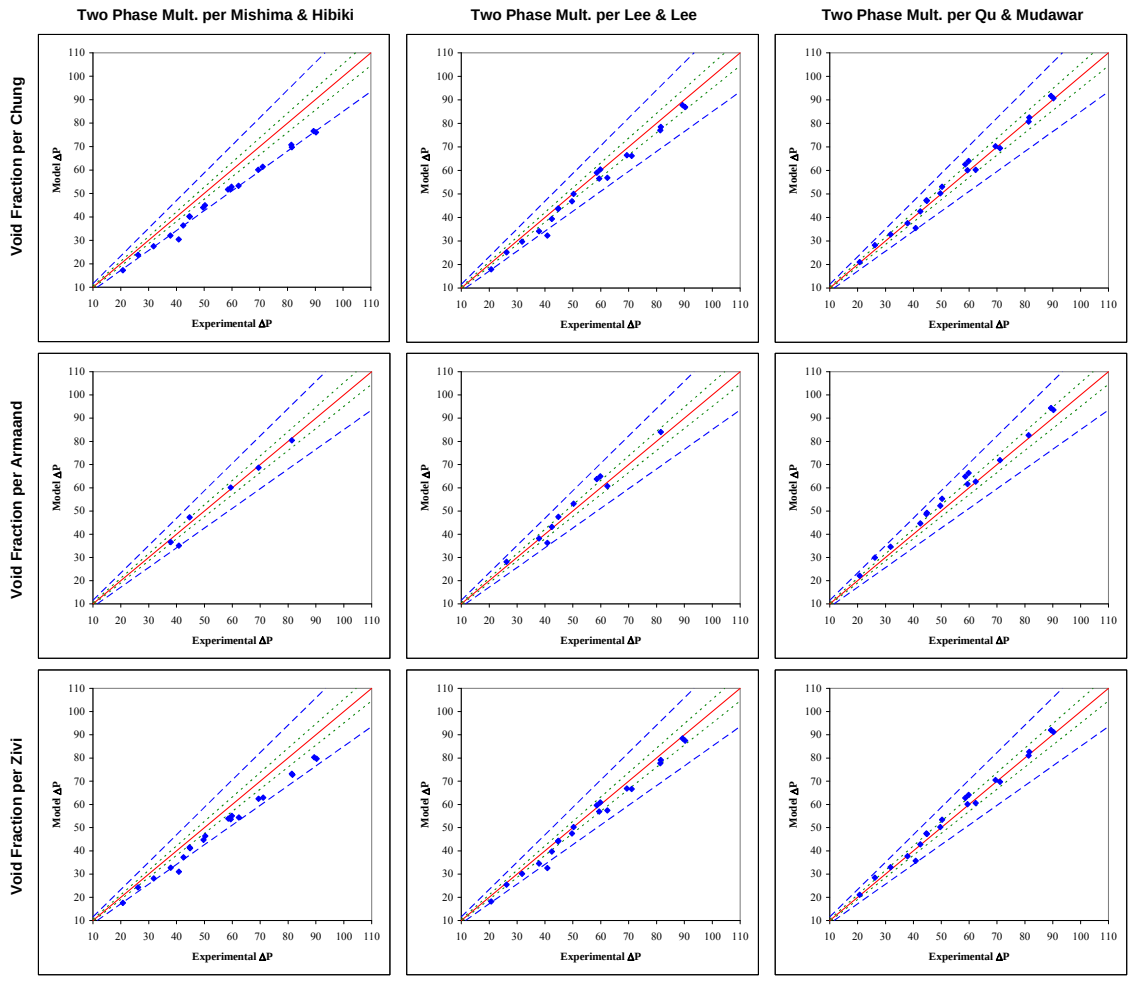


Figure 48. Comparison of experimentally measured pressure drop and model predicted pressure drops using separated flow models through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

Table XVI. Average and maximum absolute deviations of model predictions from experimental measurements expressed as a percent for homogenous flow models

		Two-phase Multiplier Model		
		Mishima & Hibiki	Lee & Lee	Qu & Mudawar
Separated Void Fraction	Chung et. al.	Avg. Dev. 13.6% Max Dev. 25.5%	Avg. Dev. 5.7% Max Dev. 20.9%	Avg. Dev. 3.5% Max Dev. 12.9%
	Armaand	Avg. Dev. 4.5% Max Dev. 14.1%	Avg. Dev. 4.5% Max Dev. 14.1%	Avg. Dev. 6.7% Max Dev. 14.4%
	Zivi	Avg. Dev. 11.0% Max Dev. 23.9%	Avg. Dev. 5.0% Max Dev. 20.2%	Avg. Dev. 3.7% Max Dev. 12.6%

Figure 49 shows a comparison between the experimental and model predicted void fractions for models based on homogenous flow conditions. Data for the 0^{th} through the 3^{rd} branching levels are shown in parts a) through d) respectively. The plots show that for the homogenous models the initial onset of vapor generation does not occur for most cases until the 2^{nd} or 3^{rd} branching levels. In the experimental data, the onset of vapor generation occurs in the 0^{th} level in all cases. The agreement between the model predicted and experimentally measured void fractions is not good for either two-phase viscosity model. Some of this is related to scatter in the experimental data, but some is related to the poor ability of these models to accurately predict the behavior of the flow along the channels as indicated by the discrepancy of the point of initial vapor generation between the model and experimental results.

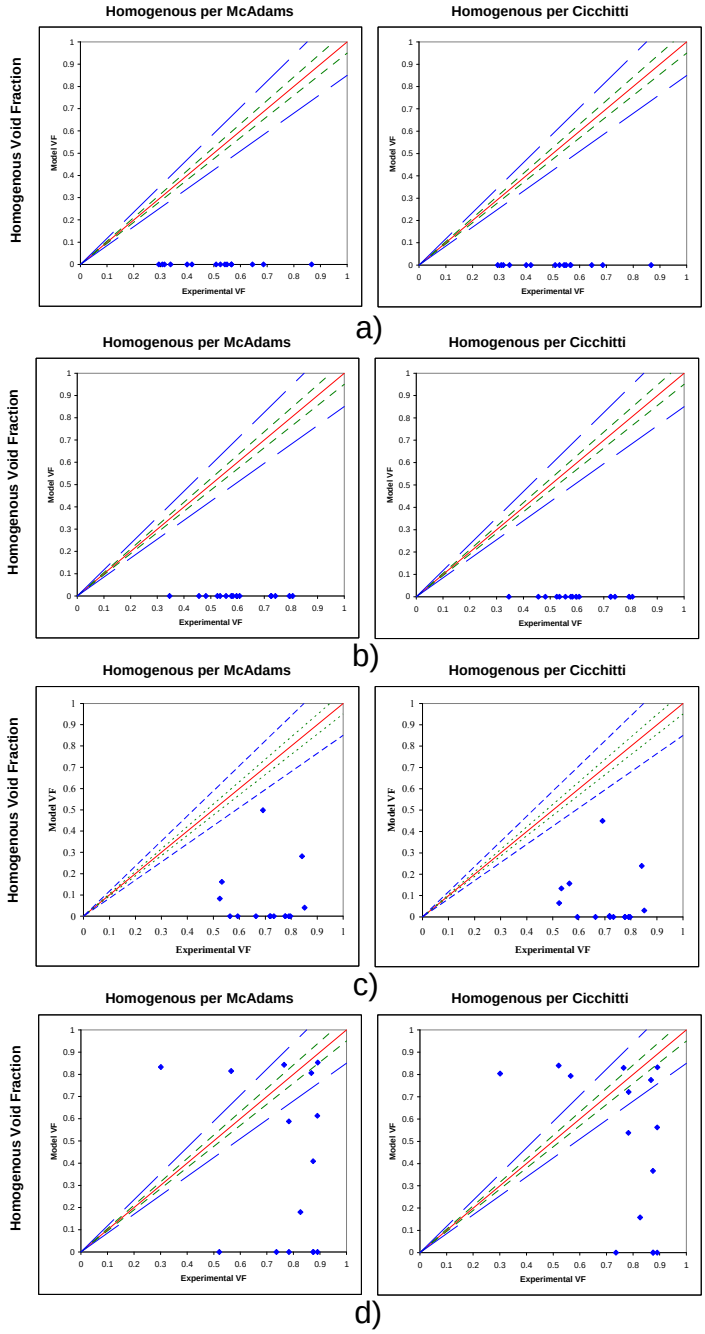


Figure 49. Comparison of experimentally determined void fraction to model predicted void fraction using homogenous flow models through fractal like branching channels. Results for a) the 0^{th} branching level, b) the 1^{st} branching level c)the 2^{nd} branching level and d) the 3^{rd} branching level. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

Figures 50 - 53 show a similar comparisons between the experimental void fraction and the model predicted void fraction for the 0^{th}, 1^{st}, 2^{nd}, and 3^{rd} branching levels respectively. This time however separated flow models were used for the void fraction and the two phase multiplier. The agreement between the model predicted void fraction and the experimentally measured void fraction still is not very good. The 0^{th} level void fraction is severely under predicted with several of the models predicting single phase flow, for many if not all the flow conditions. The two model combinations which predict two phase flow for all the flow conditions for the 0^{th} level both use the Mishima & Hibiki [32] friction model with the Chung et. al.[50-55] and Zivi [27] void fraction models. The void fractions predicted however are still much lower than those measured from the experimental data. All of the models do better in predicting two-phase flow for most the flow conditions in the 1^{st} branching level, however most still under predict the void fraction levels as compared to the experimental values. Two models still stand out from the rest, these again are the combination of the Mishima & Hibiki [32] friction model with the Chung et. al [50-55] and Zivi [27]model.

The agreement between the predictive models and experimental data is better for the 2^{nd} branching level, however this time the driving factor seems to be the scatter in the experimental data, rather than a dramatic systematic difference between the model and the experimental results. There are still some systematic shifts, such as for the combination of the Chung et al [50-55] void fraction model and the two phase multiplier model by Qu & Mudawar [57]. This plot shows that the experimental void fractions are systematically higher than the model predictions. The systematic shifts for the separated

flow models are not as large as those for the homogeneous flow models. Again, combinations of the void fraction model by Chung et al. [50-55] and Zivi [27] and the friction model by Mishima & Hibiki [32] show very little systematic shift, indicating that they are likely doing a good job of modeling the local quality and void fraction. The data pattern tends to be more cloud like rather than linear like the pressure drop data indicating more scatter in the void fraction measurements.

The 3rd branching level, Fig. 53, shows that the model in general is still under predicting the measured void fractions. None of the model combinations stand out as particularly good, however, it is clear that the void fraction model by Armaand [26] does a particularly poor job matching the experimental data.

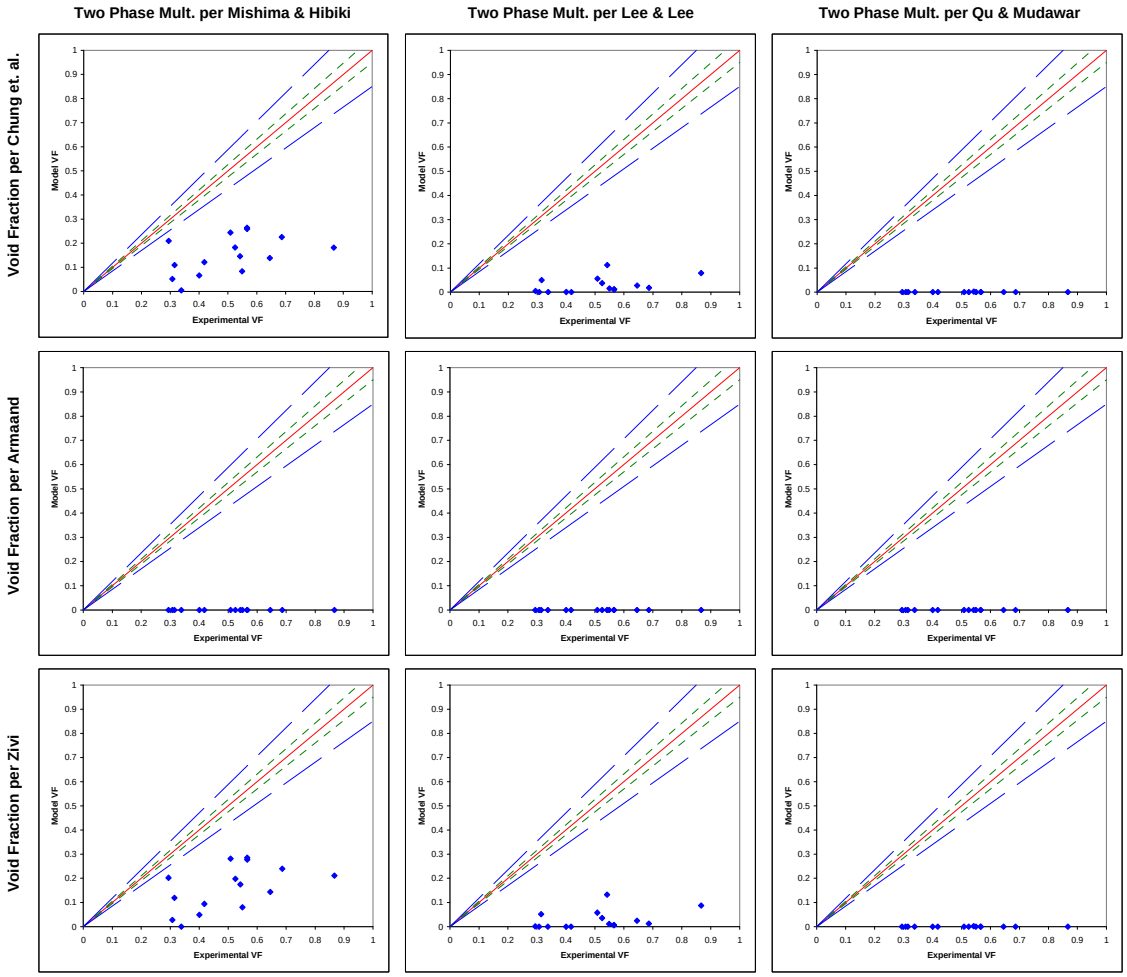


Figure 50. Comparison of experimentally measured void fraction to model predicted void fraction for the 0th branching level using separated flow models for through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

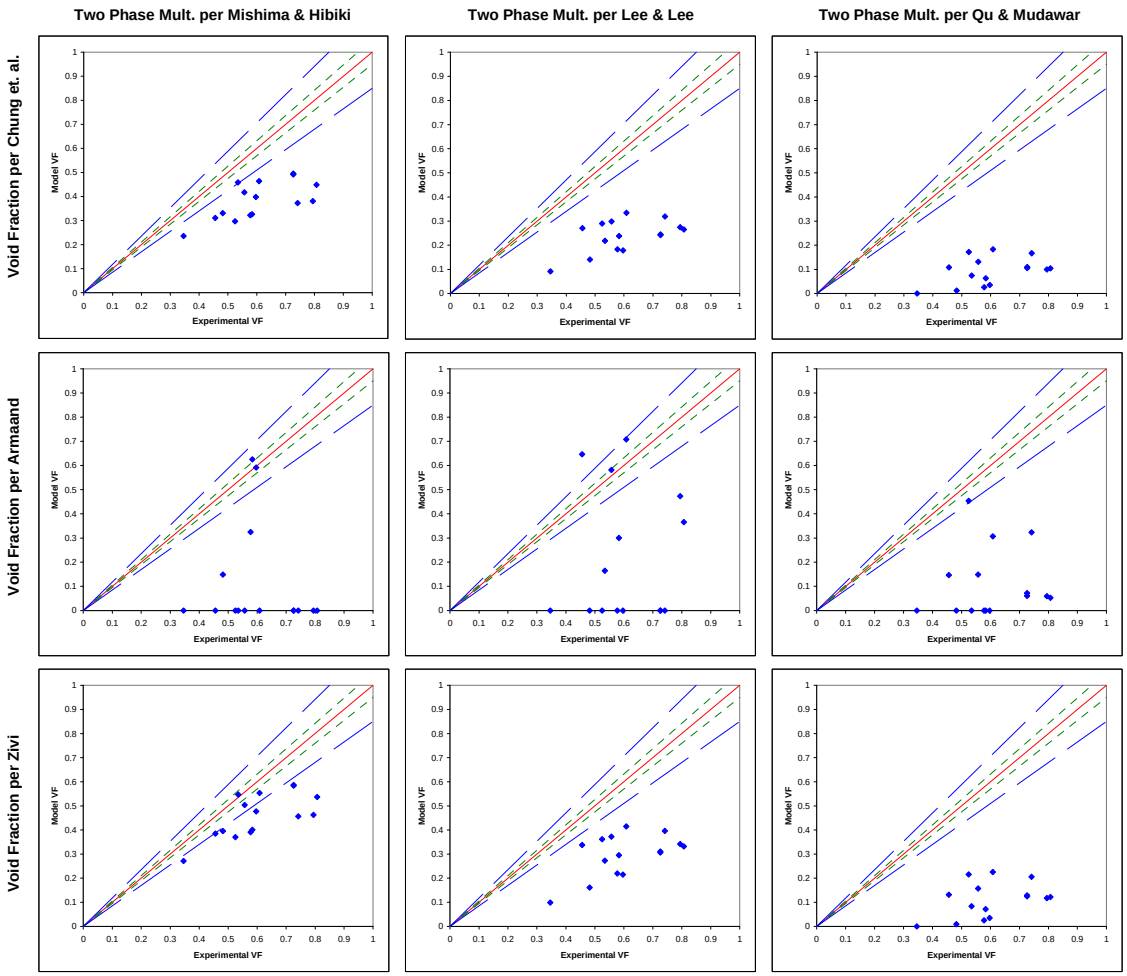


Figure 51. Comparison of experimentally measured void fraction to model predicted void fraction for the 1st branching level using separated flow models for through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

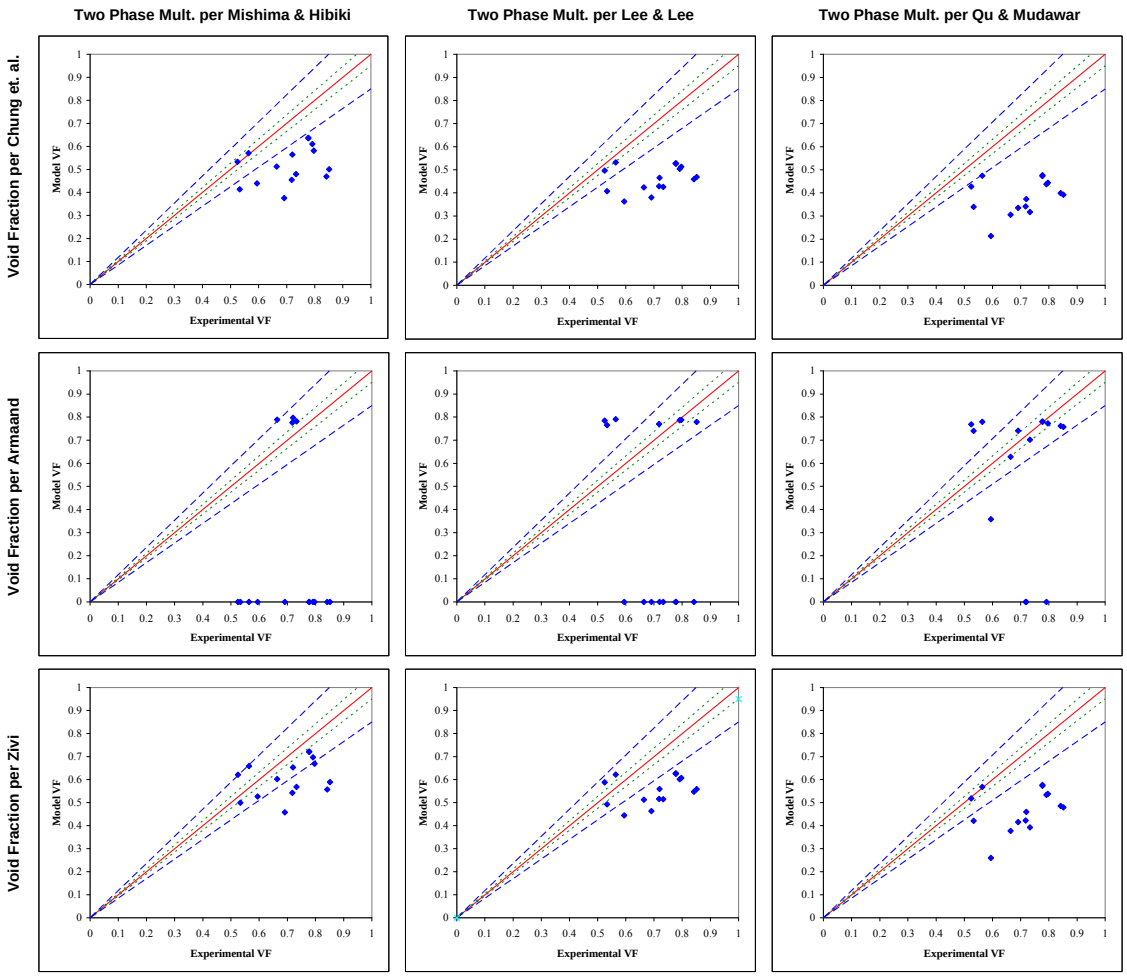


Figure 52. Comparison of experimentally measured void fraction to model predicted void fraction for the 2nd branching level using separated flow models for through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

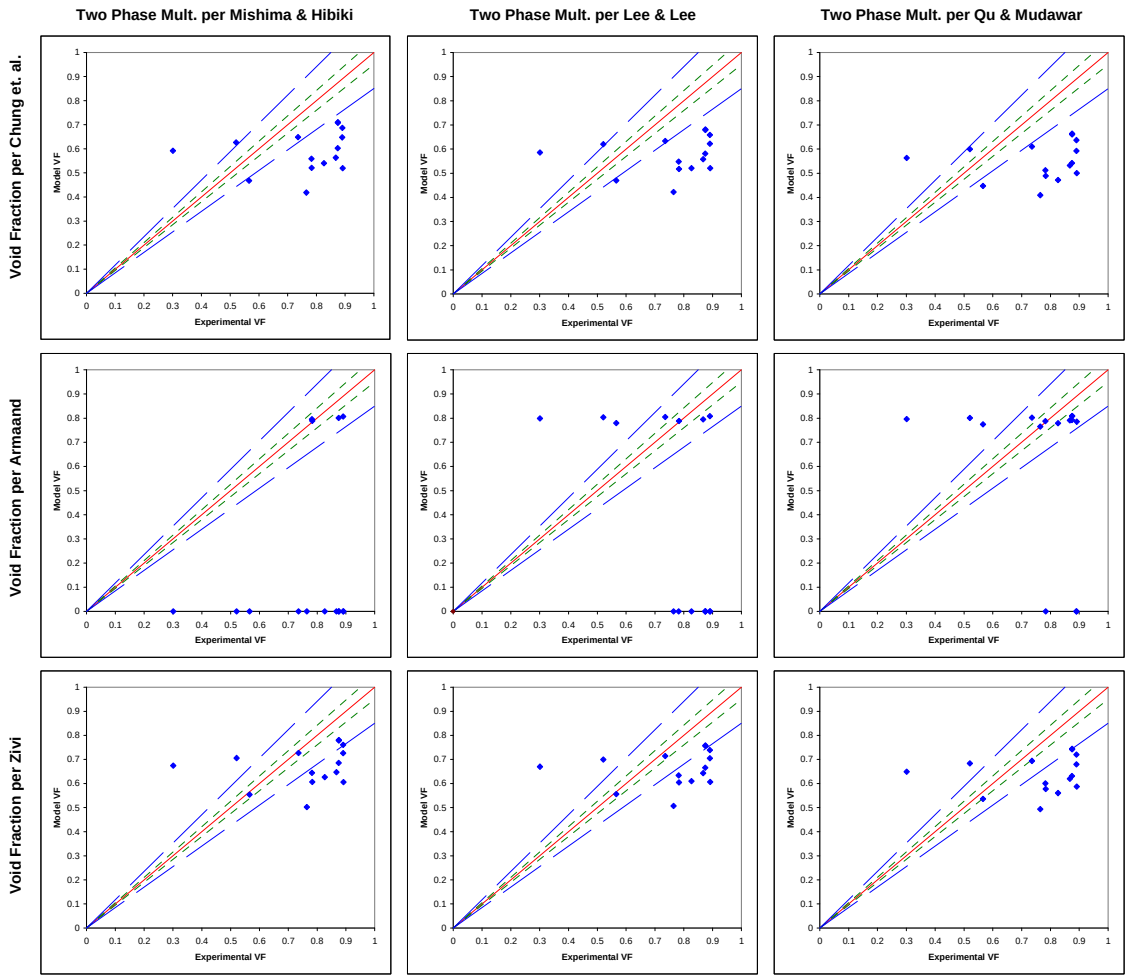


Figure 53. Comparison of experimentally measured void fraction to model predicted void fraction for the 3rd branching level using separated flow models for through fractal like branching channels. Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm.

Void fraction data from both the experiments and the model are shown in Fig. 54 as a function of exit quality for each of the branching levels for which experimental void fraction data was collected. For the model predicted data, the void fraction model of Zivi [27] and the two-phase multiplier model of Mishima and Hibiki [32], on the other hand, a trend develops quite noticeably by the 1st branching level. By the 3rd branching level there is a very strong trend showing a monotonic increase in void fraction as a function of exit quality. The model data for the other branching levels show the same trend of increasing void fraction with exit quality, they however seem to have increasing levels of noise as the branching levels decreases towards the inlet.

The decrease in scatter of the model data as the branching level increases is explained by the fact that the void fraction is plotted as a function of exit quality. The local void fraction will correlate well with the local quality due to its functional dependence on the local quality. The local quality in the higher branching levels also will correlate better with the exit quality since they are physically closer to each other.

Noticing a trend in the experimental data is more difficult due to the increased scatter of the data. However, the experimental data for the 1st branching level through the 3rd branching level do seem to follow the same general trend although with a higher mean void fraction and significantly more scatter. The experimental data for the 0th branching level seems to be at a significantly higher average void fraction than the model predicted data. Neither of which have a strong trends.

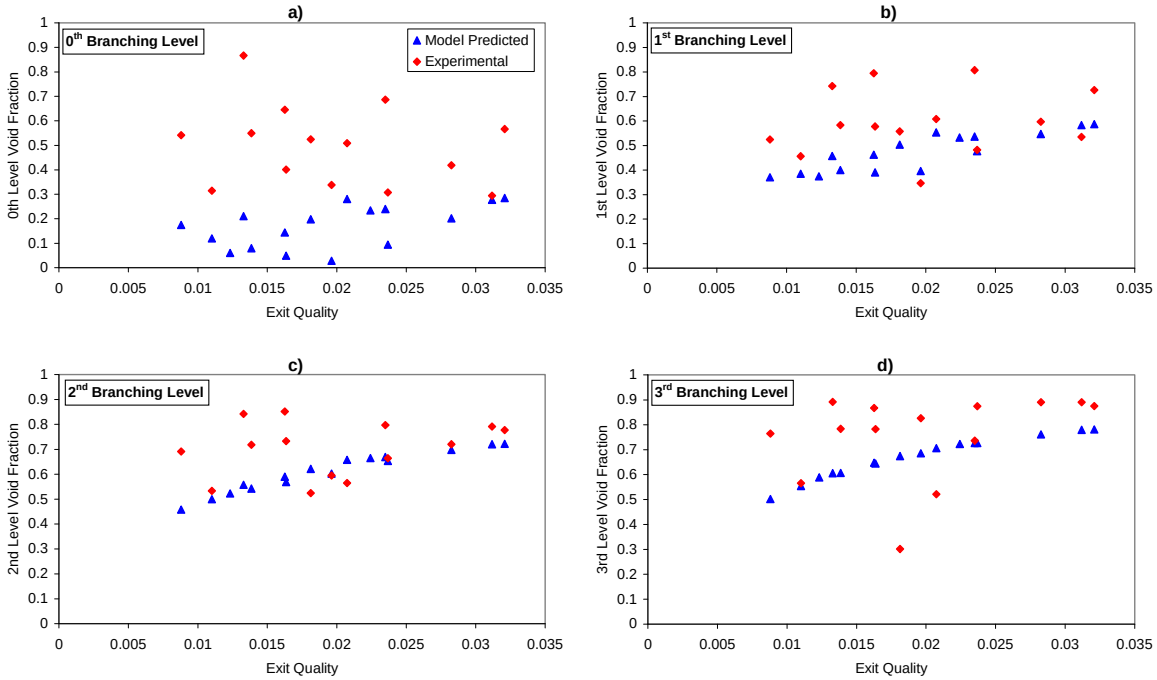


Figure 54. Experimental and Model predicted void fraction for adiabatic flow boiling through fractal like branching channels as a function of exit quality. Data for a) 0^{th} branching level, b) 1^{st} branching level, c) 2^{nd} branching level, and d) 3^{rd} branching level Channel geometry: 100 μm terminal channel width, 4 branching levels, width and length ratios of 0.7071, a total length of 180 mm, and a channel height of 150 mm

.

Conclusions

Diabatic flow results:

A 1-D model of the flow boiling in microchannels has been developed. The model includes the effects of variable fluid properties and developing flow. The model has been used to simulate the single and two-phase heat transfer in fractal-like branching channel and parallel channel heat sinks. A comparison of the performance of single and two-phase flow in microchannel fractal-like branching channel heat sinks has been performed. The comparison of two-phase flow through fractal-like branching channel heat sinks has also been made with parallel channel heat sinks. From these comparison, it has been shown that:

- The two-phase flow pressure drops are equivalent to the single phase flow pressure drops, however the single phase mass flow rates are 4-5 times greater than the two-phase flow rates.
- The two-phase pumping powers are equivalent to the single phase pumping powers for single phase mass flow rates 2-3 times greater than the two-phase mass flow rates.
- The performance parameter for single phase flows were larger that the comparable two-phase flow conditions.
- The temperature change across the heat sink for single phase flows are larger than two-phase flows, however, at large single phase flow rates, the temperature changes between the two flow types are not significantly different.

All this indicates that under certain conditions, single phase flows do hold an advantage over two-phase flows in performance. For example if the goal is to minimize

pressure drop for a fixed mass flow, single phase flow would be better as long as the temperature rise across the heat sink was not important. However if the temperature rise must be minimized, then two-phase flow must be employed to minimize the pressure drop and pumping power.

Two-phase flow in parallel and branching channel heat sinks has also been compared, with the following conclusions:

- The branching channel pressure drop is significantly less than the parallel channel pressure drop, particularly at high wall heat fluxes.
- The performance parameter for the branching channel flows are as much as an order of magnitude larger than the parallel channel flows
- The temperature change for the parallel channel flows is larger than the branching channel flows, as would be expected from the pressure drop behavior.

These indicate that from a flow network perspective, the fractal-like branching channels have an advantage over parallel channels as measured by pressure drop, pumping power, performance parameter and temperature change across the heat sink.

Adiabatic Flow Results

The model was also employed to compare the performance of adiabatic flow in fractal-like branching channels to adiabatic flow in parallel channels. Results show that the fractal-like flow networks are very sensitive to the length ratio used. For a large length ratio design, the pressure drops and exit qualities for the fractal-like branching channels are significantly larger than the comparable parallel channel case with the same total channel surface area. However, using a small length ratio, the pressure drop and exit

quality can be reduced compared to a parallel channel flow network with the same channel surface area. Further work is recommended to obtain an optimal geometry for a desired pressure drop and/or exit quality.

Experimental Conclusions

Experiments of adiabatic flow boiling have been performed on a fractal-like branching channel network. Pressure drop and void fraction measurements were made for a range of mass flow rates and inlet subcooling levels. These results were compared to model predictions using several different models for both the void fraction and two-phase multiplier.

The pressure drop results show that the primary factor determining the pressure drop at these flow rates is the exit quality. The flow rate also has the expected parabolic behavior with mass flow rate. This effect however is smaller than the effect of exit quality. The models tended to do a good job predicting the measured pressure drops, with the separated flow models doing a better job than the homogenous flow models. The two best combinations of models both used the two phase multiplier correlation of Qu & Mudawar [57], and the void fraction correlations used were those of Chung et. al. [50-55] and Zivi [27].

The experimental void fraction measurements did not agree as well with the model predictions. The model predictions of void fraction show a definite trend with exit quality for the higher branching levels that the experimental void fraction data did not seem to have. The primary cause of this was the large amount of scatter in the experimental data. The scatter also shows up in the direct comparison of the model void

fractions to the experimental void fractions for the various combinations of void fraction and two-phase multiplier correlations. Most models show some sort of systematic shift of the model data relative to the experimental data. The combination of the Mishima & Hibiki [32] two-phase multiplier correlation and the Zivi [27] void fraction correlation show the smallest systematic shift.

Given what was found in this work there are several suggestions for future work to either extend or improve upon the work accomplished here. First it would be interesting to compare the model results against diabatic flow boiling result from the same fractal-like branching geometry. It would also be interesting to compare the results against other fractal-like branching channel geometries and parallel channel geometries for both adiabatic and diabatic conditions to examine the models limitations. Along with these new experimental results, improved void fraction measurements would also be desirable to validate the model behavior along the channels.

Improvements in the void fraction measurement can be made in a couple of ways. First priority would be to continue to improve on preventing the fluorescent liquid from interfering with the view of the fractal channels. This would include continued use of the deflector cone, bonding of a thicker glass cover to the top of the fractal disk in such a way that would prevent liquid from seeping between them. For example a UV cure epoxy could be used. Second priority would be to prevent the liquid which has exited the area of the fractal but is still in the field of view from interfering with the imaging. This could be accomplished by using a mask of some sort to only allow the camera to see the

fractal device. Or the conical deflector could be made from some sort of opaque material, again masking the view of the areas outside the fractal device.

The last set of improvements concern the 1-D model. There is at least one new friction model available in the literature. These could be incorporated and evaluated against the experimental data to determine if they performs better than Qu & Mudawar [57] or Mishima & Hibiki [32]. The other model improvement would be to incorporate an adaptive step size algorithm based on how quickly the pressure is changing along the channel. In regions where the pressure gradient along the channel is high, the step size along the channel would be reduced. This should improve stability and convergence. At higher flow rates this model behaves as though it were mathematically stiff, which it likely is due to the acceleration caused by the phase change. Reducing the step size in these areas would help improve the stability at least by improving the resolution of where the initial phase change occurs. Currently the model can only resolve this location to within the grid spacing. If this is course enough that a shift of one grid location causes the exit pressure to oscillate outside the convergence criteria, the model will never converge. The model does currently exhibit behavior that is consistent with this hypothesis.

References

[1] Peng, X.F. and Wang, B.X., "Forced convection and flow boiling heat transfer for liquid flowing through micro-channels". *International Journal of Heat and Mass Transfer*, 1993. **36**(14): p. 3421-3427.

[2] Bowers, M.B. and Mudawar, I., "Two-phase electronic cooling using mini-channel and micro-channel heat sinks: Part 2 - flow rate and pressure drop constraints". *Journal of Electronic Packaging, Transactions of the ASME*, 1994. **116**(4): p. 298-305.

[3] Bowers, M.B. and Mudawar, I., "High flux boiling in low flow rate, low pressure drop mini-channels and micro-channel heat sinks". *International Journal of Heat and Mass Transfer*, 1994. **37**(2): p. 321-332.

[4] Bowers, M.B. and Mudawar, I., "Two phase electronic cooling using mini-channel and micro-channel heat sinks: Part I Design criteria and heat diffusion constraints". *Journal of Electronic Packaging, Transactions of the ASME*, 1994. **116**: p. 290-297.

[5] Cuta, J., Benett, W., McDonald, C.E., and Ravigururajan, T.S. "Fabrication and performance testing of micro-heat exchangers". in *SPIE international conference*. 1995. Austin, TX.

[6] Ravigururajan, T.S., "Impact of channel geometry on two-phase flow heat transfer characteristics of refrigerants in microchannel heat exchangers". *Journal of Heat Transfer, Transactions of the ASME*, 1998. **120**(2): p. 485-491.

[7] Ravigururajan, T.S., "Two-phase flow characteristics of refrigerant flows in a microchannel heat exchanger". *Journal of Enhanced Heat Transfer*, 1999. **6**(6): p. 419-427.

[8] Ravigururajan, T.S., Cuta, J., McDonald, C.E., and Drost, M.K. "Effects of heat flux on two-phase flow characteristics of refrigerant flows in a micro-channel heat exchanger". in *Proceedings of the 1996 31st ASME National Heat Transfer Conference*. 1996. Houston, Tx.

[9] Jiang, L., Koo, J.M., Zeng, S., Mikkelsen, J.C., Zhang, L., Zhou, P., Santiago, J.G., Kenny, T.W., and Goodson, K.E. "Two-phase microchannel heat sinks for an electrokinetic VLSI chip cooling system". in *17th Annual IEEE Semiconductor Thermal Measurement Symposium*. 2001. San Jose, CA.

[10] Koo, J.M., Jiang, L., Zhang, L., Zhou, P., Banerjee, S.S., Kenny, T.W., Santiago, J.G., and Goodson, K.E. "Modeling of two-phase microchannel heat sinks for VLSI chips". in *MEMS-2001: presented at the 2001 SME International Mechanical Engineering Congress and Exposition*. 2001. New York, NY.

[11] Zhang, L., Koo, J.M., Jiang, L., Asheghi, M., Goodson, K.E., Santiago, J.G., and Kenny, T.W., "Measurements and modeling of two-phase flow in microchannels with nearly constant heat flux boundary conditions". *Journal of Micoroelectromechanical Systems*, 2002. **11**(1): p. 12-19.

[12] Pence, D.V., "Reduced pumping power and wall temperature in microchannel heat sinks with fractal-like branching channel networks". *Microscale Thermophysical Engineering*, 2002. **6**: p. 319-330.

[13] Collier, J.G., Convective Boiling and Condensation. 2nd ed. 1981, New York: McGraw-Hill.

[14] Stanley, R.S., Barron, R.F., and Ameel, T.A. "Two-phase flow in microchannels". in DSC-Vol. 62/HTD-Vol. 354, *Microelectromechanical Systems (MEMS) ASME* 1997. 1997.

[15] Dukler, A.E., Wicks, M., III, and Cleveland, R.G., "Frictional Pressure Drop in Two-Phase Flow: A. A Comparison of Existing Correlations for Pressure Loss and Holdup". *AIChE Journal*, 1964. **10**(1): p. 38-43.

[16] McAdams, W.H., Woods, W.K., and Heroman, L.C.J., "Vaporization Inside Horizontal Tubes -II Benzene-Oil Mixtures". *Transactions of the ASME*, 1942. 64: p. 193-200.

[17] Cicchitti, A., "Two Phase Cooling Experiments - Pressure Drop, Heat Transfer, and Burnout Measurements". *Energia Nucleare*, 1960. **7**(6): p. 407-425.

[18] Lin, S., Kwok, C.C.K., Li, R.Y., Chen, Z.H., and Chen, Z.Y., "Local frictional pressure drop during vaporization of R-12 through capillary tubes". *International Journal of Multiphase Flow*, 1991. **17**(1): p. 95-102.

[19] Wallis, G.B., *One-dimensional Two-phase Flow*. 1969, New York: McGraw-Hill.

[20] Chisholm, D., *Two-phase flow in pipelines and heat exchangers*. 1983: George Godwin.

[21] Lockhart, R.W. and Martinelli, R.C., "Proposed correlation of data for isothermal two-phase, two-component flow in pipes". *Chemical Engineering Progress*, 1949. **45**: p. 39-48.

[22] Martinelli, R.C. and Nelson, D.B., "Prediction of pressure drop during forced-circulation boiling of water". *Transactions of the ASME*, 1948. **70**: p. 695-702.

[23] Chisholm, D., "Friction pressure gradient during the flow of boiling water". *Engineering and Boiler House Review*, 1963. **78**(8): p. 287-289.

[24] Chisholm, D., "A theoretical basis for the Lockhart-Martinelli correlation for two-phase flow". *International Journal of Heat and Mass Transfer*, 1967. **10**: p. 1767-1778.

[25] Chisholm, D. and Laird, A.D.K., "Two-Phase flow in rough tubes". *Transactions of the ASME*, 1958. p. 276-286.

[26] Armand, A.A., "The resistance during the movement of a two-phase system in horizontal pipes". *Izvestiya Vsesoyuznogo Teplotekhnicheskogo Instituta (A.E.R.E. Transactions 828)*, 1946. **1**: p. 16-23.

[27] Zivi, S.M., "Estimation of steady-state steam void-fraction by means of the principle of minimum entropy production". *Journal of Heat Transfer*, 1964. **86**: p. 247-252.

[28] Zhao, T.S. and Bi, Q.C., "Pressure drop characteristics of gas-liquid two-phase flow in vertical miniature triangular channels". *International Journal of Heat and Mass Transfer*, 2001. **44**: p. 2523-2534.

[29] Triplett, K.A., Ghiaasiaan, S.M., Abdel-Khalik, S.I., LeMouel, A., and McCord, B.N., "Gas-liquid two-phase flow in microchannels Part II: void fraction and pressure drop". *International Journal of Multiphase Flow*, 1999. **25**: p. 395-410.

[30] Bao, Z.Y., Bosnich, M.G., and Haynes, B.S., "Estimation of void fraction and presssure drop for two-phase flow in fine passages". *Transactions of the Institue of Chemical Engineers*, 1994. **72**, Part A: p. 625-632.

[31] Lee, H.J. and S.Y., L., "Pressure drop correlations for two-phase flow within horizontal rectangular channels with small heights". *International Journal of Multiphase Flow*, 2001. **27**: p. 783-796.

[32] Mishima, K. and Hibiki, T., "Some characteristics of air-water two-phase flow in small diameter vertical tubes". *International Journal of Multiphase Flow*, 1996. **22**: p. 703-712.

[33] Shuai, J., Kulenovic, R., and Groll, M. "Heat transfer and pressure drop for flow boiling of water in narrow vertical rectangular channels". in *First International Conference on Microchannels and Minichannels*. 2003. Rochester, NY.

[34] Kaminaga, F., Sumith, B., and Matsumura, K. "Pressure drop in a capillary tuber in boiling two-phase flow". in *First International Conference on Microchannels and Minichannels*. 2003. Rochester, NY.

[35] Whitesel, H.A., "Capillary two-phase flow, Part I". *Refrigerating Engineering*, 1957. **65**(4): p. 42-44, 98-99.

[36] Whitesel, H.A., "Capillary two-phase flow, Part II". *Refrigerating Engineering*, 1957. **65**(9): p. 35-40.

[37] Mikol, E.P., "Adiabatic single and two-phase flow in small bore tubes". *ASHRAE Journal*, 1963. **5**: p. 75-86.

[38] Mikol, E.P. and Dudley, J.C., "A visual and photographic study of the inception of vaporization in adiabatic flow". *Transactions of the ASME*, 1964. **7**: p. 257-264.

[39] Meyer, J.J. and Dunn, W.E., "New insights into the behavior of the metastable region of an operating capillary tube". *International Journal of Heating, Ventilating, Air-Conditioning and Refrieration Research*, 1998. **4**(1): p. 105-115.

[40] Bittle, R.R., Carter, J.A., and Oliver, J.V., "Extended insight into the metastable liquid region behavior in an adiabatic capillary tube". *ASHRAE Transactions*, 2001. **107**(2): p. 281-297.

[41] Chen, D.K. and Lin, S., "Underpressure of vaporization of refrigerant R-134a through a diabatic capillary tube". *International Journal of Refrigeration*, 2001. **24**(3): p. 261-271.

[42] Bittle, R.R. and Pate, M.B., "A theoretical model for predicting capillary tube performance with alternative refrigerants". *ASHRAE Transactions*, 1996. **102**(2): p. 52-64.

[43] Lackme, C., "Incompleteness of the flashing of a supersaturated liquid and sonic ejection of the produced phases". *International Journal of Multiphase Flow*, 1979. **5**: p. 131-141.

[44] Hardy, P. and Mali, P., "Validation and development of a model describing subcooled critical flow through long tubes". *Energie Primaire*, 1983. **18**: p. 5-23.

[45] Feburie, V., Goit, M., Granger, S., and Seynhaeve, J.M., "A model for choked flow through cracks with inlet subcooling". *International Journal of Multiphase Flow*, 1993. **19**(4): p. 541-562.

[46] Wong, T.N. and Ooi, K.T., "Refrigerant flow in capillary tube: An assessment of the two-phase viscosity correlations on model prediction". *International Communications in Heat and Mass Transfer*, 1995. **22**(4): p. 595-604.

[47] Melo, C., Neto, C.B., and Ferreria, R.T.S., "Empirical correlations for the modeling of R-134a flow through adiabatic capillary tubes". *ASHRAE Transactions*, 1999. **105**(2): p. 51-59.

[48] Kuehl, S.J. and Goldschmidt, V.W., "Modeling of steady flows of R-22 through capillary tubes". *ASHRAE Transactions*, 1991. **97**(1): p. 139-148.

[49] Serizawa, A., Feng, Z., and Kawara, Z., "Two-phase flow in microchannels". *Experimental Thermal and Fluid Science*, 2002. **26**: p. 703-714.

[50] Chung, P.M.Y. and Kawaji, M., "The effect of channel diameter on adiabatic two-phase flow characteristics in microchannels". *International Journal of Multiphase Flow*, 2004. **30**: p. 735-761.

[51] Chung, P.M.Y., Kawaji, M., Kawahara, A., and Shibata, Y. "Two-phase flow through square and circular microchannels - effect of channel geometry". in *4th ASME-JSME Joint Fluids Engineering Conference*. 2003. Honolulu, HI.

[52] Kawahara, A., Chung, P.M.Y., and Kawaji, M., "Investigation of two-phase flow pattern, void fraction and pressure drop in a microchannel". *International Journal of Multiphase Flow*, 2002. **28**: p. 1411-1435.

[53] Kawaji, M. and Chung, P.M.Y., "Adiabatic gas-liquid flow in microchannels". *Microscale Thermophysical Engineering*, 2004. **8**: p. 239-257.

[54] Kawaji, M. and Chung, P.M.Y. "Unique characteristics of adiabatic gas-liquid flows in microchannels: Diameter and shape effects on flow patterns, void fractions and pressure drop". in *First International Conference on Microchannels and Minichannels*. 2003. Rochester, NY.

[55] Chung, P.M.Y., Kawaji, M., Kawahara, A., and Shibata, Y., "Two-Phase flow through square and circular microchannels - Effects of channels geometry". *Journal of Fluids Engineering*, 2004. **126**: p. 546-552.

[56] Kawahara, A., Sadatomi, M., Okayama, K., Kawaji, M., and Chung, P.M.Y., "Effects of channel diameter and liquid properties on void fraction in adiabatic two-phase flow through microchannels". *Heat Transfer Engineering*, 2005. **26**(3): p. 13-19.

[57] Qu, W. and Mudawar, I., "Measurement and prediction of pressure drop in two-phase micro-channel heat sinks". *International Journal of Heat and Mass Transfer*, 2003. **46**: p. 2737-2753.

[58] Qu, W. and Mudawar, I., "Flow boiling heat transfer in two-phase micro-channel heat sinks - II. Annular two-phase flow model". *International Journal of Heat and Mass Transfer*, 2003. **46**: p. 2773-2784.

[59] Qu, W. and Mudawar, I., "Flow boiling heat transfer in two-phase micro-channel heat sinks - I. Experimental investigation and assessment of correlation methods". *International Journal of Heat and Mass Transfer*, 2003. **46**: p. 2755-2771.

[60] Lee, J. and I. Mudawar (). "Two-phase flow in high-heat-flux micro-channel heat sink for refrigeration cooling applications: Part I - pressure drop characteristics." *International Journal of Heat and Mass Transfer*, 2005 **48**: p. 928-940.

[61] Revellin, R. and Thome, J.R., "Adiabatic two-phase frictional pressure drops in microchannels." *Experimental Thermal and Fluid Science*, 2007, **31**: p. 673-685

[62] Shah, R.K., "A correlation for laminar hydrodynamic entry length solutions for circular and non circular ducts", *J. Fluids Eng*, 1978, **100**, p 177-179

[63] Kandlikar, S. G,. "Two-phase flow patterns, pressure drop and heat transfer during boiling in minichannel and microchannel flow passages of compact evaporators." *Engineering Foundation Conference on Compact Heat Exchangers*, Davos, Switzerland, 2001.

[64] Irvine, T. F. and P. E. Liley. *Steam and gas tables with computer equations*. New York, Academic Press, 1984.

[65] Moran, M. J. and H. N. Shapiro, *Fundamentals of Engineering Thermodynamics*. New York, John Wiley & Sons, 2004)

[66] Rogers, C. F. C. and Y. R. Mayhew. *Thermodynamic and Transport Properties of Fluids*. Cambridge, Blackwell, 1995.

[67] Cengel, Y. A., *Heat Transfer; a Practical Approach*, McGraw-Hill, 1998.

[68] White, F. M., *Viscous Fluid Flow*. Boston, MA, McGraw-Hill, 1991.

APPENDICIES

APPENDIX A

Uncertainty Analysis

Measured Variables

Repeated Measurements: $\dot{m}_{IN}$, ΔP_c, T_{IN}

Single Measurements: P_1, D_2, D_3, D_4, L, P_{ATM}

Calculated Variables

Exit Quality: $$w = \frac{h_{IN} - h_{l,exit}}{h_{v,exit} - h_{l,exit}}$$

$$h_l = f_1(T_{SAT}) \; ; \; h_v = f_2(T_{SAT})$$

$$T_{SAT} = g(P) \quad P = \text{Pressure}$$

Channel Inlet Pressure: $$P_{IN} = P_{ATM} + \Delta P$$

Channel Pressure Drop: $$\Delta P = \Delta P_c - \Delta P_{ic}$$

Inlet Pressure Drop Correction: $$\Delta P_{ic} = \Delta P_{1-2} + \Delta P_{2-3} + \Delta P_{3-4}$$

$$\Delta P_{1-2} = 2 f_f \frac{L}{D_1} \rho V_1^2 \quad \Delta P_{2-3} = \frac{1}{2} \rho V_3^2 K_1 \quad \Delta P_{3-4} = \frac{1}{2} \rho V_3^2 K_2$$

$$f_f = \frac{0.079}{Re^{1/4}} \quad \mu = f_3(T_{IN}) \quad \rho = f_4(T_{IN}) \quad Re = \frac{\rho V_1 D_1}{\mu}$$

$$V_1 = \frac{\dot{m}_{IN}}{\rho A_1} \quad V_3 = \frac{\dot{m}_{IN}}{\rho A_3} \quad A_1 = \frac{\pi D_1^2}{4} \quad A_3 = \frac{\pi D_3^2}{4}$$

$$K_1 = 1 - \frac{D_3}{D_2} + \frac{1}{2}\left(1 - \frac{D_3}{D_2}\right) \quad K_2 = \left(\frac{D_3}{D_4}\right)^2 - 1 + \left(1 - \frac{D_3}{D_4}\right)^2$$

Inlet Sub-cooling: $$T_{Sub} = T_{SAT}\Big|_{P=P_{IN}} - T_{IN}$$

Bias Uncertainties

Measured Variables

$\dot{m}_{IN}$ - Based on Instrument Specs

Zero Stability Flow: $\dot{m}_{ZS} = 0.0333\overline{3}$ g/min

if $\dot{m}_{IN} \leq \frac{\dot{m}_{ZS}}{0.001} = 33.33\overline{3}$ g/min

* This condition is never satisfied for testing performed.

if $\dot{m}_{IN} > \frac{\dot{m}_{ZS}}{0.001} = 33.33\overline{3}$ g/min

Then

$$B_{\dot{m}_{IN_A}} = 0.001 \cdot \dot{m}_{IN}$$

$$B_{\dot{m}_{IN_R}} = 0.0005 \cdot \dot{m}_{IN}$$

$$B_{\dot{m}_{IN}} = \sqrt{B_{\dot{m}_{IN_A}}^2 + B_{\dot{m}_{IN_R}}^2}$$

$$= 0.001118 \cdot \dot{m}_{IN}$$

T_{IN} - Based on RTD Calibration

Calibration Data File:
\\Mohr\U7\Cytec\Temp Calibration\RTD Cal.xls

Std error of Fit: SEF = 0.135244 °C
Ref Std Accuracy: STD = 0.3 °C

$$B_{T_{IN}} = \sqrt{SEF^2 + STD^2}$$

$$= 0.3291\,^{\circ}C$$

ΔP_c - Based on Instrument Calibration

Calibration Data File:
\\Mohr\U7\Cytec\Press Calibration\
Press Xdcr Cal - CP 68074-08 - SN 130G023.xls

Std error of fit : SEF = 0.101656 kPa
Ref Std Accuracy : STD = 0.344738 kPa

$$B_{\Delta P_c} = \sqrt{SEF^2 + STD^2}$$

$$= 0.359414 \text{ kPa}$$

D_1 - Based on Instrument Resolution
Calipers used : Smallest Increment = 0.001 in

$$B_{D_1} = 0.0005_{in} = 0.0000127 m$$

D_2 - Based on Instrument Resolution
Calipers used : Smallest Increment = 0.001 in

$$B_{D_2} = 0.0005_{in} = 0.0000127 m$$

D_3 - Based on Instrument Resolution
Calipers used : Smallest Increment = 0.001 in

$$B_{D_3} = 0.0005_{in} = 0.0000127 m$$

D_4 - Based on Instrument Resolution
Calipers used : Smallest Increment = 0.001 in

$$B_{D_4} = 0.0005_{in} = 0.0000127 m$$

L - Based on Instrument Resolution
Calipers used : Smallest Increment = 0.001 in

$$B_L = 0.0005_{in} = 0.0000127 m$$

P_{ATM} - Based on Instrument Resolution
Smallest Increment = 0.1 kPa

$$B_{P_{ATM}} = 0.1\ kPa$$

Calculated Variables

$$\frac{B_{A_1}}{A_1} = \sqrt{\left(2\frac{B_{D_1}}{D_1}\right)^2} = \frac{2B_{D_1}}{D_1}$$

$$B_{A_1} = \frac{2B_{D_1}}{D_1} A_1 = \frac{2B_{D_1}}{D_1} \frac{\pi D_1^2}{4_2} = \frac{\pi B_{D_1} D_1}{2}$$

$$B_{A_3} = A_3\sqrt{\left(2\frac{B_{D_3}}{D_3}\right)^2} = \frac{\pi D_3^2}{2\,4} 2\frac{B_{D_3}}{D_3} = \frac{\pi B_{D_3} D_3}{2}$$

$$B_\rho = \sqrt{\left(\frac{d\rho}{dT_{in}} B_{T,i}\right)^2} = \frac{d\rho}{dT_{in}} B_{T_{in}} \; ; \; \frac{d\rho}{dT_{in}} \cong \frac{\rho|_{T_{in}+B_{T_{in}}} - \rho|_{T_{in}}}{B_{T_{in}}}$$

$$= \rho|_{T_{in}+B_{T_{in}}} - \rho|_{T_{in}}$$

$$B_\mu = \mu|_{T_{in}+B_{T_{in}}} - \mu|_{T_{in}} \quad \text{By Analogy to } B_\rho$$

$$B_{V_1} = V_1\sqrt{\left(\frac{B_{\dot m}}{\dot m}\right)^2 + \left(-\frac{B_\rho}{\rho}\right)^2 + \left(-\frac{B_{A_1}}{A_1}\right)^2}$$

$$B_{V_3} = V_3\sqrt{\left(\frac{B_{\dot m}}{\dot m}\right)^2 + \left(-\frac{B_\rho}{\rho}\right)^2 + \left(\frac{B_{A_3}}{A_3}\right)^2}$$

$$B_{RE} = R_E\sqrt{\left(\frac{B_\rho}{\rho}\right)^2 + \left(\frac{B_{V_1}}{V_1}\right)^2 + \left(\frac{B_{D_1}}{D_1}\right)^2 + \left(-\frac{B_\mu}{\mu}\right)^2}$$

$$B_{f_f} = f_f \sqrt{\left(-\frac{1}{4}\frac{B_{Re}}{Re}\right)^2} = f_f \frac{1}{4}\frac{B_{Re}}{Re}$$

$$B_{\Delta P_{1-2}} = \Delta P_{1-2}\sqrt{\left(\frac{B_{f_f}}{f_f}\right)^2 + \left(\frac{B_L}{L}\right)^2 + \left(-\frac{B_{D_1}}{D_1}\right)^2 + \left(\frac{B_\rho}{\rho}\right)^2 + \left(2\frac{B_{V_1}}{V_1}\right)^2}$$

$$B_{K_1} = \sqrt{\left(\frac{dk_1}{dD_3}B_{D_3}\right)^2 + \left(\frac{dk_1}{dD_2}B_{D_2}\right)^2}$$

$$\frac{dk_1}{dD_3} = -\frac{1}{D_2} - \frac{1}{2}\frac{1}{D_2} = -\frac{3}{2}\frac{1}{D_2}$$

$$\frac{dk_1}{dD_2} = \frac{D_3}{D_2^2} + \frac{1}{2}\frac{D_3}{D_2^2} = \frac{3}{2}\frac{D_3}{D_2^2}$$

$$= \sqrt{\left(-\frac{3}{2}\frac{1}{D_2}B_{D_3}\right)^2 + \left(\frac{3}{2}\frac{D_3}{D_2}B_{D_2}\right)^2}$$

$$B_{\Delta P_{2-3}} = \Delta P_{2-3}\sqrt{\left(\frac{B_\rho}{\rho}\right)^2 + \left(2\frac{B_{V_3}}{V_3}\right)^2 + \left(\frac{B_{K_1}}{K_1}\right)^2}$$

$$B_{K_2} = \sqrt{\left(\frac{dk_2}{dD_3}B_{D_3}\right)^2 + \left(\frac{dk_2}{dD_2}B_{D_4}\right)^2}$$

$$\frac{dk_2}{dD_3} = 4\frac{D_3}{D_4^2} - 2\frac{1}{D_4}$$

$$\frac{dk_2}{dD_4} = -4\frac{D_3^2}{D_4^3} + 2\frac{D_3}{D_4^2}$$

$$= \sqrt{\left[\left(4\frac{D_3}{D_4^2} - 2\frac{1}{D_4}\right)B_{D_3}\right]^2 + \left[\left(-4\frac{D_3^2}{D_4^3} + 2\frac{D_3}{D_4}\right)B_{D_4}\right]^2}$$

$$B_{\Delta P_{1c}} = \sqrt{\left(\frac{d\Delta P_{1c}}{d\Delta P_{1-2}}B_{\Delta P_{1-2}}\right)^2 + \left(\frac{d\Delta P_{1c}}{d\Delta P_{2-3}}B_{\Delta P_{2-3}}\right)^2 + \left(\frac{d\Delta P_{1c}}{d\Delta P_{3-4}}B_{\Delta P_{3-4}}\right)^2}$$

$$\frac{\partial \Delta P_{1C}}{\partial \Delta P_{1-2}} = 1 \quad \frac{\partial \Delta P_{1C}}{\partial \Delta P_{2-3}} = 1 \quad \frac{\partial \Delta P_{1C}}{\partial \Delta P_{3-4}} = 1$$

$$B_{\Delta P_{1C}} = \sqrt{B^2_{\Delta P_{1-2}} + B^2_{\Delta P_{2-3}} + B^2_{\Delta P_{3-4}}}$$

$$B_{\Delta P} = \sqrt{B^2_{\Delta P_2} + \left(-B_{\Delta P_{1C}}\right)^2}$$

$$B_{P_{IN}} = \sqrt{B^2_{P_{ATM}} + B^2_{\Delta P}}$$

ASSIDE: $h = f(T_{SAT})$ $T_{SAT} = g(P)$

LET $\ell = f(g(P))$

SO: $h = \ell_1(P)$

$$B_{h_{IN}} = \sqrt{\left(\frac{\partial h_{IN}}{\partial P_{IN}} B_{P_{IN}}\right)^2} \; ; \; \frac{\partial h_{IN}}{\partial P_{IN}} \cong \frac{h_\ell|_{P_{IN}+B_{P_{IN}}} - h_\ell|_{P_{IN}}}{B_{P_{IN}}}$$

$$= h_\ell|_{P_{IN}+B_{P_{IN}}} - h_\ell|_{P_{IN}}$$

$$h_{L,EXIT} = \ell_1(P_{ATM}) \; , \; h_{V,EXIT} = \ell_2(P_{ATM})$$

$$B_{h_{\ell,EXIT}} = h_{\ell,EXIT}|_{P_{ATM}+B_{P_{ATM}}} - h_{\ell,EXIT}|_{P_{ATM}}$$ By analogy to $B_{h_{IN}}$

$$B_{h_{V,EXIT}} = h_{V,EXIT}|_{P_{ATM}+B_{P_{ATM}}} - h_{V,EXIT}|_{P_{ATM}}$$

$$B_W = \sqrt{\left(\frac{\partial W}{\partial h_{IN}} B_{h_{IN}}\right)^2 + \left(\frac{\partial W}{\partial h_{\ell,EXIT}} B_{h_{\ell,EXIT}}\right)^2 + \left(\frac{\partial W}{\partial h_{V,EXIT}} B_{h_{V,EXIT}}\right)^2}$$

$$\frac{\partial W}{\partial h_{IN}} = \frac{1}{h_{V,EXIT} - h_{L,EXIT}} \qquad \frac{\partial W}{\partial h_{V,EXIT}} = -\frac{h_{IN} - h_{\ell,EXIT}}{\left(h_{V,EXIT} - h_{\ell,EXIT}\right)^2}$$

$$\frac{\partial W}{\partial h_{e,exit}} = \frac{-1}{(h_{v,exit} - h_{e,exit})} + \frac{h_{in} - h_{e,exit}}{(h_{v,exit} - h_{e,exit})^2}$$

$$= \frac{-(h_{v,exit} - h_{e,exit}) + (h_{in} - h_{e,exit})}{(h_{v,exit} - h_{e,exit})^2} = \frac{h_{in} - h_{v,exit}}{(h_{v,exit} - h_{e,exit})^2}$$

$$B_w = \sqrt{\left(\frac{1}{h_{v,exit} - h_{l,exit}} B_{h_{in}}\right)^2 + \left(\frac{h_{in} - h_{v,exit}}{(h_{v,exit} - h_{e,exit})^2} B_{h_{e,exit}}\right)^2 + \left(-\frac{h_{in} - h_{e,exit}}{(h_{v,exit} - h_{e,exit})^2} B_{h_{v,exit}}\right)^2}$$

$$B_{T_{sub}} = \sqrt{B_{T_{sat}}^2 + B_{T_{in}}^2}$$

Precision Uncertainties

Definitions

Finite Statistics:

- $S \equiv$ Standard Deviation of Data Series
- $M \equiv$ Number of Data Points in Data Series

Pooled Statistics:

- $N \equiv$ Number of Data Points in Recording Interval
- $M \equiv$ Number of Recording Intervals
- $S_i \equiv$ Standard Deviation of Data in i^{th} Recording Interval

$$\langle S \rangle \equiv \sqrt{\frac{1}{M} \sum_{i=1}^{M} S^2}$$

Measured Variables

$\dot{M}_{in}$ - Finite Statistics

$$P_{\dot{M}_{in}} = \frac{S_{\dot{M}_{in}}}{\sqrt{M_{\dot{M}_{in}}}}$$

T_{IN} - Pooled Statistics

$$P_{T_{IN}} = \frac{\langle S_{T_{IN}} \rangle}{\sqrt{M_{T_{IN}} \cdot N_{T_{IN}}}} \qquad N_{T_{IN}} = f_{T_{IN}} \cdot \Delta t_{T_{IN}} = 1\,Hz \cdot 5s = 5$$

ΔP_c - Pooled Statistics

$$P_{\Delta P_c} = \frac{\langle S_{\Delta P_c} \rangle}{\sqrt{M_{\Delta P_c} \cdot N_{\Delta P_c}}} \qquad N_{\Delta P_c} = f_{\Delta P_c} \cdot \Delta t_{\Delta P_c} = 200\,Hz \cdot 5s = 1000$$

Calculated Variables

All precision error formulas are based on bias error formulas for the same variables except variables with no repeated measurements are removed

$$P_{A_1} = \frac{\pi \cancel{P_{D_1}} \overset{0}{D_1}}{2} = \phi$$

$$P_{A_3} = \frac{\pi \cancel{P_{D_3}} \overset{0}{D_3}}{2} = \phi$$

$$P_\rho = \rho|_{T_{IN}+P_{T_{IN}}} - \rho|_{T_{IN}}$$

$$P_\mu = \mu|_{T_{IN}+P_{T_{IN}}} - \mu|_{T_{IN}}$$

$$P_{V_1} = V_1 \sqrt{\left(\frac{P_{\dot{m}_{IN}}}{\dot{m}_{IN}}\right)^2 + \left(-\frac{P_\rho}{\rho}\right)^2 + \left(-\frac{\cancel{P_{A_1}}}{A_1}\right)^2} = V_1 \sqrt{\left(\frac{P_{\dot{m}_{IN}}}{\dot{m}_{IN}}\right)^2 + \left(-\frac{P_\rho}{\rho}\right)^2}$$

$$P_{V_3} = V_3 \sqrt{\left(\frac{P_{\dot{m}_{IN}}}{\dot{m}_{IN}}\right)^2 + \left(-\frac{P_\rho}{\rho}\right)^2 + \left(-\frac{P_{A_3}}{A_3}\right)^2} = V_3 \sqrt{\left(\frac{P_{\dot{m}_{IN}}}{\dot{m}_{IN}}\right)^2 + \left(-\frac{P_\rho}{\rho}\right)^2}$$

$$P_{RE} = R_E \sqrt{\left(\frac{P_\rho}{\rho}\right)^2 + \left(\frac{P_{V_1}}{V_1}\right)^2 + \left(\frac{P_{D_1}}{D_1}\right)^2 + \left(\frac{P_\mu}{\mu}\right)^2}$$

$$= R_E \sqrt{\left(\frac{P_\rho}{\rho}\right)^2 + \left(\frac{P_{V_1}}{V_1}\right)^2 + \left(\frac{P_\mu}{\mu}\right)^2}$$

$$P_{f_f} = f_f \frac{1}{4} \frac{P_{RE}}{R_E}$$

$$P_{\Delta P_{1-2}} = \Delta P_{1-2} \sqrt{\left(\frac{P_{f_f}}{f_f}\right)^2 + \left(\frac{P_L}{L}\right)^2 + \left(-\frac{P_{D_1}}{D_1}\right)^2 + \left(\frac{P_\rho}{\rho}\right)^2 + \left(2\frac{P_{V_1}}{V_1}\right)^2}$$

$$= \Delta P_{1-2} \sqrt{\left(\frac{P_{f_f}}{f_f}\right)^2 + \left(\frac{P_\rho}{\rho}\right)^2 + \left(2\frac{P_{V_1}}{V_1}\right)^2}$$

$$P_{K_1} = \sqrt{\left(-\frac{3}{2}\frac{1}{D_1}P_{D_3}\right)^2 + \left(\frac{3}{2}\frac{D_3}{D_2^2}P_{D_2}\right)^2} = \emptyset$$

$$P_{\Delta P_{2-3}} = \Delta P_{2-3} \sqrt{\left(\frac{P_\rho}{\rho}\right)^2 + \left(2\frac{P_{V_3}}{V_3}\right)^2 + \left(\frac{P_{K_1}}{K_1}\right)^2} = \Delta P_{2-3} \sqrt{\left(\frac{P_\rho}{\rho}\right)^2 + \left(2\frac{P_{V_3}}{V_3}\right)^2}$$

$$P_{K_2} = \sqrt{\left[\left(4\frac{D_3}{D_4} - 2\frac{1}{D_4}\right)P_{D_3}\right]^2 + \left[\left(-4\frac{D_3^2}{D_4^3} + 2\frac{D_4}{D_3^2}\right)P_{D_4}\right]^2} = \emptyset$$

$$P_{\Delta P_{3-4}} = \Delta P_{3-4} \sqrt{\left(\frac{P_\rho}{\rho}\right)^2 + \left(2\frac{P_{V_3}}{V_3}\right)^2 + \left(\frac{P_{K_2}}{K_2}\right)^2} = \Delta P_{3-4} \sqrt{\left(\frac{P_\rho}{\rho}\right)^2 + \left(2\frac{P_{V_3}}{V_3}\right)^2}$$

$$P_{\Delta P_{1C}} = \sqrt{P_{\Delta P_{1-2}}^2 + P_{\Delta P_{2-3}}^2 + P_{\Delta P_{3-4}}^2}$$

$$P_{\Delta P} = \sqrt{P_{\Delta P_2}^2 + (-\Delta P_{1C})^2}$$

$$P_{P_{IN}} = \sqrt{P_{P_{ATM}}^2 + P_{\Delta P}^2} = P_{\Delta P}$$

$$P_{h_{IN}} = h_f\Big|_{P_{IN}+P_{P_{IN}}} - h_f\Big|_{P_{IN}}$$

$$P_{h_{f,EXIT}} = h_{f,EXIT}\Big|_{P_{ATM}+P_{ATM}} - h_{f,EXIT}\Big|_{P_{ATM}} = \phi$$

$$P_{h_{V,EXIT}} = h_{V,EXIT}\Big|_{P_{ATM}+P_{ATM}} - h_{V,EXIT}\Big|_{P_{ATM}} = \phi$$

$$P_{UJ} = \sqrt{\left(\frac{1}{h_{V,EXIT}-h_{f,EXIT}} P_{h_{IN}}\right)^2 + \left(\frac{h_{IN}-h_{f,EXIT}}{(h_{V,EXIT}-h_{f,EXIT})^2} P_{h_{f,EXIT}}\right)^2 + \left(\frac{-h_{IN}-h_{f,EXIT}}{(h_{V,EXIT}-h_{f,EXIT})^2} P_{h_{V,EXIT}}\right)^2}$$

$$= \sqrt{\left(\frac{1}{h_{V,EXIT}-h_{f,EXIT}} P_{h_{IN}}\right)^2}$$

$$P_{T_{SAT}} = T_{SAT}\Big|_{P_{IN}-P_{P_{IN}}} - T_{SAT}\Big|_{P_{IN}}$$

$$P_{T_{SUB}} = \sqrt{P_{T_{SAT}}^2 + P_{T_{IN}}^2}$$

Combining Bias & Precision Errors

Only need to combine bias & precision errors for a small number of the variables shown so far. These are:

Exit Quality: w

Inlet Subcooling: T_{SUB}

Channel Pressure Drop: ΔP

All the rest of the variables are intermediate variables.

Definitions

t ≡ Students-t distribution parameter, used to account for the fact that the precision error is determined from a finite sample set.

C ≡ Confidence interval for Students-t parameter in this case 95% is used

D ≡ Degrees of freedom in precision error. This is typically calculated using a complicated weighted average of the variables feeding the variable being calculated. I am not going to use this method. I am going to use the degrees of freedom of the variable with the minimum degrees of freedom of the variables feeding the variable being calculated.

Calculated Variables

$$U_W = \sqrt{B_W^2 + \left(t_{\nu_W, 95\%} P_W\right)^2}$$

$$\nu_W = \nu_{in} = M_{in} - 1$$

$$U_{DP} = \sqrt{B_{DP}^2 + \left(t_{\nu_{DP}, 95\%} P_{DP}\right)^2}$$

$$\nu_{DP} = \nu_{in} = M_{in} - 1$$

$$U_{T_{Sub}} = \sqrt{B_{T_{Sub}}^2 + \left(t_{\nu_{T_{Sub}}, 95\%} P_{T_{Sub}}\right)^2}$$

$$\nu_{T_{Sub}} = \nu_{in} = M_{in} - 1$$

APPENDIX B

1-D Model Program Listing

```
%   Pdrop_batch

% inpbfname='Pdropin_batch_modelP2_';
%
% inpmfname='B';
% strtcond=1;
% endcond=36;

% inpmfname='A';
% strtcond=101;
% endcond=154;

inpbfname='Pdropin_batch_expt_cond_';

for j=1:3
    if j==1
        subprt='_a';
    elseif j==2
        subprt='_b';
    elseif j==3
        subprt='_c';
    end
    inpmfname='';
    strtcond=1;
    endcond=12;

    for i=strtcond:endcond
        if i<10
            inpfname=[inpbfname inpmfname '00' num2str(i) subprt];
        elseif i>=10&i<100
            inpfname=[inpbfname inpmfname '0' num2str(i) subprt];
        elseif i>=100
            inpfname=[inpbfname inpmfname num2str(i) subprt];
        end

        StartPdrop_func(inpfname)
    end
end
```

```
function StartPdrop_func(batchfname);

%    StartPdrop_func.m

%    ***  Results File name Input  ***
warning off MATLAB:fzero:UndeterminedSyntax
% [sndi,spdi]=wavread('chimes.wav');
% [sndf,spdf]=wavread('tada.wav');
clc

% Pdropin_batch_adb
% Pdropin_batch_liqonly
% Pdropin_batch_modelP2_B001
% Pdropin_batch_Props
% Pdropin_batch_val_sp
% Pdropin_batch_val_tp
% Pdropin_batch_grid
% Pdropin_batch_valid
eval(batchfname);
% whos

% resfile=input('Enter name for results file: ','s');
resfilem=[resfile_sub];
resfilemc=[resfile];
resfile=[resfile_sub '.csv'];
fid=fopen(resfile,'a');


contl='';
condn=1;
close all

for b=1:length(Psize);
% for b=1:length(wtsize);
% for b=1:length(ssize);
% for b=1:length(gamsize);
% for b=1:length(qsize);
% while isempty(contl)


if chmod==2
    % Rectangular Channels
    clear('R', 'tr', 'pr', 'totT', 'tolP', 'quf', 'bmflg', 'vimod', 'vfmod', 'flmod', 'vpflg', ...
        'pdflg', 'npflg', 'dfflg', 'Pout_t', 'Tsub', 'Pin_i', 'mdot', 'Cip', 'casnum', ...
        'Af', 'Gin', 'qin_0', 'Pin', 't1', 'NCflg', 'Tin', 'Tout_t', 'hin', 'nconvflg', 'jZ', ...
        'Aw', 'Af', 'hc', 'wc', 'alph', 'Per', 'D', 'dz', 'z', 'lamre', 'kinf', 'C', 'G', ...
        'Re', 'h', 'xi', 'lam', 'P', 'dPat', 'dPft', 'T', 'Ts', 'hsl', 'hsv', 'W', 'VF', 'Gm', ...
        'phi', 'phifl', 'chisq', 'Relo', 'Rel', 'Reg', 'v', 'vl', 'vv', 'mul', 'muv', 'lamm', ...
        'hfm', 'cnt', 'tpflg', 'delP', 'Ci', 'GM', 'delP_2last', 'delP_last', 'Tw', 'htc', ...
        'dPc', 'Kc', 'Gc', 'Ac', 't2', 'Pout', 'fricmod', 'acelmod', 'stitl', 'DP', 'dPa', ...
        'dPf', 'Vdot', 'Wout', 'VFend', 'uv', 'ul');
elseif chmod==3
    % Fractal Channels
    clear('R', 'tr', 'pr', 'totT', 'tolP', 'quf', 'bmflg', 'flmod', 'vpflg', ...
        'pdflg', 'npflg', 'dfflg', 'Pout_t', 'Tsub', 'Pin_i', 'mdot', 'Cip', 'casnum', ...
        'Af', 'Gin', 'qin_0', 'Pin', 't1', 'NCflg', 'Tin', 'Tout_t', 'hin', 'nconvflg', ...
        'jZ', 'jk', 'Aw', 'Af', 'hc', 'wc', 'alph', 'Per', 'D', 'dz', 'z', 'Zk', 'lamre', ...
        'kinf', 'C', 'lev', 'G', 'Re', 'h', 'xi', 'lam', 'P', 'dPat', 'dPft', 'dPr', 'T', 'Ts', ...
        'hsl', 'hsv', 'W', 'VF', 'Gm', 'phi', 'phifl', 'chisq', 'Relo', 'Rel', 'Reg', 'v', 'vl', 'vv', ...
        'mul', 'muv', 'lamm', 'hfm', 'cnt', 'tpflg', 'delP', 'Ci', 'Tw', 'htc', 't2', 'Pout', ...
        'fricmod', 'acelmod', 'stitl', 'rr', 'spcng', 're', 'ri', 'DP', 'dPa', 'dPf', 'Vdot', ...
        'Wout', 'VFend', 'uv', 'ul');
end

%
%    ***  Constants  ***
%


%
%    ***  Inputs  ***
%

% Pdropinp_single
% Pdropinp_adb
% Pdropinp_liqonly
% Pdropinp_modelP2
% Pdropinp_Props
% Pdropinp_val_sp
% Pdropinp_val_tp
% Pdropinp_grid
% Pdropinp_valid
if chmod==2
    if exist('qsize','var')
        Pdropinp_modelP2p
    else
        pdropinp_modelP2pa
    end
else
    if exist('qsize','var')
        Pdropinp_modelP2f
```

```
    else
        pdropinp_modelP2fa
    end
end

t1=clock;


%
%    ***  Initial Calculations  ***
%

NCflg=0;

Pin=Pin*1000;               %    [kPa] -> [Pa]
Pout_t=Pout_t*1000;         %    [kPa] -> [Pa]
if isnan(Tsub)
    Tin=Tin_t+273.15;
else
     Tin=Tsw(Pin)-Tsub;     %    [deg K]
end
Tout_t=Tsw(Pout_t);         %    [deg K]
hin=hlw(Tin);               %    [J/kg]
qin=qin*10000;              %    [W/cm^2] -> [W/m^2]

if chmod==1
     D=D*1e-6;              %    [micron] - [m]
elseif chmod==2
%      wcin=wcin*1e-6;             %    [micron] - [m]
%      hcin=hcin*1e-6;             %    [micron] - [m]
elseif chmod==3
%      hcin=hcin*1e-6;           %    [micron] - [m]
%      wct=wct*1e-6;             %    [micron] - [m]
end

%
%    ***  Pressure Calculations  ***
%

% whos

delP=0; delP_last=0;
padj=0; padj_last=0;
padjflg=0;

if chmod==1                   % Single Round Channel
    nconvflg=0;
    [jZ,Aw,Af,Per,D,dz,z,lamre,kinf,C]=circgeom(D,Z,dzn);
    Wout=((qin*Aw/Gin*Af)-hin)/(hvw(Tout_t)-hlw(Tout_t));
    if qin==0;
        [G,Re,h,xi,lam]=...
            circflow_adb(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
        [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
            circPcal_w_f_adb_2(jZ,Pin,Tin,qin,G,h,D,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
    else
        [G,Re,h,xi,lam]=...
            circflow_f(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
        [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
            circPcal_w_f_2(jZ,Pin,Tin,qin,G,h,D,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
    end
    delP=abs(P(end)-Pout_t); delP_last=2*delP; delP_2last=3*delP;
    Ci=itercoef(Cip,delP);
    clc
    disp(sprintf('\n'))
    disp(sprintf('T_sub = %4.1f; q"_wall = %5.2f;  G_in = %6.2f; D_c = %4.1f',...
        Tsub,qin_0,Gin,D(1)*1e6))
            disp(sprintf('Pin=%6.2fkPa; Pout=%6.2fkPa; Wout=%4.3f; delP=%7.4fkPa; delP_l=%7.4fkPa; nconvflg=%2.0f;
Pajd=%6.1f; Padjc=%d',...
                P(1)/1000,P(end)/1000,W(end),delP/1000,delP_last/1000,nconvflg, padj, padjflg))
    while delP>=tolP & npflg==0
        padj_last=padj;
        padj=Ci*(P(end)-Pout_t);
        if abs(padj)>abs(padj_last) & padj_last~=0;
            if padjflg==0
                padj=(abs(padj_last)/10)*(padj/abs(padj));
                padjflg=1;
                nconvflg=0;
            else
                if delP>delP_last
                    padj=(abs(padj_last)/10)*(padj/abs(padj));
                else
                    padj=padj_last;
                end
            end
        end
        Pin=Pin-padj;
        if W(end)>1.05;
            npflg=1;
            NCflg=3;
            break
        end
```

```
        if ~isreal(P(end))
            NCflg=4
            break
        end
        if Pin>=2000*1000
            NCflg=5;
            break
        else
            if isnan(Tsub)
                Tin=Tin_t+273.15;
            else
                Tin=Tsw(Pin)-Tsub;                              %    [deg K]
            end
            hin=hlw(Tin);
            if qin==0;
                [G,Re,h,xi,lam]=...
                    circflow_adb(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
                [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
                    circPcal_w_f_adb_2(jZ,Pin,Tin,qin,G,h,D,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
            else
                [G,Re,h,xi,lam]=...
                    circflow_f(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
                [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
                    circPcal_w_f_2(jZ,Pin,Tin,qin,G,h,D,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
            end
            delP_last=delP;
            delP=abs(P(end)-Pout_t);
            Ci=itercoef(Cip,delP);
            disp(sprintf('Pin=%6.2fkPa; Pout=%6.2fkPa; Wout=%4.3f; delP=%7.4fkPa; delP_l=%7.4fkPa; nconvflg=%2.0f;
Pajd=%6.1f; Padjc=%d',...
                P(1)/1000,P(end)/1000,W(end),delP/1000,delP_last/1000,nconvflg, padj, padjflg))
            if delP>delP_last
                nconvflg=nconvflg+1;
                if nconvflg>3
                    NCflg=1;
                    break
                end
            end
        end
    end
    [Tw,htc]=frachcal_w_b(G,T,xi,W,jZ,vl,vv,qin,D,ones(size(z)),tpflg);
    if dz==10
        ressave3b
    end

elseif chmod==2              % Single Rectangular Channel
    nconvflg=0;
    [jZ,Aw,Af,hc,wc,alph,Per,D,dz,z,lamre,kinf,C]=rectgeom(hcin,wcin,Z,dzn);
    if qin==0;
        [G,Re,h,xi,lam]=...
            rectflow_adb(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
        [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
            rectPcal_w_f_adb_2(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
    else
        [G,Re,h,xi,lam]=...
            rectflow_f(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
        [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
            rectPcal_w_f_2(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);

    end
    delP=abs(P(end)-Pout_t); delP_last=2*delP; delP_2last=3*delP;
    Ci=itercoef(Cip,delP);
    clc
    disp(sprintf('\n'))
    disp(sprintf('Case Number: %d',casnum))
    disp(sprintf('T_sub = %4.1f; q"_wall = %5.2f;  G_in = %6.2f; w_c = %4.1f; h_c = %5.1f',...
        Tsub,qin_0,Gin,wc(1)*1e6,hc(1)*1e6))
            disp(sprintf('Pin=%6.2fkPa; Pout=%6.2fkPa; Wout=%4.3f; delP=%7.4fkPa; delP_l=%7.4fkPa; nconvflg=%2.0f;
Pajd=%6.1f; Padjc=%d',...
                P(1)/1000,P(end)/1000,W(end),delP/1000,delP_last/1000,nconvflg, padj, padjflg))

    while delP>=tolP & npflg==0
        padj_last=padj;
        padj=Ci*(P(end)-Pout_t);
        if abs(padj)>abs(padj_last) & padj_last~=0;
            if padjflg==0
                padj=(abs(padj_last)/10)*(padj/abs(padj));
                padjflg=1;
                nconvflg=0;
            else
                if delP>delP_last
                    padj=(abs(padj_last)/10)*(padj/abs(padj));
                else
                    padj=padj_last;
                end
            end
        end
        Pin=Pin-padj;
        if W(end)>1.05;
            npflg=1;
            NCflg=3;
```

```
            break
        end
        if ~isreal(P(end))
            NCflg=4
            break
        end
        if Pin>=2000*1000
            NCflg=5;
            break
        else
            if isnan(Tsub)
                Tin=Tin_t+273.15;
            else
                Tin=Tsw(Pin)-Tsub;                              %    [deg K]
            end
            hin=hlw(Tin);
            if qin==0;
                [G,Re,h,xi,lam]=...
                    rectflow_adb(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
                [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
                    rectPcal_w_f_adb_2(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
            else
                [G,Re,h,xi,lam]=...
                    rectflow_f(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg);
                [P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=...
                    rectPcal_w_f_2(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg);
            end
%             GM=sqrt(-1/((1-W(end))/dPdvl(Pout_t)+W(end)/dPdvv(Pout_t)));
%             if GM<=G(1);
%                 NCflg=2;
%                 break
%             end
            delP_last=delP;
            delP=abs(P(end)-Pout_t);
            Ci=itercoef(Cip,delP);
            disp(sprintf('Pin=%6.2fkPa; Pout=%6.2fkPa; Wout=%4.3f; delP=%7.4fkPa; delP_l=%7.4fkPa; nconvflg=%2.0f;
Pajd=%6.1f; Padjc=%d',...
                P(1)/1000,P(end)/1000,W(end),delP/1000,delP_last/1000,nconvflg, padj, padjflg))
            if delP>delP_last
                nconvflg=nconvflg+1;
                if nconvflg>3
                    NCflg=1;
                    break
                end
            end
        end
    end
    [Tw,htc]=frachcal_w_b(G,T,xi,W,jZ,vl,vv,qin,D,ones(size(z)),tpflg);
    if dz==10
        ressave3b
    end
    if quf==1;
        [dPc, Kc, Gc, Ac]=qu_correction(vl(1), v(end), Gin);
    end

elseif chmod==3              % Fractal Channel Network
    nconvflg=0;
    [jZ,jk,Aw,Af,hc,wc,alph,Per,D,dz,z,Zk,lamre,kinf,C,lev]=...
        fracgeom(dzn,Levs,hcin,wct,Z,gama,betta,pdflg);
    if qin==0
        [G,Re,h,xi,lam]=fracflow_adb...
            (jZ,jk,Tin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,lamre,kinf,C,pdflg,vpflg,dfflg);

[P,dPat,dPft,dPr,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=fracPcal_w_f_adb
_2...
            (jZ,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,Re,tolT,lev,dfflg,vfmod,vimod,vpflg);
    else
        [G,Re,h,xi,lam]=fracflow_f...
            (jZ,jk,Tin,qin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,lamre,kinf,C,pdflg,vpflg,dfflg);

[P,dPat,dPft,dPr,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=fracPcal_w_f_2..
.
            (jZ,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,Re,tolT,lev,dfflg,vfmod,vimod,vpflg);
    end
    delP=abs(P(end)-Pout_t); delP_last=2*delP;
    Ci=itercoef(Cip,delP);
    clc
    disp(sprintf('\n'))
    disp(sprintf('Case Number: %d',casnum))
    disp(sprintf('T_sub = %4.1f; q"_wall = %5.2f;  G_in = %6.2f; k = %2.0f; w_ct = %4.1f; h_c = %5.1f',...
        Tsub,qin_0,Gin,Levs,wc(end)*1e6,hc(1)*1e6))
            disp(sprintf('Pin=%6.2fkPa; Pout=%6.2fkPa; Wout=%4.3f; delP=%7.4fkPa; delP_l=%7.4fkPa; nconvflg=%2.0f;
Pajd=%6.1f; Padjc=%d',...
                P(1)/1000,P(end)/1000,W(end),delP/1000,delP_last/1000,nconvflg, padj, padjflg))
    while delP>=tolP & npflg==0
        padj_last=padj;
        padj=Ci*(P(end)-Pout_t);
        if abs(padj)>abs(padj_last) & padj_last~=0;
            if padjflg==0
                padj=(abs(padj_last)/10)*(padj/abs(padj));
                padjflg=1;
```

```
                nconvflg=0;
            else
                if delP>delP_last
                    padj=(abs(padj_last)/10)*(padj/abs(padj));
                else
                    padj=padj_last;
                end
            end
        end
        Pin=Pin-padj;
        if W(end)>1.05
            npflg=1;
            NCflg=3;
            break
        end
        if ~isreal(P(end))
            NCflg=4
            break
        end
        if Pin>=2000*1000
            NCflg=5;
            break
        else
            if isnan(Tsub)
                Tin=Tin_t+273.15;
            else
                Tin=Tsw(Pin)-Tsub;                              %    [deg K]
            end
            hin=hlw(Tin);
            if qin==0
                [G,Re,h,xi,lam]=fracflow_adb...
                    (jZ,jk,Tin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,lamre,kinf,C,pdflg,vpflg,dfflg);

[P,dPat,dPft,dPr,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=fracPcal_w_f_adb
_2...
                    (jZ,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,Re,tolT,lev,dfflg,vfmod,vimod,vpflg);
            else
                [G,Re,h,xi,lam]=fracflow_f...
                    (jZ,jk,Tin,qin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,lamre,kinf,C,pdflg,vpflg,dfflg);

[P,dPat,dPft,dPr,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=fracPcal_w_f_2..
.
                    (jZ,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,Re,tolT,lev,dfflg,vfmod,vimod,vpflg);
            end
            delP_last=delP;
            delP=abs(P(end)-Pout_t);
            Ci=itercoef(Cip,delP);
            disp(sprintf('Pin=%6.2fkPa; Pout=%6.2fkPa; Wout=%4.3f; delP=%5.3fkPa; delP_l=%5.3fkPa; nconvflg=%2.0f;
Pajd=%6.1e; Padjc=%d',...
                P(1)/1000,P(end)/1000,W(end),delP/1000,delP_last/1000,nconvflg, padj, padjflg))
            if delP>delP_last
                nconvflg=nconvflg+1;
                if nconvflg>3
                    NCflg=1;
                    break
                end
            end

        end
    end
    [Tw,htc]=frachcal_w_b(G,T,xi,W,jZ,vl,vv,qin,D,alph,tpflg);
%     if dz==10
%         ressave3b
%     end
    ressave3b
end


t2=clock;

resprt3
ressave3
% pause
% resplot3
% wavplay(sndi,spdi,'sync')

% contl=input('Do you want to continue (CR=Yes, n=No):','s');
% if ~isempty(contl)
%      break
% end

condn=condn+1;

%   This is end that goes with for statement, comment for single case
end

ressave3a;

if qin==0 & chmod==3
%     ressave3a
%     respost3
```

```
end

% wavplay(sndf,spdf);
fclose(fid);
```

```
%    Pdropin_f_batch_modelP.m
%
%
%


%   *** Rectangular Channel ***

%   *** Fractal Channel
%   6-21-05

resfile='actual_expt_cond_f10_a4_dz10';
resdir='C:\My Documents\Experimental Conditions\f10_a4'
cd(resdir);

chmod=3;

vimod=10;
vfmod=4;

wct=100e-6;       % Units: [m];
hcin=150e-6;      % Units: [m];
Z=.018;         % Units: [m];
Levs=4;
No=16;
betta=1/sqrt(2);
gama=1/sqrt(2);
dzn=10e-6;     % Units: [m]; [m];



%   Single Phase
msize=[100.15 ...
        124.81 124.97 ...
        150.11 150.84 150.12 149.77 ...
        175.09 174.05 149.36 175.56 ...
        199.72 199.82 200.07 200.16 199.79 200.89 ...
        225.85 225.43 225.72]/(60*No);
Tsize=[104.62 ...
        107.02 105.8 ...
        111.02 109.62 108.59 107.33 ...
        112.58 112.33 106.86 108.84 ...
        117.31 117.15 115.56 113.11 110.94 108.66 ...
        114.46 112.17 110.1];
Cipsize=[1 1 1 1 1 1 1 1 1 1 ...
        1 1 1 1 1 1 1 1]*10;
%               End First Line Here.........................v
Psize=[121.5 ...
        132.62 127.02 ...
        151.21 145.73 143.29 138.77 ...
        161.15 161.08 143.04 146.29 ...
        202.18 204.27 195.47 183.34 173.34 163.57 ...
        195.3 184.89 176.17];
casen=[10 ...
        20 21 ...
        30 31 32 33 ...
        40 41 42 43 ...
        50.1 50.2 51 52 53 54 ...
        63 64 65];
```

```
%    Pdropinp_f.m
%
%    Defines the input variables and parameters for StartPdrop_f.m
%
%    *** Model Configuration Parameters:
%
%    chmod     -    Channel Geometry Definition:  1=Single Circular
%                                                 2=Single Rectangular
%                                                 3=Fractal Branching
%    vimod     -    Pressure Drop Model to use:   1=Homogenous-Lin et.al.;
%                                                 2=Homogenous-McAdams;
%                                                 3=Homoggenous-Cicchitti;
%                                                 4=Homogemous-Dukler;
%                                                 5=Homogenous-Stanley et al.
%                                                 6=Homogenous-Collier
%                                                 7=Separated-Lockhart & Martinelli;
%                                                 8=Separated-Mishima & Hibiki;
%                                                 9=Separated-Lee & Lee
%                                                10=Separated-Qu & Mudawar
%    vfmod     -    Void fraction Correlation:    0=homogenous model;
%                                                 1=separated model per Chung et al.;
%                                                 2=Separated model per Armaand
%                                                 3=Separated model per Lockhart & Martinelli
%                                                 4=Separated model per Zivi
%    dfflg     -    Developing Flow Flag:         0=Fully Developed Flow
%                                                 1=Developing Flow
%    flmod     -    Working Fluid Definition:     1=Water
%                                                 2=R-113
%                                                 3=Ammonia-Water
%    vpflg     -    Variable Properties Flag:     0=Constant Properties
%                                                 1=Temperature Dependent
%    pdflg     -    Propagation Direction Flag:   0=Forward Propagation
%                                                 1=Backward Propagation
%    npflg     -    physically possible flow condition Flag:
%                                                 0=
%                                                 1=
%
%    *** Convergence Tolerances
%
%    tolT      -    Temperature Convergence Tolerance
%    topP      -    Pressure Convergence Tolerance
%
%    *** Channel Geometry Definitions
%
%    D         -    Diameter for Circular Channel in Microns
%    hc        -    Channel height for Rectangular channel in microns
%    wc        -    Channel width for rectangular channel in microns
%    hcin      -    Channel height for fractal channels in microns
%    wct       -    Terminal Channel width for fractal channels in microns
%    Z         -    Total channel lenght in meters
%    dzn       -    Nominal discretization distance in meters
%    Levs      -    Number of branching levels for fractal channels
%    n         -    Number of branches per branching level for fractal channels
%    betta      -    Width ratio between brancing levels for fractal channels
%    gama      -    Length ratio between branching levels for fractal channels
%
%    *** Flow Condition Definintions
%
%    Pin       -    Initial guess for inlet Pressure
%    Pout_t    -    Outlet Pressure target (Saturated liquid vapor mixture)
%    Gin       -    Channel Inlet liquid mass flux
%    qin       -    Channel wall heat flux
%    Tsub      -    Channel Inlet liquid subcooling

%    ***  Thermodynamic Property Non-Dimenstionalization  ***
R=8314; tr=100; pr=1000000; %Units: [J/kmol-K]; [K]; [Pa]
tolT=.01;                     % Units: [Deg K]
tolP=100;                    % Units: [Pa]

quf=0;    bmflg=0;


% % %   *** Rectangular Channel ***
%
%                   vimod=10;
%                   vfmod=4;
%
%                   flmod=1;          vpflg=1;
%                   pdflg=0;          npflg=0;
%                   if vimod==5|vimod==6
%              dfflg=0;
%                   else
%              dfflg=1;
%                   end
%         dfflg=1;
%
%
%
% %   *** Single Case / Constant Data ***
%                   Pout_t=101.3;                % Units: [kPa]
%                   if isnan(Tin_t)
```

```
%             Tsub=Tsize(b);                      % Units: [Deg C]
%         else
%             Tsub=NaN;
%         end
%
% %   *** Batch Input Data ***
%                   Pin_i=Psize(b);                     % Units: [kPa]
%                   mdot=msize(b);
%       qin=qsize(b);
%                   Cip=Cipsize(b);
%                   casnum=casen(b);
%
%       Af=(hcin)*(wcin);
%                   Gin=(mdot/(1000*60))/(No*Af);
%                   qin_0=qin;
%                   Pin=Pin_i;                   % Units: [kPa]
%
% %   *** Fractal Channel ***


                    flmod=1;          vpflg=1;
                    pdflg=0;          npflg=0;
                    if vimod==5|vimod==6
              dfflg=0;
                    else
              dfflg=1;
                    end
                    dfflg=1;


%   *** Single Case / Constant Data ***
                    Pout_t=101.3;                % Units: [kPa]
%                   if isnan(Tin_t)
%             Tsub=Tsize(b);                      % Units: [Deg C]
%         else
%             Tsub=NaN;
%         end

%   *** Batch Input Data ***
                    Pin_i=Psize(b);                     % Units: [kPa]
        Tsub=NaN;
        Tin_t=Tsize(b);
                    mdot=msize(b);
        qin=0;
        Cip=Cipsize(b);
        casnum=casen(b);

        Af=hcin*(wct/(betta^Levs));
        Gin=(mdot/(1000))/(Af);
        qin_0=qin;
                    Pin=Pin_i;                   % Units: [kPa]
```

```
function  [jZ,Aw,Af,Per,D,dz,z,lamre,kinf,C]=circgeom(D,Z,dzn)

%    function  [jZ,Aw,Af,Per,D,dz,z,lamre,kinf,C]=circgeom(D,Z,dzn)
%
%    Creates arrays containing information on the channel geometry for the
%    single circular channel geometry
%
%    Inputs:
%         D    -    Channel diameter
%         Z    -    Channel Length
%         dzn  -    Nominal discritization length
%
%    Outputs:
%         jZ   -
%         Aw   -
%         Af   -
%         Per  -
%         D    -
%         dz   -
%         z    -
%         lamre-
%         kinf -
%         C    -
%

%    ***  Calculating Constants  ***

Aw=pi*D*Z;
Af=(pi*D^2)/4;
dz=Z/(round(Z/dzn));


%    ***  Creating Variable Arrays  ***

z=[0:dz:Z];
dz=ones(size(z))*dz;
D=ones(size(z))*D;
Per=(pi.*D);

lamre=ones(size(z))*64;
kinf=ones(size(z))*1.25;
C=ones(size(z))*0.000212;

jZ=length(z);
```

```
function  [G,Re,h,xi,lam]=circflow_f(jZ,Tin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)

%    function  [G,Re,h,xi,lam]=circflow_f(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)

%
%    Calculates the flow parameters in single circular channel.
%
%    Inputs:
%         jZ   -
%         Tin  -
%         qin  -
%         Gin  -
%         hin  -
%         Aw   -
%         Per  -
%         D    -
%         dz   -
%         z    -
%         lamre-
%         kinf -
%         C    -
%         pdflg-
%         vpflg-    variable property flag 0=constant prop, 1=variable prop
%         dfflg-
%    Outputs:
%         G    -
%         Re   -
%         h    -
%         xi   -
%         lam  -
%


%
%    ***  Variable Array Creation/Initialization  ***
%

G=ones(size(z))*Gin;
h=ones(size(z));
xi=zeros(size(z));
lam=zeros(size(z));

xi(1)=eps;

Re=G.*D./muliq_w(Tin,vpflg);

h=h*hin;

%
%    ***  Friction Factor Calculation  ***
%

xi(2:jZ)=(z(2:jZ)./D(2:jZ))./Re(2:jZ);

if dfflg==0
    lam=64./Re;
else
    lam(2:jZ)=lamda_loc(xi(2:jZ),z(2:jZ),lamre(2:jZ),kinf(2:jZ),C(2:jZ),D(2:jZ),Re(2:jZ));
    lam(1)=lam(2)-5*((lam(3)-lam(2))/dz(2))*z(2);
end
```

```
function
[P,dPat,dPft,T,Ts,hsl,hsv,W,VF,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=circPcal_w_f(jZ,Pin,Tin
,qin,G,h,D,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg)

%    function
[P,dPa,dPf,T,Ts,hsl,hsv,W,VF,phi,v,vl,vv,hfm,cnt,tpflg]=circPcal_w_f(jZ,Pin,Tin,qin,G,h,D,dz,z,lam,tolT,dfflg,vfmod,vim
od,vpflg)
%
%    Calculates the pressure along the circular channel from the end towards
%    the beginning based for water as the fluid.
%
%    Inputs:
%         jZ   -
%         Pout -
%         Wout -
%         G    -
%         h    -
%         D    -
%         dz   -
%         z    -
%         lam  -
%         dfflg-
%         vimod-
%         vpflg-
%    Outputs:
%         P    -
%         T    -
%         Ts   -
%         hs   -
%         W    -
%         VF   -
%         phi  -
%         v    -
%         vl   -
%         vv   -
%         mul  -
%         muv  -
%         mutp -
%         tpflg-
%

%
%    ***   Constants   ***
%
Pmin=70000;

%
%    ***  Array Initialization  ***
%

%    ***  Creation  ***
P=zeros(size(z));        T=zeros(size(z));
dPat=zeros(size(z));     dPft=zeros(size(z));
mul=zeros(size(z));      muv=zeros(size(z));
lamm=zeros(size(z));
vl=zeros(size(z));       vv=zeros(size(z));
v=zeros(size(z));
Ts=zeros(size(z));
hsl=zeros(size(z));      hsv=zeros(size(z));
W=zeros(size(z));        VF=zeros(size(z));
phi=ones(size(z));       phifl=zeros(size(z));
hfm=ones(size(z));
Relo=zeros(size(z));
Rel=zeros(size(z));      Reg=zeros(size(z));
chiz=zeros(size(z));     chisq=zeros(size(z));

tpflg=zeros(size(z));
cnt=zeros(size(z));

%    ***  Initial Values  ***
P(1)=Pin;
T(1)=Tin;                Ts(1)=Tsw(Pin);
hsl(1)=hlw(Ts(1));
vl(1)=vlw(T(1));
v(1)=vl(1);
W(1)=0;
lamm(1)=lam(1)
%
%    ***  Calculations  ***
%

bar=waitbar(0,['Percent of Channel Length Completed']);

if dfflg==1                                %  Developing Flow
    for j=2:jZ
        waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],P1);
            vl(j)=Vlw(T(j));    v(j)=vl(j);
```

```
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))               %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*(Re(j)/Relo(j));
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
        end
    end
elseif dfflg==0                             %  Fully Developed flow
    for j=2:jZ
        waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                   %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],P1);
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))                %  Two-phase flow
            tpflg(j)=3;
            y(j)=1;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
```

```
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*(Re(j)/Relo(j));
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
        end
    end
end

close(bar)
```

```
function  [G,Re,h,xi,lam]=circflow_f(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)

%    function  [G,Re,h,xi,lam]=circflow_f(jZ,Tin,qin,Gin,hin,Aw,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)

%
%    Calculates the flow parameters in single circular channel.
%
%    Inputs:
%         jZ   -
%         Tin  -
%         qin  -
%         Gin  -
%         hin  -
%         Aw   -
%         Per  -
%         D    -
%         dz   -
%         z    -
%         lamre-
%         kinf -
%         C    -
%         pdflg-
%         vpflg-    variable property flag 0=constant prop, 1=variable prop
%         dfflg-
%    Outputs:
%         G    -
%         Re   -
%         h    -
%         xi   -
%         lam  -
%


%
%    ***  Variable Array Creation/Initialization  ***
%

G=ones(size(z))*Gin;
h=zeros(size(z));
xi=zeros(size(z));
lam=zeros(size(z));

xi(1)=eps;

Re=G.*D./muliq_w(Tin,vpflg);

for j=1:jZ
     if j==1
          h(j)=hin;
     else
          h(j)=h(j-1)+qin*Per(j)*dz(j)/(G(j)*((pi*D(j)^2)/4));
     end
end

%
%    ***  Friction Factor Calculation  ***
%

xi(2:jZ)=(z(2:jZ)./D(2:jZ))./Re(2:jZ);

if dfflg==0
    lam=64./Re;
else
    lam(2:jZ)=lamda_loc(xi(2:jZ),z(2:jZ),lamre(2:jZ),kinf(2:jZ),C(2:jZ),D(2:jZ),Re(2:jZ));
    lam(1)=lam(2)-5*((lam(3)-lam(2))/dz(2))*z(2);
end
```

```
function
[P,dPat,dPft,T,Ts,hsl,hsv,W,VF,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=circPcal_w_f(jZ,Pin,Tin
,qin,G,h,D,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg)

%    function
[P,dPa,dPf,T,Ts,hsl,hsv,W,VF,phi,v,vl,vv,hfm,cnt,tpflg]=circPcal_w_f(jZ,Pin,Tin,qin,G,h,D,dz,z,lam,tolT,dfflg,vfmod,vim
od,vpflg)
%
%    Calculates the pressure along the circular channel from the end towards
%    the beginning based for water as the fluid.
%
%    Inputs:
%         jZ   -
%         Pout -
%         Wout -
%         G    -
%         h    -
%         D    -
%         dz   -
%         z    -
%         lam  -
%         dfflg-
%         vimod-
%         vpflg-
%    Outputs:
%         P    -
%         T    -
%         Ts   -
%         hs   -
%         W    -
%         VF   -
%         phi  -
%         v    -
%         vl   -
%         vv   -
%         mul  -
%         muv  -
%         mutp -
%         tpflg-
%

%
%    ***   Constants   ***
%
Pmin=70000;

%
%    ***  Array Initialization  ***
%

%    ***  Creation  ***
P=zeros(size(z));        T=zeros(size(z));
dPat=zeros(size(z));     dPft=zeros(size(z));
mul=zeros(size(z));      muv=zeros(size(z));
lamm=zeros(size(z));
vl=zeros(size(z));       vv=zeros(size(z));
v=zeros(size(z));
Ts=zeros(size(z));
hsl=zeros(size(z));      hsv=zeros(size(z));
W=zeros(size(z));        VF=zeros(size(z));
phi=ones(size(z));       phifl=zeros(size(z));
hfm=ones(size(z));
Relo=zeros(size(z));
Rel=zeros(size(z));      Reg=zeros(size(z));
chiz=zeros(size(z));     chisq=zeros(size(z));

tpflg=zeros(size(z));
cnt=zeros(size(z));

%    ***  Initial Values  ***
P(1)=Pin;
T(1)=Tin;                Ts(1)=Tsw(Pin);
hsl(1)=hlw(Ts(1));
vl(1)=vlw(T(1));
v(1)=vl(1);
W(1)=0;
lamm(1)=lam(1)
%
%    ***  Calculations  ***
%

bar=waitbar(0,['Percent of Channel Length Completed']);

if dfflg==1                                %  Developing Flow
    for j=2:jZ
        waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],P1);
            vl(j)=Vlw(T(j));    v(j)=vl(j);
```

```
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))               %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*(Re(j)/Relo(j));
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
        end
    end
elseif dfflg==0                             %  Fully Developed flow
    for j=2:jZ
        waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                   %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],P1);
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))                %  Two-phase flow
            tpflg(j)=3;
            y(j)=1;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
```

```
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*(Re(j)/Relo(j));
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
        end
    end
end

close(bar)
```

```
function Ci=itercoef(Cip,delP)

%    function Ci=itercoef(delP)
%
%    Sets the multiplicitave interation coefficient based on the size of
%    the difference.

delP=delP/1000;

% C=10;

if delP>=200
     Ci=.5/Cip;
elseif delP>=100 & delP<200
     Ci=.5/Cip;
elseif delP>=60 & delP<100
     Ci=.5/Cip;
elseif delP>=.2 & delP<60
     Ci=.5/Cip;
elseif delP>=.005 & delP<.2
%      Ci=.5/(50*C);
     Ci=.5/Cip;
else
     Ci=.5/Cip;
end
```

```
function [Tw,htc]=frachcal_w_b(G,T,xi,W,jZ,vl,vv,qin,D,alph,tpflg);

%    function [Tw,h]=frachcal_w_b(G,T,xi,W,jZ,vl,vv,qin,D,alph);

Tw=zeros(size(T));
htc=zeros(size(T));

Pr = Prlw(T);
kl = klw(T);
zstar = xi./Pr;

load sd_data_table;                %load coefficient data - already determined from sd_td_table_formulation.m
ARtab=SD_table(:,1);               %reassignment of data
SD(:,:)=SD_table(:,2:4);

Fk = 1.0;   % for water.  Fluid dependent.
% bar=waitbar(0,['Wall Temperature Calculation']);
for j = 2:jZ
%      waitbar((j)/jZ);
     ARtemp=max((round(alph(j)*100))/100,0.25);
     sd(1)=interp1(ARtab,SD(:,1),ARtemp);
     sd(2)=interp1(ARtab,SD(:,2),ARtemp);
     sd(3)=interp1(ARtab,SD(:,3),ARtemp);
     Nu(j) = 1./(sd(1)*(1-exp(sd(2)*zstar(j).^sd(3))));
     h_1p(j)=Nu(j).*kl(j)./D(j);
     if tpflg(j)==0
          htc(j)=h_1p(j);
     elseif tpflg(j)==3
          Wt = min(W(j),1.0);
          Co(j) = (((1-Wt)./Wt).^0.8).*((vl(j)./vv(j)).^0.5);
          Bo(j) = qin./G(j)./(hvw(T(j))-hlw(T(j)));
          Fr(j) = ((G(j).*vv(j)).^2)./9.81./D(j);
          if Co(j) < 0.65
               CC = [1.1360 -0.9 667.2 0.7 0.3];
          else
               CC = [0.6683 -0.2 1058.0 0.7 0.3];
          end
          if Fr(j) > 0.04
               CC(5) = 0;
          end
          htc(j) = h_1p(j)*(CC(1)*(Co(j)^CC(2))*((25*Fr(j))^CC(5)) + CC(3)*(Bo(j)^CC(4))*Fk);
     end
     Tw(j)=T(j)+qin./htc(j);
end
% close(bar)
```

```
%    ressave3b.m
%
%    saves data arrays for each case
%
%
%
%
%
%

if chmod==1
    save([resfilemc '-Case-' num2str(casnum)],'P', 'T', 'W', 'VF', 'vl', 'vv', ...
        'dPft', 'dPat', 'phi', 'dz', 'z', 'G', 'lam', 'D');
elseif chmod==2
    save([resfilemc '-Case-' num2str(casnum)],'P', 'T', 'Tw', 'W', 'VF', 'vl', 'vv', ...
        'h', 'hsl', 'hsv', 'mul', 'muv', 'dPft', 'dPat', 'phi','phifl','chisq', ...
        'Re', 'Relo', 'Rel', 'Reg', 'tpflg', ...
        'dz', 'z','alph', 'G', 'Gm', 'lam', 'D', 'No', 'wc', 'hc');
elseif chmod==3
    save([resfilemc '-Case-' num2str(casnum)],'P', 'T', 'Tw', 'W', 'VF', 'vl', 'vv', ...
        'h', 'hsl', 'hsv', 'mul', 'muv' ,'dPft', 'dPat', 'dPr', 'phi','phifl','chisq', ...
        'Re', 'Relo', 'Rel', 'Reg', 'tpflg', ...
        'dz', 'z', 'lev', 'alph', 'G', 'Gm', 'lam', 'D', 'wc', 'hc', 'gama', 'betta');
end
```

```
function  [jZ,Aw,Af,hc,wc,alph,Per,D,dz,z,lamre,kinf,C]=rectgeom(hc,wc,Z,dzn)

%    function  [jZ,Aw,Af,hc,wc,alph,Per,D,dz,z,lamre,kinf,C]=rectgeom(hc,wc,Z,dzn)
%
%    Creates arrays containing information on the channel geometry for the
%    single rectangular channel geometry
%
%    Inputs:
%         hc   -    Channel height
%         wc   -    Channel width
%         Z    -    Channel Length
%         N    -    number of discritization divisions
%
%    Outputs:
%         jZ   -
%         Aw   -
%         Af   -
%         hc   -
%         wc   -
%         alph-
%         Per  -
%         D    -
%         dz   -
%         z    -
%         lamre-
%         kinf -
%         C    -
%

%    ***  Calculating Constants  ***

Aw=2*(hc+wc)*Z;
Af=hc*wc;
D=4*(wc*hc)/(2*(wc+hc));
Per=(2*(wc+hc));
if hc<wc
     alph=hc/wc;
elseif hc>=wc
     alph=wc/hc;
end
dz=Z/(round(Z/dzn));

%    ***  Creating Variable Arrays  ***

z=[0:dz:Z];
dz=ones(size(z))*(dz);
hc=ones(size(z))*hc;
wc=ones(size(z))*wc;
alph=ones(size(z))*alph;
Per=ones(size(z))*Per;
D=ones(size(z))*D;
[lamre,kinf,C]=Kinf(alph(1));
lamre=ones(size(z))*lamre;
kinf=ones(size(z))*kinf;
C=ones(size(z))*C;
jZ=length(z);
```

```
function  [G,Re,h,xi,lam]=rectflow_f(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)

%    function  [G,Re,h,xi,lam]=rectflow_f(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)
%
%    Calculates the flow parameters in single rectangular channel.
%
%    Inputs:
%         jZ   -
%         Tin  -
%         qin  -
%         Gin  -
%         hin  -
%         Aw   -
%         hc   -
%         wc   -
%         alph -
%         Per  -
%         D    -
%         dz   -
%         z    -
%         lamre-
%         kinf -
%         C    -
%         pdflg-
%         vpflg-    variable property flag 0=constant prop, 1=variable prop
%         dfflg-
%    Outputs:
%         G    -
%         Re   -
%         h    -
%         xi   -
%         lam  -
%

%
%    ***  Variable Array Creation/Initialization  ***
%

G=ones(size(z))*Gin;
h=ones(size(z));
xi=zeros(size(z));
lam=zeros(size(z));

xi(1)=eps;

Re=G.*D./muliq_w(Tin,vpflg);

h=h*hin;

%
%    ***  Friction Factor Calculation  ***
%
xi(2:jZ)=(z(2:jZ)./D(2:jZ))./Re(2:jZ);

if dfflg==0
    lam=64./Re;
else
    lam(2:jZ)=lamda_loc(xi(2:jZ),z(2:jZ),lamre(2:jZ),kinf(2:jZ),C(2:jZ),D(2:jZ),Re(2:jZ));
    lam(1)=lam(2)-5*((lam(3)-lam(2))/dz(2))*z(2);
end
```

```
function
[P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=rectPcal_w_f_2(jZ,Pi
n,Tin,qin,G,h,D,alph,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg)

%    function
[P,dPa,dPf,T,Ts,hsl,hsv,W,VF,phi,v,vl,vv,hfm,cnt,tpflg]=rectPcal_w_f(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,lam,tolT,dfflg,vfmo
d,vimod,vpflg)
%
%    Calculates the pressure along the rectangular channel from the end towards
%    the beginning based for water as the fluid.
%
%    Inputs:
%         jZ   -
%         Pout -
%         Wout -
%         G    -
%         h    -
%         D    -
%         dz   -
%         z    -
%         lam  -
%         dfflg-
%         vimod-
%         vpflg-
%    Outputs:
%         P    -
%         T    -
%         Ts   -
%         hs   -
%         W    -
%         VF   -
%         phi  -
%         v    -
%         vl   -
%         vv   -
%         mul  -
%         muv  -
%         mutp -
%         tpflg-

%
%    ***   Constants   ***
%
Pmin=70000;

%
%    ***  Array Initialization  ***
%

%    ***  Creation  ***
P=zeros(size(z));        T=zeros(size(z));
dPat=zeros(size(z));     dPft=zeros(size(z));
mul=zeros(size(z));      muv=ones(size(z))*NaN;
lamm=zeros(size(z));
vl=zeros(size(z));       vv=zeros(size(z));
v=zeros(size(z));
Ts=zeros(size(z));
hsl=zeros(size(z));      hsv=zeros(size(z));
W=zeros(size(z));        VF=zeros(size(z));
phi=ones(size(z));       phifl=ones(size(z));
hfm=ones(size(z));
Relo=zeros(size(z));
Rel=zeros(size(z));      Reg=zeros(size(z));
Gm=NaN*ones(size(z));
chiz=zeros(size(z));     chisq=zeros(size(z));

tpflg=zeros(size(z));
cnt=zeros(size(z));

%    ***  Initial Values  ***
P(1)=Pin;
T(1)=Tin;                Ts(1)=Tsw(Pin);
hsl(1)=hlw(Ts(1));
vl(1)=vlw(T(1));
v(1)=vl(1);
W(1)=0;
lamm(1)=lam(1);
%
%    ***  Calculations  ***
%

% bar=waitbar(0,['Percent of Channel Length Completed']);

if dfflg==1                                %  Developing Flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
```

```
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))                %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            mul(j)=muliq_w(T(j),vpflg);     muv(j)=muvap_w(T(j),vpflg);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*(Re(j)/Relo(j));
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
    end
elseif dfflg==0                             %  Fully Developed flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                   %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*Re(j)/Relo(j);
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))                %   Two Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            mul(j)=muliq_w(T(j),vpflg);     muv(j)=muvap_w(T(j),vpflg);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*Re(j)/Relo(j);
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
```

```
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*Re(j)/Relo(j);
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
    end
end

% close(bar)
```

```
function  [G,Re,h,xi,lam]=rectflow_f(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)

%    function  [G,Re,h,xi,lam]=rectflow_f(jZ,Tin,qin,Gin,hin,Aw,wc,hc,alph,Per,D,dz,z,lamre,kinf,C,pdflg,vpflg,dfflg)
%
%    Calculates the flow parameters in single rectangular channel.
%
%    Inputs:
%         jZ   -
%         Tin  -
%         qin  -
%         Gin  -
%         hin  -
%         Aw   -
%         hc   -
%         wc   -
%         alph -
%         Per  -
%         D    -
%         dz   -
%         z    -
%         lamre-
%         kinf -
%         C    -
%         pdflg-
%         vpflg-    variable property flag 0=constant prop, 1=variable prop
%         dfflg-
%    Outputs:
%         G    -
%         Re   -
%         h    -
%         xi   -
%         lam  -
%

%
%    ***  Variable Array Creation/Initialization  ***
%

G=ones(size(z))*Gin;
h=zeros(size(z));
xi=zeros(size(z));
lam=zeros(size(z));

xi(1)=eps;

Re=G.*D./muliq_w(Tin,vpflg);

for j=1:jZ
     if j==1
          h(j)=hin;
     else
          h(j)=h(j-1)+qin*Per(j)*dz(j)/(G(j)*(hc(j)*wc(j)));
     end
end

%
%    ***  Friction Factor Calculation  ***
%
xi(2:jZ)=(z(2:jZ)./D(2:jZ))./Re(2:jZ);

if dfflg==0
    lam=64./Re;
else
    lam(2:jZ)=lamda_loc(xi(2:jZ),z(2:jZ),lamre(2:jZ),kinf(2:jZ),C(2:jZ),D(2:jZ),Re(2:jZ));
    lam(1)=lam(2)-5*((lam(3)-lam(2))/dz(2))*z(2);
end
```

```
function
[P,dPat,dPft,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=rectPcal_w_f_2(jZ,Pi
n,Tin,qin,G,h,D,alph,dz,z,lam,Re,tolT,dfflg,vfmod,vimod,vpflg)

%    function
[P,dPa,dPf,T,Ts,hsl,hsv,W,VF,phi,v,vl,vv,hfm,cnt,tpflg]=rectPcal_w_f(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,lam,tolT,dfflg,vfmo
d,vimod,vpflg)
%
%    Calculates the pressure along the rectangular channel from the end towards
%    the beginning based for water as the fluid.
%
%    Inputs:
%         jZ   -
%         Pout -
%         Wout -
%         G    -
%         h    -
%         D    -
%         dz   -
%         z    -
%         lam  -
%         dfflg-
%         vimod-
%         vpflg-
%    Outputs:
%         P    -
%         T    -
%         Ts   -
%         hs   -
%         W    -
%         VF   -
%         phi  -
%         v    -
%         vl   -
%         vv   -
%         mul  -
%         muv  -
%         mutp -
%         tpflg-

%
%    ***   Constants   ***
%
Pmin=30000;

%
%    ***  Array Initialization  ***
%

%    ***  Creation  ***
P=zeros(size(z));        T=zeros(size(z));
dPat=zeros(size(z));     dPft=zeros(size(z));
mul=zeros(size(z));      muv=ones(size(z))*NaN;
lamm=zeros(size(z));
vl=zeros(size(z));       vv=zeros(size(z));
v=zeros(size(z));
Ts=zeros(size(z));
hsl=zeros(size(z));      hsv=zeros(size(z));
W=zeros(size(z));        VF=zeros(size(z));
phi=ones(size(z));       phifl=ones(size(z));
hfm=ones(size(z));
Relo=zeros(size(z));
Rel=zeros(size(z));      Reg=zeros(size(z));
Gm=NaN*ones(size(z));
chiz=zeros(size(z));     chisq=zeros(size(z));

tpflg=zeros(size(z));
cnt=zeros(size(z));

%    ***  Initial Values  ***
P(1)=Pin;
T(1)=Tin;                Ts(1)=Tsw(Pin);
hsl(1)=hlw(Ts(1));
vl(1)=vlw(T(1));
v(1)=vl(1);
W(1)=0;
lamm(1)=lam(1);
%
%    ***  Calculations  ***
%

% bar=waitbar(0,['Percent of Channel Length Completed']);

if dfflg==1                                %  Developing Flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
```

```
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))               %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            mul(j)=muliq_w(T(j),vpflg);     muv(j)=muvap_w(T(j),vpflg);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*(Re(j)/Relo(j));
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*(Re(j)/Relo(j));
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
    end
elseif dfflg==0                             %  Fully Developed flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                   %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j)*D(j)/mul(j);
            lamm(j)=lam(j)*Re(j)/Relo(j);
            dPat(j)=G(j)^2*(vl(j)-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))               %   Two Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            mul(j)=muliq_w(T(j),vpflg);     muv(j)=muvap_w(T(j),vpflg);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(j)=lam(j)*Re(j)/Relo(j);
            dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
            dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
```

```
            P(j)=P(1)-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(j)=lam(j)*Re(j)/Relo(j);
                dPat(j)=G(j)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(1));
                dPft(j)=(G(j)^2*dz(j)*trapz(phi(1:j).*vl(1:j).*lamm(1:j)))/(2*D(j));
                P(j)=P(1)-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
    end
end

% close(bar)
```

```
function [jZ,jk,Aw,Af,hc,wc,alph,Per,D,dz,z,Zk,lamre,kinf,C,lev]=fracgeom(dzn,Levs,hc,wct,Z,gama,betta,pdflg);

%    function [jZ,jk,Aw,Af,hc,wc,alph,Per,D,dz,z,Zk,lamre,kinf,C,lev]=fracgeom(dzn,Levs,hc,wct,Z,gama,betta,pdflg);
%
%    creates arrays containing information on the channel geometry for the
%    fractal-like channel network
%
%    Inputs:
%         N    -    Number of divisions per branching level
%         dzn  -    Nominal discritization lenght
%         Levs -    Number of branching levels (1st level is 0th)
%         hc   -    Height of channels [m]
%         wct  -    Terminal channel width [m]
%         Z    -    Total lenght of channels [m]
%         gama -    Ratio of branching level lengths
%         betta -    Ratio of branching level hydraulic diamters
%         pdflg-    Stepping direction flag
%   Outputs:
%         jZ   -    Index of end of axial position array
%         jk   -    Array of index of branching level changes
%         Aw   -    Total fractal channel wall surface area
%         Af   -
%         hc   -    Channel height array
%         wc   -    Channel width array
%         alph-    Aspect ratio array
%         Per  -    Channel perimeter array
%         D    -    Hydraulic diameter array
%         dz   -    Discretization distance array
%         z    -    Axial position array
%         Zk   -    Array of total channel length by branching level
%         lamre-
%         kinf -    Developing flow parameter array
%         C    -    Developing flow parameter array
%         lev  -    Branching level array
%    Intermediate Variables
%         wk   -    Array of channel widths by branching level
%         Dk   -    Array of channel hydraulic diameters by branching level
%         Perk -    Array of channel perimeter by branching level
%         Lenk -    Array of branching channel lengths by branching level
%         Awk  -    Array of flow area per branching level
%         Sgama-
%

%
%  ***   Channel Dimensions   ***
%
% jZ=(N*(Levs+1))+1;

%     ***   Initializing Variables   ***
% dz=zeros(1,jZ);     hc=ones(1,jZ)*hc;
% wc=zeros(1,jZ);     Per=zeros(1,jZ);
% D=zeros(1,jZ);      alph=zeros(1,jZ);
% kinf=zeros(1,jZ);   C=zeros(1,jZ);
% lev=zeros(1,jZ);

Dk=ones(1,Levs+1);       wk=ones(1,Levs+1);
alpk=ones(1,Levs+1);     dzk=ones(1,Levs+1);
Perk=zeros(1,Levs+1);    Lenk=ones(1,Levs+1);
Awk=zeros(1,Levs+1);     Zk=zeros(1,Levs+2);

Sgama=0;
for k=0:Levs
     Sgama=Sgama+(1/(gama^k));
end

for k=Levs+1:-1:1
     if k==Levs+1
          Lenk(k)=Z/Sgama;    wk(k)=wct;
          Dk(k)=(4*wk(k)*hc(1))/(2*(wk(k)+hc(1)));
     else
          Lenk(k)=Lenk(k+1)/gama;  wk(k)=wk(k+1)/betta;
          Dk(k)=(4*wk(k)*hc(1))/(2*(wk(k)+hc(1)));
     end

     dzk(k)=Lenk(k)/(round(Lenk(k)/dzn));
%      dzk(k)=Lenk(k)/N;

     Perk(k)=2*(hc(1)+wk(k));
     Awk(k)=Lenk(k)*Perk(k)*2^(k-1);

     Nk(k)=Lenk(k)/dzk(k);
     if wk(k)<=hc(1)
          alpk(k)=wk(k)/hc(1);
     elseif wk(k)>hc(1)
          alpk(k)=hc(1)/wk(k);
     end
end

for k=1:Levs+1
     Zk(k+1)=sum(Lenk(1:k));
     if k==1
```

```
          z=[Zk(k):dzk(k):Zk(k+1)];
     else
          z=[z Zk(k)+dzk(k):dzk(k):Zk(k+1)];
     end
end

jZ=length(z);
dz=zeros(1,jZ);     hc=ones(1,jZ)*hc;
wc=zeros(1,jZ);     Per=zeros(1,jZ);
D=zeros(1,jZ);      alph=zeros(1,jZ);
kinf=zeros(1,jZ);   C=zeros(1,jZ);
lev=zeros(1,jZ);


Aw=sum(Awk);
jk(1)=1;
if pdflg==0              %    Forward propagation
     lev(jZ)=Levs;
     for k=1:Levs+1
          for j=1:jZ-1
               if z(j)>=Zk(k) & z(j)<Zk(k+1)
                    lev(j)=k-1;
                    jk(k+1)=j+1;
               end
          end
     end
end
D=Dk(lev+1);        wc=wk(lev+1);
Per=Perk(lev+1);    dz=dzk(lev+1);
alph=alpk(lev+1);  zref=Zk(lev+1);

[lamre,kinf,C]=Kinf(alph);

Af=wc(1)*hc(1);
```

```
function [G,Re,h,xi,lam]=fracflow_adb(jZ,jk,Tin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,lamre,kinf,C,pdflg,vpflg,dfflg)

%    function [G,Re,h,xi,lam]=fracflow(jZ,Tin,Gin,hin,Aw,wc,hc,D,Per,alph,dz,z,Zk,lev,kinf,C,pdflg,vpflg,dfflg)
%
%    Calculates the flow parameters in each branching level of the
%    fractal-like channel network.
%
%    Inputs:
%         jZ   -
%         jk   -
%         Tin  -    Inlet fluid temperature
%         qin  -    Wall heat flux
%         hin  -    Enthalpy of inlet fluid
%         hout -    Enthalpy of outlet fluid
%         Aw   -    Total wall surface area
%         wc   -    Array of channel width
%         hc   -    Array of channel height
%         D    -    Array of channel hydraulic diameter
%         Per  -    Array of channel perimeter
%         alph-    Array of aspect ratio
%         dz   -    Array discretization distance
%         z    -    Array of axial position
%         Zk   -
%         lev  -    Array of branching level
%         kinf -
%         C    -
%         pdflg-
%         vpflg-    variable property flag 0=constant prop, 1=variable prop
%         dfflg-
%    Outputs:
%         G    -    Mass flux Array
%         Re   -    Reynolds Number Array
%         h    -    fluid enthalpy array
%         xi   -    Array of dimensionless distance from beginning of branching level
%         lam  -    Friction factor array
%

%
%  ***   Variable Array Creation/Initialization   ***
%

G=zeros(size(z));
Re=zeros(size(z));
h=ones(size(z));
xi=zeros(size(z));
lam=zeros(lev(jZ)+1,length(z));

xi(1)=eps;

G=Gin*(wc(2)*hc(1))./((2.^lev).*(wc.*hc));
Re=G.*D./muliq_w(Tin,vpflg);

h=h*hin;

%
%  ***   Friction Factor Calculations   ***
%

xi(2:jZ)=((z(2:jZ)-Zk(lev(1:jZ-1)+1))./D(1:jZ-1))./Re(1:jZ-1);

for k=1:lev(jZ)+1
    if dfflg==0
        lam(k,jk(k)+1:jk(k+1))=lamre(jk(k)+1:jk(k+1))./Re(jk(k)+1:jk(k+1));
    else
        lam(k,jk(k)+1:jk(k+1))=lamda_loc(xi(jk(k)+1:jk(k+1)),z(jk(k)+1:jk(k+1))-
Zk(lev(jk(k)+1:jk(k+1))+1),lamre(jk(k)+1:jk(k+1)),kinf(jk(k)+1:jk(k+1)),C(jk(k)+1:jk(k+1)),D(jk(k)+1:jk(k+1)),Re(jk(k)+
1:jk(k+1)));
        lam(k,jk(k))=lam(k,jk(k)+1)-((lam(k,jk(k)+2)-lam(k,jk(k)+1))/dz(jk(k)+1))*(z(jk(k)+1)-Zk(k));
    end
end
```

```
function
[P,dPat,dPft,dPr,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=fracPcal_w_f_2(j
Z,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,Re,tolT,lev,dfflg,vfmod,vimod,vpflg);

%    function
[P,dPa,dPf,dPr,T,Ts,hs,W,VF,phi,v,vl,vv,hfm,cnt,tpflg]=fracPcal_w_f(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,tolT,lev,dffl
g,vimod,vpflg)
%
%    Calculates the pressure along the fractal channel from the end towards
%    the beginning based for water as the fluid.
%
%    Inputs:
%         jZ   -
%         Pout -
%         Wout -
%         G    -
%         h    -
%         D    -
%         dz   -
%         z    -
%         Zk   -
%         lam  -
%         lev  -
%         dfflg-
%         vimod-
%         vpflg-
%    Outputs:
%         P    -
%         T    -
%         Ts   -
%         hs   -
%         W    -
%         VF   -
%         phi  -
%         v    -
%         vl   -
%         vv   -
%         mul  -
%         muv  -
%         mutp -
%         tpflg-
%

%
%    ***   Constants   ***
%
Clc=.6;
Pmin=50000;

%
%    ***  Array Initialization  ***
%

%    ***  Creation  ***
P=zeros(size(z));        T=zeros(size(z));
dPat=zeros(size(z));     dPft=zeros(size(z));
dPr=zeros(size(z));
mul=zeros(size(z));      muv=zeros(size(z));
lamm=lam;
vl=zeros(size(z));       vv=zeros(size(z));
v=zeros(size(z));
Ts=zeros(size(z));
hsl=zeros(size(z));      hsv=zeros(size(z));
W=zeros(size(z));        VF=zeros(size(z));
phi=ones(size(z));       phifl=ones(size(z));
hfm=ones(size(z));
Relo=zeros(size(z));
Rel=zeros(size(z));      Reg=zeros(size(z));
Gm=NaN*ones(size(z));
chiz=zeros(size(z));     chisq=zeros(size(z));

tpflg=zeros(size(z));
cnt=zeros(size(z));

%    ***  Initial Values  ***
P(1)=Pin;
T(1)=Tin;
Ts(1)=Tsw(Pin);
hsl(1)=hlw(Ts(1));
vl(1)=vlw(T(1));
v(1)=vl(1);
W(1)=0;

Pref=P(1);
vlref=vl(1);
jref=1;
k=1;

%
%    ***  Calculations  ***
%
```

```
% bar=waitbar(0,['Percent of Channel Length Completed']);

if dfflg==1                                %  Developing Flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j-1)*D(j-1)/mul(j);
            lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
            dPat(j)=G(j-1)^2*(vl(j)-vlref);
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))               %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
            if VF(jref)==0
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
            else
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                    -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
            end
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
                if VF(jref)==0
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
                else
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                        -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
                end
                dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
                P(j)=Pref-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
        if lev(j-1)~=lev(j)
            dPr(j)=Clc*(G(j)^2-G(j-1)^2)*vl(j);
            % P(j)=P(j)-dPr(j);
            Pref=P(j);
            vlref=vl(j);
            jref=j;
            k=k+1;
        end
    end
elseif dfflg==0                            %  Fully Developed flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
```

```
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j-1)*D(j-1)/mul(j);
            lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
            dPat(j)=G(j-1)^2*(vl(j)-vlref);
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))                %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            Relo(j)=G(j-1)*D(j-1)/mul(j);
            if VF(jref)==0
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
            else
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                    -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
            end
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                Relo(j)=G(j-1)*D(j-1)/mul(j);
                if VF(jref)==0
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
                else
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                        -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
                end
                dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
                P(j)=Pref-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
        if lev(j-1)~=lev(j)
            dPr(j)=Clc*(G(j)^2-G(j-1)^2)*vl(j);
            % P(j)=P(j)-dPr(j);
            Pref=P(j);
            vlref=vl(j);
            jref=j;
            k=k+1;
        end
    end
end

% close(bar)
```

```
function
[G,Re,h,xi,lam]=fracflow_f(jZ,jk,Tin,qin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,lamre,kinf,C,pdflg,vpflg,dfflg)

%    function
[G,Re,h,xi,lam]=fracflow_f(jZ,jk,Tin,qin,hin,Gin,Aw,hc,wc,D,Per,alph,dz,z,Zk,lev,kinf,C,pdflg,vpflg,dfflg)
%
%    Calculates the flow parameters in each branching level of the
%    fractal-like channel network.
%
%    Inputs:
%         jZ   -
%         jk   -
%         Tin  -    Inlet fluid temperature
%         qin  -    Wall heat flux
%         hin  -    Enthalpy of inlet fluid
%         hout -    Enthalpy of outlet fluid
%         Aw   -    Total wall surface area
%         wc   -    Array of channel width
%         hc   -    Array of channel height
%         D    -    Array of channel hydraulic diameter
%         Per  -    Array of channel perimeter
%         alph-    Array of aspect ratio
%         dz   -    Array discretization distance
%         z    -    Array of axial position
%         Zk   -
%         lev  -    Array of branching level
%         kinf -
%         C    -
%         pdflg-
%         vpflg-    variable property flag 0=constant prop, 1=variable prop
%         dfflg-
%    Outputs:
%         G    -    Mass flux Array
%         Re   -    Reynolds Number Array
%         h    -    fluid enthalpy array
%         xi   -    Array of dimensionless distance from beginning of branching level
%         lam  -    Friction factor array
%

%
%  ***   Variable Array Creation/Initialization   ***
%

G=zeros(size(z));
Re=zeros(size(z));
h=ones(size(z));
xi=zeros(size(z));
lam=zeros(lev(jZ)+1,length(z));

xi(1)=eps;

G=Gin*(wc(1)*hc(1))./((2.^lev).*(wc.*hc));
Re=G.*D./muliq_w(Tin,vpflg);

for j=1:jZ
     if j==1
          h(j)=hin;
     else
          h(j)=h(j-1)+qin*Per(j)*dz(j)/(G(j)*hc(j)*wc(j));
     end
end

%
%  ***   Friction Factor Calculations   ***
%

xi(2:jZ)=((z(2:jZ)-Zk(lev(1:jZ-1)+1))./D(1:jZ-1))./Re(1:jZ-1);

for k=1:lev(jZ)+1
    if dfflg==0
        lam(k,jk(k)+1:jk(k+1))=lamre(jk(k)+1:jk(k+1))./Re(jk(k)+1:jk(k+1));
    else
        lam(k,jk(k)+1:jk(k+1))=lamda_loc(xi(jk(k)+1:jk(k+1)),z(jk(k)+1:jk(k+1))-
Zk(lev(jk(k)+1:jk(k+1))+1),lamre(jk(k)+1:jk(k+1)),kinf(jk(k)+1:jk(k+1)),C(jk(k)+1:jk(k+1)),D(jk(k)+1:jk(k+1)),Re(jk(k)+
1:jk(k+1)));
        lam(k,jk(k))=lam(k,jk(k)+1)-((lam(k,jk(k)+2)-lam(k,jk(k)+1))/dz(jk(k)+1))*(z(jk(k)+1)-Zk(k));
    end
end
```

```
function
[P,dPat,dPft,dPr,T,Ts,hsl,hsv,W,VF,Gm,phi,phifl,chisq,Relo,Rel,Reg,v,vl,vv,mul,muv,lamm,hfm,cnt,tpflg]=fracPcal_w_f_2(j
Z,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,Re,tolT,lev,dfflg,vfmod,vimod,vpflg);

%    function
[P,dPa,dPf,dPr,T,Ts,hs,W,VF,phi,v,vl,vv,hfm,cnt,tpflg]=fracPcal_w_f(jZ,Pin,Tin,qin,G,h,D,alph,dz,z,Zk,lam,tolT,lev,dffl
g,vimod,vpflg)
%
%    Calculates the pressure along the fractal channel from the end towards
%    the beginning based for water as the fluid.
%
%    Inputs:
%         jZ   -
%         Pout -
%         Wout -
%         G    -
%         h    -
%         D    -
%         dz   -
%         z    -
%         Zk   -
%         lam  -
%         lev  -
%         dfflg-
%         vimod-
%         vpflg-
%    Outputs:
%         P    -
%         T    -
%         Ts   -
%         hs   -
%         W    -
%         VF   -
%         phi  -
%         v    -
%         vl   -
%         vv   -
%         mul  -
%         muv  -
%         mutp -
%         tpflg-
%

%
%    ***   Constants   ***
%
Clc=.6;
Pmin=50000;

%
%    ***  Array Initialization  ***
%

%    ***  Creation  ***
P=zeros(size(z));        T=zeros(size(z));
dPat=zeros(size(z));     dPft=zeros(size(z));
dPr=zeros(size(z));
mul=zeros(size(z));      muv=zeros(size(z));
lamm=lam;
vl=zeros(size(z));       vv=zeros(size(z));
v=zeros(size(z));
Ts=zeros(size(z));
hsl=zeros(size(z));      hsv=zeros(size(z));
W=zeros(size(z));        VF=zeros(size(z));
phi=ones(size(z));       phifl=ones(size(z));
hfm=ones(size(z));
Relo=zeros(size(z));
Rel=zeros(size(z));      Reg=zeros(size(z));
Gm=NaN*ones(size(z));
chiz=zeros(size(z));     chisq=zeros(size(z));

tpflg=zeros(size(z));
cnt=zeros(size(z));

%    ***  Initial Values  ***
P(1)=Pin;
T(1)=Tin;
Ts(1)=Tsw(Pin);
hsl(1)=hlw(Ts(1));
vl(1)=vlw(T(1));
v(1)=vl(1);
W(1)=0;

Pref=P(1);
vlref=vl(1);
jref=1;
k=1;

%
%    ***  Calculations  ***
%
```

```
% bar=waitbar(0,['Percent of Channel Length Completed']);

if dfflg==1                              %  Developing Flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j-1)*D(j-1)/mul(j);
            lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
            dPat(j)=G(j-1)^2*(vl(j)-vlref);
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))               %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
            if VF(jref)==0
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
            else
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                    -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
            end
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
                if VF(jref)==0
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
                else
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                        -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
                end
                dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
                P(j)=Pref-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
            if lev(j-1)~=lev(j)
                dPr(j)=Clc*(G(j)^2-G(j-1)^2)*vl(j);
                % P(j)=P(j)-dPr(j);
                Pref=P(j);
                vlref=vl(j);
                jref=j;
                k=k+1;
            end
    end
elseif dfflg==0                          %  Fully Developed flow
    for j=2:jZ
%         waitbar(j/jZ);
        if h(j)<=hlw(Tsw(P(j-1)))                  %  Single Phase Flow
            W(j)=0;
```

```
            y(j)=1;
            T(j)=fzero('h2T_w_2',T(j-1),[],h(j));
            vl(j)=Vlw(T(j));    v(j)=vl(j);
            mul(j)=muliq_w(T(j),vpflg);
            Relo(j)=G(j-1)*D(j-1)/mul(j);
            lamm(k,j)=lam(k,j)*(Re(j-1)/Relo(j));
            dPat(j)=G(j-1)^2*(vl(j)-vlref);
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            Ts(j)=Tsw(P(j));
            hsl(j)=hlw(Ts(j));  hsv(j)=hvw(Ts(j));
        elseif h(j)>hlw(Tsw(P(j-1)))                %   Two-Phase Flow
            tpflg(j)=3;
            T(j)=Tsw(P(j-1));   Ts(j)=T(j);
            W(j)=max(0,(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j))));
            vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
            v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
            VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
            hfm(j)=dia_mult(qin,G(j));
            Relo(j)=G(j-1)*D(j-1)/mul(j);
            if VF(jref)==0
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
            else
                dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                    -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
            end
            dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
            P(j)=Pref-dPft(j)-dPat(j);
            if P(j)<=Pmin;
                P(j)=Pmin;
            end
            dT=abs(T(j)-Tsw(P(j)));
            cnt(j)=cnt(j)+1;
            while dT>tolT
                T(j)=Tsw(P(j));     Ts(j)=T(j);
                W(j)=(h(j)-hlw(T(j)))/(hvw(T(j))-hlw(T(j)));
                vl(j)=vlw(T(j));    vv(j)=vvw(T(j));
                v(j)=(1-W(j))*vl(j)+W(j)*vv(j);
                VF(j)=Voidfrac(W(j),vl(j),vv(j),T(j),vfmod);

[phi(j),phifl(j),chisq(j),Relo(j),Rel(j),Reg(j)]=phisq_w(vl(j),vv(j),T(j),W(j),G(j),D(j),lam(j),Re(j),vimod,vpflg);
                hfm(j)=dia_mult(qin,G(j));
                Relo(j)=G(j-1)*D(j-1)/mul(j);
                if VF(jref)==0
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))-vl(jref));
                else
                    dPat(j)=G(j-1)^2*((W(j)^2*vv(j)/VF(j))+((1-W(j))^2*vl(j)/(1-VF(j)))...
                        -((W(jref)^2*vv(jref)/VF(jref))+((1-W(jref))^2*vl(jref)/(1-VF(jref)))));
                end
                dPft(j)=(G(j-1)^2*dz(j-1)*trapz(phi(jref:j).*vl(jref:j).*lamm(k,jref:j)))/(2*D(j-1));
                P(j)=Pref-dPft(j)-dPat(j);
                if P(j)<=Pmin;
                    P(j)=Pmin;
                end
                dT=abs(T(j)-Tsw(P(j)));
                cnt(j)=cnt(j)+1;
                if cnt(j)==600
                    disp('Exceeded Temp interation count')
                    break
                end
            end
            hsl(j)=hlw(Ts(j));      hsv(j)=hvw(Ts(j));
            Gm(j)=sqrt(-1/((1-W(j))/dPdvl(P(j))+W(j)/dPdvv(P(j))));
        end
        if lev(j-1)~=lev(j)
            dPr(j)=Clc*(G(j)^2-G(j-1)^2)*vl(j);
            % P(j)=P(j)-dPr(j);
            Pref=P(j);
            vlref=vl(j);
            jref=j;
            k=k+1;
        end
    end
end

% close(bar)
```

```
%    resprt3
%
%    Performs initial Calc's of results from pressure drop model and Prints them out
%
%
%

Pout=P(end);

if vimod==1
     fricmod='Homogenous per Lin et.al.';
elseif vimod==2
     fricmod='Homogenous per McAdams';
elseif vimod==3
     fricmod='Homogenous per Cicchitti';
elseif vimod==4
     fricmod='Homogemous per Dukler';
elseif vimod==5
     fricmod='Homogemous per Stanley et al.';
elseif vimod==6
     fricmod='Homogenous per Collier';
elseif vimod==7
     fricmod='Separated per Lockhart & Martinelli';
elseif vimod==8
     fricmod='Separated per Mishima & Hibiki';
elseif vimod==9
     fricmod='Separated per Lee & Lee';
elseif vimod==10
     fricmod='Separated per Qu & Mudawar';
end

if vfmod==0
     acelmod='Homogenous';
elseif vfmod==1
     acelmod='Separated - V.F. per chung et al.';
elseif vfmod==2
     acelmod='Separated - V.F. per Armaand';
elseif vfmod==3
     acelmod='Separated - V.F. per Lockhart & Martinelli';
elseif vfmod==4
     acelmod='Separated - V.F. per Zivi';
end

if chmod==3
     if pdflg==0
          stitl='Fractal-like branching network - Forward solution';
     elseif pdflg==1
          stitl='Fractal-like branching network - Backward solution';
     end
%      [No,rr,spcng,re,ri]=frachsgeom(Z,hcin,gama,betta,Levs,wct);
          rr=NaN; spcng=NaN; re=NaN; ri=NaN;
     DP=(P(1)-P(end));
     dPa(1)=dPat(1); dPf(1)=dPft(1);
     dPa(2)=dPat(2); dPf(2)=dPft(2);
     for j=3:length(z);
         if lev(j-2)~=lev(j-1)
             dPa(j)=dPat(j); dPf(j)=dPft(j);
         else
             dPa(j)=dPat(j)-dPat(j-1);
             dPf(j)=dPft(j)-dPft(j-1);
         end
     end
     mdot=G(1)*hc(1)*wc(1);
     Vdot=mdot*v(1);
     Tin=T(1);
     Wout=W(end); VFend=VF(end);
     uv=NaN;
%      uv=G(end)*Wout*vv(end)/VF(end);
     ul=G(end)*(1-Wout)*vl(end)/(1-VF(end));
elseif chmod==2
     if pdflg==0
          stitl='Single Rectangular Channel - Forward solution';
     elseif pdflg==1
          stitl='Single Rectangular Channel - Backward solution';
     end
     DP=(P(1)-P(end));
     for j=1:length(z);
         if j==1
             dPa(j)=dPat(j); dPf(j)=0;
         else
             dPa(j)=dPat(j)-dPat(j-1);
             dPf(j)=dPft(j)-dPft(j-1);
         end
     end
     mdot=G(1)*hc(1)*wc(1);
     Vdot=mdot*v(1);
     Tin=T(1);
     Wout=W(end); VFend=VF(end);
     uv=G(end)*Wout*vv(end)/VF(end);
     ul=G(end)*(1-Wout)*vl(end)/(1-VF(end));
elseif chmod==1
```

```
     if pdflg==0
          stitl='Single Circular Channel - Forward solution';
     elseif pdflg==1
          stitl='Single Circular Channel - Backward solution';
     end
     DP=(P(1)-P(end));
     for j=1:length(z);
         if j==1
             dPa(j)=0; dPf(j)=0;
         else
             dPa(j)=dPat(j)-dPat(j-1);
             dPf(j)=dPft(j)-dPft(j-1);
         end
     end
     mdot=G(1)*pi*D(1)^2;
     Vdot=mdot*v(1);
     Tin=T(1);
     Wout=W(end); VFend=VF(end);
     uv=G(end)*Wout*vv(end)/VF(end);
     ul=G(end)*(1-Wout)*vl(end)/(1-VF(end));
end

if chmod==3
     %    Standard Output
%      disp(sprintf('\n'))
%      disp(stitl)
%      disp(['Friction Model: ' fricmod ])
%      disp(['Acceleration Model: ' acelmod])
%      disp(sprintf('\nInlet Mass Flux = %5.1f kg/m^2-s;   Nominal Grid = %5.1f micron' ,G(1), dzn*1e6))
%      disp(sprintf('Wall Heat Flux = %4.1f W/cm^2;   Inlet Sub-Cooling = %3.1f Deg C',qin_0, Tsub))
%      disp(sprintf('Friction Pressure Drop = %6.2f kPa;   \nAcceleration Pressure Drop = %6.2f kPa;   \nTotal Pressure
Recovery = %6.2f kPa',sum(dPf)/1000,sum(dPa)/1000,sum(dPr)/1000))
%      disp(sprintf('Pout = %5.2f kPa;   Ts_out - Tout = %3.1f Deg C;',P(end)/1000,Ts(end)-T(end)))
%      disp(sprintf('\nExit Quality = %5.4f;   \nPressure Drop = %6.3f kPa;   Pin = %6.3f kPa;   Tin = %5.2f Deg
C;',W(end),(P(1)-P(end))/1000, P(1)/1000, T(1)-273.15))
%      disp(sprintf('Elapsed time: %2.0f minutes, %4.2f seconds\n',floor(etime(t2,t1)/60),etime(t2,t1)-
floor(etime(t2,t1)/60)*60))

     %    Vinod's Optimiztion Output
     disp(sprintf('\n'))
     disp(['Friction Model: ' fricmod ])
     disp(['Acceleration Model: ' acelmod])
     disp(sprintf('\nNominal Step Size: %6.0f micron', dzn*1e6))
     disp(sprintf('Inlet Mass Flux = %5.1f kg/m^2-s;   Wall Heat Flux = %5.1f W/cm^2' ,G(1), qin_0))
     disp(sprintf('Gamma = %5.4f;     Beta = %5.4f;     Levels = %1.0f; ',gama,betta,Levs))
     disp(sprintf('hc_in = %5.0f microns;     w_ct = %5.0f microns; ', hcin*1e6, wct*1e6))
     disp(sprintf('\nPressure Drop = %6.3f kPa;     dPf = %6.3f kPa;  dPa = %6.3f kPa;  dPr = %6.3f kPa; ',DP/1000,
sum(dPf)/1000, sum(dPa)/1000, sum(dPr)/1000))
     disp(sprintf('\nExit Quality = %5.4f;   Exit VF = %5.4f;   T_in = %5.2f Deg C;   Tw_max = %5.2f Deg C; ', Wout,
VFend, Tin-273.15, max(Tw)-273.15))
     disp(sprintf('\nm_dot = %6.3f g/s; Pumping Power = %6.3f mW; Wall Area = %6.3f mm^2; Disk Area = %6.1f mm^2',
mdot*1000, Vdot*DP*1000, Aw*1e6, pi*re^2*1e6))
     disp(sprintf('Performance Parameter = %6.1f; ',qin*Aw/(Vdot*DP)))
     disp(sprintf('Liquid Velocity = %7.2f m/s,  Vapor Velocity = %7.2f m/s', ul, uv))
     disp(sprintf('\nNo = %3.0f;    Disk Radius = %5.2f mm     Ri/Ro = %5.4f;     Perimeter Channel Spacing = %3.2f;
\n', No, re*1000, rr, spcng))

elseif chmod==2
% %         Standard Output
%      disp(sprintf('\n'))
%      disp(stitl)
%      disp(['Friction Model: ' fricmod ])
%      disp(['Acceleration Model: ' acelmod])
%      disp(sprintf('\nInlet Mass Flux = %5.1f kg/m^2-s;   Nominal Grid = %5.1f micron' ,G(1), dzn*1e6))
%      disp(sprintf('Wall Heat Flux = %4.1f W/cm^2;   Inlet Sub-Cooling = %3.1f Deg C',qin_0, Tsub))
%      disp(sprintf('Friction Pressure Drop = %6.2f kPa;   \nAcceleration Pressure Drop = %6.2f kPa;',sum(dPf)/1000,
sum(dPa)/1000))
%      disp(sprintf('Pout = %5.2f kPa;   Ts_out - Tout = %3.1f Deg C;',P(end)/1000,Ts(end)-T(end)))
%      disp(sprintf('\nExit Quality = %5.4f;   \nPressure Drop = %6.3f kPa;   Pin = %6.3f kPa;   Tin = %5.2f Deg
C;',W(end),(P(1)-P(end))/1000, P(1)/1000, T(1)-273.15))
%      disp(sprintf('Elapsed time: %2.0f minutes, %4.2f seconds\n',floor(etime(t2,t1)/60),etime(t2,t1)-
floor(etime(t2,t1)/60)*60))

     %         Detailed Data Sheet Output
     disp(sprintf('\n'))
     disp(['Friction Model: ' fricmod ])
     disp(['Acceleration Model: ' acelmod])
     disp(sprintf('\nNominal Step Size: %4.1f micron', dzn*1e6))
     disp(sprintf('Inlet Mass Flux = %6.2f kg/m^2-s;   Wall Heat Flux = %6.2f W/cm^2' ,G(1), qin_0))
     disp(sprintf('wc = %4.1f micron;  hc = %4.1f micron; Ltot = %4.1f mm; ',hc(1)*1e6, wc(1)*1e6, Z*1000))
     disp(sprintf('\nPressure Drop = %6.3f kPa; dPf = %6.3f kPa; dPa = %6.3f kPa; ',DP/1000, sum(dPf)/1000,
sum(dPa)/1000))
     if quf==1
          disp(sprintf('Corrections:  contraction = %6.3f kPa;  expansion = %6.3f kPa;  Total = %6.3f
kPa',(dPc(1)+dPc(2))/1000, (dPc(3)+dPc(4))/1000, sum(dPc)/1000))
     end
%      disp(sprintf('\nExit Quality = %5.4f;   T_in = %5.2f Deg C;   Tw_max = %5.2f Deg C; ', Wout, Tin-273.15,
max(Tw)-273.15))
     disp(sprintf('\nExit Quality = %5.4f;   Exit VF = %5.4f   T_in = %5.2f Deg C;   ', Wout, VFend, Tin-273.15))
     disp(sprintf('\nm_dot = %6.4f g/s; Pumping Power = %6.3f mW; Wall Area = %6.3f mm^2;', mdot*1000, Vdot*DP*1000,
Aw*1e6))
```

```
     disp(sprintf('Performance Parameter = %6.1f; ',qin*Aw/(Vdot*DP)))
     disp(sprintf('Elapsed time: %2.0f minutes, %4.2f seconds\n',floor(etime(t2,t1)/60),etime(t2,t1)-
floor(etime(t2,t1)/60)*60))


elseif chmod==1
%         Standard Output
     disp(sprintf('\n'))
     disp(stitl)
     disp(['Friction Model: ' fricmod ])
     disp(['Acceleration Model: ' acelmod])
     disp(sprintf('\nNominal Step Size: %6.0f micron', dzn*1e6))
     disp(sprintf('Inlet Mass Flux = %5.1f kg/m^2-s;   Nominal Grid = %5.1f micron' ,G(1), dzn*1e6))
     disp(sprintf('Wall Heat Flux = %4.1f W/cm^2;   Inlet Sub-Cooling = %3.1f Deg C',qin_0, Tsub))
     disp(sprintf('\nPressure Drop = %6.3f kPa; dPf = %6.3f kPa; dPa = %6.3f kPa; ',DP/1000, sum(dPf)/1000,
sum(dPa)/1000))
     disp(sprintf('Pout = %5.2f kPa;   Ts_out - Tout = %3.1f Deg C;',P(end)/1000,Ts(end)-T(end)))
     disp(sprintf('\nExit Quality = %5.4f;   Exit VF = %5.4f   \nPressure Drop = %6.3f kPa;   Pin = %6.3f kPa;   Tin =
%5.2f Deg C;',W(end),VFend,(P(1)-P(end))/1000, P(1)/1000, T(1)-273.15))
     disp(sprintf('Elapsed time: %2.0f minutes, %4.2f seconds\n',floor(etime(t2,t1)/60),etime(t2,t1)-
floor(etime(t2,t1)/60)*60))

end
```

```
%    ressave3.m
%
%    Saves results to the User Specified file name
%
%
%
%
%
%
%


if chmod==1
    if bmflg==1
        if condn==1
            fprintf(fid,'\nFile Name:, %s',resfile);
            fprintf(fid,'\n%s',stitl);
            fprintf(fid,'\nFriction Mode:, %s',fricmod);
            fprintf(fid,'\nAcceleration Mode:, %s',acelmod);
            fprintf(fid,'\nTemperature tolerance:, %4.3f, Deg C \nPressure tolerance:, %5.1f, Pa',tolT,tolP);
            fprintf(fid,'\n,,,D,L,N_0,dzn,Pout,Pin_i,Tsub,NC flag,Elapsed Time,
,Gin,q_wall,DP,dPf,dPa,Wout,VFout,Pin,T_in,mdot,Vdot,PP,Aw,epsilon,ul,uv,Padj');
            fprintf(fid,'\n,,,[micron],[mm], ,[micron],[kPa],[kPa],[Deg C], ,[s], ,[kg/m^2-
s],[W/cm^2],[kPa],[kPa],[kPa], , ,[kPa],[Deg C],[g/s],[ml/s],[mW],[mm^2], , ,[Pa] ');

fprintf(fid,'\n,,,%5.1f,%5.2f,%3.0f,%5.1f,%6.2f,%6.2f,%3.1f,%1.0f,%8.2f,%s,%8.4f,%6.2f,%6.3f,%6.3f,%6.3f,%5.4f,%5.4f,%6
.3f,%5.2f,%6.4f,%6.4f,%6.3f,%6.3f,%6.1f,%6.3f,%6.3f,%6.3f',...
                D(1)*1e6,Z*1000,N0,dzn*1e6,P(end)/1000,Pin_i,Tsub,NCflg,etime(t2,t1),'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[D(1)*1e6,N0,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/10
00,...
                    Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
        else

fprintf(fid,'\n,,,%5.1f,%5.2f,%3.0f,%5.1f,%6.2f,%6.2f,%3.1f,%1.0f,%8.2f,%s,%8.4f,%6.2f,%6.3f,%6.3f,%6.3f,%5.4f,%5.4f,%6
.3f,%5.2f,%6.4f,%6.4f,%6.3f,%6.3f,%6.1f,%6.3f,%6.3f,%6.3f',...
                D(1)*1e6,Z*1000,N0,dzn*1e6,P(end)/1000,Pin_i,Tsub,NCflg,etime(t2,t1),'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[D(1)*1e6,N0,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/10
00,...
                    Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
        end
    elseif bmflg==0
        if condn==1
            fprintf(fid,'\nFile Name:, %s',resfile);
            fprintf(fid,'\n%s',stitl);
            fprintf(fid,'\nFriction Mode:, %s',fricmod);
            fprintf(fid,'\nAcceleration Mode:, %s',acelmod);
            fprintf(fid,'\nTemperature tolerance:, %4.3f, Deg C \nPressure tolerance:, %5.1f, Pa',tolT,tolP);
            fprintf(fid,'\n,,,D,L,N_0,dzn,Pout,Pin_i,Tsub,NC flag,Elapsed Time,
,Gin,q_wall,DP,dPf,dPa,Wout,VFout,Pin,T_in,mdot,Vdot,PP,Aw,epsilon,ul,uv,Padj');
            fprintf(fid,'\n,,,[micron],[mm], ,[micron],[kPa],[kPa],[Deg C], ,[s], ,[kg/m^2-
s],[W/cm^2],[kPa],[kPa],[kPa], , ,[kPa],[Deg C],[g/s],[ml/s],[mW],[mm^2], , ,[Pa], ');

fprintf(fid,'\n,,,%5.1f,%5.2f,%3.0f,%5.1f,%6.2f,%6.2f,%3.1f,%1.0f,%8.2f,%s,%8.4f,%6.2f,%6.3f,%6.3f,%6.3f,%5.4f,%5.4f,%6
.3f,%5.2f,%6.4f,%6.4f,%6.3f,%6.3f,%6.1f,%6.3f,%6.3f,%6.3f',...
                D(1)*1e6,Z*1000,N0,dzn*1e6,P(end)/1000,Pin_i,Tsub,NCflg,etime(t2,t1),'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[D(1)*1e6,N0,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/10
00,...
                    Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
        else

fprintf(fid,'\n,,,%5.1f,%5.2f,%3.0f,%5.1f,%6.2f,%6.2f,%3.1f,%1.0f,%8.2f,%s,%8.4f,%6.2f,%6.3f,%6.3f,%6.3f,%5.4f,%5.4f,%6
.3f,%5.2f,%6.4f,%6.4f,%6.3f,%6.3f,%6.1f,%6.3f,%6.3f,%6.3f',...
                D(1)*1e6,Z*1000,N0,dzn*1e6,P(end)/1000,Pin_i,Tsub,NCflg,etime(t2,t1),'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[D(1)*1e6,N0,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/10
00,...
                    Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
        end
    end
elseif chmod==2
    if quf==1
        if condn==1
            fprintf(fid,'\nFile Name:, %s',resfile);
            fprintf(fid,'\n%s',stitl);
            fprintf(fid,'\nFriction Mode:, %s',fricmod);
            fprintf(fid,'\nAcceleration Mode:, %s',acelmod);
            fprintf(fid,'\nTemperature tolerance:, %4.3f, Deg C \nPressure tolerance:, %5.1f, Pa',tolT,tolP);
            fprintf(fid,'\n,,,wc,hc,L,N_0,dzn,Pout,Pin_i,Tsub,NC flag,Elapsed Time,case #,
,Gin,q_wall,DP,dPf,dPa,dPc(contraction),dPc(expansion),dPc(total),Wout,VFout,P_in,T_in,mdot,Vdot,PP,Aw');
            fprintf(fid,'\n,,,[micron],[micron],[mm], ,[micron],[kPa],[kPa],[Deg C], ,[s], ,[kg/m^2-
s],[W/cm^2],[kPa],[kPa],[kPa],[kPa],[kPa],[kPa], , ,[kPa],[Deg C],[g/s],[ml/s],[mW],[mm^2], ');
```

```
fprintf(fid,'\n,,,%5.3f,%5.3f,%5.3f,%5.1f,%5.1f,%6.4f,%6.4f,%3.2f,%1.0f,%8.2f,%d,%s,%8.6f,%6.3f,%6.4f,%6.4f,%6.4f,%6.4f
,%6.4f,%6.4f,%5.4f,%5.4f,%5.4f,%6.4f,%6.6f,%6.3f,%6.1f,%8.4f',...
                wc(1)*1e6,hc(1)*1e6,Z*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,sum(dPc(1:2))/1000,sum(dPc(3:4))/1000,sum(dPc)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6);

A(condn,:)=[wc(1)*1e6,hc(1)*1e6,No,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,Gin,qin_0,DP/1000,sum(dPf
)/1000,sum(dPa)/1000,...
                    Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6];
        else

fprintf(fid,'\n,,,%5.3f,%5.3f,%5.3f,%5.1f,%5.1f,%6.4f,%6.4f,%3.2f,%1.0f,%8.2f,%d,%s,%8.6f,%6.3f,%6.4f,%6.4f,%6.4f,%6.4f
,%6.4f,%6.4f,%5.4f,%5.4f,%5.4f,%6.4f,%6.6f,%6.3f,%6.1f,%8.4f',...
                wc(1)*1e6,hc(1)*1e6,Z*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,sum(dPc(1:2))/1000,sum(dPc(3:4))/1000,sum(dPc)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6);

A(condn,:)=[wc(1)*1e6,hc(1)*1e6,No,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,Gin,qin_0,DP/1000,sum(dPf
)/1000,sum(dPa)/1000,...
                    Wout,VFend,P(1)/1000,Tin-273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6];
        end
    else
        if condn==1
            fprintf(fid,'\nFile Name:, %s',resfile);
            fprintf(fid,'\n%s',stitl);
            fprintf(fid,'\nFriction Mode:, %s',fricmod);
            fprintf(fid,'\nAcceleration Mode:, %s',acelmod);
            fprintf(fid,'\nTemperature tolerance:, %4.3f, Deg C \nPressure tolerance:, %5.1f, Pa',tolT,tolP);
            fprintf(fid,'\n,,,wc,hc,L,N_0,dzn,Pout,Pin_i,Tsub,NC flag,Elapsed Time,Case #,
,Gin,q_wall,DP,dPf,dPa,Wout,VFend,Pin,T_in,T_out,T_max,Tw_max,mdot,Vdot,PP,Aw,epsilon,Ul,Uv,Padj');
            fprintf(fid,'\n,,,[micron],[micron],[mm], ,[micron],[kPa],[kPa],[Deg C], ,[s], , ,[kg/m^2-
s],[W/cm^2],[kPa],[kPa],[kPa], , ,[kPa],[Deg C],[Deg C],[Deg C],[g/s],[ml/s],[mW],[mm^2], ,[m/s],[m/s],[Pa]');

fprintf(fid,'\n,,,%5.3f,%5.3f,%5.3f,%5.1f,%5.1f,%6.4f,%6.4f,%3.2f,%1.0f,%8.2f,%d,%s,%8.6f,%6.3f,%6.4f,%6.4f,%6.4f,%5.4f
,%5.4f,%6.4f,%5.4f,%5.4f,%5.4f,%5.4f,%6.4f,%6.6f,%6.3f,%6.3f,%6.1f,%5.2f,%5.2f,%6.3f',...
                wc(1)*1e6,hc(1)*1e6,Z*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[wc(1)*1e6,hc(1)*1e6,No,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,Gin,qin_0,DP/1000,sum(dPf
)/1000,sum(dPa)/1000,...
                    Wout,VFend,P(1)/1000,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
        else

fprintf(fid,'\n,,,%5.3f,%5.3f,%5.3f,%5.1f,%5.1f,%6.4f,%6.4f,%3.2f,%1.0f,%8.2f,%d,%s,%8.6f,%6.3f,%6.4f,%6.4f,%6.4f,%5.4f
,%5.4f,%6.4f,%5.4f,%5.4f,%5.4f,%5.4f,%6.4f,%6.6f,%6.3f,%6.3f,%6.1f,%5.2f,%5.2f,%6.3f',...
                wc(1)*1e6,hc(1)*1e6,Z*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,'
',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,...
                Wout,VFend,P(1)/1000,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[wc(1)*1e6,hc(1)*1e6,No,Z,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2,t1),casnum,Gin,qin_0,DP/1000,sum(dPf
)/1000,sum(dPa)/1000,...
                    Wout,VFend,P(1)/1000,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
        end
    end
elseif chmod==3
    if condn==1
        fprintf(fid,'\nFile Name:, %s',resfile);
        fprintf(fid,'\n%s',stitl);
        fprintf(fid,'\nFriction Mode:, %s',fricmod);
        fprintf(fid,'\nAcceleration Mode:, %s',acelmod);
        fprintf(fid,'\nTemperature tolerance:, %4.3f, Deg C \nPressure tolerance:, %5.1f, Pa',tolT,tolP);
        fprintf(fid,'\n,,,wc_i,wc_t,hc,gamma,betta,k,L,R_disk,R_inlet,N_0,dzn,Pout,Pin_i,Tsub,NC flag,Elapsed Time,case
#, ,Gin,q_wall,DP,dPf,dPa,dPr,Wout,VFout,T_in,T_out,T_max,Tw_max,Pin,mdot,Vdot,PP,Aw,epsilon,Ul,Uv,Padj');
        fprintf(fid,'\n,,,[micron],[micron],[micron], , , ,[mm],[mm],[mm], ,[micron],[kPa],[kPa],[Deg C], ,[s], ,
,[kg/m^2-s],[W/cm^2],[kPa],[kPa],[kPa],[kPa], , ,[Deg C],[Deg C],[Deg C],[Deg C],[kPa],[g/s],[ml/s],[mW],[mm^2],
,[m/s],[m/s],[Pa]');

fprintf(fid,'\n,,,%5.3f,%5.3f,%5.3f,%5.4f,%5.4f,%2.0f,%5.3f,%4.3f,%4.3f,%3.0f,%5.3f,%6.4f,%6.4f,%3.1f,%1.0f,%8.2f,%d,%s
,%8.6f,%6.3f,%6.4f,%6.4f,%6.4f,%6.4f,%5.4f,%5.4f,%6.3f,%6.3f,%6.3f,%6.3f,%6.4f,%6.6f,%6.4f,%6.3f,%6.3f,%6.1f,%5.2f,%5.2
f,%6.3f',...

wc(1)*1e6,wc(end)*1e6,hc(1)*1e6,gama,betta,Levs,Z*1000,re*1000,ri*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2
,t1),casnum,' ',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,sum(dPr)/1000,...
            Wout,VFend,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,P(1)/1000,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[wc(1)*1e6,wc(end)*1e6,hc(1)*1e6,gama,betta,Levs,Z*1000,re*1000,ri*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NC
flg,etime(t2,t1),casnum,Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,sum(dPr)/1000,...
                Wout,VFend,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,P(1)/1000,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];
    else

fprintf(fid,'\n,,,%5.3f,%5.3f,%5.3f,%5.4f,%5.4f,%2.0f,%5.3f,%4.3f,%4.3f,%3.0f,%5.3f,%6.4f,%6.4f,%3.1f,%1.0f,%8.2f,%d,%s
,%8.6f,%6.3f,%6.4f,%6.4f,%6.4f,%6.4f,%5.4f,%5.4f,%6.3f,%6.3f,%6.3f,%6.3f,%6.4f,%6.6f,%6.4f,%6.3f,%6.3f,%6.1f,%5.2f,%5.2
f,%6.3f',...
```

```
wc(1)*1e6,wc(end)*1e6,hc(1)*1e6,gama,betta,Levs,Z*1000,re*1000,ri*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NCflg,etime(t2
,t1),casnum,' ',Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,sum(dPr)/1000,...
            Wout,VFend,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,P(1)/1000,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv,padj);

A(condn,:)=[wc(1)*1e6,wc(end)*1e6,hc(1)*1e6,gama,betta,Levs,Z*1000,re*1000,ri*1000,No,dzn*1e6,Pout_t/1000,Pin_i,Tsub,NC
flg,etime(t2,t1),casnum,Gin,qin_0,DP/1000,sum(dPf)/1000,sum(dPa)/1000,sum(dPr)/1000,...
                Wout,VFend,Tin-273.15,T(end)-273.15,max(T)-273.15,max(Tw)-
273.15,P(1)/1000,mdot*1000,Vdot*1e6,Vdot*DP*1000,Aw*1e6,(qin*Aw)/(Vdot*DP),ul,uv];

    end
end
```

```
%    ressave3a.m
%
%    Saves results to the User Specified file name
%
%
%
%
%
%

save(resfilem,'A');
```

```
function VF=Voidfrac(W,vl,vv,T,vfmod)

%
%
%  Calculates the void fraction (percentage of vapor by volume) assuming
%  homogenous flow.
%
%
%
%

%
%  ***   Calculation   ***
%

v=W.*vv+(1-W).*vl;
hvf=W.*vv./v;

if vfmod==0         %    Homogenous Model

     VF=hvf;

elseif vfmod==1     %    Separated Model per Chung et al. 2003

     VF=0.03*sqrt(hvf)./(1-0.97*sqrt(hvf));

elseif vfmod==2     %    Separated Model per Armaand

     VF=.833*hvf;

elseif vfmod==3     %    Separated Model per Lockhart & Martinelli

     X=sqrt((muliq_w(T,1)/muvap_w(T,1)).*((1-W)./W).*(vl/vv));
     VF=1-1./(sqrt(1+20./X+1./X.^2));

elseif vfmod==4     %    Separated Model per Zivi

     VF=1./(1+((1-W)./W).*((vl/vv)^(2/3)));

end
```

```
function [phi,phifl,chisq,Relo,Rel,Reg]=phisq_w(vl,vv,T,W,G,D,lam,Re,vimod,vpflg);

%  function [phi,Relo,Rel,Reg]=phisq_w(vl,vv,T,W,G,D,lam,Re,vimod,vpflg);
%
%   [phi,phifl,chisq,Relo,Rel,Reg]=phisq_w(vl,vv,T,W,G,D,lam,Re,vimod,vpflg);
%
%  Calculates the two-phase friction multiplier for water  given  the
%  temperature and vapor fraction of the two phase mixture.
%
%  Inputs
%     vl -  Specific volume of liquid phase
%     vv -  Specific volume of vapor phase
%     T  -  Temperature of the liquid and vapor
%     W  -  Vapor fraction of the two phase mixture
%     G  -  Mass flux in channel
%     D  -  Channel hydaulic Diameter
%    lam  -
%     Re  -
%  vimod -  Two phase multiplier model to be used
%  vpflg -  Variable property flag 1=temperature dependent, 0=constant properties
%
%  Outputs
%     phi-  two-phase friction multiplier
%    Relo-
%     Rel-
%     Reg-
%
%  Calls
%     muliq_w.m  -  calculates the viscosity of the liquid mixture
%     muvap_w.m  -  calculates the viscosity of the vapor mixture
%     LamRe.m    -  calculates the fully developed friction factor as a
%                   function of aspect ratio
%

%
%  ***   Constants   ***
%

Recrit=2300;

%
%  ***   Preliminary Calculations   ***
%
mul=muliq_w(T,1);
muv=muvap_w(T,1);
v=(1-W)*vl+W*vv;

Relo=G*D/mul;
Rel=G*(1-W)*D/mul;
Reg=G*W*D/muv;

%
% ***   Calculations   ***
%

%    ***   Two Phase Viscosity   ***

if vimod<7     %    Homogemous model - Laminar flow
    phifl=NaN;
    chisq=NaN;
     if vimod==1
          %    Per Lin et al.
          n=1.4;
          mutp=mul*muv./(muv+W.^n*(mul-muv));

     elseif vimod==2
          %    Per McAdams
          mutp=1./(W./muv+(1-W)./mul);

     elseif vimod==3
          %    Per  Cicchitti
          mutp=W*muv+(1-W)*mul;

     elseif vimod==4
          %    Per Dukler
          v=(1-W)*vl+W*vv;
          mutp=((1-W)*vl*mul+W*vv*muv)/v;

     elseif vimod==5
          %    Per Stanley et al.
          nul=mul*vl;
          nuv=muv*vv;
          nutp=(1-W)*nul+W*nuv;
          Retp=G*v*D/nutp;
          lamtp=97/Retp;

     elseif vimod==6
          %    Per Collier
          lamtp=4*0.003;

     end
```

```
     if vimod==5|vimod==6
          phi=(lamtp/lam)*(1+W*(vv/vl-1));
     else
         if Relo < Recrit
             phi=(mutp./mul).*(1+W*(vv/vl-1));
         else
             phi=(mutp./mul)^.25.*(1+W*(vv/vl-1));
         end
     end

elseif vimod>=7     %    Seperated Flow Model - Laminar/Laminar Flow

     n=.25;
     if Rel < Recrit & Reg < Recrit
         chisq=(mul/muv)*((1-W)/W)*(vl/vv);
%      elseif Rel < Recrit & Reg >= Recrit
%          chisq=((mul/muv)^n)*(((1-W)/W)^(2-n))*(vl/vv)*(64/.316)*(Rel^(n-1));
%      elseif Rel >= Recrit & Reg < Recrit
%          chisq=((mul/muv)^n)*(((1-W)/W)^(2-n))*(vl/vv)*(.316/64)*(Reg^-(n-1));;
     else
         chisq=((mul/muv)^n)*(((1-W)/W)^(2-n))*(vl/vv);
     end

     if vimod==7
          %    Per Lockhart & Martinelli
          Chiz=21;
          phifl=(1+Chiz/sqrt(chisq)+1/chisq);
          phi=phifl*((1-W));


     elseif vimod==8
          %    Per Mishima & Hibiki
          Chiz=21*(1-exp(-0.319*D*1000));
          phifl=(1+(Chiz/sqrt(chisq))+(1/chisq));
          phi=phifl*(1-W);

     elseif vimod==9
          %    Per Lee & Lee
%           Rel=2000; Reg=2500;
          sig=5.89e-2;        %[N/m] at 100 Deg C.
          l=mul^2/(sig*D/vl);      psi=mul*G*(1-W)*((1-W)*vl+W*vv)/sig;      Re=G*D/mul;
          if Rel < Recrit & Reg < Recrit
              q=-1.317;               r=0.719;                s=0.557;
              A=6.833e-8;
              Chiz=A*(l^q)*(psi^r)*(Re^s);
          elseif Rel < Recrit & Reg >= Recrit
%               Reg=G*W*D/muv;
%               Rel=G*(1-W)*D/mul;
              chisq=((1-W)/W)^2*(vl/vv)*((lam*Reg^.25)/.316);
              s=.726;
              A=6.185e-2;
              Chiz=A*(Re^s);
          elseif Rel >= Recrit & Reg < Recrit
              s=.174;
              A=3.627;
              Chiz=A*(Re^s);
          else
              s=.451;
              A=.408;
              Chiz=A*(Re^s);
          end
          phifl=(1+(Chiz/sqrt(chisq))+(1/chisq));
          phi=phifl*(1-W);

     elseif vimod==10
          %    Per Qu & Mudawar
          Chiz=21*(1-exp(-0.319*D*1000))*(.00418*G+0.0613);
          phifl=(1+(Chiz/sqrt(chisq))+(1/chisq));
          phi=phifl*(1-W);

      elseif vimod==11
          %    Per Chisholm
          Gam=((vv/vl)^.5)*((muv/mul)^(n/2))
          %  B=((21*Gam-2^(2-n)+2)/(Gam^2-1));
          B=1
          phi=1+((Gam^2)-1)*((B*W^((2-n)/2))*((1-W)^((2-n)/2))+W^(2-n));
      end


end
```

```
function hfm=dia_mult(q,G)

%    function hfm=dia_mult(q,G)
%
%    Calculates the ratio between the adiabatic two phase multipier and the
%    diabatic two phase multiplier per the correlation proposed by Tarasova
%    as shown in Collier's book
%
%    Inputs:
%         q    -    Heat flux in W/m^2
%         G    -    Mass flux in kg/m^2-s
%
%    Outputs:
%         hfm  -    ratio of two-phase multiliers
%
%


hfm=1+.00435*(q./G).^.7;
```

```
function [lamre,kinf,c]=Kinf(alpha);

%  function [lamre,kinf,C]=Kinf(alpha);
%
%  Calculates the coefficients kinf and C used in the apparent friction
%  factor formula for the rectangular channel.  The calculation is based
%  on the aspect ratio of the rectangular channel, alpha.
%
%
%

%
%  ***   Constants   ***
%

Alph=[1 0.5 0.2 0.0];
K=[1.430 1.280 0.931 0.674];
C=[0.000290 0.000210 0.000076 0.000029];
LAMRe=4*[14.23 15.55 19.07 24];

for i=1:length(alpha)

    %
    %  ***   Interpolation Limits   ***
    %

    if alpha(i)<=Alph(1) & alpha(i)>Alph(2)
        low=1; high=2;
    elseif alpha(i)<=Alph(2) & alpha(i)>Alph(3)
        low=2; high=3;
    elseif alpha(i)<=Alph(3) & alpha(i)>=Alph(4)
        low=3; high=4;
    end

    %
    %  ***   Interpolations   ***
    %

    kinf(i)=((K(high)-K(low))/(Alph(high)-Alph(low)))*(alpha(i)-Alph(low))+K(low);
    c(i)=((C(high)-C(low))/(Alph(high)-Alph(low)))*(alpha(i)-Alph(low))+C(low);
    lamre(i)=((LAMRe(high)-LAMRe(low))/(Alph(high)-Alph(low)))*(alpha(i)-Alph(low))+LAMRe(low);

end
```

```
function [ff]=lamda_loc(xi,L,lamre,K,C,D,Re);



dfappdl=(-13.76./(2*xi.^1.5)+(2*lamre.*C./xi.^3-K.*(1-C./xi.^2)...
    ./(xi.^2)+13.76*(1-3*C./xi.^2)./(2*xi.^1.5))./(1+C./xi.^2).^2)./(D.*Re.^2);
ff=lamda_app(xi,lamre,K,C,Re)+L.*dfappdl;
```

```
function [ffapp]=lamda_app(xi,lamre,K,C,Re)



fapp1=13.76./xi.^(1/2);
fapp2u=(lamre)+(K./xi)+(-13.76./xi.^(1/2));
fapp2l=(1+C./xi.^2);

ffapp=(1./Re).*(fapp1+fapp2u./fapp2l);
```

```
function C=LamRe(alpha)

%  function C=LamRe(alpha)
%
%  Calculates the constant value of the product of the friction factor and
%  the Reynolds Number for fully developed flow in a rectangular duct with
%  the aspect ratio calculated from the channel width and height
%
%
%

%
%  ***   Constants   ***
%

p=[-0.2537 0.9564 -1.7012 1.9467 -1.3553 1.0000];

%
%  ***   Calculation   ***
%

C=96*polyval(p,alpha);
```

```
function hl=hlw(T);

%
%  Calculates the enthalpy in J/kg for saturated liquid water at the
%  provided temperature in Kelvin
%
%
%
%
%


%
%  ***   Calculation   ***
%


%    Per Curvfit using Table2D, Temperature in Deg C;
%         T=T-273.15;
%         C=[1.276398 -0.0026137329 4.1160052 2.115243e-7 -0.010241881];
%         hl=(C(1)+C(3)*T+C(5)*T.^2)./(1+C(2)*T+C(4)*T.^2);
%         hl=hl*1000;

%    Property Formula per Irvine & Liley; 'Steam and Gas Tables with
%    Computer Equations'; 1984.
          Tcr=647.3;     Hcr=2099.3;
          for i=1:length(T)
         Tc=(Tcr-T(i))/Tcr;
         if T(i)>=273.16 & T(i)<300
              A=0;      B=0;       C=0;      D=0;
              E=[624.698837 -2343.85369 -9508.12101 71628.7928 -163535.221 166531.093 -64785.4585];
         elseif T(i)>=300 & T(i)<600
              A=.8839230108;      B=0;       C=0;      D=0;
              E=[-2.67172935 6.22640035 -13.1789573 -1.91322436 68.7937653 -124.819906 72.1435404];
         elseif T(i)>=600 & T(i)<=647.3
              A=1;      B=-.441057805;       C=-5.52255517;        D=6.43994847;
              E=[-1.64578795 -1.30574143 0 0 0 0 0];
         else
              A=NaN;    B=NaN;              C=NaN;              D=NaN;
              E=[NaN NaN NaN NaN NaN NaN NaN];
         end
         Ht=0;
         for j=1:7
              Ht=Ht+E(j)*Tc^j;
         end
         hl(i)=(Ht+A+B*Tc^(1/3)+C*Tc^(5/6)+D*Tc^(7/8))*Hcr*1000;
          end
```

```
function hv=hvw(T);

%
%  Calculates the enthalpy in J/kg for saturated water vapor at the
%  provided temperature in Kelvin
%
%
%
%
%
%

%
%  ***   Calculation   ***
%

%    Per Curvfit using Table2D, Temperature in Deg C;
%         T=T-273.15;
%         C=[2501.28 -0.0049660766 -10.569546 8.0542799e-6 0.010491081 -5.0051184e-9];
%         hv=(C(1)+C(3)*T+C(5)*T.^2)./(1+C(2)*T+C(4)*T.^2+C(6)*T.^3);
%         hv=hv*1000;

%    Property Formula per Irvine & Liley; 'Steam and Gas Tables with
%    Computer Equations'; 1984.
          Tcr=647.3;     Hcr=2099.3;
          for i=1:length(T)
         Tc=(Tcr-T(i))/Tcr;
         if T(i)>=273.16 & T(i)<=647.3
              A=1;      B=.457874342;       C=5.08441288;      D=-1.48513244;
              E=[-4.81351884 2.69411792 -7.39064542 10.4961689 -5.46840036 0 0];
         else
              A=NaN;    B=NaN;              C=NaN;              D=NaN;
              E=[NaN NaN NaN NaN NaN NaN NaN];
         end
         Ht=0;
         for j=1:7
              Ht=Ht+E(j)*Tc^j;
         end
         hv(i)=(Ht+A+B*Tc^(1/3)+C*Tc^(5/6)+D*Tc^(7/8))*Hcr*1000;
          end
```

```
function mu=muliq_w(T,varprop)

%  function mu=muliq_w(T,varprop)
%
%  Calculates the viscosity of  water
%  based on the temperature.
%
%  Inputs
%     T  -  Temperature of mixture in Kelvin

%
%  ***   Calculation   ***
%

if varprop==0

   mu=6.53e-4;

elseif varprop==1
%    Excell curvefit - Temperatuer input in Kelvin
%    mu=T^4*3.247317e-11 - T^3*4.453606e-8 + T^2*2.294031e-5 - T*5.265866e-3 + 4.554164e-1;  % Units: [kg/m-s]; Liquid
water viscosity

%    Tablecurve 2D curvefit - Temperature input in Deg C
     T=T-273.15;
     c=[-6.3465546 0.0075044209 -0.079802656];
     mu=exp((c(1)+c(3)*T)/(1+c(2)*T));

end
```

```
function mu=muvap_w(T,varprop)

%    function mu=muvap_w(T,varprop)
%
%    Calculates the viscosity of vapor water based on the temperature input
%    in Kelvin
%
%
%
%



%
%    ***  Calculations  ***
%

if varprop==0     % Constant Viscosity

     mu=1.5e-5; % Units: [kg/m-s]; water vapor viscosity

elseif varprop==1 % Temperature Dependent Viscosity
     T=T-273.15;
%      c=[8.3733035e-6 3.6818645e-8];
%      mu=c(1)+c(2)*T;
     c=[0.0029025072 6.3263251e-6 -7.8178558e-9 1.2555444e-11];
     mu=(c(1)+c(2)*T+c(3)*T^2+c(4)*T^3)^2; % Units: [kg/m-s]; water vapor viscosity

end

iii
```

```
function Ts=Tsw(P)

% Calculates the saturation temperature of water in Kelvin given the
% pressure in Pa.
%
%
%
%
%
%
%


%    Excell curvefit, P in curvefit must be in in kPa, T is given in Kelvin
%         P=P/1000;
%         Ts= -4.971679e-11*P^4 + 2.0306949e-7*P^3 - 3.1318331e-4*P^2 + 2.6100634e-1*P + 351.24244;

%    Tablecurve 2D curvefit, P in curvefit must be in Bar, T is provided in Deg C
%         c=[88.5518244 -0.14776411 0.964692203 11.38087502 22.38381106];
%         P=P./100000;        % Units: [Pa] -> [bar]
%         Ts=c(1)+c(2)*P+c(3)*(log(P)).^2+c(4)*P.^0.5+c(5)*log(P);    % Units: [Deg C]
%         Ts=Ts+273.15;         % Units: [Deg C] -> [K]

%    Property Formula per Irvine & Liley; 'Steam and Gas Tables with
%    Computer Equations'; 1984.
         P=P./1e6;           % Units: [Pa] -> [MPa];
     for i=1:length(P)
          if P(i)>=.000611 & P(i)<12.33
               A(i)=42.6776;     B(i)=-3892.7;     C(i)=-9.48654;
          elseif P(i)>=12.33 & P(i)<=22.1
               A(i)=-387.592;    B(i)=-12587.5;    C(i)=-15.2578;
          else
               A(i)=NaN;    B(i)=NaN;    C(i)=NaN;
          end
     end
     Ts=A+B./(log(P)+C);      % Units: [K]
```

```
function vl=Vlw(T);

%
%    Calculates the specific volume of saturated liquid water given the
%    temperature in Kelvin
%
%
%
%
%
%

% vl=ones(size(T))*0.00104693497435;

%
%  ***   Calculation   ***
%

%    Per Curvfit using Table2D, Temperature in Deg C;
%         T=T-273.15;
%         C=[-6.9155638 -0.0021065942 0.014997507];
%         vl=exp((C(1)+C(3)*T)./(1+C(2)*T));

%    Property Formula per Irvine & Liley; 'Steam and Gas Tables with
%    Computer Equations'; 1984.
          Tcr=647.3;     Vcr=3.155e-3;
          for i=1:length(T)
         Tc=(Tcr-T(i))/Tcr;
         if T(i)>=273.16 & T(i)<=647.3
              A=1;      B=-1.9153882;       C=12.015186;        D=-7.8464025;
              E=[-3.888614 2.0582238 -2.0829991 .82180004 .47549742 0 0];
         else
              A=NaN;    B=NaN;              C=NaN;              D=NaN;
              E=[NaN NaN NaN NaN NaN NaN NaN];
         end
         Vt=0;
         for j=1:7
              Vt=Vt+E(j)*Tc^j;
         end
         vl(i)=(Vt+A+B*Tc^(1/3)+C*Tc^(5/6)+D*Tc^(7/8))*Vcr;
          end
```

```
function vv=Vvw(T);

%
%    Calculates the specific volume of saturated water vapor given the
%    temperature in Kelvin
%
%
%
%
%
%

% vv=ones(size(T))*1.45676573551827;

%
%  ***   Calculation   ***
%

%    Per Curvfit using Table2D, Temperature in Deg C;
%         T=T-273.15;
%         C=[5.3292323 0.0037122296 -0.049242051 -1.0540567e-6 2.9086784e-5 -9.0842686e-9];
%         vv=exp((C(1)+C(3)*T+C(5)*T.^2)./(1+C(2)*T+C(4)*T.^2+C(6)*T.^3));

%    Property Formula per Irvine & Liley; 'Steam and Gas Tables with
%    Computer Equations'; 1984.
          Tcr=647.3;     Vcr=3.155e-3;       Pcr=22.089;
          for i=1:length(T)
         Tc=(Tcr-T(i))/Tcr;
         if T(i)>=273.16 & T(i)<=647.3
              A=1;      B=1.6351057;       C=52.584599;        D=-44.694653;
              E=[-8.9751114 -.43845530 -19.179576 36.765319 -19.462437 0 0];
         else
              A=NaN;    B=NaN;              C=NaN;              D=NaN;
              E=[NaN NaN NaN NaN NaN NaN NaN];
         end
         Vt=0;
         for j=1:7
              Vt=Vt+E(j)*Tc^j;
         end
         vv(i)=(Vt+A+B*Tc^(1/3)+C*Tc^(5/6)+D*Tc^(7/8))*Vcr*Pcr/(Psw(T(i))*1e-6);
          end
```

```
function Ps=Psw(T)

% Calculates the saturation Pressure of water in Pascals given the
% Temperature in Kelvin.
%
%
%
%
%
%
%


%    Property Formula per Irvine & Liley; 'Steam and Gas Tables with
%    Computer Equations'; 1984.

for i=1:length(T)
     if T(i)>=273.16 & T(i)<=647.3
          A=[10.4592 -.00404897 -.417520e-4 .368510e-6 -.101520e-8 .865310e-12 ...
                    .903668e-15 -.19969e-17 .779287e-21 .191482e-24 -3968.06 39.5735];
          N=[0 1 2 3 4 5 6 7 8 9];
     else
          A=[NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN];
     end
     Pt=0;
     for j=0:9
          Pt=Pt+A(j+1)*T(i)^j;
     end
     Ps(i)=exp(Pt+A(11)/(T(i)-A(12)));
end
Ps=Ps*1e6;
```

```
function Pr=Prlw(T)

%    Calculates the Prandtl number of liquid water given the temperature of
%    the water in Kelvin.

%
%    Calculation
%

T=T-273.15;

C=[12.965541 0.034803388 -0.0092351876 0.00038937263 4.436557e-5 -1.161299e-6];

Pr=(C(1)+C(3)*T+C(5)*T.^2)./(1+C(2)*T+C(4)*T.^2+C(6)*T.^3);
```

```
function k=klw(T)

%    Calculates the thermal conductivity of liquid water given the temperature of
%    the water in Kelvin.

%
%    Calculation
%

T=T-273.15;

C=[-1.1833196e-11 1.0523503e-8 -8.7330833e-6 0.0018992789 0.56875667];

k=polyval(C,T);
```

```
function T=h2T_w(h);

%
%  Calculates the saturation temperature of liquid water in Kelvin given
%  the enthalpy of the liquid water in J/kg.
%
%
%
%
%
%

%
%  ***   Calculation   ***
%

h=h/1000;
% disp(sprintf('h = %10.7f', h))
Hcr=2099.3;

% C=[-.10148895 -.00033915519 0.23960264 2.4907428e-8 -8.0336702e-5];
% T=(C(1)+C(3)*h+C(5)*h.^2)./(1+C(2)*h+C(4)*h.^2);


if h<=112.39
     p=[-64785.4585 166531.093 -163535.221 71628.7928 -9508.12101 -2343.85369 624.698837 -h/Hcr];
     r=roots(p);
     for i=1:length(r)
          if isreal(r(i))
               if r(i)>.536 & r(i)<.5781;
                    T=647.3*(1-r(i));
                    break
               end
          end
     end
elseif h>=112.39 & h<=1505.9
     p=[72.1435404 -124.819906 68.7937653 -1.91322436 -13.1789573 6.22640035 -2.67172935 .8839230108-h/Hcr];
     r=roots(p);
     T=647.3*(1-real(r(7)));
%      for i=1:length(r)
%           if isreal(r(i))
%                if r(i)>.073 & r(i)<.5366;
%                     T=647.3*(1-r(i));
%                     break
%                end
%           end
%      end
%      T=T+273.15;
elseif h>1505.9
     C=[-4.3268659e-15 1.1828037e-11 -2.5154668e-8 1.2732779e-5 0.23695752 0.031284377];
     T=polyval(C,h);
     T=T+273.15;
end

% disp(sprintf('T = %5.2f Deg C',T))
```