AN ABSTRACT OF THE THESIS OF

Salvador Martin Aceves-Saborio for the degree of Master of Science in Mechanical Engineering presented October 29, 1986

Title: Analysis of Ice Formation With Flow Reversal for Application to an Ice-Maker Heat Pump

Abstract Approved: Redacted for Privacy
Gordon M. Reistad

The problems of freezing and melting present special mathematical complexity and, except for the most elementary one-dimensional cases, require a numerical solution. However, these problems have a growing importance in engineering problems because of their application to metallurgical operations, food preservation, heating and air conditioning, etc.

In the present work, a numerical model for ice formation from flowing water in a finned annular sector is developed. The model predicts ice profiles as a function of axial position and time, and energy transfer values as a function of time. The model also allows flow direction reversals in the annular sector. The flow reversal melts part of the ice existing in the annulus. Also, part of the ice detaches, allowing it to flow out with the liquid water. This enhances the energy transfer values. The model

uses an arbitrary criterion for melting and detaching.

The model is used for a preliminary evaluation of an ice-maker heat pump evaporator operating with flow reversal. Flow reversal de-ices the evaporator surfaces and enhances the heat transfer. Therefore, the flow reversal method is suggested as an alternative for de-icing without the losses commonly associated with the methods currently used to accomplish this.

The results of the preliminary evaporator evaluation show a large energy transfer gain for the reversal case, as compared to the energy transfer obtained without reversal. Most of the gain comes from an increased cooling of the liquid water (more sensible energy from the water is being transferred to the refrigerant). The model predicts only a small amount of ice detachment from the evaporator.

Further study in this area is recommended for completing the evaluation of the evaporator. A better ice melting-detaching criterion and a more complete parametric evaluation of the evaporator are required.

ANALYSIS OF ICE FORMATION WITH FLOW REVERSAL FOR APPLICATION TO AN ICE-MAKER HEAT PUMP

by

Salvador Martin Aceves-Saborio

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

MASTER OF SCIENCE

Completed October 29, 1986

Commencement June 1987

APPROVED:

Redacted for Privacy

Professor of Mechanical Engineering in Charge of Major

Redacted for Privacy

Head of Department of Mechanical Engineering

Redacted for Privacy

Dean of Graduate School

Date thesis is presented October 29, 1986

ACKNOWLEDGEMENTS

Many persons helped me in completing this work. First of all, I want to thank Dr. Gordon M. Reistad for all the help and guidance that I recieved from him in the past two years. Dr Hajime Nakamura's advice represented a significant contribution to the development of the numerical model and I want to express my gratitude to him. I am very grateful for the efforts of my committee members, Dr. Robert Wilson, Dr. Lorin Davis and Dr. John Peterson.

I also want to thank Dr. James Welty and Dr. Arturo Lara, to whom I owe the opportunity of being here. I acknowledge the financial support recieved from CONACYT, Mexico, during the second year of my studies.

I acknowledge the great help that I have recieved from my parents, Salvador and Alicia, and my grandmother, Judith. All my work has been much easier thanks to their help.

Finally, I would like to thank Virginia Betz for her help in completing the figures and editing the text.

Nomenclature

a,b,c,d	coefficients in the Laplacian
A	flow area
Cp	specific heat
F	radial position of interface
Fn	normal position of interface
F', F''	succesive derivatives of F respect to ϕ.
h	refrigerant heat transfer coefficient
K	thermal conductivity
L	latent heat of solidification
n	coordinate in the normal direction
Nu	Nusselt number
p	pressure
P	heat transfer perimeter
Q	volumetric fluid flow
r	radial polar coordinate
r^+	non-dimensional radial coordinate
Ri	internal pipe radius
Ro	external pipe radius
Ste	Stefan number
Sx	generalized scale factor
t	time
T	temperature
to	pipe wall thickness
tw	ice thickness at external pipe wall
U	fluid velocity

Nomenclature - Cont.

x	coordinate along the metallic wall
Z	axial coordinate
z'	coordinate along the solid-liquid interface
α	thermal diffusivity
α, β	angles between the r, ϕ ,Z and n, ϕ ',z' coordinate systems
$\Delta\phi$,Δr	radial and transversal stepsizes
ϕ	polar transversal coordinates
ϕ '	coordinate along the solid-liquid interface
ϕ o	half of the angle between two consecutive fins
μ	fluid viscosity
ν	fluid kinematic viscosity
ρ	density

Subscripts

a	average value
c	freezing condition
f	liquid property
i	solid property
s	metallic wall property
x	generalized parameter
∞	ambient condition

Nomenclature - Cont.

Superscripts

*	non-dimensional variable

TABLE OF CONTENTS

LIST OF FIGURES

ANALYSIS OF ICE FORMATION WITH FLOW REVERSAL FOR APPLICATION TO AN ICE-MAKER HEAT PUMP

I. INTRODUCTION

1.1 Historical Background

A heat pump is a system that transfers energy from a region of low temperature to a region of high temperature. This upgrade of thermal energy requires work input, according to the Second Law of Thermodynamics. Heat pumps make it possible to supply more thermal energy to the heated space than the work that they require. They are also capable of being reversed in most cases, this is, they can be used to provide air conditioning during the summer months.

Heat pumps have their origins in the early years of the 19th century. The growing understanding of physical processes led to interest in the possibility of pumping heat energy to a higher temperature. Sadi Carnot described in 1824 the theoretical concept of the heat pump. William Thomson (Lord Kelvin), 1852, was the first in proposing a "heat multiplier", as he called the heat pumps.

First heat pump applications were considered in the 1920s, with improvements on Thomson's paper by Krauss, 1921. Haldane, 1930, analyzed data from a number of refrigerating plants. From his analysis, Haldane was able to recommend the heat pump as an efficient device for building

heating.

In the early 1960s, reversible domestic air-to-air heat pumps achieved an appreciable sales success in the U.S.A. Unfortunately, they were not reliable, because it had not been recognized that a reversible heat pump needs to be more than just an air conditioner with a refrigerant reversing valve added. The models built broke down easily as the heating load increased by an outdoor temperature decrease. These experiences almost destroyed the heat pump industry.

From 1973 to date, heat pump production has experienced a constant growth. This is mainly the result of the increased awareness of the world's limited available energy, high energy costs, and the design and construction of more efficient and reliable heat pump systems.

1.2 Heat Pump Operation

The heat pump cycle is illustrated in figure 1.1. The cycle consists of four basic operations: evaporation, compression, condensation and expansion. A heat pump accomplishes its task of transferring energy from a low temperature reservoir to a high temperature space by using a refrigerant. Space heating is accomplished by transferring energy from the low temperature reservoir (generally water or ambient air) to the even lower-temperature liquid refrigerant. The refrigerant evaporates by the effect of

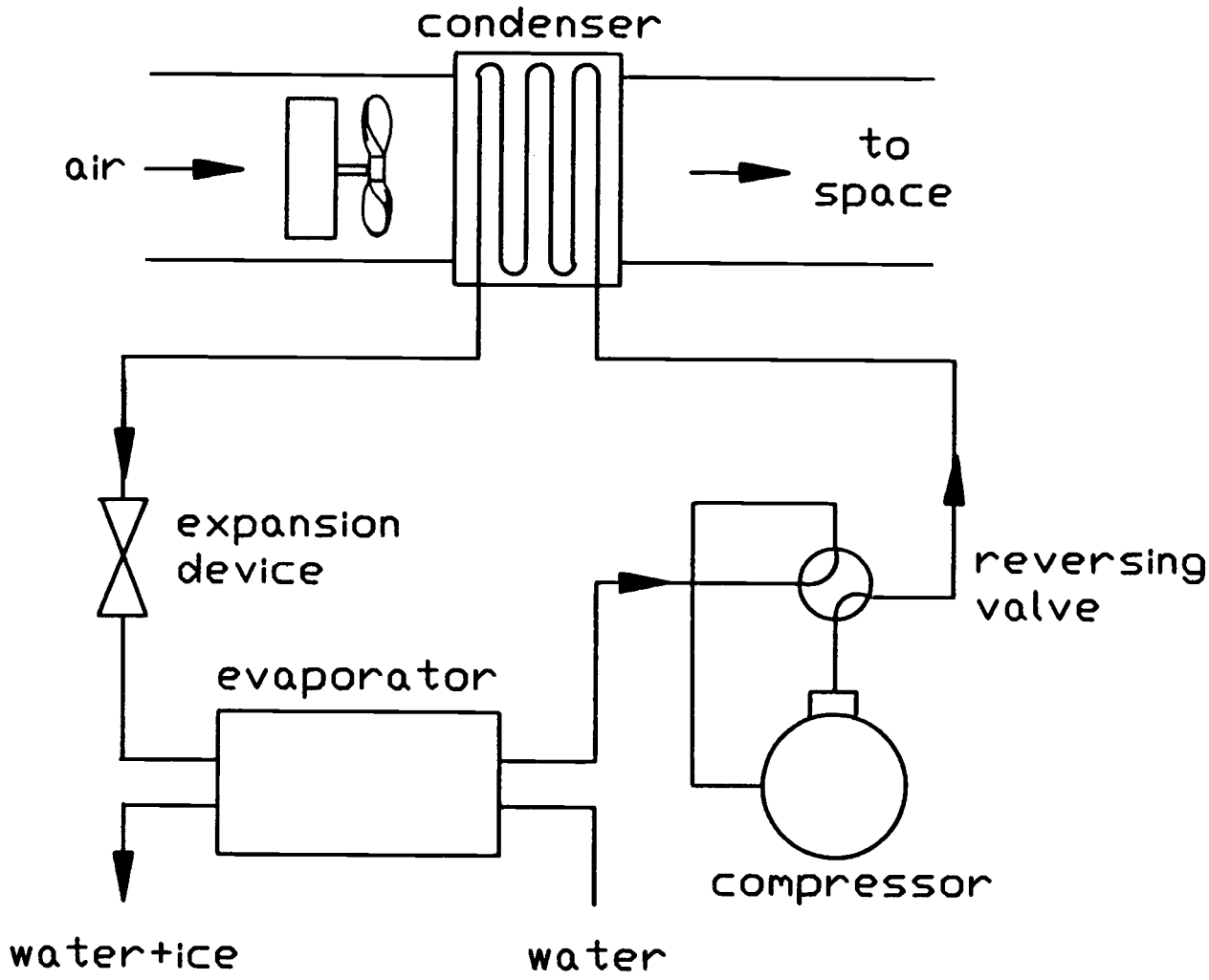

Figure 1.1 Schematic of an ice-maker heat pump operating in the heating mode.

the thermal energy transferred to it. Work is then done on the vapor by a compressor. This increases the vapor's temperature along with its pressure. The vapor is then condensed and the heat of condensation is released to the heated space. The advantage of this type of heating is that more energy is made available for heating than the work required to operate the heat pump.

A heat pump cycle can be reversed to provide space cooling during the summer months. To accomplish this, most heat pumps have a four-way valve as shown in figure 1.1. A movement of this valve transforms the indoor heat exchanger into the evaporator, while the outdoor heat exchanger acts as the condenser.

Heat pumps are classified according to their low-temperature reservoir as air-source heat pumps or water-source heat pumps. Air-source heat pumps present the advantage of having a readily available and free low-temperature reservoir. However, the efficiency of air-source heat pumps decreases rapidly as ambient temperature drops below $5^{o}C$, because frost forms in the evaporator coils insulating them. The insulation reduces the capacity of the heat pump, making it necessary to use backup electrical heating to meet the required heating load. This considerably decreases the heat pump effectiveness.

Water-source heat pumps do not encounter trouble with frosting in the evaporator surfaces, and they are the ideal

choice if a stream of warm water is available. However, the water temperature is generally comparable to the outdoor temperature, and the amount of energy that may be obtained from the water without freezing it may be very small. A very large volume of water would be needed to meet the heating requirements of the heated space, and the cost of this water may make the use of a heat pump uneconomical.

Ice-maker heat pumps are a special kind of water-source heat pump in which the flowing water is allowed to freeze in the evaporator surfaces. This reduces the water consumption by increasing the amount of energy obtained from the water. Ice-maker heat pumps also present the disadvantage of requiring a periodic interruption of the cycle to de-ice the surfaces.

A very efficient system based on an ice-maker heat pump is called the Annual Cycle Energy System (ACES) (Fisher and Nephew, 1976). In this system, the ice made by the heat pump during the winter months is stored in an insulated reservoir and it is used to provide practically free air conditioning during the summer months.

1.3 The Flow Reversal Method

Insulation of evaporator surfaces caused by ice or frost formation is an important problem in the operation of ice-maker and air-source heat pumps respectively. De-frosting of the surfaces can be accomplished in a number of

ways, but it always reduces the effectiveness of the heat pumps, because the heating cycle is interrupted and energy is consumed to defrost the surfaces.

The most common method of defrosting the surfaces is to circulate a stream of warm refrigerant in the evaporator. The refrigerant temperature must be above the freezing point for de-icing to occur. The warm refrigerant melts the superficial layer of ice, leaving the rest of the ice free to fall from the surface by gravitational effect or free to flow out with the water.

An alternative method was tested by Rinaldi et al., 1977. They tried to supercool water in the evaporator without freezing it. The supercooled water was supposed to flow out of the evaporator to a reservoir where ice would form from the supercooled water. This method would provide steady-state operation of the heat pump without the need of interrupting the cycle for de-icing. Their experimental work consisted of measuring the maximum supercooling attainable with water flowing over the evaporator surfaces. The surfaces were coated with different substances to reduce the chances of ice formation on the wall. However, the results indicated very low water supercooling (the maximum value recorded was 2.5^{o}C), so that the amount of ice obtained in the reservoir would be negligible, if any. There is also the possibility of having ice build-up in the evaporator, which would translate into a drop in

effectiveness.

The purpose of this thesis is to develop a numerical model for the freezing of flowing water in an annular sector. The model allows the flow direction to be reversed. This model is used to evaluate an ice-maker heat pump that operates according to the flow reversal method. In this method, water is circulated along the evaporator surfaces in one direction. The water-side of the evaporator has long fins that allow the ice to build on them in a slender profile. To avoid thick ice layers near the pipe exit that insulate the surfaces considerably, the evaporator should not be excessively long.

At given time intervals, the water flow to the evaporator is reversed. The water outlet then suddenly becomes the water inlet and it is exposed to a warm water flow. The metallic wall has a very high conductivity and a very low heat capacity. This makes it react immediately to a reversal, suddenly increasing its temperature and melting the superficial layer of ice. The melting of the superficial layer of ice leaves part of the remaining ice too weak to stand the pressure exerted by the water flow and breaks free from the rest of the ice. The free ice flows out with the water, increasing the amount of heat obtained from the water and reducing the thickness of the ice insulation in the evaporator.

II. LITERATURE SURVEY

2.1 Introduction

The design of an ice-maker heat pump that can operate in steady state requires a previous estimate of physical parameters, such as the amount of ice formed, ice layer thickness, average power and wall temperature. This makes it important to develop mathematical methods to solve for phase change problems. What follows is a review of some procedures currently used to accomplish this.

2.2 Methods For Calculating Ice Profiles

The presence of a moving boundary whose position is not known a priori mathematically represents a non-linear boundary condition. This nonlinearity complicates greatly the solution for moving boundary problems.

The general problem is stated by the next set of equations, written for the existing domains:

Liquid phase, continuity equation:

$$\nabla \cdot U = 0 \tag{2.1}$$

Liquid phase, momentum equation:

$$\frac{DU}{Dt} = -\frac{1}{\rho} \nabla p + \nu \nabla^2 U \tag{2.2}$$

Liquid phase, energy equation:

$$\frac{DT}{Dt} = \alpha \nabla^2 T \qquad (2.3)$$

Solid phase, energy equation:

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T \qquad (2.4)$$

Boundary condition at the interface:

$$Ki \frac{\partial T}{\partial n} = Kf \frac{\partial T}{\partial n} + \rho L \frac{dFn}{dt} \qquad (2.5)$$

where n is the coordinate along the normal to the interface and dFn is the solid phase growth in the normal direction.

The initial condition and additional boundary conditions depend on the geometry and physical dimensions of the system.

2.2.1 Exact Analytical Solutions: Because of the nonlinear boundary condition, only a very limited number of freezing-melting problems admit an exact analytic solution. These include the classic one-dimensional Stefan and Neumann problems and their variants (Carslaw and Jaeger, 1959) in

which a semi-infinite body, initially in the liquid state, starts to freeze when its face temperature is suddenly changed to a value below the freezing point. For other problems, approximate or numerical methods are required.

2.2.2 Analytical Approximate Methods: This section discusses a few of the approximate analytical methods that are available. These methods have the advantage of giving the results in a closed form that can be readily applied to solve the problems. However, these methods can only be used with advantage for solving simplified problems involving freezing and melting. The most useful of these methods are described below:

The Integral Method: This is basically a Von Karman-Pohlhausen method as applied to boundary layer-type problems. With this method, the temperature penetration depth is defined as a thickness within which the temperature is going to vary, and the temperature is considered as a constant beyond that point. The governing equations are integrated from zero to the penetration depth. This implies that the governing equations will be satisfied only on the average for each one of the elements that constitute the field of solution. The method, as originally applied to melting and freezing problems by Goodman, 1958, is limited to one-dimensional cases where there is no liquid flow and the temperature in the solid phase is always constant and

equal to the freezing temperature. More recent investigators have been able to relax these restrictions slightly (Savino and Siegel, 1969; El-Genk and Cronenberg, 1979). Epstein, 1976, found a solution for a semi-infinite domain in which a flowing liquid freezes on a cooled wall when a constant heat transfer coefficient is given. However, the integral method is not powerful enough to solve the more general problems of multidimensional freezing in finite geometries.

The Perturbation Method: In 1939, Pekens and Schilter considered the ice formation on the outside of a long pipe, the temperature of which varied in some prescribed manner below the freezing temperature. Their work employed a series expansion in which the zeroth order solution corresponds to the complete neglect of sensible energy in the ice. Higher order terms incorporated the sensible energy, which was assumed to be a small fraction of the latent heat. Since that time, the method remains essentially unchanged. It is applied by proposing a solution for the temperature profile (or any other unknown which solution is sought) as a power series in terms of a parameter, which is usually a non-dimensional combination of physical properties of the system. The proposed series would then be expressed as:

$$T = To + \alpha T_1 + \alpha^2 T_2 + ... \qquad (2.6)$$

where To, T_1, T_2, ... are unknown functions of position and time. This expansion is substituted in the governing equation and in the boundary conditions, and these are separated into equations that contain the same power of the parameter α.

This method allows an easy solution for the first order approximation function To. The successive equations can still be solved analytically, although the algebraic difficulty increases with every new function that is calculated. The solutions beyond the third order are generally too complicated to be calculated and used. For this reason, the expansions apply only to small values of the parameter α, so that the functions T_4, T_5, ... become negligibly small when multiplied by their respective powers of α.

This method cannot be used advantageously to solve for phase change problems in multiple dimensions or problems involving fluid flow, but it is very convenient for one-dimensional, unsteady problems. Additional examples of the use of this method are the articles by Huang and Shih, 1975, and Weinbaum and Jiji, 1977.

Other approximate methods: Crank, 1981, cites a number of additional approximate methods. These include the embedding method, the moving source method and a method employing series expansions of Green's functions. Another important method was developed by Zerkle and Sunderland,

1968. They solved a steady-state ice formation problem with laminar flow inside a pipe with a constant wall temperature below the freezing point.

The method by Zerkle and Sunderland is based on the assumption that the ice existing in the pipe is smooth enough so that the velocity profile in the fluid remains parabolic. This assumption, used along with a coordinate transformation that immobilizes the solid-liquid interface, allows them to reduce the energy equation to the standard Graetz problem, whose solution is well known and tabulated (Sellars et al., 1956). This makes it possible to obtain the steady-state ice profile and the pressure drop along the pipe. Some generalizations of this solution have appeared more recently. For example, Sampson and Gibson, 1981, used the same assumptions to solve for the unsteady state problem, while Sadeghipour, et al., 1984, solved the problem including a convective boundary condition at the pipe wall.

2.2.3 Numerical Methods: The development of fast computers and efficient numerical schemes make numerical methods the preferred procedure for solving problems in heat transfer with phase change that are too complicated to be solved by any other method. This section reviews the early work in this field and some of its later applications.

Landau, 1950, first presented the coordinate transformation in which the moving boundary can be eliminated, that

is, fixed. He solved the problem of a semi-infinite solid that melts because a given amount of heat is applied to its surface. The power applied to the surface is allowed to vary with time. The formulation considers that the melt is removed immediately on formation.

Two analytical solutions were obtained for the simplified cases in which the ratio of the sensible energy in the solid to the latent energy in the solid-liquid transformation is either very small or very large. A numerical solution is necesary for intermediate values of the energy ratio. This was accomplished using a finite difference explicit method.

Douglas and Gallie, 1955, solved numerically a modification of Stefan's problem, considering a case with uniform cooling at the surface and an initial liquid temperature equal to the freezing temperature. This last simplification transforms the problem into a single-domain problem, because the liquid phase remains at the freezing temperature all the time. Convergence and stability proofs were also given.

Lotkin, 1960, solved the problem of a slab that is insulated on one side and heated on the opposite side by using the Landau transformation (Landau, 1950) and a finite difference method. The solution allowed physical properties to vary with temperature. The results were compared to the analytical solution for the case of a semi-infinite slab

with constant physical properties, and the results were found to be in good agreement.

More recent research includes solutions such as Tao's, 1967, in which a solution is found for the problem of internal solidification in a convectively-cooled sphere or cylinder. Martinez and Beaubouef, 1972, solved the problem of freezing inside of a pipe when the flow is supplied by a linear pump. The velocity profile is assumed to be parabolic regardless of the presence of the ice, as proposed by Zerkle and Sunderland, 1968. The results are used to predict ice blockage in the pipe as a function of the cooling temperature ratio and the pump characteristics. Saitoh, 1978, approached the solution of multidimensional freezing problems in arbitrary enclosures by using the Landau transformation (Landau, 1950).

Crank, 1981, cites in his review many of the different numerical methods that can be used to solve moving boundary problems. Some methods are based on the application of the Landau transformation, while there are also methods where a fixed-size grid is used. Other applicable methods are the isotherm migration method, the enthalpy method, and the method of lines.

III. DEVELOPMENT OF THE NUMERICAL METHOD

This chapter deals with the development of a numerical model that can solve for ice formation in an annular sector and test the possibilities of the flow reversal method as an alternative for operating an ice-maker heat pump in steady state.

The chapter covers the development of the governing equations, the substitutions employed and the procedure for solving for temperatures and velocities for each one of the three domains. The section on the iteration process explains how all the particular solutions are linked together to obtain the overall results.

3.1 Physical Description of the Model

The evaporator under consideration is shown in figure 3.1. This evaporator consists of two concentric pipes. The refrigerant flows inside the internal pipe, while the water flows in the annulus. There are four straight, uniformly-spaced fins in the annulus. These fins provide a large area for ice build-up. The fins and pipe walls are thin, and the external pipe wall is insulated.

The refrigerant evaporates along the internal pipe, so that its temperature remains constant throughout the evaporator. The refrigerant heat transfer coefficient is found to be a function of the refrigerant quality, but a constant, average value, is used here.

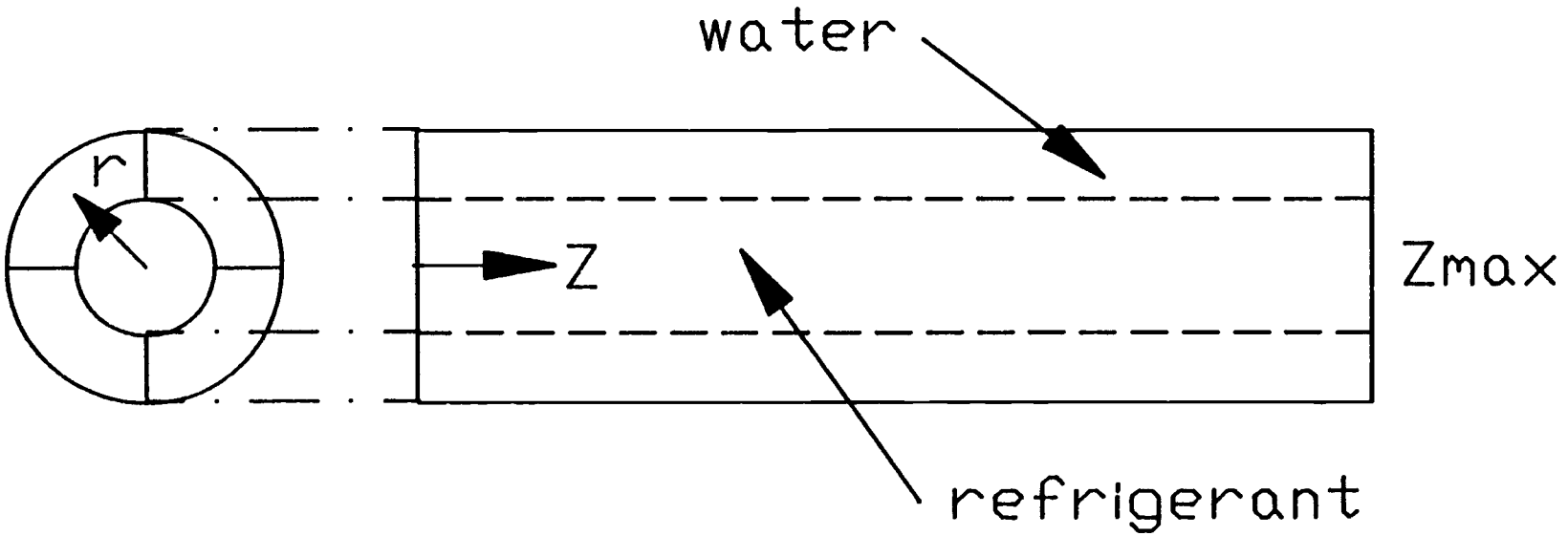

Figure 3.1 Diagram of the evaporator analyzed in the model.

The water flow is laminar throughout the evaporator. The flow is sufficiently fast so that axial conduction is negligible. The ice forms on the wall as soon as the wall temperature becomes lower than the freezing temperature.

At given time intervals, the water flow is reversed. The ice in contact with the metallic wall melts very fast and part of the remaining ice detaches and flows out with the water.

3.2 Mathematical Description of the Model

The equations governing the flow and heat transfer in the evaporator are developed in this section. These equations include the energy equation for the solid and liquid phases and the metallic wall, as well as the momentum and continuity equation for the liquid phase.

This section also discusses the boundary conditions. The assumptions required to simplify the model are indicated as they are used in the process. All the assumptions used are summarized in the last section of the chapter for reference.

The annulus in which the water flows is divided in equal sectors by the fins. This means that, by symmetry, it is only necessary to solve for the flow and heat transfer for half of the area between two consecutive fins. The same solution will apply to all the existing sectors. Figure 3.2 shows the solution domain as it is limited by

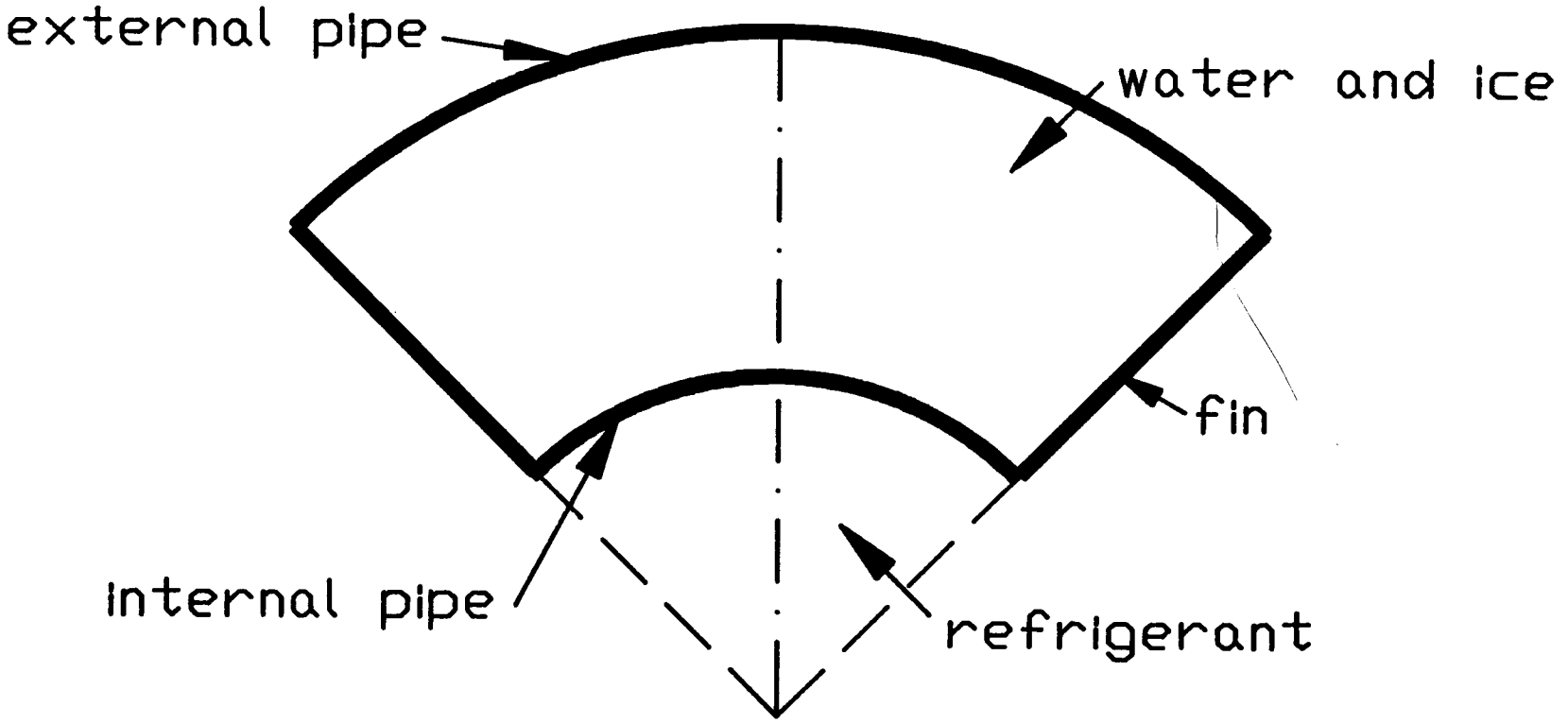

Figure 3.2 Solution domain.

the metallic walls, the fin and the symmetry line between two fins.

3.2.1 Governing Equations: The water flow rate through the evaporator is assumed to be constant. Therefore, the continuity equation can be written in integral form, as expressed by equation 3.1. This equation expresses the requirement that the flow rate is the same at all sections all the time.

$$Q = \int U dA = \text{constant} \tag{3.1}$$

The momentum equation for the liquid phase is:

$$\nabla^2 U = \frac{1}{\mu} \frac{dp}{dZ} \tag{3.2}$$

where the assumptions of laminar, fully developed and steady-state flow are made.

The energy equation for the liquid and solid phases are respectively:

$$\frac{\partial T}{\partial t} + U \frac{\partial T}{\partial Z} = \alpha \nabla^2 T \tag{3.3}$$

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T \tag{3.4}$$

The axial heat conduction is neglected from both equations.

The metallic wall is considered to be very thin. This makes it possible to lump the temperature distribution across the wall thickness. Therefore, the temperature distribution in the wall is assumed to be one-dimensional, and it must conform to the following equation:

$$KsA \frac{\partial^2 T}{\partial x^2} - hP\,(T-T_\infty) = ACps\,\rho s \frac{\partial T}{\partial t} \qquad (3.5)$$

The convective term is rewritten below (section 3.3.3) in terms of heat conduction either to the liquid or solid phase.

3.2.2 Initial and Boundary Conditions: Initially the pipe is assumed to have a uniform temperature To. So the initial condition is written as:

$$T = To \quad \text{at } t=0 \qquad (3.6)$$

where T represents both the fluid and wall temperatures.

The fluid coming into the pipe has a uniform temperature To, equal to the initial temperature. The next equation represents the entrance temperature condition when the

axial conduction is neglected:

$$T = To \quad \text{at } Z=0 \tag{3.7}$$

The boundary conditions for the momentum equation are:

$$\begin{aligned} U &= 0 \quad \text{at the walls and solid interface} \\ \frac{\partial U}{\partial \phi} &= 0 \quad \text{at the symmetry line} \end{aligned} \tag{3.8}$$

The boundary conditions for the energy equation are: for the metal:

$$\frac{\partial T}{\partial \phi} = 0 \text{ at the symmetry line} \tag{3.9}$$

for the solid and liquid:

$$\begin{aligned} T &= Tm \quad \text{at the walls} \\ T &= Tc \quad \text{at the interface} \\ \frac{\partial T}{\partial \phi} &= 0 \quad \text{at the symmetry line} \end{aligned} \tag{3.10}$$

The heat balance at the interface is:

$$Ki \frac{\partial T}{\partial n} = Kf \frac{\partial T}{\partial n} + \rho L \frac{dFn}{dt} \qquad (3.11)$$

where n is the coordinate along the normal to the interface and dFn is the solid phase growth in the normal direction.

3.2.3 Model Assumptions: The assumptions made in the process of writing the model deal mostly with the neglect of second order factors. Some of these have already been used to write the equations in this section and the rest will be applied during the development of the solution method in the last sections. These are:

1. The liquid flow is laminar everywhere and has a fully developed, steady-state velocity profile and a uniform temperature at the entrance.
2. The liquid is newtonian, incompressible and a pure substance.
3. The physical properties of each phase are independent of temperature.
4. Axial heat conduction, viscous energy dissipation, radiant heat transfer and free convection are negligible.
5. The temperature at the liquid-solid interface is constant and equal to the freezing temperature.
6. The metallic walls are thin and the temperature profile in them can be described accurately as one-dimensional.
7. The refrigerant has a constant temperature throughout the pipe and the heat transfer coefficient between the

refrigerant and the pipe wall is constant along the pipe.
8. The external pipe wall is perfectly insulated.
9. The ice layer that forms in the external pipe wall is very thin and grows very slowly.

3.3 Solution Scheme

This section starts with a global view to the methods and the order in which all the operations are done. The later sections indicate the solution procedure for each one of the domains of the problem.

3.3.1 Iteration Process: This section shows how the program works as a whole and the order in which the different operations are done. The presentation is based on the flow chart of the program (figure 3.3). Many blocks in this figure indicate solutions for velocities, temperatures and profiles. These solutions are complex subprograms that are described in the sections 3.3.3 to 3.3.6.

The program starts by solving for the temperature distribution in the metallic wall and for the velocity and temperature distribution in the liquid phase. If there is no freezing (that is, if the water temperature is above the freezing point throughout the domain), the calculation for the current position is finished and the program moves to the next axial step.

If there is freezing, the program assumes an initial

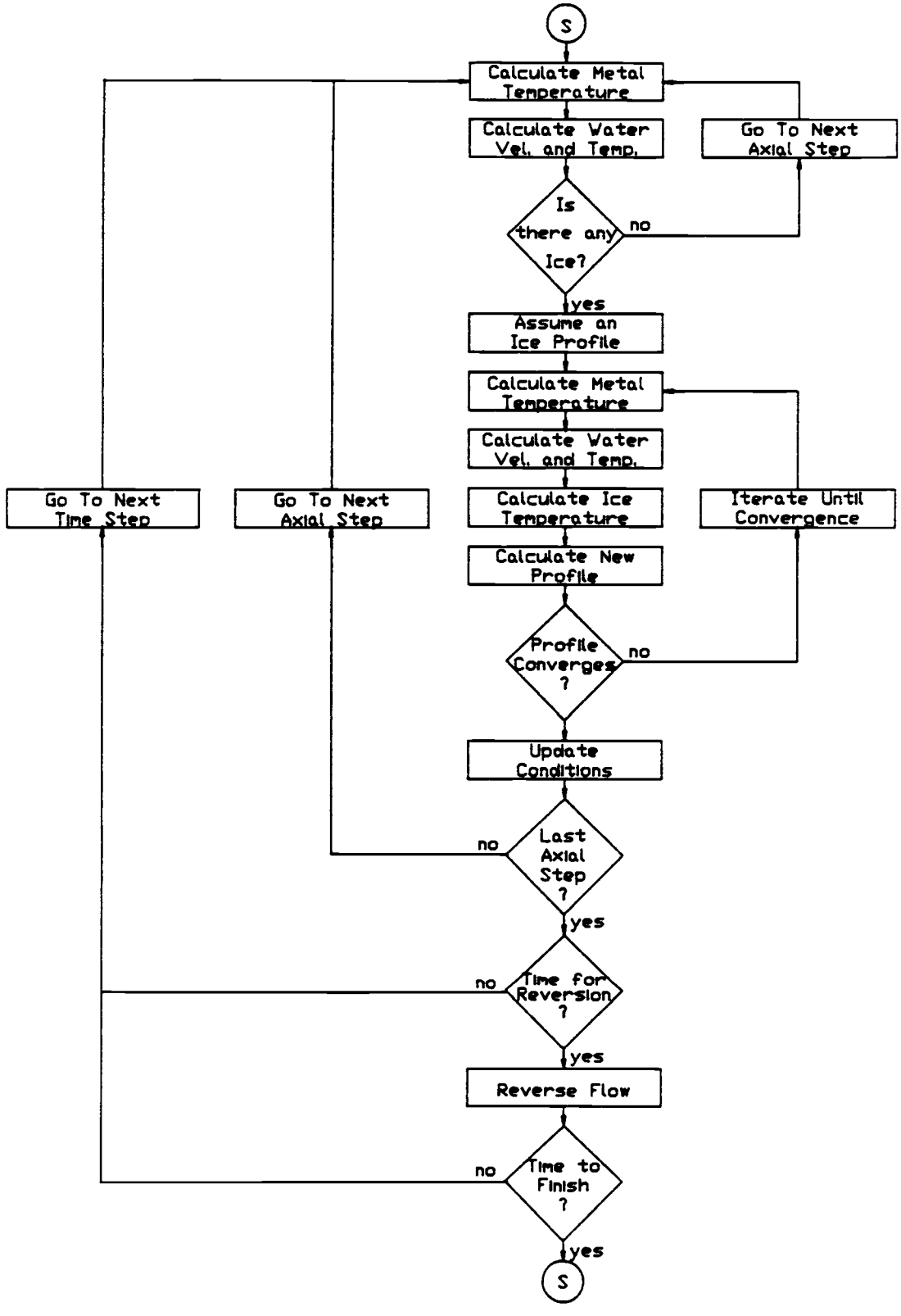

Figure 3.3 Flowchart of the computer program.

ice profile. This step is required because the transformation equation (3.13) is singular when the ice thickness is equal to zero.

The wall temperature and the liquid velocity and temperature are then recalculated. This is necessary because the existence of ice in the pipe alters both the temperature and velocity distributions.

The next step is to calculate the temperature distribution in the ice. With this information, an improved solid-liquid interface position is calculated. The program iterates on the profile until the results of two successive iterations are in close agreement.

The last part of the program updates the temperatures and velocities to the new values. Then the program moves to the next time step or axial step, depending on whether the marching process along the pipe is finished or not. This part also simulates flow reversals at given time intervals.

3.3.2 Coordinate Transformation and Grid Generation: The solution method is based on the Landau transformation (Landau, 1950). In this method, a coordinate transformation is used to fix the position of the interface. The coordinate transformations used here are:

for the liquid phase:

$$r^* = \frac{Ro - r}{Ro - F(\phi, Z, t)}$$
$$\phi^* = \frac{\phi}{\phi_o} \qquad (3.12)$$

for the solid phase:

$$r^* = \frac{r - Ri}{F(\phi, Z, t) - Ri}$$
$$\phi^* = \frac{\phi}{\phi_o} \qquad (3.13)$$

where F is the radial position of the interface, Ri is the internal radius of the pipe and Ro is the external radius of the pipe. All the existing dimensions are shown in figure 3.4.

The partial derivatives appearing in the governing equations are transformed to derivatives in the new system of coordinates by using the chain rule. For example, for the liquid phase:

$$\frac{\partial U}{\partial \phi} = \frac{\partial U}{\partial \phi^*} \frac{\partial \phi^*}{\partial \phi} + \frac{\partial U}{\partial r^*} \frac{\partial r^*}{\partial \phi} \qquad (3.14)$$

and

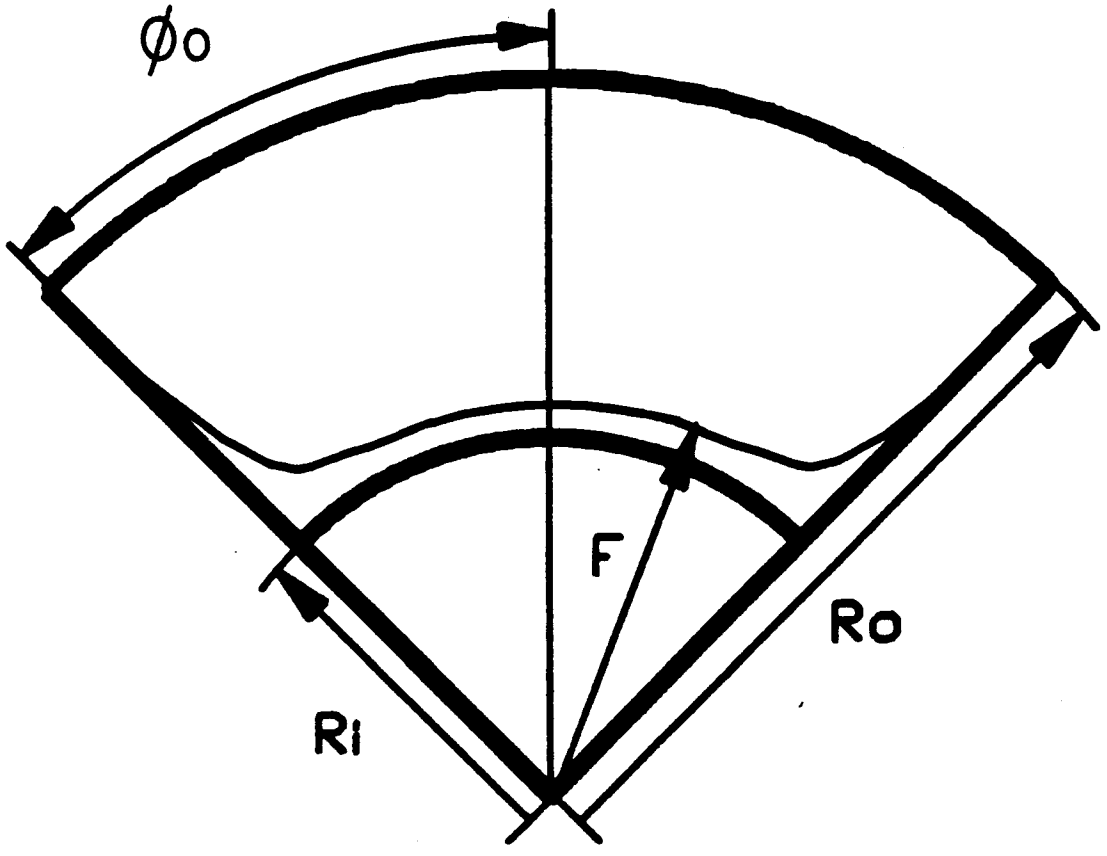

Figure 3.4 Solution domain indicating the pertinent dimensions.

$$\frac{\partial \phi^*}{\partial \phi} = \frac{1}{\phi_o}, \qquad \frac{\partial r^*}{\partial \phi} = \frac{r^* F'}{\phi_o (1-F)} \tag{3.15}$$

The same procedure is used for all the other derivatives.

There is a big advantage to working the solution with non-dimensional variables. The next equations define the non-dimensional parameters that are used during the procedure:

$$Cp^* = \frac{Cp}{Cpf} \qquad \text{specific heat} \tag{3.16}$$

$$F^* = \frac{F}{Ro} \qquad \text{interface position} \tag{3.17}$$

$$K^* = \frac{K}{Kf} \qquad \text{thermal conductivity} \tag{3.18}$$

$$Nu = \frac{h\ Ro}{Kf} \qquad \text{Nusselt number} \tag{3.19}$$

$$r^+ = \frac{r}{Ro} \qquad \text{radius} \tag{3.20}$$

$$Ri^* = \frac{Ri}{Ro} \qquad \text{radius ratio} \tag{3.21}$$

$$Ste = \frac{Cpf\ (To-Tc)}{L} \qquad \text{Stefan number} \tag{3.22}$$

$$t^* = \frac{2Kf\ (To-Tc)\ t}{\rho\ L\ Ro^2} \qquad \text{time} \tag{3.23}$$

$$to^{*} = \frac{to}{Ro} \qquad \text{wall thickness} \qquad (3.24)$$

$$T^{*} = \frac{T - Tc}{To - Tc} \qquad \text{temperature} \qquad (3.25)$$

$$V^{*} = \frac{\pi U (Ro^{2} - Ri^{2})}{Q} \qquad \text{velocity} \qquad (3.26)$$

$$Z^{*} = \frac{\alpha (Ro^{2} - Ri^{2}) Z}{Ro^{2} Q} \qquad \text{axial coordinate} \qquad (3.27)$$

$$\rho^{*} = \frac{\rho}{\rho f} \qquad \text{density} \qquad (3.28)$$

The use of different coordinate transformations for the solid and liquid phases yields two different expressions for the governing equations and the boundary conditions. One of them applies when solid is present and the other applies when liquid is present. However, these two different forms are similar and they can be written as a single equation in terms of the following generalized parameters:

$$Kx^{*} = \frac{Kx}{Kf}, \text{ and } Kx = \begin{cases} Ki & \text{if solid is present} \\ Kf & \text{if liquid is present} \end{cases} \qquad (3.29)$$

$$Sx^{*} = \begin{cases} \dfrac{1}{F^{*} - Ri^{*}} & \text{if solid is present} \\ -\dfrac{1}{1 - F^{*}} & \text{if liquid is present} \end{cases} \qquad (3.30)$$

The division of the domain in subintervals is shown in figure 3.5. This figure also shows the values of the subindexes i,j in the domain.

The transformed coordinates r^* and ϕ^* are valued in the following ranges:

$$0< r^* < 1$$
$$0< \phi^* < 1 \qquad (3.31)$$

The domain is divided in M subintervals in the ϕ - direction, N subintervals in the radial direction for the liquid phase and Q subintervals in the radial direction for the solid phase. The program allows an irregular partition in the ϕ-direction to improve the accuracy, because the ice profiles present a steep slope near the fin. This situation requires the use of a small value of $\Delta\phi$ near the wall, while a larger value of $\Delta\phi$ can be used for the other points.

3.3.3 Solution for the metallic wall: The metallic wall is divided into grid points according to one of the three partitions shown in figure 3.6. These possibilities exist so that the grid points in the wall correspond exactly to the boundary grid points in the liquid and solid phases.

The equation for the wall is the one-dimensional fin

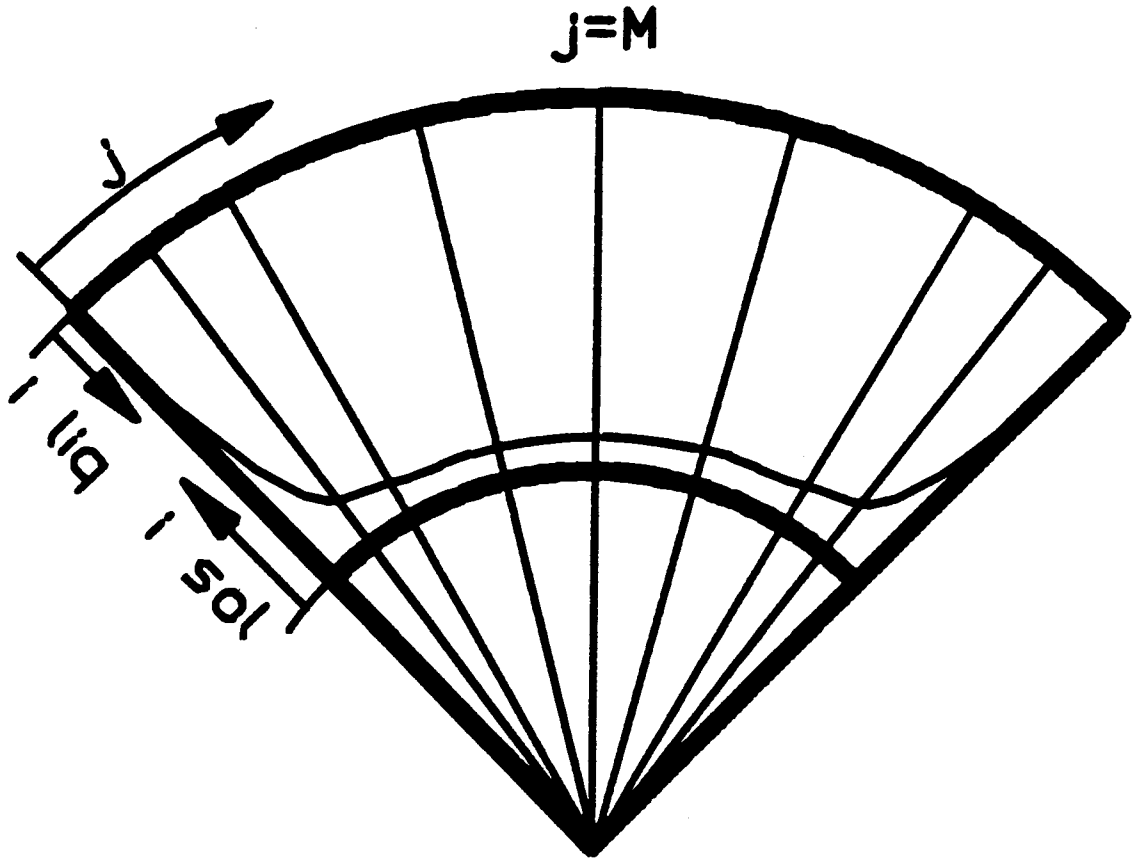

Figure 3.5 Divison of the domain in angular subintervals.

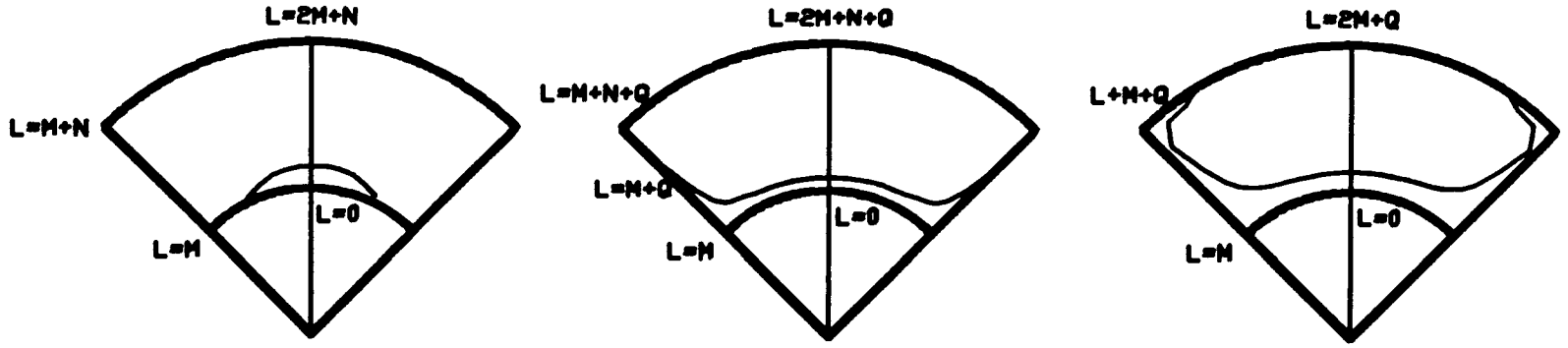

Figure 3.6 The three possible partitions of the domain for the metallic wall.

equation:

$$K_s A \frac{\partial^2 T}{\partial x^2} - hP\,(T - T_\infty) = ACp_s \rho_s \frac{\partial T}{\partial t} \qquad (3.32)$$

This equation takes different forms depending on the grid point location and whether the solid or liquid phase is in contact with the wall. These forms are shown next for all the existing possibilities. The equations are shown first in regular r-ϕ coordinates and then in the transformed coordinates r^*-ϕ^*. The equations are written in terms of the generalized parameters (equations 3.29 and 3.30). This makes it possible to write a single equation that is valid for both cases, either solid in contact with the wall or liquid in contact with the wall.

Next there is a list of the locations along the wall and the equations that apply for each location.

At the internal pipe wall:

$$\frac{toKs}{Ri^2} \frac{\partial^2 T}{\partial \phi^2} = - Kx \frac{\partial T}{\partial r} + h\,(T - Tw) \qquad (3.33)$$

The transformed equation is:

$$\frac{to^* Ks^*}{\phi_o{}^2 Ri^{*2}} \frac{\partial^2 T^*}{\partial \phi^{*2}} = - Kx^* Sx^* \frac{\partial T^*}{\partial r^*} + Nu\,(T^* - Tw^*) \qquad (3.34)$$

At the inner corner, a heat balance yields:

$$\frac{toKs}{2}\frac{\partial T}{\partial r} = -\frac{Ksto}{Ri}\frac{\partial T}{\partial \phi} + hRi\frac{\Delta\phi}{2}(T - Tw) \qquad (3.35)$$

The transformed equation is:

$$\frac{to^*Ks^*Sx^*}{2}\frac{\partial T^*}{\partial r^*} = -\frac{Ks^*to^*}{Ri^*\phi o}\frac{\partial T^*}{\partial \phi^*} + \frac{NuRi^*\Delta\phi^*\phi o}{2}(T^* - Tw^*) \qquad (3.36)$$

At the fin, $\phi = 0$, the equation is:

$$\frac{toKs}{2}\frac{\partial^2 T}{\partial r^2} = -\frac{Kx}{r}\frac{\partial T}{\partial \phi} + \frac{toCp_s\rho_s}{2}\frac{\partial T}{\partial t} \qquad (3.37)$$

And the transformed equation is:

$$\frac{to^*Ks^*Sx^{*2}}{2}\frac{\partial^2 T^*}{\partial r^{*2}} = -\frac{Kx^*}{r^+\phi o}\frac{\partial T^*}{\partial \phi^*} + \frac{Kx^*r^*F'^*Sx^*}{r^+\phi o}\frac{\partial T^*}{\partial r^*} + Ste\, to^*Cps^*\rho s^*\frac{\partial T^*}{\partial t^*} \qquad (3.38)$$

The energy balance at the external corner yields:

$$\frac{4toKs}{Ro}\frac{\partial T}{\partial \phi} - 2Ksto\frac{\partial T}{\partial r} = to\ \rho sCps\frac{\partial T}{\partial t}(2Ro\Delta\phi + \Delta r) \tag{3.39}$$

The transformed equation is:

$$\frac{Ks^*}{\phi_o}\frac{\partial T^*}{\partial \phi^*} - \frac{Ks^*Sx^*}{2}\frac{\partial T^*}{\partial r^*} = \rho s^*Cps^*Ste\frac{\partial T^*}{\partial t}\left(\Delta\phi^*\ \phi o + \frac{\Delta r^*}{2\ abs(Sx^*)}\right) \tag{3.40}$$

where $abs(Sx^*)$ represents the absolute value of Sx^*.

At the external pipe wall:

$$\frac{toKs}{Ro^2}\frac{\partial^2 T}{\partial \phi^2} = Kx\frac{\partial T}{\partial r} + toCp_s\ \rho_s\frac{\partial T}{\partial t} \tag{3.41}$$

The transformed equation is:

$$\frac{to^*Ks^*}{\phi_o{}^2}\frac{\partial^2 T^*}{\partial \phi^{*2}} = -\frac{Kf^*}{1-F^*}\frac{\partial T^*}{\partial r^*} + 2Ste\ to^*Cp_s{}^*\ \rho_s{}^*\frac{\partial T^*}{\partial t^*} \tag{3.42}$$

This equation is not written in terms of the general parameters because the assumption of a thin layer of ice at the top surface makes it possible to use the values for the liquid phase even if there is ice on it. See section 3.3.6 for the analysis of this point.

All the metallic wall equations are transformed into

finite difference equations. The resulting matrix is tri-diagonal and a U-L decomposition method is used (Atkinson, 1978).

3.3.4 Solution for the Liquid Phase: There are two things to solve for in the liquid phase. One is the velocity field and the other is the temperature field. The continuity and momentum equations are used to solve for the velocity profile, as indicated by the next equations.

The momentum equation is:

$$\nabla^2 U = \frac{1}{\mu} \frac{dp}{dZ} \tag{3.43}$$

The laplacian in cylindrical coordinates is:

$$\nabla^2 U = \frac{\partial^2 U}{\partial r^2} + \frac{1}{r} \frac{\partial U}{\partial r} + \frac{1}{r^2} \frac{\partial^2 U}{\partial \phi^2} \tag{3.44}$$

Transforming into r^*- ϕ^* coordinates, the result is:

$$Ro^2 \nabla^2 U = a \frac{\partial^2 U}{\partial r^2} + b \frac{\partial^2 U}{\partial \phi^* \partial r^*} + c \frac{\partial^2 U}{\partial \phi^{*2}} + d \frac{\partial U}{\partial r^*} \tag{3.45}$$

where the coefficients a,b,c,d are given by:

$$a = Sx^{*2} + \frac{Sx^{*2} r^{*2} F'^{*2}}{r^{+2} \phi_o^{2}} \tag{3.46}$$

$$b = - \frac{2 r^{*} F'^{*} Sx^{*}}{r^{+2} \phi_o^{2}} \tag{3.47}$$

$$c = \frac{1}{r^{+2} \phi_o^{2}} \tag{3.48}$$

$$d = \frac{Sx^{*}}{r^{+}} - \frac{Sx^{*} r^{*} F^{*''}}{r^{+2} \phi_o^{2}} + \frac{2 Sx^{*2} r^{*} (F^{*'})^{2}}{r^{+2} \phi_o^{2}} \tag{3.49}$$

The generalized parameter Sx^{*} is used so that these equations are valid also for the solid phase.

The substitution

$$U^{*} = \frac{U \mu}{Ro^{2} \frac{dP}{dZ}} \tag{3.50}$$

is used to adimensionalize the equation. The result is:

$$a \frac{\partial^2 U^{*}}{\partial r^{*2}} + b \frac{\partial^2 U^{*}}{\partial \phi^{*} \partial r^{*}} + c \frac{\partial^2 U^{*}}{\partial \phi^{*2}} + d \frac{\partial U^{*}}{\partial r^{*}} = 1 \tag{3.51}$$

And the boundary conditions are:

$$
\begin{aligned}
r^* &= 0, & U^* &= 0 \\
r^* &= 1, & U^* &= 0 \\
\phi^* &= 0, & U^* &= 0 \\
\phi^* &= 1, & \frac{\partial U^*}{\partial \phi^*} &= 0
\end{aligned}
\qquad (3.52)
$$

This is enough information to solve for U^*. The matrix resulting from equation (3.51) is solved iteratively using the succesive overrelaxation (SOR) method. However, it is still necessary to solve for the dimensional value of the velocity U and the pressure drop along the pipe is not known. The continuity equation can be used for this purpose. This equation is:

$$Q = Ua\ A = \int U\ dA = \text{constant} \qquad (3.53)$$

Substituting U by U^* (equation 3.50):

$$Q = Ua\ A = \frac{Ro^2 \dfrac{dP}{dx}}{\mu} \int U^*\ dA \qquad (3.54)$$

Defining

$$Ua^* = \frac{1}{A} \int U^*\ dA \qquad (3.55)$$

Equation (3.54) results:

$$Ua\ A = Ua^*\ A\ \frac{Ro\ \frac{dp}{dZ}}{\mu} \qquad (3.56)$$

Dividing equation (3.50) by equation (3.56):

$$\frac{U}{Ua} = \frac{U^*}{Ua^*} \qquad (3.57)$$

The values of Ua and Ua^* are calculated from their definitions (equations 3.53 and 3.55 respectively). The integral defining Ua^* is calculated numerically. The area element dA=rdrd is first transformed to coordinates r^*-ϕ^* by using the relation $dA = J\ dA^*$, where J is the Jacobian of the transformation. The result is:

$$dA = Ro^2\ (1-F^*)\ r^+\ \phi o\ dr^* d\phi^* \qquad (3.58)$$

The result for Ua^* is:

$$Ua^* = \frac{1}{A} \int U^*\ Ro^2\ (1-F^*)\ r^+\ \phi o\ dr^* d\phi^* \qquad (3.59)$$

The integration is done by using the two-dimensional Simpson's method. Now Ua, U^* and Ua^* are known and U can be calculated from equation (3.57).

This completes the calculations. Now the velocity field is known and it is possible to solve for the temperature. The energy equation is:

$$\frac{\partial T}{\partial t} + U \frac{\partial T}{\partial Z} = \alpha \nabla^2 T \qquad (3.60)$$

or, in non-dimensional form:

$$2Ste \frac{\partial T^*}{\partial t^*} + V^* \frac{\partial T^*}{\partial Z^*} = a \frac{\partial^2 T^*}{\partial r^{*2}} + b \frac{\partial^2 T^*}{\partial r^* \partial \phi^*} + c \frac{\partial^2 T^*}{\partial \phi^{*2}} + \qquad (3.61)$$

$$+ \left(d - 2Ste \frac{r^*}{1-F^*} \frac{\partial F^*}{\partial t^*} - V^* \frac{r^*}{1-F^*} \frac{\partial F^*}{\partial Z^*} \right) \frac{\partial T^*}{\partial r^*}$$

There are three different kinds of boundary conditions:

$$\begin{array}{lll} 1. \ \dfrac{\partial T^*}{\partial \phi^*} = 0 & \text{at} \quad \phi^* = 0 & \text{symmetry line} \\ 2. \ T^* = 0 & \text{at} \ r^* = 1 & \text{ice-water interface} \\ 3. \ T^* = Ts^* & & \text{at wall-liquid interface} \end{array} \qquad (3.62)$$

Equation (3.61) is written in finite differences and

the results for the temperature are found by using the SOR method. An upwind finite difference formula is used for the derivative in the axial direction, as given by Patankar, 1980.

3.3.5 Solution for the solid phase: The energy equation for the solid phase is:

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T \tag{3.63}$$

or, in non-dimensional form:

$$\frac{2Ste}{Ki^*}\frac{\partial T^*}{\partial t^*} = a\frac{\partial^2 T^*}{\partial r^{*2}} + b\frac{\partial^2 T^*}{\partial r^* \partial \phi^*} + c\frac{\partial^2 T^*}{\partial \phi^{*2}} + \left(d + \frac{2Ste\, r^*}{Ki^*(F^* - Ri^*)}\frac{\partial F^*}{\partial t^*}\right)\frac{\partial T^*}{\partial r^*} \tag{3.64}$$

The boundary conditions are the same as those for the liquid phase:

$$\begin{aligned} &1.\ \frac{\partial T^*}{\partial \phi^*} = 0 \quad \text{at} \quad \phi^* = 0 \quad \text{symmetry line} \\ &2.\ T^* = 0 \quad \text{at} \quad r^* = 1 \quad \text{solid-liquid interface} \\ &3.\ T^* = Ts^* \quad \text{at wall-solid interface} \end{aligned} \tag{3.65}$$

The equation is again solved iteratively with a SOR method.

3.3.6 Solution for the Interface Position: The interface position is found by using the enthalpy balance equation at the interface. This equation is:

$$Ki \frac{\partial T}{\partial n} = Kf \frac{\partial T}{\partial n} + \rho L \frac{dFn}{dt} \qquad (3.66)$$

The derivatives in equation (3.66) have to be written as derivatives in the radial direction instead of derivatives in the normal direction. To accomplish this, it is necessary to write a coordinate transformation between two orthogonal systems. One of them is the regular polar coordinate system r, ϕ, z. The second one, n, ϕ', z', has the first coordinate axis n normal to the interface. The other two coordinate axis lie on the interface, ϕ' on the $r-\phi$ plane and z' on the r-z plane. The normal and radial directions and the angle α between them are shown in figure 3.7.

The transformed coordinate system is the result of first rotating the original system r, ϕ, z an angle α around the z axis and then an angle β around the ϕ axis.

The coordinate transformations can be written in terms of matrix operations as follows:

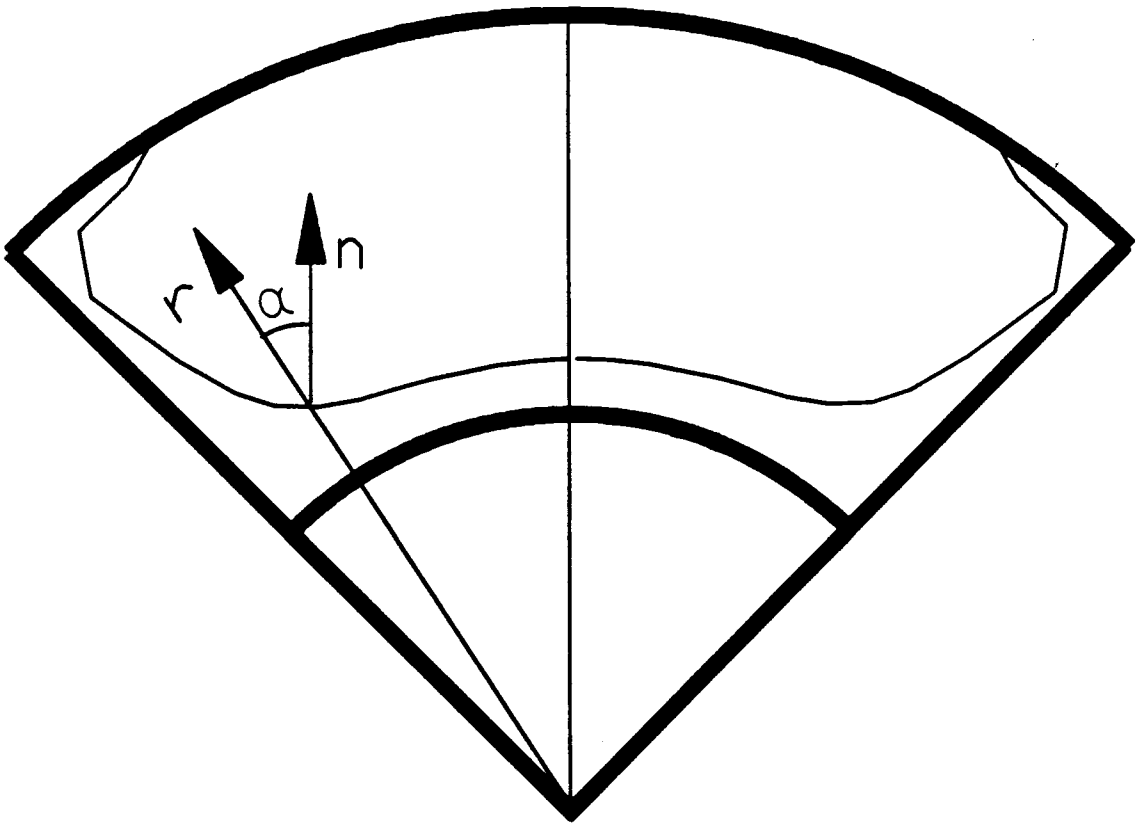

Figure 3.7 Radial and normal directions.

$$\begin{bmatrix} r \\ \phi \\ z \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix} \begin{bmatrix} n \\ \phi' \\ z' \end{bmatrix} \tag{3.67}$$

The result of this transformation is:

$$r = \cos\alpha \cos\beta\, n - \sin\alpha\, \phi' - \cos\alpha \sin\beta\, z' \tag{3.68}$$

The transformation required can be obtained by using the chain rule, as follows:

$$\frac{\partial T}{\partial r} = \frac{\partial T}{\partial n}\frac{\partial n}{\partial r} + \frac{\partial T}{\partial \phi'}\frac{\partial \phi'}{\partial r} + \frac{\partial T}{\partial z'}\frac{\partial z'}{\partial r} \tag{3.69}$$

but ϕ' and z' lie on the interface, where the temperature is always equal to the freezing temperature, then:

$$\frac{\partial T}{\partial \phi'} = \frac{\partial T}{\partial z'} = 0 \tag{3.70}$$

and:

$$\frac{\partial T}{\partial r} = \frac{\partial T}{\partial n} \cos\alpha \cos\beta \tag{3.71}$$

The right hand side of the enthalpy balance equation

(3.66) has to be written in terms of the ice growth in the radial direction, instead of the growth in the normal direction. The transformation equation 3.68 is used for this purpose. The differential form of equation 3.68 is:

$$dr = \cos\alpha\cos\beta\, dn - \sin\alpha\, d\phi' - \cos\alpha\, \sin\beta\, dz' \quad (3.72)$$

Since there is no ice growth in the ϕ' and z' directions, $d\phi'=0$ and $dz'=0$. Observing also that dr=dF, because F is measured in the radial direction, the final result is:

$$Ki \frac{\partial T}{\partial r} - Kf \frac{\partial T}{\partial r} = \rho L \cos^2\alpha \cos^2\beta \frac{\partial F}{\partial t} \quad (3.73)$$

The pipe is very long with respect to its diameter and the amount of ice does not increase very abruptly along the pipe. This implies that the value of $\cos\beta$ will be very close to 1. This simplification is used along the program. The non-dimensional form of (3.73) is:

$$\frac{Ki^*}{F^*-Ri^*} \frac{\partial T^*}{\partial r^*} + \frac{1}{1-F^*} \frac{\partial T^*}{\partial r^*} = 2\cos^2\alpha \frac{\partial F^*}{\partial t^*} \quad (3.74)$$

The first derivative on the left hand side of equation 3.74 is evaluated in the solid phase, while the second

derivative is evaluated in the liquid phase.

The coordinate transformation used to solve this problem (equations 3.12 and 3.13) does not allow for ice formation on the external pipe wall, because F is assumed to be a single-valued function of the angle ϕ. However, the transformations 3.12 and 3.13 can still be applied if there is ice on the external pipe wall, as long as the ice is not very thick and does not grow very fast. This is, tu<< Ro-F. This situation is illustrated in figure 3.8.

The ice growing at the external pipe wall constitutes a fourth domain for which it is necessary to solve in this analysis. The presence of a fourth domain would greatly complicate the solution procedure. However, an approximate solution can be used for the external layer of ice, if this is thin and grows slowly.

Consider equation 3.73. If the ice grows very slowly, the derivative of F in time is very nearly equal to zero and then equation 3.73 becomes:

$$Ki \frac{\partial T}{\partial r} = Kf \frac{\partial T}{\partial r} \qquad (3.75)$$

The term on the left hand side of equation 3.62 appears as a boundary condition for the metallic wall equation. Equation (3.62) makes it possible to write this term as a function of derivatives of the liquid phase, as

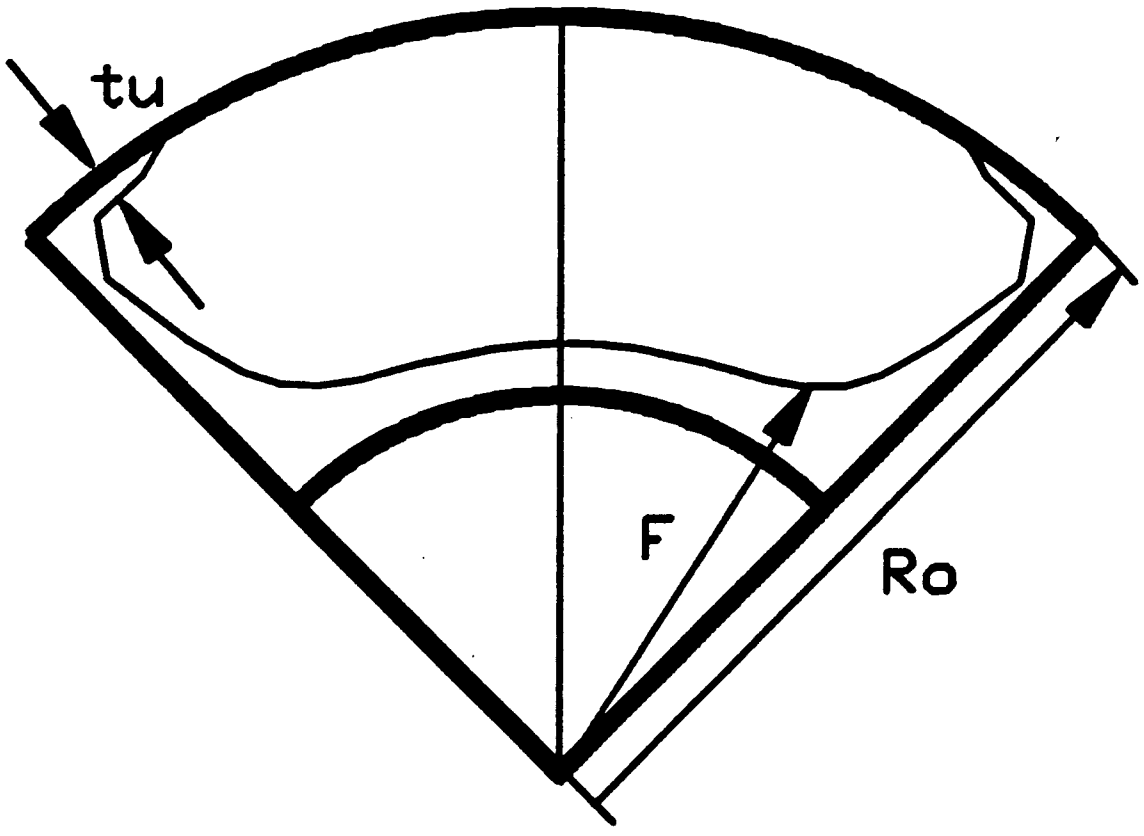

Figure 3.8 Ice formation in the external pipe wall.

stated in section 3.3.3. This makes it unnecessary to define the ice at the top as a fourth domain.

3.3.7 Melting and detaching criterion: The metallic walls have a very low specific heat and a very high thermal conductivity. This means that the layer of ice in contact with the wall reacts immediately to the warming effect of a flow reversion and melts very fast, leaving a portion of the remaining ice structurally too weak to support the pressure exerted by the flow, causing it to break and flow out with the water. The program uses a criterion to simulate this condition. This criterion is just a preliminary one, subject to further study and research. The criterion is:

The part of the existing ice that will detach upon reversal is determined by how far the superficial layer of ice melts. If the superficial layer of ice only melts up to a point along the fin (figure 3.9), the ice which exists between this point and the external pipe wall detaches, while the rest of the ice remains in the pipe. If the superficial layer of ice melts down to a point in the internal pipe wall, all the ice existing between this point and the fin detaches, while the rest of the ice remains in the pipe.

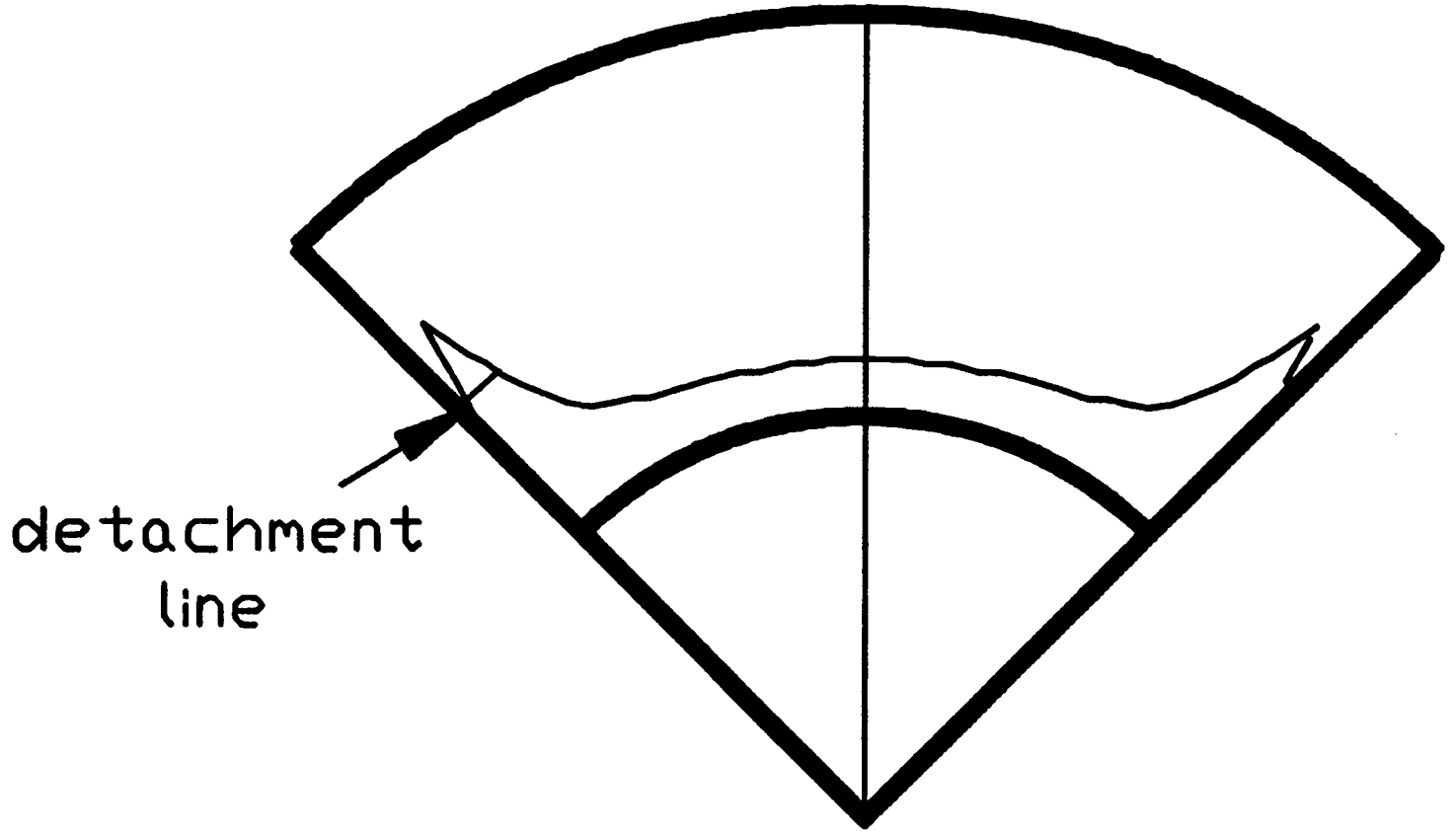

Figure 3.9 Portion of the ice that breaks away from the pipe.

IV. RESULTS AND CONCLUSIONS

4.1 Results

The computer program was executed for different values of the refrigerant heat transfer coefficient while keeping fixed all the evaporator dimensions and water volumetric flow rate. The physical dimensions of the evaporator are given in table 4.1.

The program was executed twice for each value of the heat transfer coefficient. considering first a case of flow reversal every 200 sec. and then a case with no flow reversal.

The results are shown in figures 4.1 to 4.15. Figures 4.1 to 4.5 show the ice profiles obtained from the program as a function of axial position along the evaporator. For the non-reversal case, the profiles change until they reach their steady-state configurations. These are shown in figure 4.1 for a heat transfer coefficient h=1500 $W/m^{2o}C$. Figure 4.1 corresponds to a time t=1200 sec. after the process of cooling the evaporator started. Steady-state conditions were practically reached for t=1000 sec. for all the values of the heat transfer coefficient. For flow reversal, the ice profiles change constantly, but after a few (usually 6 or 7) reversal cycles the profiles vary periodically, so that the ice shape at time=t is the same as that for time=t+400 sec. This periodical variation of

Table 4.1 Evaporator dimensions and physical conditions used during program execution.

Dimension	Value
External pipe radius	0.02 m
Internal pipe radius	0.01 m
Number of fins in annulus	4
Pipe wall thickness	0.001 m
Fin thickness	0.001 m
Initial temperature	$5^{o}C$
Refrigerant temperature	$-5^{o}C$
Water flow rate	0.0001 m^3
Reversal time	200 sec.

Pipe walls and fins are made of copper.

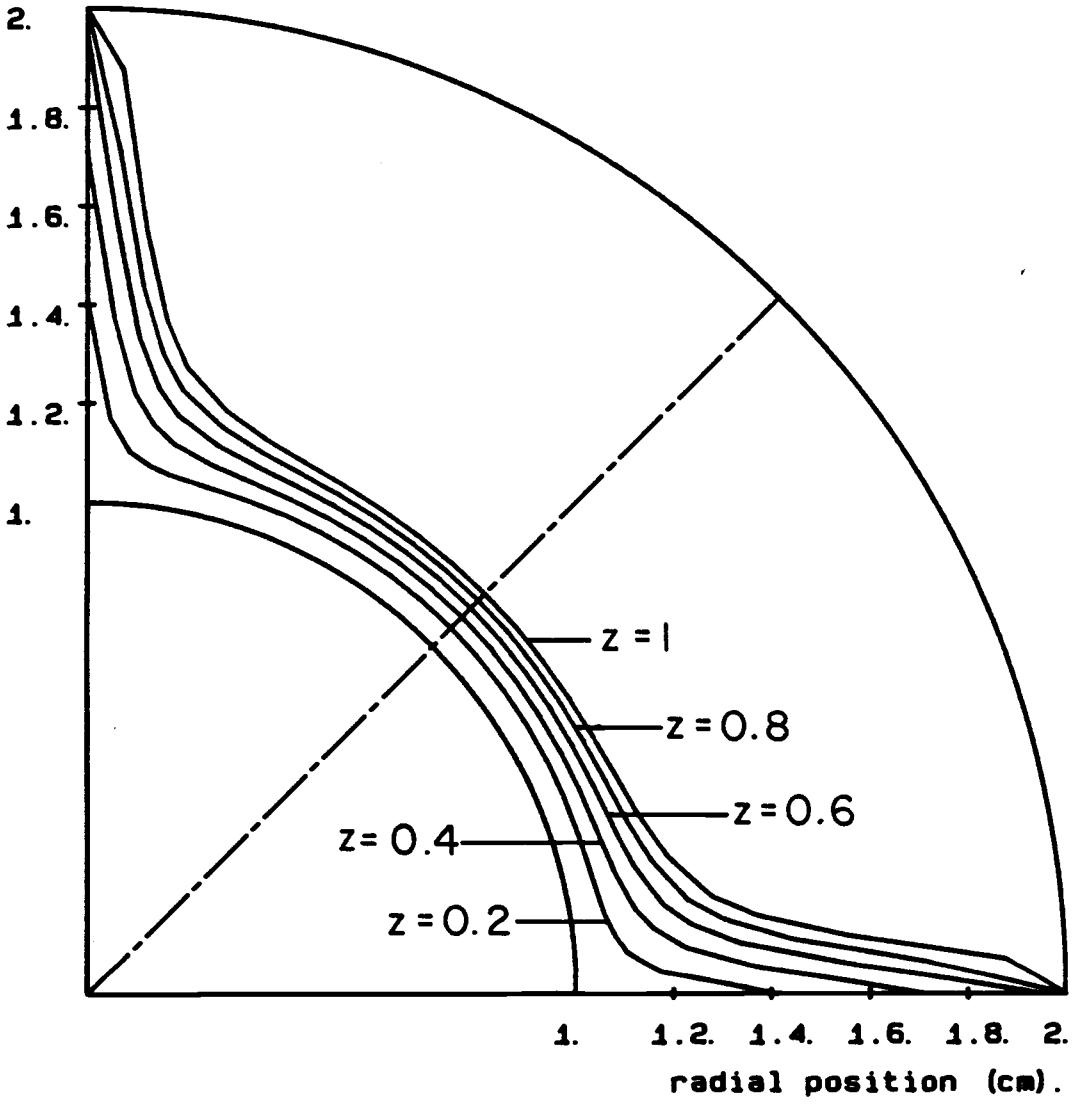

Figure 4.1 Steady-state ice profiles for h = 1500 $w/m^2\ ^oC$ as a function of the axial position z.

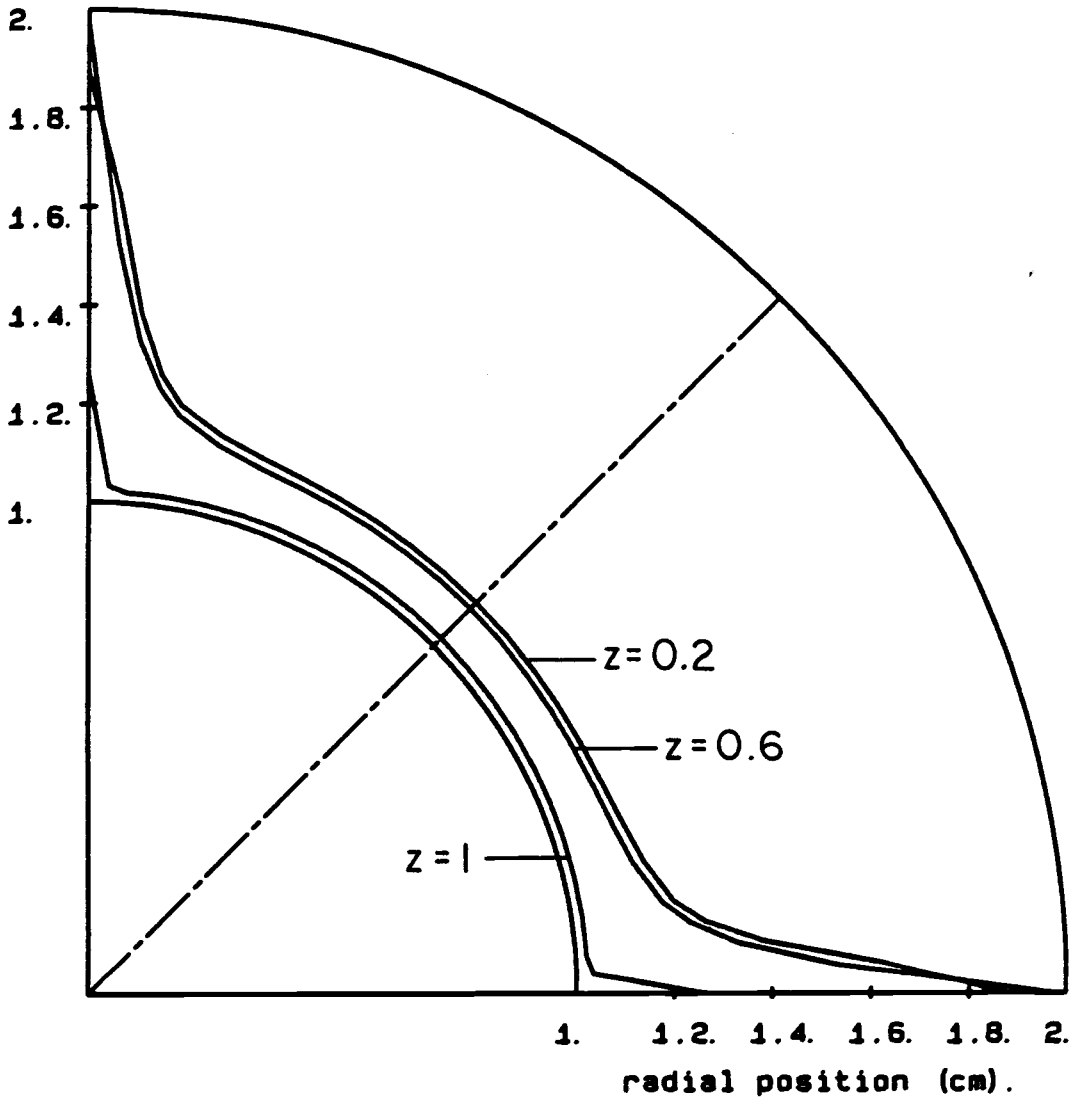

Figure 4.2 Periodical variation of the profiles. Ice profiles at t = 10 sec after reversal.

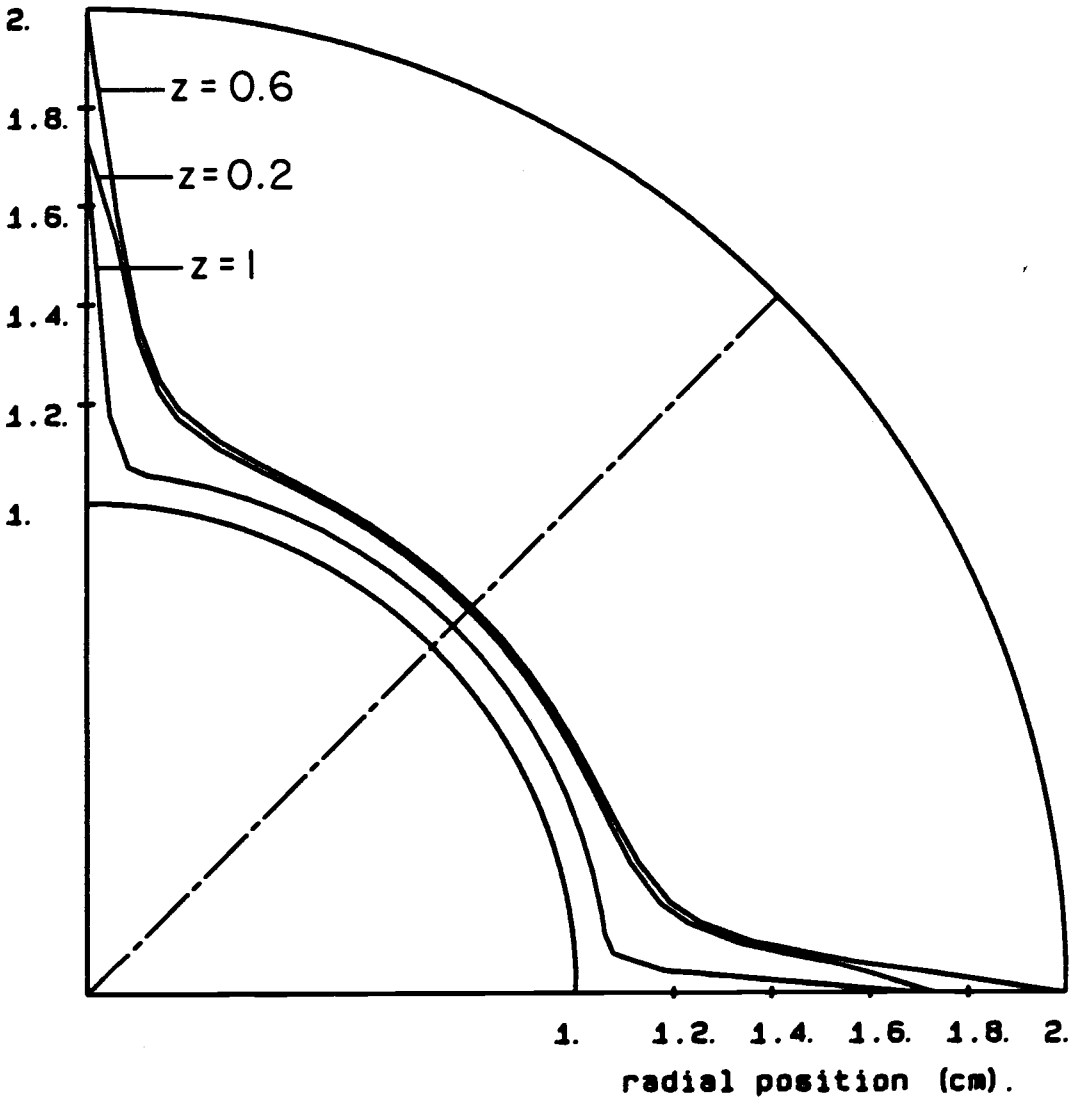

Figure 4.3 Periodical variation of the profiles. Ice profiles at t = 50 sec after reversal.

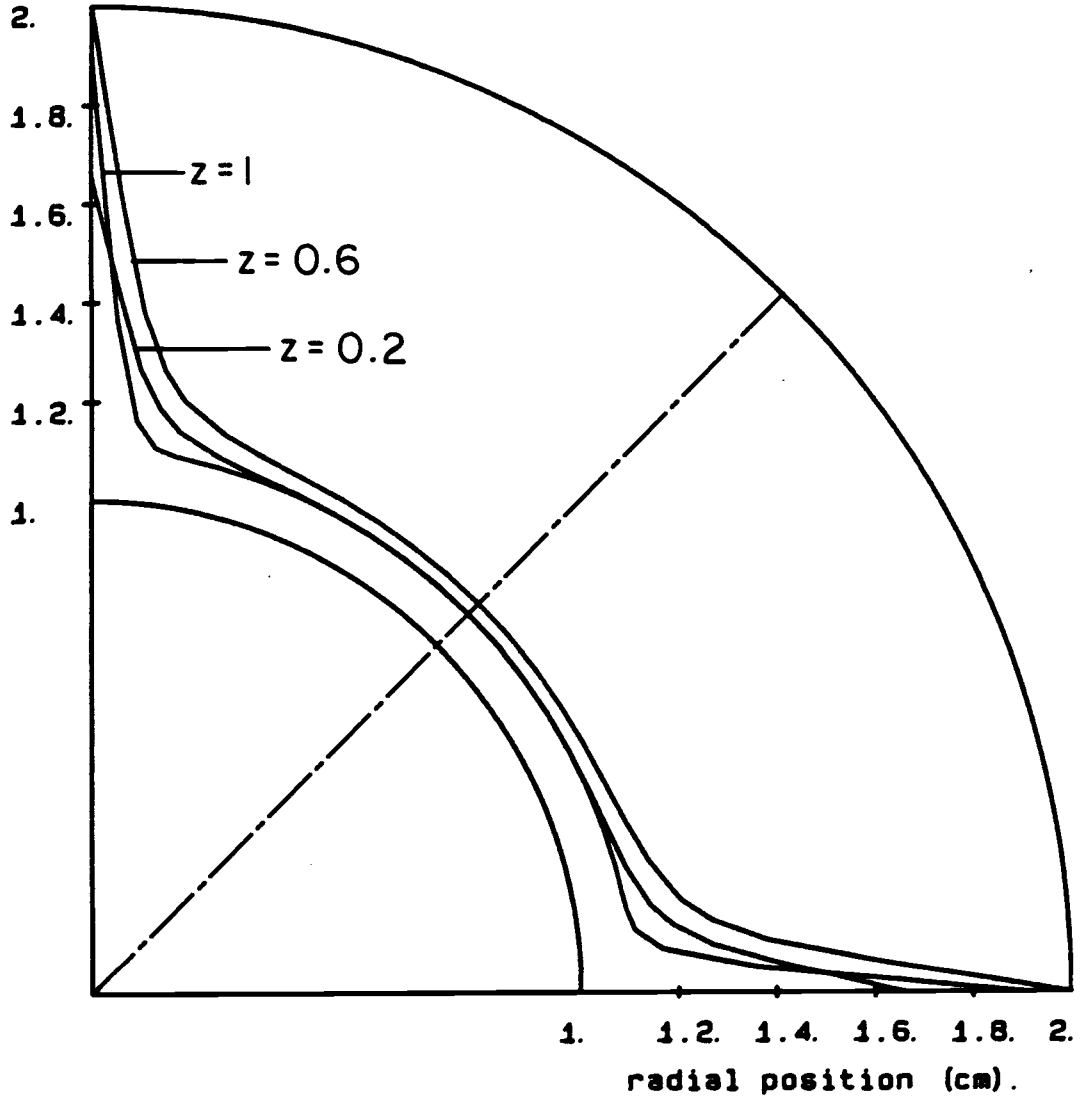

Figure 4.4 Periodical variation of the profile. Ice profiles at t = 100 sec after reversal.

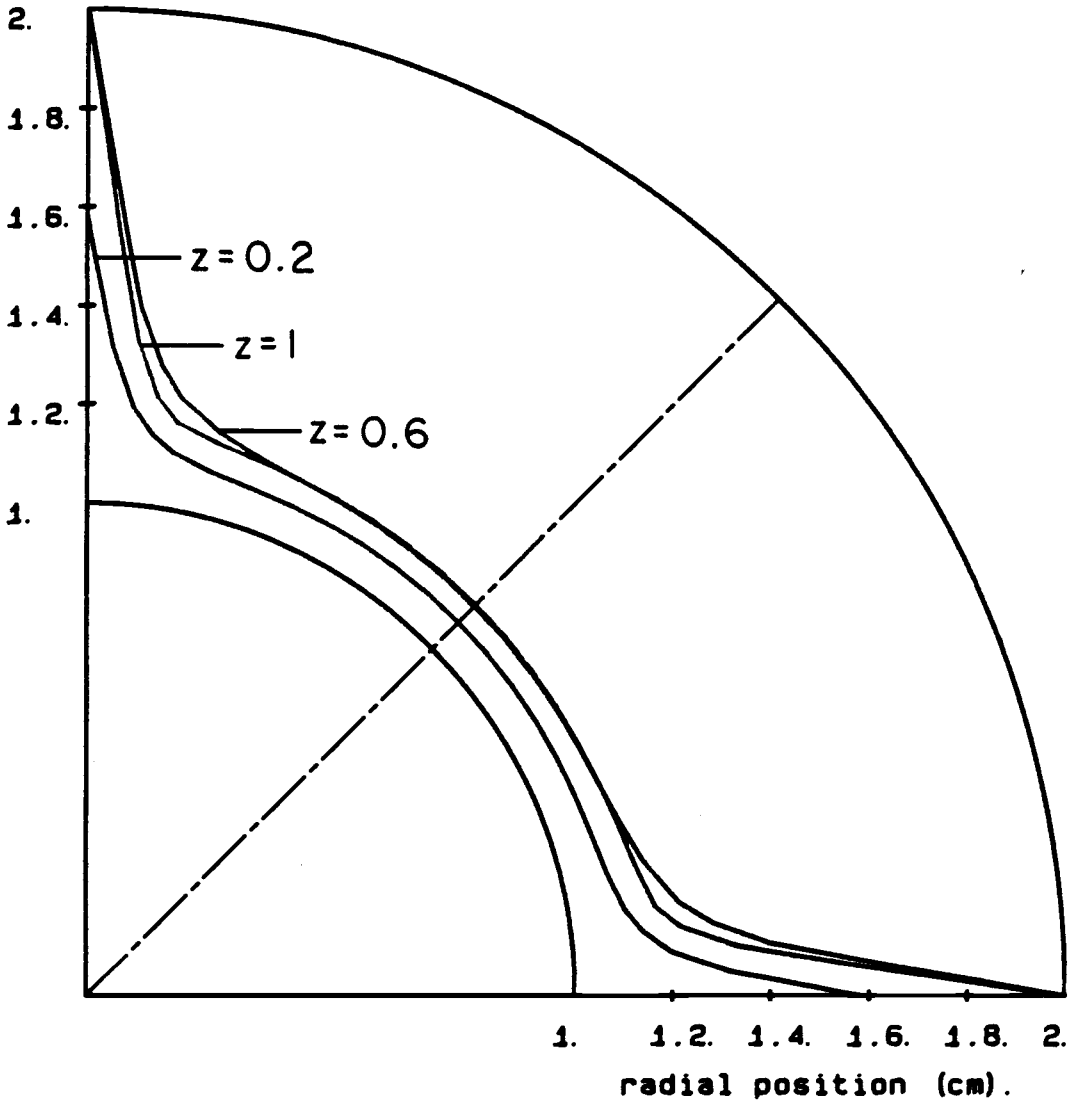

Figure 4.5 Periodical variation of the profile. Ice profiles at t = 200 sec after reversal.

the profile is shown in figures 4.2 to 4.5. These figures show the ice profiles for h=1500 $W/m^{2o}C$ at times after the last reversal of 10, 50, 100 and 200 seconds. Figure 4.5, corresponding to 200 sec. after reversal, shows the ice profile at the moment in which the flow is reversed again. Figures 4.2 to 4.5 show that the evaporator operating in the reversal case has more ice at the entrance than at the exit for a significant period of time after a reversal.

Figures 4.6 and 4.7 show the heat transfer values (energy lost by the water) for heat transfer coefficients of 1500 and 2250 $W/m^{2o}C$ respectively. The curves in the figures show energy transfer values in the evaporator. The topmost curve shows the overall energy transfer. The curve labelled "sensible" indicates sensible energy lost by the water (i.e., energy lost in cooling the water). The "latent" curve indicates latent energy lost by the water stream (i.e., energy lost in freezing the water), including latent energy in ice that detaches and flows out of the pipe. The "latent in pipe" curve indicates latent energy lost by the water stream, considering only the ice that remains in the pipe.

Figures 4.8 to 4.11 show the overall average powers. The results for the reversal case are compared with those for the non-reversal case for values of h=1500, 1750, 2000 and 2250 $W/m^{2o}C$. The curves start from a power equal to zero at time t=0. The power grows very fast from this point

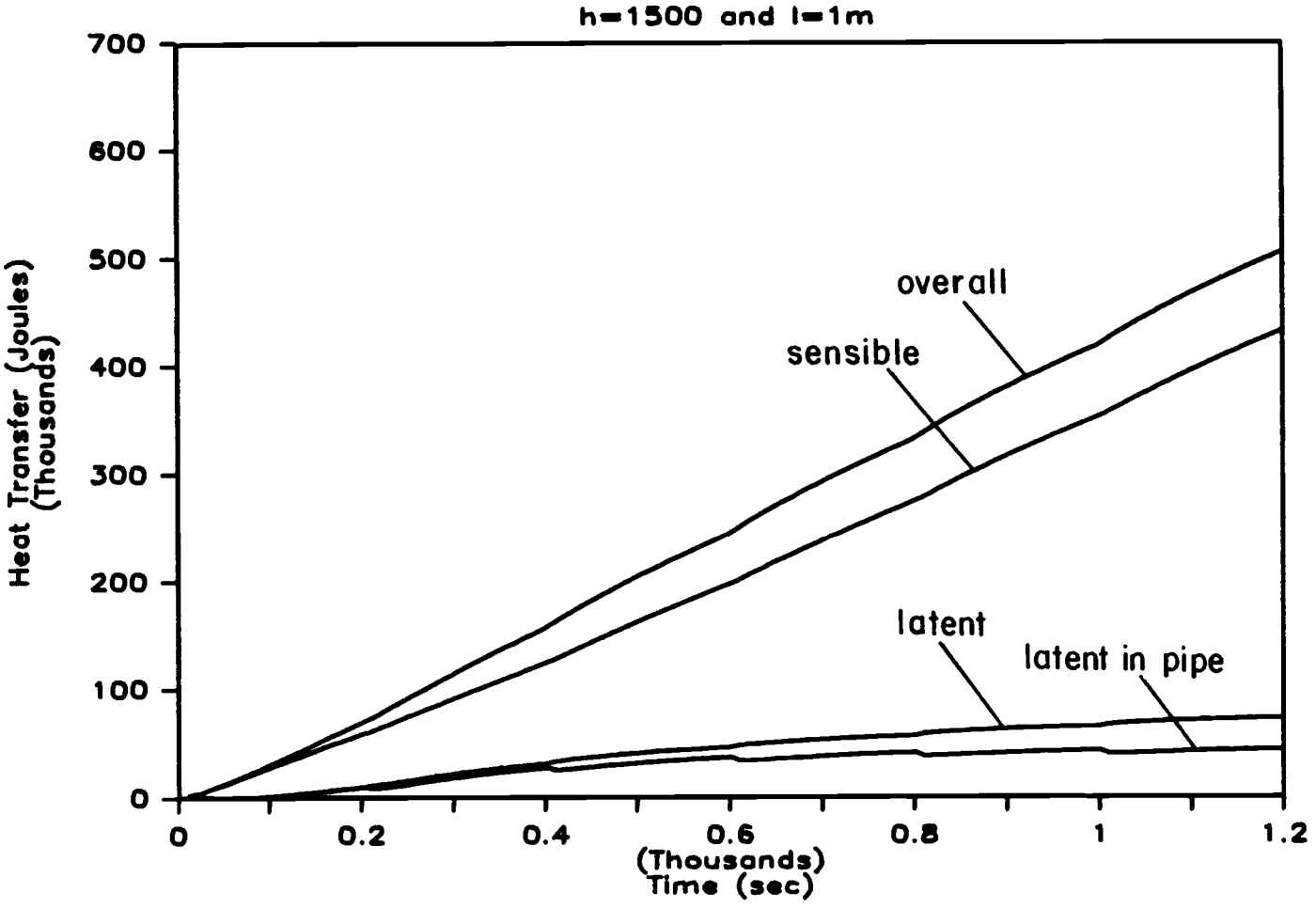

Figure 4.6 Energy transfer results for the reversal case.
h = 1500 w/m^2 ^{o}C and l = 1m.

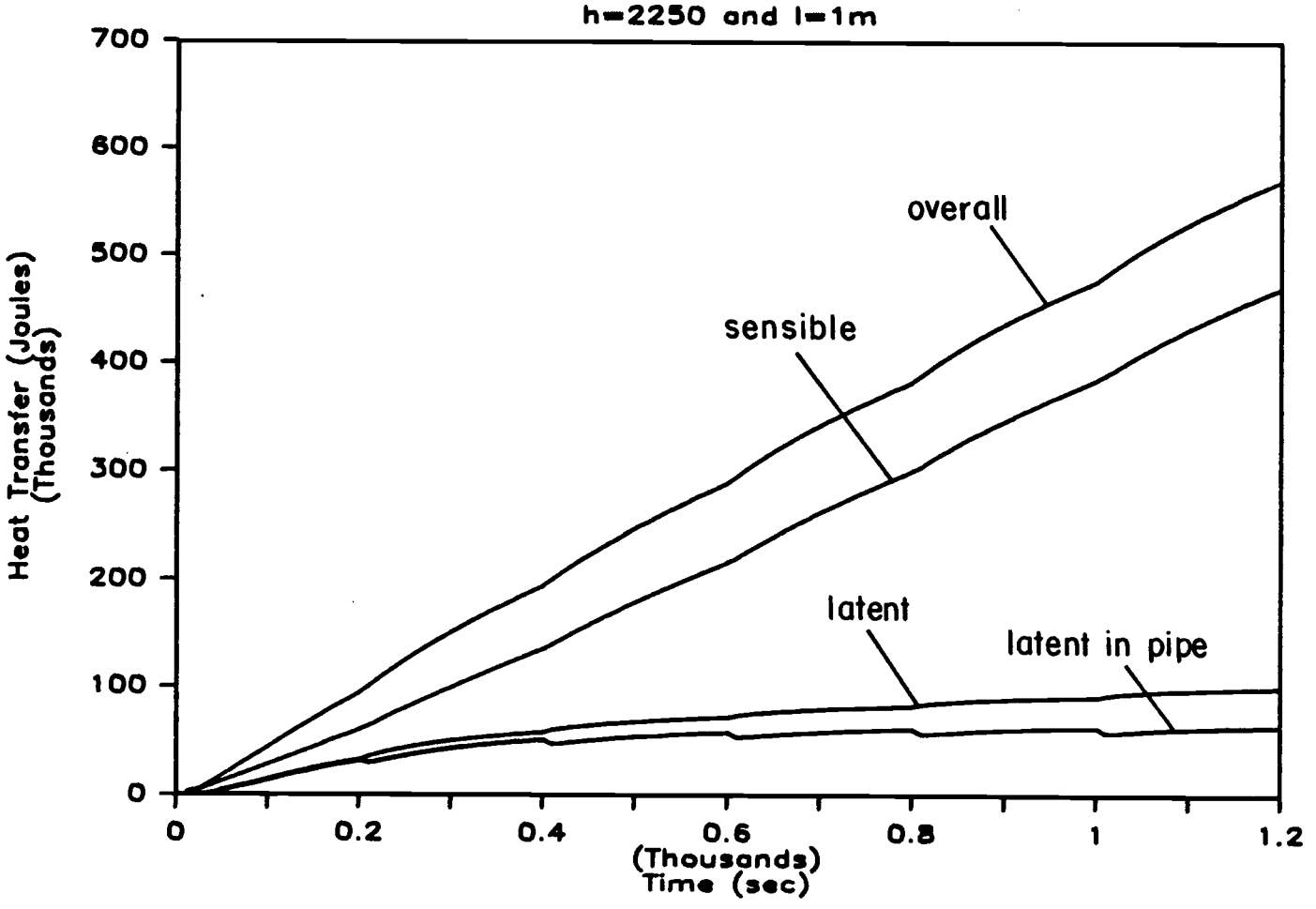

Figure 4.7 Energy transfer results for the reversal case.
h = 2250 w/m^2 ^{o}C and l = 1m.

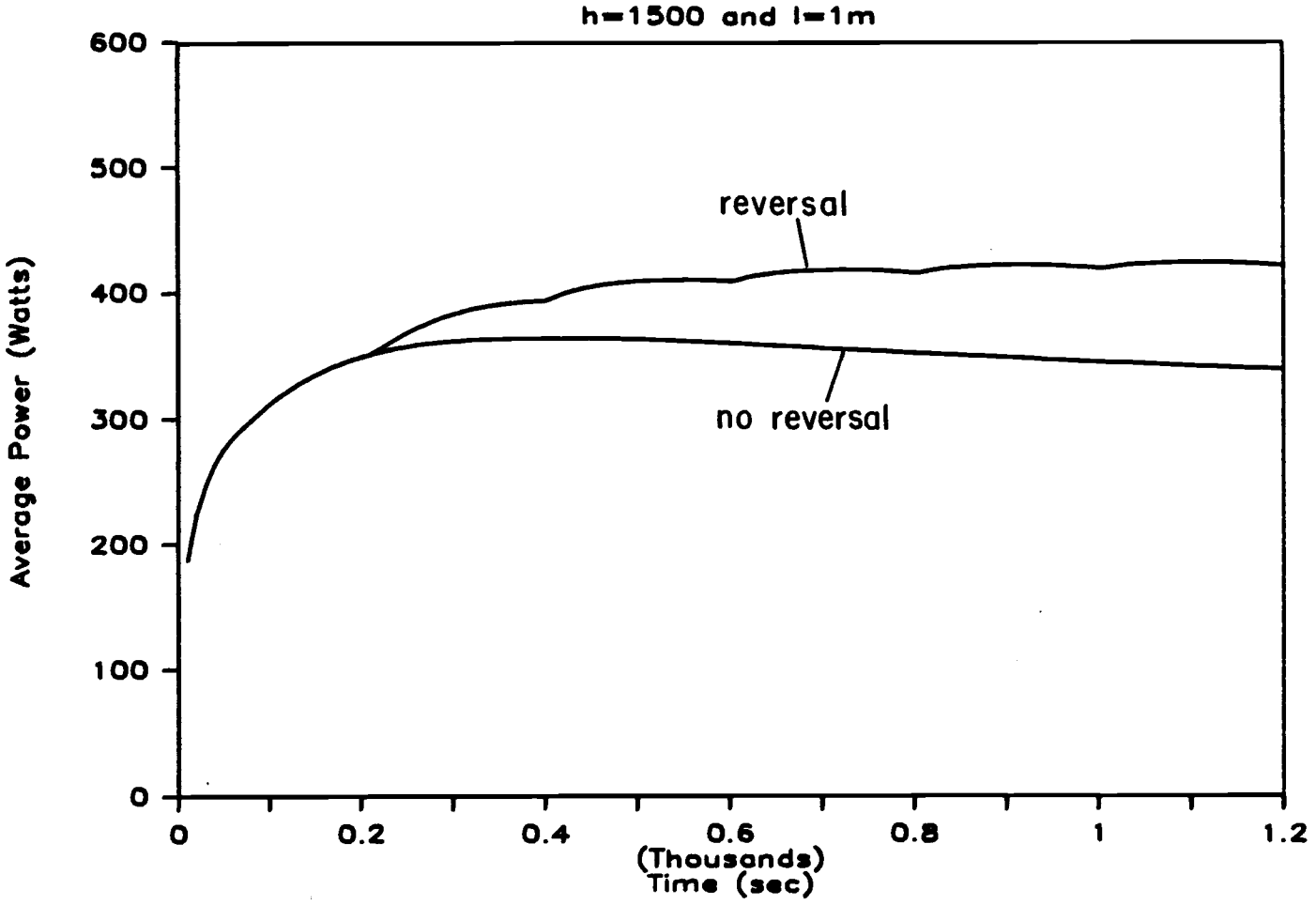

Figure 4.8 Comparison of the overall average power between reversal and no reversal cases.
h = 1500 w/m^2 ^oC and l = 1m.

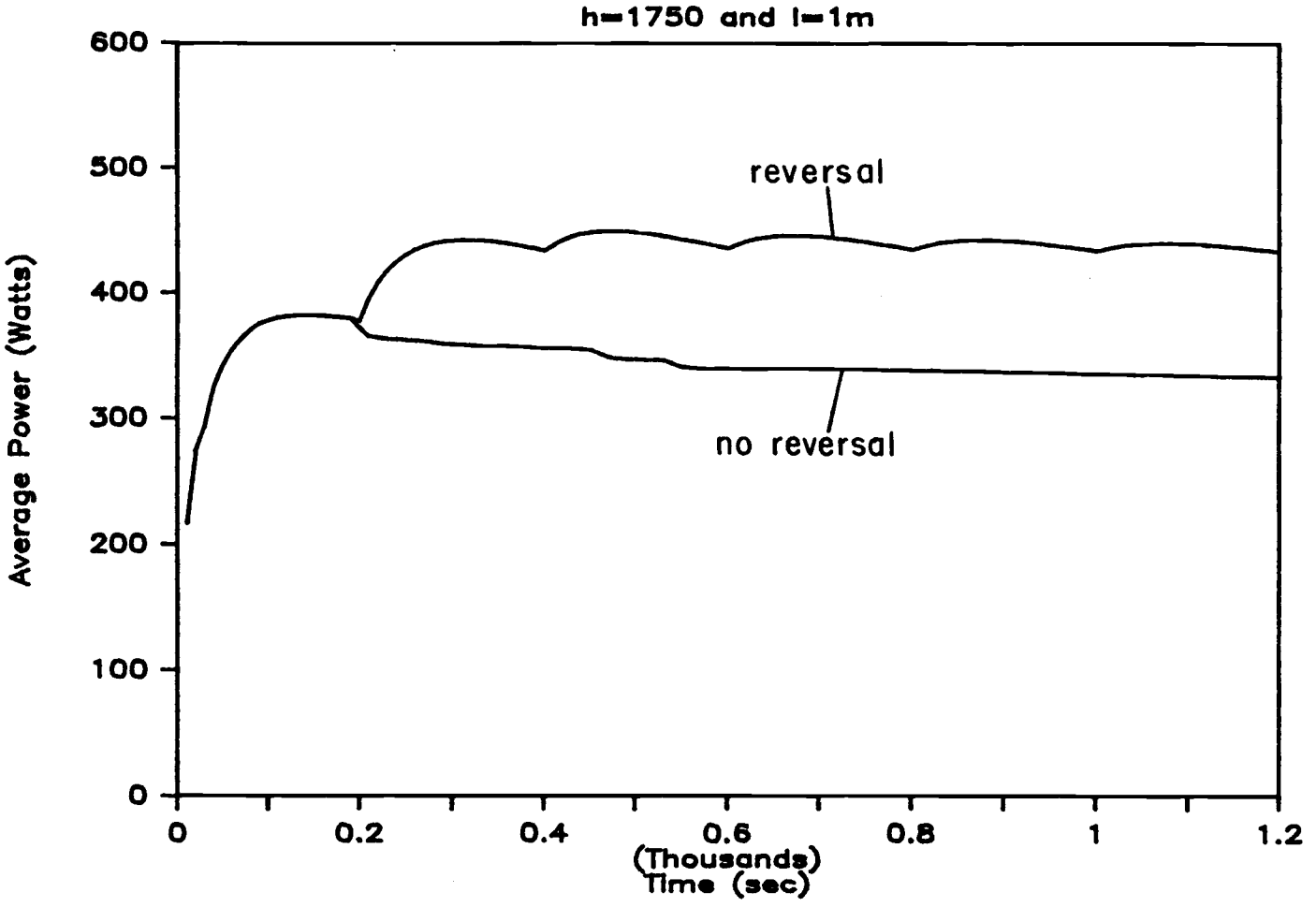

Figure 4.9 Comparison of the overall average power between reversal and no reversal cases.
h = 1750 w/m^2 ^oC and l = 1m.

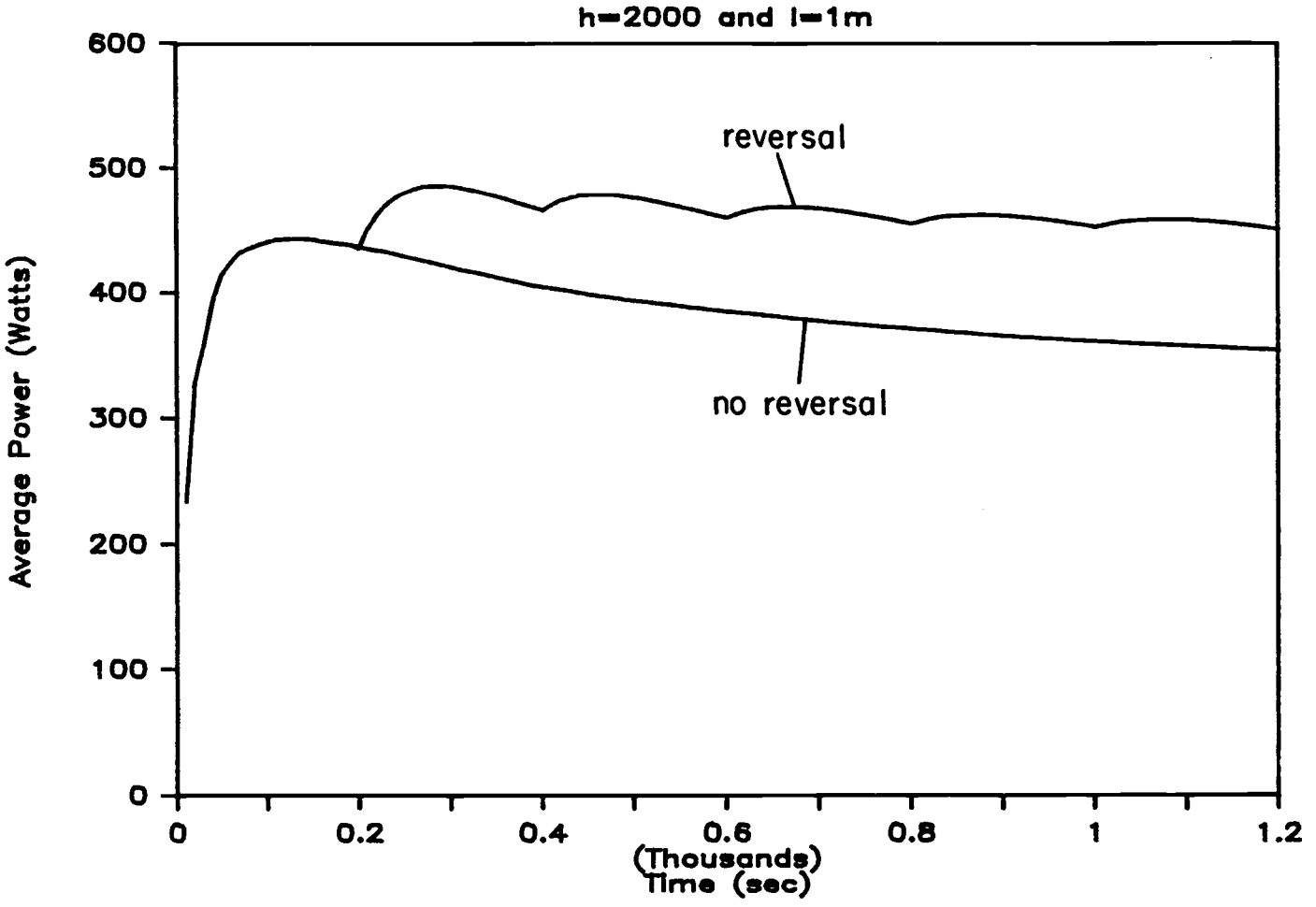

Figure 4.10 Comparison of the overall average power between reversal and no reversal cases.
h = 2000 w/m^2 ^oC and l = 1m.

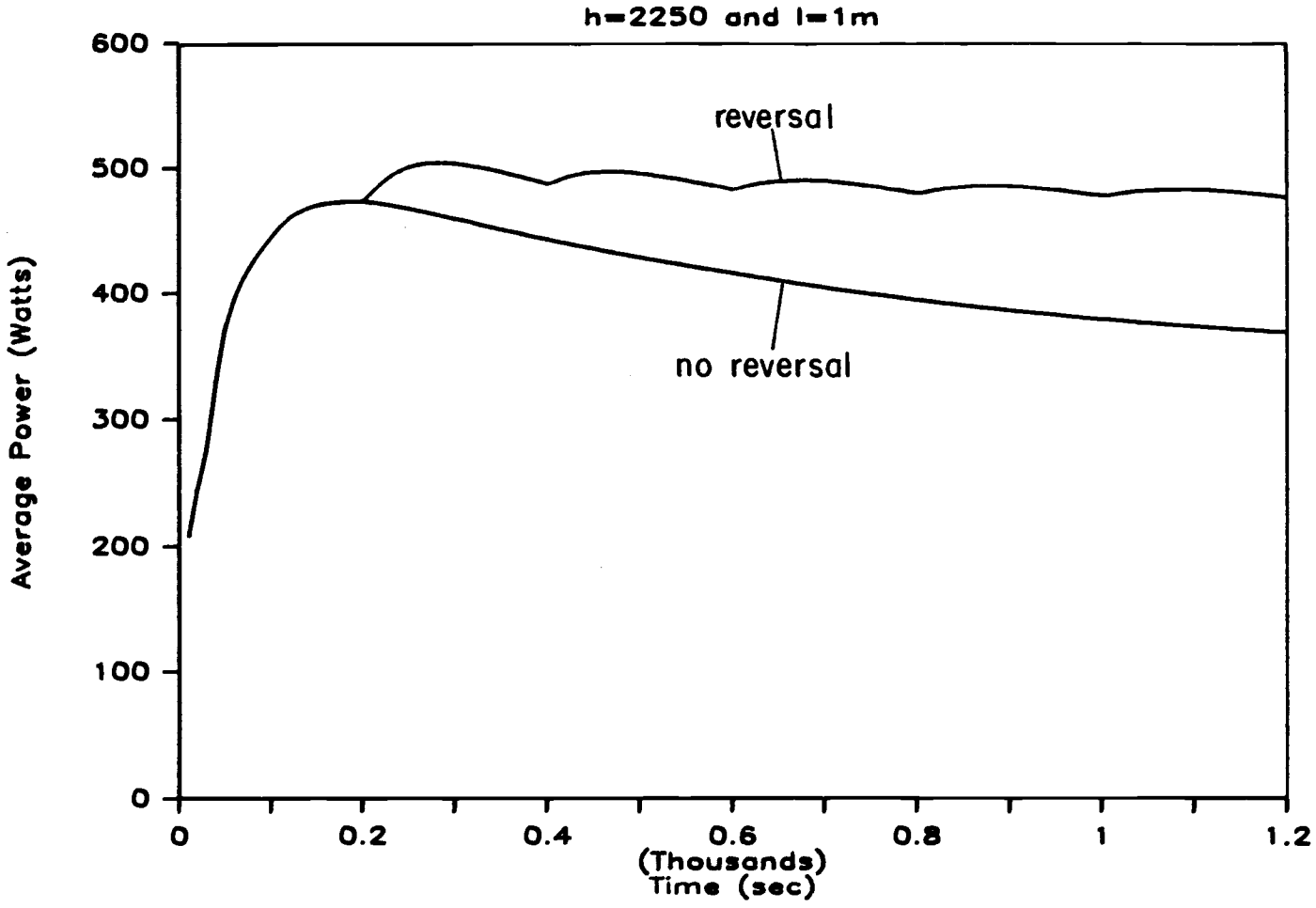

Figure 4.11 Comparison of the overall average power between reversal and no reversal cases.
h = 2250 w/m^2 ^oC and l = 1m.

until reaching a peak at about t=150 sec. The presence of the peaks is explained below in terms of the sensible and latent energy contributions to the overall power. The curves in the figures are identical up to 200 seconds, when the first reversal occurs. From that point, the curves diverge. In steady-state, the curves go parallel to one another, with the reversal curve above the non-reversal curve by about 20% of the overall reversal power. The difference between the curves increases slightly with the heat transfer coefficient.

Figures 4.12 and 4.13 show the contributions to the overall power by the latent and sensible energy transfer. Figure 4.12, for h=1500 $W/m^{2o}C$, shows a slow and smooth increase in the latent energy power followed by a slow decrease to the steady-state value. Figure 4.13, for h=2250 $W/m^{2o}C$, shows a sharp peak in the latent energy power at about 150 sec. This explains the presence of peaks in the overall power curves for high heat transfer coefficients (figures 4.9, 4.10 and 4.11).

Figure 4.14 shows a comparison between the powers obtained by reversing the flow with those obtained with no flow reversal. The values for the flow reversal case are higher, not only for the latent energy power, but also for the sensible energy power. In fact, most of the difference between the overall power for the two cases is accounted for by the sensible energy transfer, while energy in ice

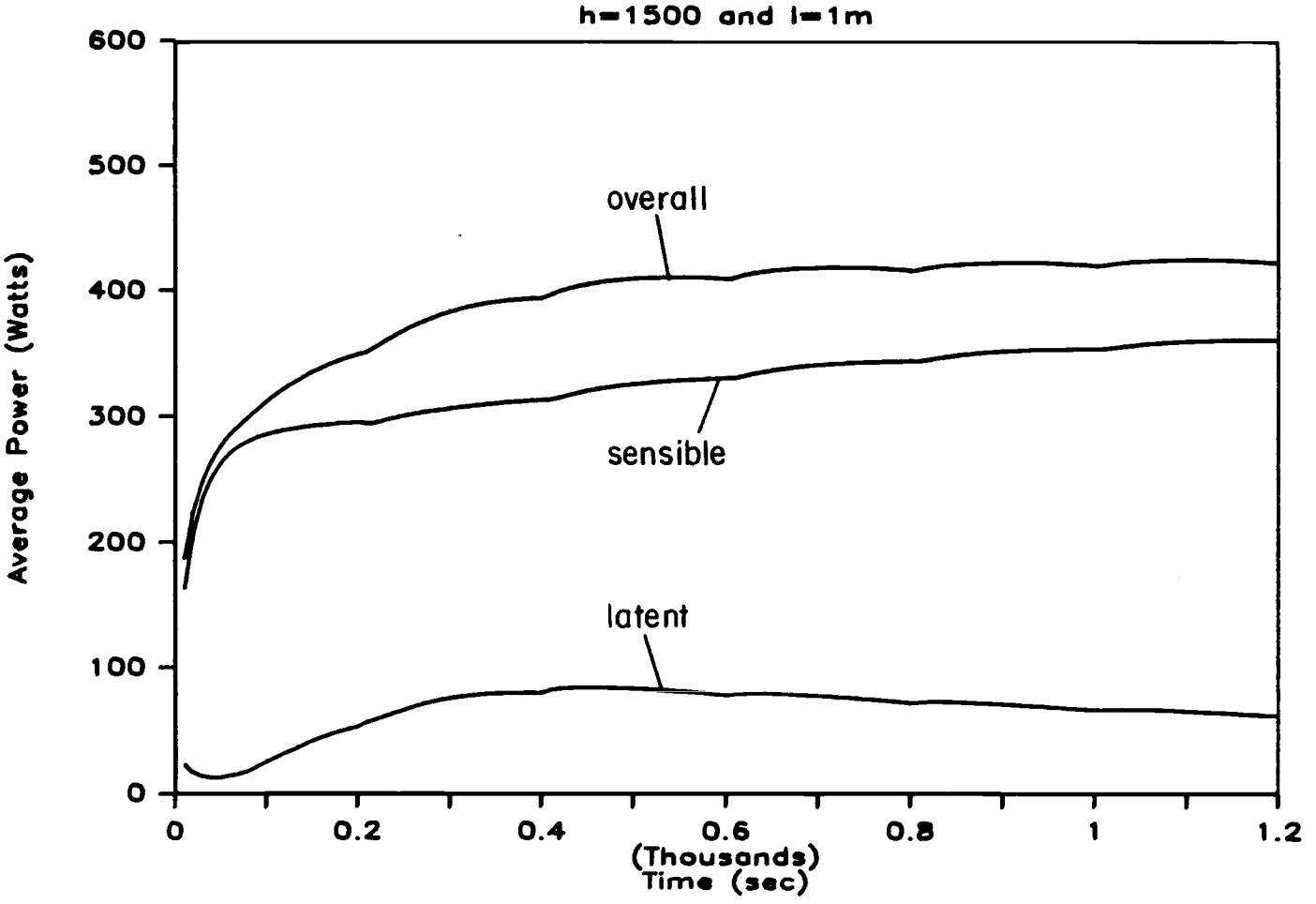

Figure 4.12 Contributions to overall power by latent and sensible energy transfer.
h = 1500 $w/m^2\ ^oC$ and l = 1m.

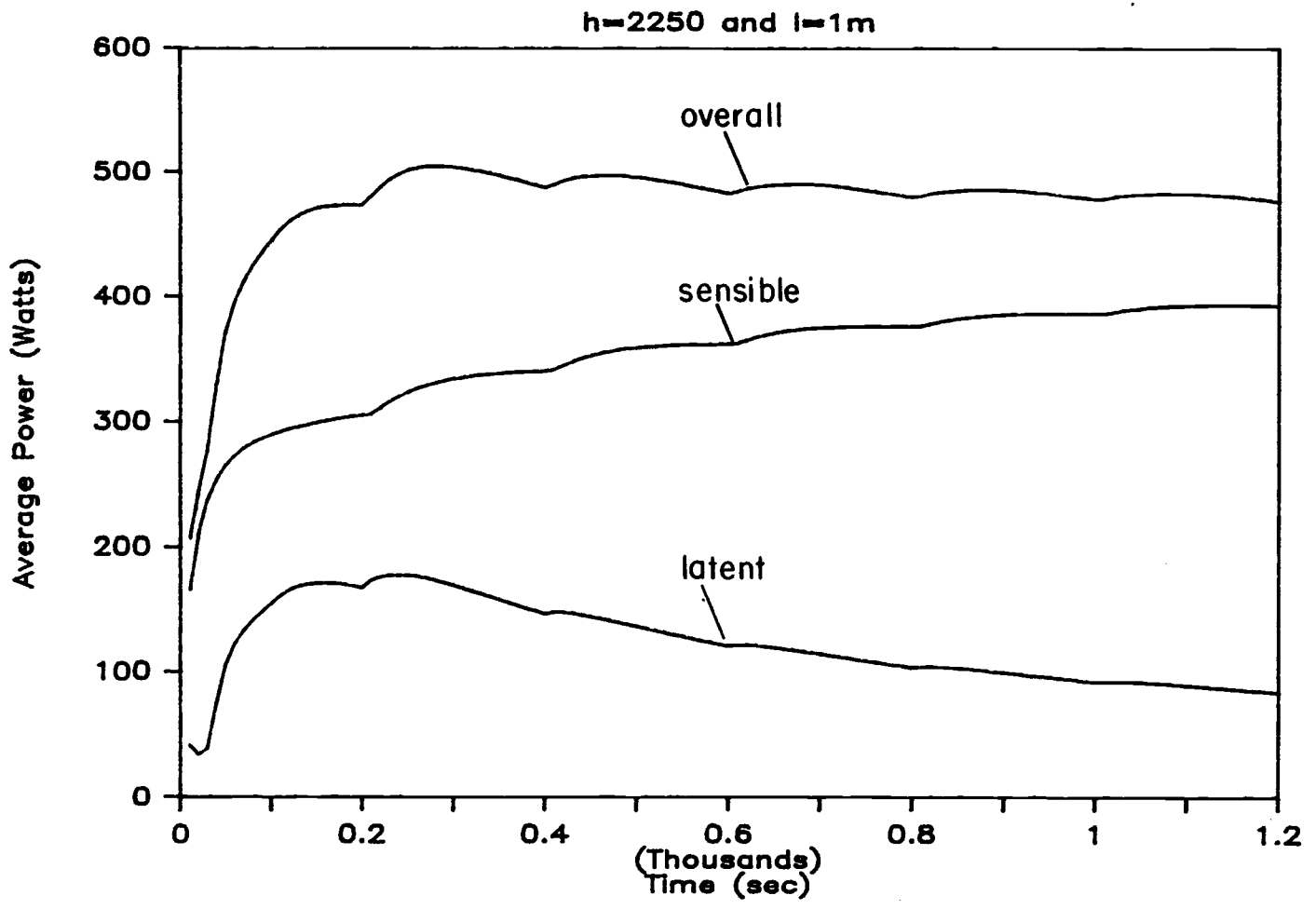

Figure 4.13 Contributions to overall power by latent and sensible energy transfer.
h = 2250 w/m^2 ^oC and l = 1m.

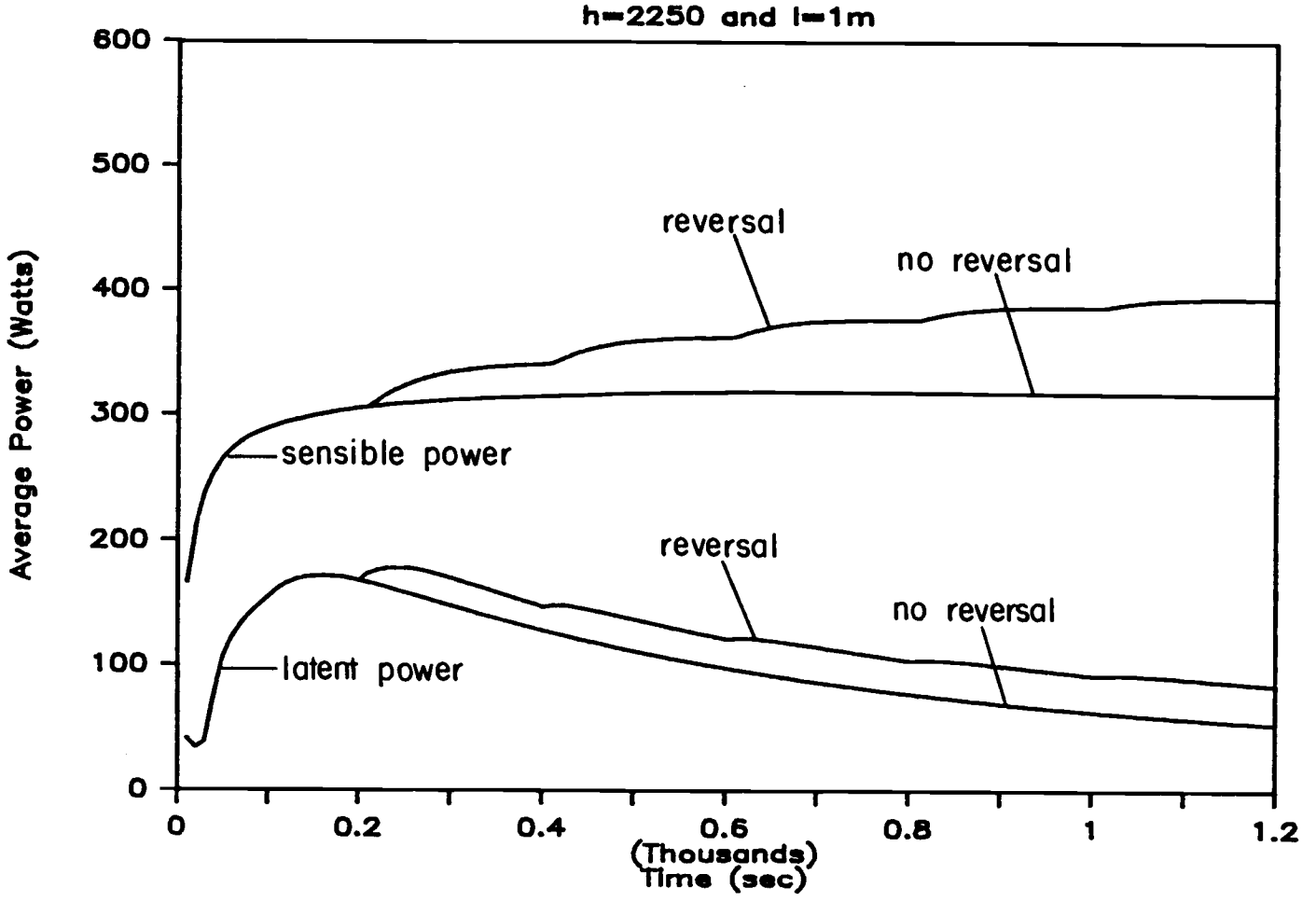

Figure 4.14 Comparison between powers obtained with reversal and with no reversal.

that detaches represents only a small part of the overall power.

Figure 4.15 is a plot of the amount of ice existing in the evaporator for the two cases, either flow reversal or non-flow reversal. The results indicate almost exactly the same amount of ice for both cases. Although the overall amount of ice is almost the same for both cases, the ice distribution along the evaporator is different. In the non-reversal case, most of the ice exists at the last section of the evaporator, while in the reversal case there are instances when there is more ice at the entrance of the evaporator than at the end. This explains the difference in sensible energy transfer between the two cases, as illustrated in figure 4.14. In the non-reversal case, water cools down in the initial section of the evaporator and then it reaches the final section, which is insulated by a thick layer of ice. Insulation by the ice layer prevents further cooling of the flowing water. In the flow reversal case, the final section has less ice insulation than in the non-reversal case and, therefore, allows the water to cool further.

4.2 Conclusions

The numerical model described here predicts unsteady ice formation in an evaporator with an annular finned geometry that is particularly well suited for allowing ice

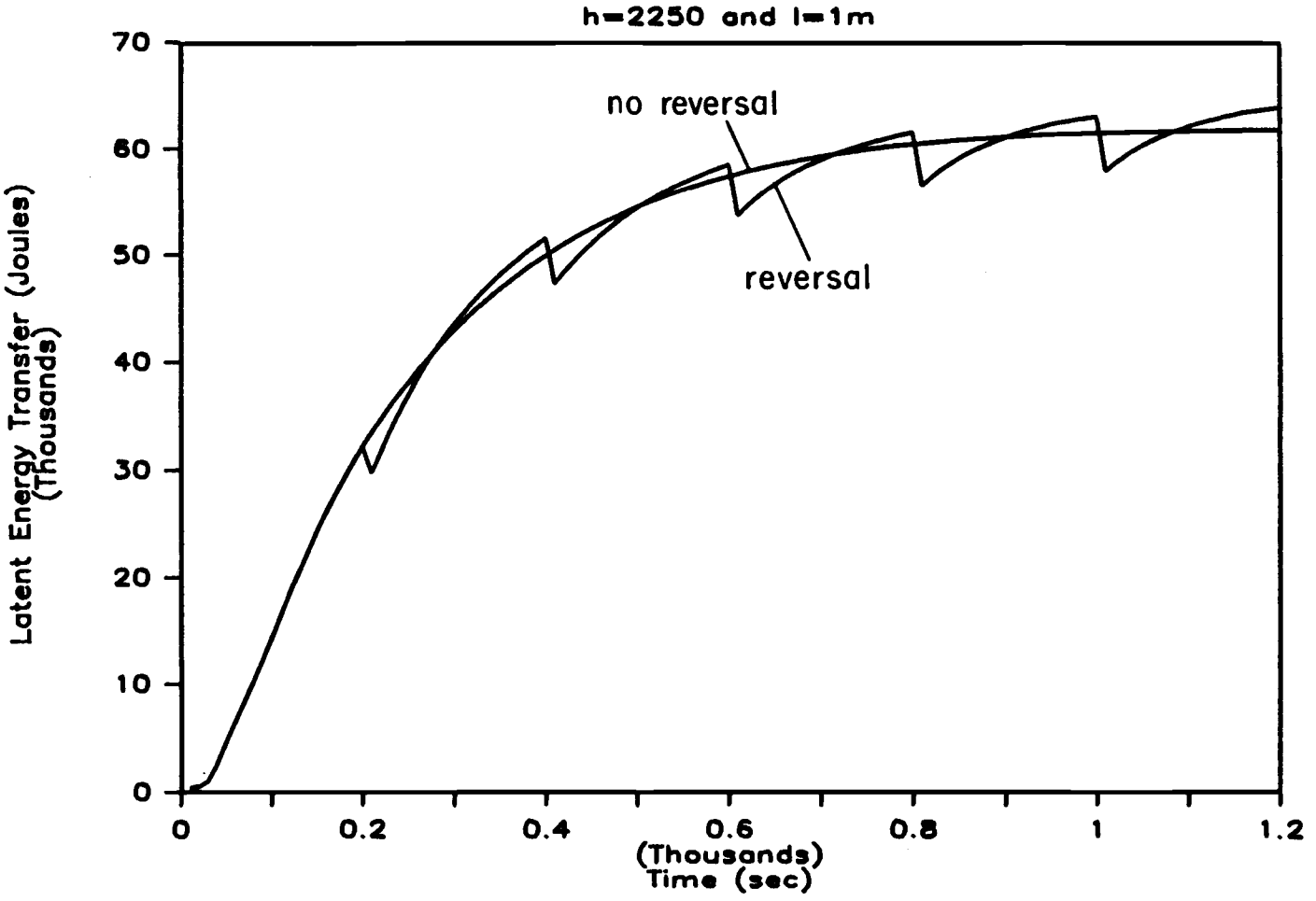

Figure 4.15 Comparison of the amount of ice existing in the pipe (expressed as the latent energy lost by the water) for reversal and no reversal cases. h = 2250 w/m^2 ^oC and l = 1m.

detachment from the wall. The program calculates ice profiles as a function of time, as well as energy transfer values.

The model has been used to evaluate an evaporator that operates under the flow reversal method, and the preliminary results indicate that this method gives a much better power output than the regular unidirectional ice formation in the evaporator. The power advantage of the flow reversal case is illustrated in figures 4.8 to 4.11. The power obtained is about 20% superior when flow reversal is used.

The model is still subject to future improvement. The most important limitations of the model are that it predicts only laminar flow ice formation and it uses an arbitrary ice detachment criterion.

The evaluation of the flow reversal method is still incomplete, because the computer program was executed for a fixed set of evaporator dimensions, while varying only the refrigerant heat transfer coefficient. A complete evaluation must include an extensive execution of the program for different evaporator dimensions and flow rates.

4.3 Recommendations for Further Work

The following are topics and areas suggested for further investigation:

1. All of the evaporator dimensions were kept constant in the program execution. It would be convenient to evaluate

the results for different sets of dimensions. Some dimensions that may affect the results significantly are the pipe wall thickness and the ratio of the internal pipe radius to the external pipe radius.

2. Flow reversal frequency is also an important parameter that should be varied to study how this affects the results.

3. A more accurate ice melting-detaching criterion is required to improve the results obtained from the program.

4. Further work should also include a comparison of the results obtained by using this method with the values obtained by using the ordinary de-icing techniques.

5. Experimental work is also necessary to determine the practical feasibility of the flow reversal method.

BIBLIOGRAPHY

1. K.E. Atkinson: An Introduction to Numerical Analysis, John Wiley and Sons, New York, NY, 1978.

2. H.S. Carslaw and J.C. Jaeger:Conduction of Heat in Solids, Oxford University Press, London, 1959.

3. J. Crank: "How to Deal With Moving Boundaries in Thermal Problems", in R.W. Lewis, K. Morgan and O.C. Zienkiewicz, editors:Numerical Methods in Heat Transfer, John Wiley and Sons, New York, NY 1981.

4. J. Douglas and T.M. Gallie: "On the Numerical Integration of a Parabolic Differential Equation Subject to a Moving Boundary Condition", Duke Mathematical Journal, Vol. 22, 1955, p. 557-571.

5. M.S. El-Genk and A.W. Cronenberg: "On the Thermal Stability of a Frozen Crust on an Insulated Finite Wall", International Journal of Heat and Mass Transfer, Vol. 22, 1979, p. 1719-1723.

6. M. Epstein: "The Growth and Decay of a Frozen Layer in Forced Flow", International Journal of Heat and Mass Transfer, Vol. 19, 1976, p. 1281-1288.

7. H.C. Fischer and E.A. Nephew: "Application of the Ice-Maker Heat Pump to an Annual Cycle Energy System", ASME paper 76-WA/Ener-4, 1976.

8. T.R. Goodman: "The Heat-Balance Integral and Its Applications to Problems Involving a Change in Phase", Journal of Heat Transfer, Vol. 80, February 1958, p. 335-342.

9. T.G.N. Haldane: "The Heat Pump, an Economical Method of Producing Low-Grade Heat From Electricity", J.I.E.E.E., 1930, p.666-675.

10. C.L. Huang and Y.P. Shih: "Perturbation Solution for Planar Solidification of a Saturated Liquid With Convection at the Wall", International Journal of Heat and Mass Transfer, Vol. 18, 1975, p. 1481-1483.

11. F. Krauss: "The Heat Pump in Theory and Practice", Power, Vol. 54, 1921, p. 298-300.

12. H.G. Landau: "Heat Conduction in a Melting Solid", Quarterly of Applied Mathematics, Vol. 8, 1950, p. 81-94.

13. M. Lotkin: "The Calculation of Heat Flow in Melting Solids", Quarterly of applied Mathematics, Vol. 18, 1960, p. 79-85.

14. E.P. Martinez and R.T. Beaubouef: "Transient Freezing in Laminar Tube Flow", The Canadian Journal of Chemical Engineering, Vol. 50, August 1972, p. 445-449.

15. S.V. Patankar:Numerical Heat Transfer and Fluid Flow, Mc Graw-Hill Book Co. New York, NY, 1980.

16. C.L. Pekeris and L.B. Slichter: "Problems of Ice Formation", Journal of Applied Physics, Vol. 10, 1939, p. 135-137.

17. G.M. Rinaldi, L.W. Bonnell and C.T. Geary: "Supercooling of Water in ACES Heat Exchanger", Oak Ridge Station paper # ORNL/MIT-263, 1977.

18. M.S. Sadeghipour, M.N. Ozisik and J.C. Mulligan: "Transient Freezing in a Convectively Cooled Tube", Journal of Heat Transfer, Vol. 104, May 1982, p. 316-322.

19. T. Saitoh: "Numerical Method for Multi-Dimensional Freezing Problems in Arbitrary Domains", Journal of Heat Transfer, Vol. 100, May 1978, p. 294-299.

20. P. Sampson and R.D. Gibson: "A Mathematical Model of Nozzle Blockage by Freezing", International Journal of Heat and Mass Transfer, Vol. 24, February 1981, p. 231-241.

21. J.M. Savino and R. Siegel: "An Analytical Solution for Solidification of a Moving Warm Liquid Onto an Isothermal Cold Wall", International Journal of Heat and Mass Transfer, Vol. 12, 1969, p. 803-809.

22. J.R. Sellars, M. Tribus and J.S. Klein: "Heat Transfer to Laminar Flow in a Round Tube or Flat Conduit-The Graetz Problem Extended", Transactions of the ASME, Vol. 78, February 1956, p. 441-448.

23. L.C. Tao: "Generalized Numerical Solutions of Freezing a Saturated Liquid in Cylinders and Spheres", A.I.Ch.E. Journal, Vol. 13, January 1967, p. 165-169.

24. W. Thomson: "On the Economy of the Heating and Cooling of Buildings by Means of Currents of Air", Glasgow Philosophical Society Proceedings, 1852, p. 269-272.

25. S. Weinbaum and L.M. Jiji: "Singular Perturbation Theory for Melting or Freezing in Finite Domains Initially Not at the Fusion Temperature", Journal of Applied Mechanics, Vol. 99, March 1977, p. 25-30.

26. R.D. Zerkle and J.E. Sunderland: "The Effect of Liquid Solidification in a Tube Upon Laminar-Flow Heat Transfer and Pressure Drop", Journal of Heat Transfer, Vol. 90, May 1968, p. 183-190.

APPENDIX

APPENDIX

Computer Program for Modelling Ice Formation in an Annular Sector

```
Program Fins;

{The following constants define physical properties
 of the system. All the values given in SI units.}
Const DExtRad=0.02;
      DIntRad=0.01;
      DFlow=0.0001;
      Fins=4;
      DTInit=5;
      DTFreeze=0;
      DTCool=-5;
      DHTCoef=1250.0;
      DKliq=0.558;
      DKsol=386;
      DKice=2.246;
      DThickness=0.001;
      DSpecHeatLiq=4180.0;
      DSpecHeatSol=385.4;
      DLatHeat=333500.0;
      DDensity=1000.0;
      DDensitySol=8890.0;
      DTimeMax=2000;
      DTimeRev=2000;

{The following constants are program parameters}
Const N=10;
      M=12;
      MPlus1=13;     {Adjust this value to M+1}
      P=10;           {Number of axial steps}
      Q=10;          {Radial steps in the ice}
      M2pNpQ=44;     {Adjust to 2*M+N+Q}
      AlphaTemp=1;   {Overrelaxation Parameter}
      DTimeSt=10;     {Initial time stepsize}
      DAxialSt=0.20;


Type Matrix=Array[0..N,0..MPlus1] of Real;
     MatrixIce=Array[0..Q,0..MPlus1] of Real;
     Vector=Array[0..MPlus1] of Real;
     Cubic=Array[0..n,0..MPlus1,0..P] of real;
     CubicIce=Array[0..Q,0..MPlus1,0..P] of Real;
     HyperMatrix=Array[0..Mplus1,0..P]of real;
     BooleanMatr=Array[0..MPlus1,0..P] of boolean;
     PVector=Array[0..P] of Integer;
     MetalMatrix=array[0..M2pNpQ, 0..P] of real;
     MetalVector=array[0..M2pNpQ] of real;


Var Velocity, VVelocity, Temp          :Matrix;
    Tice                               :MatrixIce;
    Temperature                        :Cubic;
    TempIce                            :CubicIce;
    Marker,UpMark                      :PVector;
    DeltaFi, Fiter                     :Vector;
    F, DF, DDF                         :HyperMatrix;
```

```
    I,J,K, MarkerIter                                          :Integer;
    SumResSq, SumResOld, Residue, Alpha, AlphaNew,AlphaOld :Real;
    Lambda, Eta, R, RS, FI0, A1, B1, C1, D1, FreeArea      :Real;
    AreaVel, IntRad, Integ, SumRes, SumTemp, Ks, Ki, T0    :Real;
    Nj, TW, SteL, Stes,TimeStp, Um0, NonDimZ, NonDimT, Nu  :Real;
    ZStep, Umo, Chronograph, A2, B2, C2, D2, DensSol, CpS  :Real;
    Nus,Veltemper,Tmean, SwitchChrono, Wathick, IceThick   :Real;
    Flast0, Flastit, VolumeIce, HeatWater, HeatIce         :Real;
    OverallHeat, Power, GrowthVol, IceArea, ExistingIce    :Real;
    HeatExistIce                                           :Real;
    ReqExactness, Optimum, IceFormed, IceDoesnotGrow       :Boolean;
    SlowGrowth, icemetal, JustReversion                    :Boolean;
    Velfile, Tempfile, TIceFile, Profile, metalfile        :Text;
    HeatFile                                               :Text;
    NewIce                                                 :BooleanMatr;
    Tmetal                                                 :MetalMatrix;
    T                                                      :MetalVector;

Procedure InitializeF;
{Initializes the F function (interface
 profile) to the ice free value, F=Ri/Ro }
begin
   For I:=0 to M
   do For K:=0 to P
      do begin
         F[I,K]:=IntRad;
         DF[I,K]:=0;
         DDF[I,K]:=0;
      end; {ForK}
end; {InitializeF}

Procedure InitializeU;
{Initializes to zero the velocity profile}
Begin
   For I:=0 to N
   do For J:=0 to M+1
      do Velocity[I,J]:=0;
end; {InitializeU}

Procedure InitializeDeltaFi;
{Defines the Fi direction stepsize
 that will be used  to solve the ice equation.
 The current partition of the domain is:
 Uniform partition in the R direction
 N subintervals
 in the fi direction,
4 subintervals with a stepsize 1/2n
8 subintervals with a stepsize 1/n

 If a different partition is required, modify this function accordingly}


begin
   For I:=0 to 4
      do DeltaFi[I]:=1/N/2;
```

```
   For I:=5 to 13
      do DeltaFi[I]:=1/N;
end; {InitializeDeltaFi}

Procedure InitializeMarker;
{Initializes the marker vector. The values of this vector
 indicate how much of the internal pipe wall is covered
 with ice. Its values vary from M+1 (no ice) to 0 (the
 whole surface covered with ice). The index K indicates
 the axial position along the pipe.}

begin
   For K:=0 to P
   do begin
      Marker[K]:=M+1;
      UpMark[K]:=-1;
   end; {for K}
end; {InitializeMarker}

Procedure InitializeT;
{Initializes the temperature profile to its inlet value}
begin
   For I:=0 to N
   do For J:=0 to M
      do For K:=0 to P
         do Temperature[I,J,K]:=1;
end; {Initialize}

Procedure InitializeTemp;
{The matrix temp is used to do the iterations on the temperature
distribution. At this moment, It is initialized to the old value
of the temperature, as stored in the 3-dimensional matrix Temperature}
begin
   For I:=0 to N
   do For J:=0 to M
      do Temp[I,J]:=Temperature[I,J,K];
end; {InitializeTemp}

Procedure InitializeTemperatureIce;
{Initializes the temperature of the ice}
begin
   For I:=0 to Q
   do for J:=0 to M
      do for K:=0 to P
         do TempIce[I,J,K]:=0;
end; {Initializetemperatureice}

Procedure InitializeTice;
{Initializes the temperature iteration matrix}
begin
   For I:=0 to Q
   do for J:=0 to M
      do Tice[I,J]:=TempIce[I,J,K];
end; {InitializeTice}
```

```
Procedure InitializeTmetal;
{Sets Tmetal to its initial value}
begin
   For J:=0 to M2pNpQ
   do For K:=0 to P
      Do Tmetal[J,K]:=1;
end; {InitializeMetal}

Procedure InitializeTvector;
{Initializes the metal temperature iteration vector}
begin
   for j:=0 to M2pNpQ
   do T[J]:=TMetal[J,1];
end;

Procedure InitializeFiter;
{Initializes the iteration vector Fiter}
begin
   For J:=0 to Marker[K]-1
   do Fiter[J]:=IntRad;
end; {InitializeFiter}

Procedure StartFiter;
{Initializes the iteration vector Fiter to start a new time cycle}
begin
   For J:=0 to M
   do Fiter[J]:=F[J,K];
end; {StartFiter}

Procedure DerivativeF;
{Evaluates the first and second derivatives of F respect to Fi}
begin
   DF[0,K]:=(-3*Fiter[0]+4*Fiter[1]-Fiter[2])/2/DeltaFi[1];
   DF[M,K]:=0;
   DDF[0,K]:=(Fiter[0]-2*Fiter[1]+Fiter[2])/sqr(DeltaFi[1]);
   DDF[M,K]:=0;
   For I:=1 to M-1
   do begin
      DF[I,K]:=((Fiter[I+1]-Fiter[I])*DeltaFi[I]/DeltaFi[I+1]+
               (Fiter[I]-Fiter[I-1])*DeltaFi[I+1]/DeltaFi[I])/
               (DeltaFi[I+1]+DeltaFi[I]);
      DDF[I,K]:=2*((Fiter[I+1]-Fiter[I])/DeltaFi[I+1]-
      (Fiter[I]-Fiter[I-1])/DeltaFi[I])/(DeltaFi[I]+DeltaFi[I+1]);
   end; {for I}
end; {DerivativeF}

Procedure DefineLiquid;
{Evaluates the values of the coefficients of the Laplacian, A1,B1,C1,D1}
begin
   A1:=1/sqr(1-Fiter[J])+sqr(R*DF[J,K]/RS/FI0/(1-Fiter[J]));
   B1:=2*r*DF[J,K]/sqr(RS*FI0)/(1-Fiter[J]);
   C1:=1/sqr(RS*FI0);
   D1:=-1/RS/(1-Fiter[J])+R*DDF[J,K]/sqr(RS*FI0)/(1-Fiter[J])
       +2*sqr(DF[J,K]/RS/(1-Fiter[J])/FI0)*R;
end; {DefineLiquid}
```

```
Procedure DefineSolid;
{Evaluates the values of the coefficients of the Laplacian, A2,B2,C2,D2}
{Calculations for the solid phase}
Var Icethick :real;
begin
   Icethick:=Fiter[J]-IntRad;
   A2:=1/sqr(Icethick)+sqr(R*DF[J,K]/RS/FI0/IceThick);
   B2:=-2*R*DF[J,K]/sqr(RS*FI0)/IceThick;
   C2:=1/sqr(Fi0*RS);
   D2:=1/RS/Icethick-R*DDF[J,K]/sqr(RS*FI0)/IceThick+
   2*R*sqr(DF[J,K])/sqr(RS*FI0*IceThick);
end; {DefineSolid}

Procedure IterateU;
{ Solves for the velocity distribution }
begin
   SumResSq:=0;
   ReqExactness:=True;
   For I:=0 to N
      do Velocity[I,M+1]:=Velocity[I,M-1];
   For J:=1 to M
   do begin
      For I:=1 to N-1
      do begin
         R:=I/N;
         RS:=1-R*(1-Fiter[J]);
         DefineLiquid;

         {The next lines calculate the residue}
         Residue:=A1*N*N*(Velocity[I-1,J]+Velocity[I+1,J])
         +B1*N/2/(DeltaFi[J]+DeltaFi[J+1])*(DeltaFi[J]/DeltaFi[J+1]*
         (Velocity[I+1,J+1]-Velocity[I+1,J]-Velocity[I-1,J+1]
         +Velocity[I-1,J])+DeltaFi[J+1]/DeltaFi[J]*(Velocity[I+1,J]
         -Velocity[I+1,J-1]-Velocity[I-1,J]+Velocity[I-1,J-1]))
         +2*C1/(DeltaFi[J+1]*DeltaFi[J])/(DeltaFi[J+1]+DeltaFi[J])*
         (Velocity[I,J-1]*DeltaFi[J+1]+Velocity[I,J+1]*DeltaFi[J])
         +D1/2*N*(Velocity[I+1,J]-Velocity[I-1,J])-1;
         Residue:=Residue/(2*N*N*A1+2*C1/DeltaFi[J]/DeltaFi[J+1]);
         Residue:=Residue-Velocity[I,J];

         SumResSq:=SumResSq+sqr(Residue);
         Velocity[I,J]:=Velocity[I,j]+Alpha*Residue;
         If Abs(Residue)>0.001 then ReqExactness:=False;
      end; {for I}
   end; {for j}
end; {IterateU}

Procedure SORBusiness;
{Iterates to get the optimum value of the SOR coefficient}
begin
   Lambda:=Sqrt(SumResSq)/Sqrt(SumResOld);
   Eta:=(Lambda+(Alpha-1))/Alpha/Sqrt(Lambda);
   AlphaNew:=2/(1+sqrt(abs(1-sqr(eta))));
   if Abs(AlphaNew-AlphaOld)<0.01
```

```
   then begin
      Alpha:=AlphaNew;
      Optimum:=True;
   end {then}
   else begin
      AlphaOld:=AlphaNew;
      SumResOld:=SumResSq
   end; {else}
end; {Sorbusiness}

Procedure Simpson;
{ This procedure performs a numerical double integration to calculate
  the ice-free area. A Simpson's 3-point formula is used.}

Function Weight(I,N:Integer):Integer;
Var W:integer;
begin
   If (I mod 2=1) then W:=4;
   If (I mod 2=0) then W:=2;
   If (I=0) or (I=N) then W:=1;
   Weight:=W;
end; {Weight}

begin {Simpson}
   FreeArea:=0;
   AreaVel:=0;
   Veltemper:=0;
   For J:=0 to M
   do begin
      For I:=0 to N
      do begin
         R:=I/N;
         RS:=1-R*(1-Fiter[J]);
         Integ:=RS*Weight(I,N)*Weight(J,M)*
         (1-Fiter[J])*(DeltaFi[J]+DeltaFi[J+1]);
         FreeArea:=FreeArea+Integ;
         AreaVel:=AreaVel+Integ*Velocity[I,J];
         veltemper:=Veltemper+Integ*Velocity[I,J]*Temp[I,J];
      end; {For I}
   end; {for J}
   FreeArea:=FreeArea*FI0/18/N;
   AreaVel:=AreaVel*FI0/18/N;
   Veltemper:=Veltemper*FI0/18/N;
   {the next lines calculate the amount of ice existing currently}
   {in the pipe.}
   IceArea:=pi*(sqr(DExtRad)-sqr(DintRad))-FreeArea*2*Fins*sqr(DExtRad);

end; {Simpson}

Procedure DimVelocity;
{Calculates the non-dimensional velocity as a function of the flow
 rate when the velocity as a function of the pressure drop is known.
 It also calculates the ice-water Nusselt number}
begin
   For I:=0 to N
```

```
   do For J:=0 to M
      do VVelocity[I,j]:=PI*Velocity[I,J]*(1-sqr(IntRad))
         /AreaVel/2/Fins;
   Tmean:=Veltemper/AreaVel;
   Nus:=DDensity*DFlow*DSpecHeatLiq*(1-Tmean);
   Nus:=Nus/DKLiq/Tmean/K/DAxialSt/pi;
end; {DimVelocity}

Procedure PrintVel;
begin
   For I:=0 to n
   do For J:=0 to m
      do writeln(Velfile,I,'  ',J,'  ',VVelocity[I,J]);
end; {printvel}

Procedure CalculateMetal;
{Calculates the temperature distribution in the metallic parts}
Var L,MN,Low, Factor                                    :Integer;
    Diag, Superior, Inferior, S, Derecha                :MetalVector;
    R1, RS1, RS2, T1                                    :Real;

begin
   {The next lines calculate the number of elements in the matrix}
   MN:=M2pNpQ;
   if Marker[K]>0 then MN:=MN-Q;
   if Fiter[0]=1 then MN:=MN-N;
   Low:=MN-M;

   {The next lines set up the matrix}

   {at the internal pipe wall:}
   For L:=0 to M-1
   do begin
      J:=M-L;
      wathick:=1-Fiter[J];
      IceThick:=Fiter[J]-IntRad;
      if J<Marker[K]
      then begin                {Where there is water}
         Diag[L]:=2*Ks*T0/sqr(FI0*INTRAD)/DeltaFi[J]/DeltaFi[J+1]
         +3*N/2/WaThick+Nu;
         Superior[L]:=-2*Ks*T0/sqr(IntRad*FI0)/DeltaFi[J+1]/
         (DeltaFi[J]+DeltaFi[J+1]);
         Inferior[L]:=Superior[L]*DeltaFi[J+1]/DeltaFi[J];
         Derecha[L]:=Nu*Tw+2*N*Temp[N-1,J]/WaThick-N*Temp[N-2,J]
         /2/WaThick;
      end {then}
      else begin                {Where there is ice}
         Diag[L]:=Nu+3*Ki*Q/2/IceThick+2*Ks*T0/sqr(IntRad*FI0)
         /DeltaFi[J]/DeltaFi[J+1];
         Superior[L]:=-2*Ks*T0/sqr(IntRad*FI0)/DeltaFi[J+1]/
         (DeltaFi[J]+DeltaFi[J+1]);
         Inferior[L]:=Superior[L]*DeltaFi[J+1]/DeltaFi[J];
         Derecha[L]:=Nu*Tw+2*Ki*Q/IceThick*Tice[1,J]-Ki*Q/2
         /IceThick*Tice[2,J];
      end; {else}
```

```
end; {for}
{The next line modify the values for L=0}
Superior[0]:=Superior[0]+Inferior[0];

{At the inferior corner}
If Marker[K]>0
then begin                          {if there is no ice}
   Diag[M]:=T0*Ks*N/(1-Fiter[0])/2+T0*Ks/IntRad/FI0/DeltaFi[0]+
   Nu*IntRad*DeltaFi[0]*FI0/2;
   Superior[M]:=-T0*Ks*N/(1-Fiter[0])/2;
   Inferior[M]:=-T0*Ks/intrad/FI0/DeltaFi[0];
end {then}
else begin                          {if there is ice}
   Diag[M]:=Nu*IntRad*DeltaFi[0]*FI0/2+T0*Ks/IntRad/FI0/DeltaFi[0]
   +T0*Ks*Q/2/(Fiter[0]-IntRad);
   Superior[M]:=-Q*T0*KS/2/(Fiter[0]-IntRad);
   Inferior[M]:=-T0*Ks/IntRad/FI0/DeltaFi[0];
end; {else}
Derecha[M]:=Tw*Nu*IntRad*DeltaFi[0]*FI0/2;

{at the side fin}
If marker[K]=0
then begin                        {distribution in the ice}
   For L:=M+1 to M+Q-1
   do begin
      If NewIce[0,K]
      then factor:=1000
      else factor:=1;
      I:=L-M;
      IceThick:=Fiter[0]-IntRad;
      R:=I/Q;
      Rs:=IntRad+R*IceThick;
      Diag[L]:=Ks*T0*Q*Q/sqr(IceThick)+3*ki/2/Rs/FI0/DeltaFi[0]
      +CpS*DensSol*SteL*T0/TimeStp/factor;
      Superior[L]:=Q*Ki*R*DF[0,K]/2/RS/FI0/IceThick-
      Q*Q*Ks*T0/2/sqr(IceThick);
      Inferior[L]:=-Q*Ki*R*DF[0,K]/2/RS/FI0/IceThick-
      Q*Q*Ks*T0/2/sqr(IceThick);
      Derecha[L]:=2*Ki/Rs/FI0/DeltaFi[0]*Tice[I,1]-Ki/2/Rs/FI0
      /DeltaFi[0]*Tice[I,2]+Tmetal[L,K]*CpS*DensSol*SteL
      *T0/TimeStp/factor;
   end; {for}
end {then}
else begin                  {In case that there is no ice at the fin}
   for L:=M+1 to M+N-1
   do begin
      I:=M+N-L;
      WaThick:=1-Fiter[0];
      R:=I/N;
      Rs:=1-R*Wathick;
      Diag[L]:=Ks*T0*N*N/sqr(Wathick)+3/2/Rs/Fi0/DeltaFi[0]
      +CpS*DensSol*SteL*T0/TimeStp;
      Superior[L]:=-R*DF[0,K]*N/2/RS/wathick/FI0-Ks*T0*N*N/2
      /sqr(Wathick);
      Inferior[L]:=R*DF[0,K]*N/2/RS/wathick/FI0-Ks*T0*N*N/2
```

```
      /sqr(Wathick);
      Derecha[L]:=2*Temp[I,1]/Rs/FI0/DeltaFi[0]-Temp[I,2]/2/Rs/FI0/
      DeltaFi[0]+Tmetal[L,K]*CpS*DensSol*SteL*T0/TimeStp;
   end; {for}
end; {else}

If ((Fiter[0]<1) and (Marker[K]=0))
then begin     {In case that there is both water and ice at the fin}
   For L:=M+Q+1 to M+Q+N-1
   do begin
      I:=M+Q+N-L;
      WaThick:=1-Fiter[0];
      R:=I/N;
      Rs:=1-R*Wathick;
      Diag[L]:=Ks*T0*N*N/sqr(Wathick)+3/2/Rs/Fi0/DeltaFi[0]
      +CpS*DensSol*SteL*T0/TimeStp;
      Superior[L]:=-R*DF[0,K]*N/2/RS/wathick/FI0-Ks*T0*N*N/2
      /sqr(Wathick);
      Inferior[L]:=R*DF[0,K]*N/2/RS/wathick/FI0-Ks*T0*N*N/2
      /sqr(Wathick);
      Derecha[L]:=2*Temp[I,1]/Rs/FI0/DeltaFi[0]-Temp[I,2]/2/Rs/FI0/
      DeltaFi[0]+Tmetal[L,K]*CpS*DensSol*SteL*T0/TimeStp;
   end; {for}
   {At the interface point, L=M+Q}
   Wathick:=1-Fiter[0];
   Icethick:=Fiter[0]-IntRad;
   R1:=Wathick/(1-Fiter[1]);
   if R1>1 then R1:=1;
   RS1:=1-(trunc(R*N)/N)*(1-Fiter[1]);
   RS2:=RS1-(1-Fiter[1])/N;
   T1:=(Temp[trunc(R1*N),1]*(Fiter[0]-RS2)+Temp[trunc(R1*N+1),1]
   *(RS1-Fiter[0]))*N/(1-Fiter[1]);
   Diag[M+Q]:=Ks*T0*N*Q/wathick/Icethick+CpS*DensSol*SteL*T0/TimeStp
   +1/Fiter[0]/Fi0/DeltaFi[0];
   Superior[M+Q]:=-Ks*T0*N/Wathick/(wathick/N+Icethick/Q);
   Inferior[M+Q]:=-Ks*T0*Q/Icethick/(wathick/N+Icethick/Q);
   Derecha[M+Q]:=Tmetal[M+Q,K]*CpS*DensSol*SteL*T0/TimeStp
   +T1/Fiter[0]/Fi0/DeltaFi[0];
end; {then}

{at the top corner}
if Fiter[0]<1                  {When there is no ice}
then begin
   Diag[Low]:=Ks*N/2/(1-Fiter[0])+Ks/Fi0/DeltaFi[0]+DensSol*CpS*
   SteL/TimeStp*(DeltaFi[0]*FI0+(1-Fiter[0])/2/N);
   Superior[Low]:=-Ks/Fi0/DeltaFi[0];
   Inferior[Low]:=-Ks*N/2/(1-Fiter[0]);
   Derecha[Low]:=Tmetal[Low,K]*(DeltaFi[0]*FI0+(1-Fiter[0])/2/N)
   *DensSol*CpS*SteL/TimeStp;
end {then}
else begin                     {when there is ice}
   Diag[Low]:=Ks*Q/2/(1-IntRad)+Ks/Fi0/DeltaFi[0]+DensSol*CpS*
   SteL/TimeStp*(DeltaFi[0]*FI0+(1-IntRad)/2/Q);
   Superior[Low]:=-Ks/Fi0/DeltaFi[0];
   Inferior[Low]:=-Ks*Q/2/(1-IntRad);
```

```
      Derecha[Low]:=Tmetal[Low,K]*(DeltaFi[0]*FI0+(1-IntRad)/2/Q)
      *DensSol*CpS*SteL/TimeStp;
   end; {else}

   {At the external pipe wall}
   For L:=Low+1 to Low+m
   do begin
      J:=L-Low;
      wathick:=1-Fiter[J];
      Diag[L]:=3*N/2/wathick+2*Ks*T0/sqr(Fi0)/DeltaFi[J+1]/DeltaFi[J]
      +2*CpS*DensSol*T0*SteL/TimeStp;
      Superior[L]:=-2*Ks*T0/sqr(FI0)/DeltaFi[J+1]
      /(DeltaFi[J+1]+DeltaFi[J]);
      Inferior[L]:=-2*Ks*T0/sqr(FI0)/DeltaFi[J]
      /(DeltaFi[J+1]+DeltaFi[J]);
      Derecha[L]:=2*N/(1-Fiter[J])*Temp[1,J]-N/2/(1-Fiter[J])*Temp[2,J]
      +Tmetal[L,K]*2*CpS*DensSol*T0*SteL/TimeStp;
   end; {For L}
   {The next line modifies Inferior at the symmetry line}
   Inferior[Low+m]:=Inferior[Low+M]+Superior[Low+M];

   {The setup is finally concluded}
   {Now the matrix is solved}

   {The next lines factorize the matrix}
   Superior[0]:=Superior[0]/Diag[0];
   For L:=1 to MN
   do begin
      Diag[L]:=Diag[L]-Inferior[L]*Superior[L-1];
      Superior[L]:=Superior[L]/Diag[L];
   end; {for L}

   {the next lines solve for vector S}
   S[0]:=Derecha[0]/Diag[0];
   For L:=1 to MN
   do S[L]:=(Derecha[L]-Inferior[L]*S[L-1])/Diag[L];

   {the next lines solve for vector T}
   T[MN]:=S[MN];
   For L:=MN-1 downto 0
   do T[L]:=S[L]-Superior[L]*T[L+1];
   {this ends the calculations}
end; {CalculateMetal}


Procedure TempBoundary;
{Sets temperatures at the liquid and solid boundaries}
Var Low:Integer;
begin
   For J:=0 to M                          {Internal pipe wall}
   do begin
      if J<Marker[K]
      then Temp[N,J]:=T[M-J]              {no ice present}
      else Temp[N,J]:=0;                  {ice present}
   end; {for}
```

```
   For J:=Marker[K] to M
   do Tice[0,J]:=T[M-J];

   For J:=Marker[K] to M                    {Ice-water interface}
   do Tice[Q,J]:=0;

   If Marker[K]=0                           {at the fin, ice present}
   then begin
      For I:=1 to Q
      do Tice[I,0]:=T[M+I];
   end {then}
   else begin                               {no ice present}
      For I:=0 to N
      do Temp[I,0]:=T[M+N-I];
   end; {else}

   If ((Fiter[0]<1) and (marker[K]=0)) {Ice and water at the fin}
   then begin
      For I:=0 to N
      do Temp[I,0]:=T[M+N+Q-I];
   end; {then}

   If Fiter[0]=1                            {Only ice at the fin}
   then begin
      For I:=0 to N
      do Temp[I,0]:=0;
   end; {then}

   {the next lines set Low to the appropriate value}
   Low:=M+N+Q;
   if Marker[K]>0 then Low:=Low-Q;
   if Fiter[0]=0 then Low:=Low-N;

   For J:=1 to M                            {external pipe wall}
   do begin
      if J<=UpMark[K]
      then Temp[0,J]:=0
      else Temp[0,J]:=T[Low+J];
   end; {for J}

end; {Tempboundary}

Procedure IterateT;
{Solves for the temperature distribution}
begin
   ReqExactness:=false;
   SumRes:=0;
   SumTemp:=0;

   For I:=0 to N                            {symmetry line}
   do Temp[I,M+1]:=Temp[I,M-1];

   if JustReversion
   then Alpha:=0.8
   else alpha:=AlphaTemp;
```

```
   {At the flow region}
   For I:=1 to N-1
   do begin
      For J:=1 to M
      do begin
         R:=I/N;
         RS:=1-R*(1-Fiter[J]);
         DefineLiquid;
         D1:=D1-R/(1-FIter[J])*(VVelocity[I,J]*(Fiter[J]-F[J,K-1])
         /ZStep+2*SteL*(Fiter[J]-F[J,K])/TimeStp);
         Residue:=2*SteL/TimeStp*Temperature[I,J,K]+VVelocity[I,J]/
         ZStep*Temperature[I,J,K-1]+A1*N*N*(Temp[I+1,J]+Temp[I-1,j])
         +B1*N/2/(DeltaFi[J]+DeltaFi[J+1])*(DeltaFi[J]/DeltaFi[J+1]*
         (Temp[I+1,J+1]-Temp[I+1,J]-Temp[I-1,J+1]+Temp[I-1,J])
         +DeltaFi[J+1]/DeltaFi[J]
         *(Temp[I+1,J]-Temp[I+1,J-1]-Temp[I-1,J]+Temp[I-1,J-1]))
         +2*C1/(DeltaFi[J+1]*DeltaFi[J])/(DeltaFi[J+1]+DeltaFi[J])*
         (Temp[I,J-1]*DeltaFi[J+1]+Temp[I,J+1]*DeltaFi[J])
         +D1/2*N*(Temp[I+1,J]-Temp[I-1,J]);
         Residue:=Residue/(2*N*N*A1+2*C1/DeltaFi[J]/DeltaFi[J+1]
         +2*SteL/TimeStp+VVelocity[I,J]/Zstep);
         Residue:=Residue-Temp[I,J];
         Temp[I,J]:=Temp[I,J]+Alpha*Residue;
         SumRes:=SumRes+Abs(Residue);
         SumTemp:=SumTemp+Abs(Temp[I,J]);
      end; {for J}
   end; {for I}
   if (SumRes/SumTemp)<0.0005 then ReqExactness:=true;.
writeln ('water ',k,' ',sumres/sumtemp);
end; {IterateT}

Procedure PrintMetal;
{Prints the temperature im the metal}
begin
   writeln(metalfile,'The time is:',Chronograph/NonDimT);
   for j:=0 to M2pNpQ
   do begin
      writeln (k,'  ',J,'  ',T[J]);
   end; {for}
end; {Printmetal}



Procedure PrintTemp;
{Writes into a file the temperature distribution}
begin
   writeln(tempfile,'The time is:',Chronograph/NonDimT);
   For J:=0 to 2
   do For I:=0 to N
      do Writeln(Tempfile,k,'  ',I,'  ',J,'  ',Temp[I,J]);
end; {PrintTemp}

Procedure PrintTice;
{Writes into a file the results of the ice temperature}
```

```
begin
writeln(Ticefile,'The time is',Chronograph/Nondimt);
   For J:=0 to 2
   do for I:=0 to N
      do writeln(Ticefile,k,'  ',i,'  ',j,'  ',TIce[I,J]);
end;

procedure PrintProfile;
{Writes the ice profile into a file}
begin
   writeln(Profile,'The time is:',Chronograph/NonDimT);
   For J:=0 to M
      do writeln(Profile,K,'  ',j,'  ',F[J,K]);
   Writeln(Profile,'UpperMark is:',UpMark[K]);
end;

Procedure CheckTemp;
{Checks for the presence of ice}
{It also sets UpMark[K] if there is ice on the external wall}
Var L,MarkerOld              :Integer;
    Delta, Delta1, Delta2    :Real;
begin
   IceMetal:=true;
   {the next lines determine the value of the marker}
   MarkerOld:=MarkerIter;
   J:=-1;
   Repeat
      J:=J+1;
      MarkerIter:=J;
      L:=M-J;
   Until ((T[L]<0) or (J=M));
   If T[0]>0 then MarkerIter:=M+1;
   If MarkerIter<M+1 then IceFormed:=true;
   If MarkerOld<>MarkerIter then IceMetal:=false;

   {the next lines determine the position of the ice at the fin}
   If markerIter=0
   then begin
      Flast0:=FlastIt;
      FlastIt:=0.5;
      Delta1:=(Fiter[0]-IntRad)/Q;
      Delta2:=(1-Fiter[0])/N;
      I:=0;
      Repeat
         I:=I+1;
         if ((I<=Q) and (Marker[K]=0))
         then Delta:=delta1
         else Delta:=Delta2;
         Flastit:=Flastit+Delta;
      Until ((T[M+I]>0) or (Flastit>=1));
      if T[M+I]>0
      then Flastit:=Flastit-Delta*T[M+I]/(T[M+I]-T[M+I-1]);
      if Flastit>=1 then Flastit:=1;
      Writeln(K,'  ',0,'  ',Flastit);
      if abs(Flast0-Flastit)>0.005 then IceMetal:=false;
```

```
    end; {then}

  {the next lines determine Upmark}
  if Flastit=1
  then begin
     MarkerOld:=UpMark[k];
     J:=M+1;
     Repeat
        J:=J-1;
        UpMark[K]:=J;
     Until ((T[J+M+Q]<0) or (J=-1));
     If UpMark[k]<>MarkerOld then IceMetal:=false;
     If ((abs(UpMark[K]-MarkerOld)=1) and (abs(T[J+M+Q])<0.003))
     then IceMetal:=true;
     If ((abs(UpMark[K]-MarkerOld)=1) and (abs(T[J+M+Q+1])<0.003))
     then IceMatal:=true;
  end {then}
  else begin
     UpMark[K]:=-1;
  end; {else}

end; {checktemp}

Procedure UpDateMetal;
{Transforms the old value of tha metal temperature matrix}
begin
   for j:=0 to M2pNpQ
   do Tmatal[J,K]:=T[J];
end; {UpDateMetal}

Procedure UpDateTemp;
{Transforms the old valua of the water temperature matrix}
begin
   For I:=0 to n
   do For J:=0 to M
      do Temperature[I,J,K]:=Temp[I,J];
end; {UpDateTemp}

Procedure UpDateTice;
{Transforms the old value of the ice temperature matrix}
begin
   For I:=0 to Q
   do for J:=Marker[K] to M
      do TempIce[I,J,K]:=Tice[I,J];
end; {UpDateTice}

Procadure UpDateF;
{Transforms the old valua of the ice profile matrix}
{It also checks if it is possibla to increase the
 timestep without losing accuracy}
begin
   For J:=0 to M
   do begin
      SlowGrowth:=false;
      F[J,K]:=Fiter[J];
```

```
   end; {for J}
end; {UpDateF}


Procedure AssumeF;
{The ice profile is assumed to start the calculations}
begin
   For J:=Marker[K] to M
   do begin
      if Fiter[J]=IntRad
      then begin
         Fiter[J]:=Fiter[J]+0.001;
         NewIce[J,K]:=true;
      end; {then}
   end; {for}
end; {AssumeF}

Procedure IceTemperature;
{Calculates the temperature profile in the ice}
Ver Factor:integer;
begin
   ReqExactness:=False;
   SumRes:=0;
   SumTemp:=0.001;

   {At the symmetry line}
   For I:=0 to Q
   do Tice[I,M+1]:=Tice[I,M-1];

   if Marker[K]>0
   then begin
      For I:=1 to Q-1
      do Tice[I,Marker[K]]:=Tice[0,Marker[K]]*(1-I/Q);
   end; {then}

   if JustReversion
   then Alpha:=0.8
   else alpha:=AlphaTemp;

   {The next lines calculate the temperature distribution in the
    ice core. A fully implicit method is used.}
   For I:=1 to Q-1
   do for J:=Marker[K]+1 to M
      do begin
         If newice[J,K]
         then factor:=1000
         else fector:=1;
         R:=I/Q;
         RS:=IntRed+R*(Fiter[J]-IntRad);
         DefineSolid;
         D2:=D2+R/(FIter[J]-IntRad)*2*SteS/Ki*(Fiter[J]-F[J,K])/TimeStp;
         Residue:=TempIce[I,J,K]*(2*Stes/Ki/TimeStp/factor)
         +A2*Q*q*(Tice[I+1,J]+Tice[I-1,J])
         +B2*Q/2/(DeltaFi[J]+DeltaFi[J+1])*(DeltaFi[J]/DeltaFi[J+1]*
         (Tice[I+1,J+1]-Tice[I+1,J]-Tice[I-1,J+1]+Tice[I-1,J])
```

```
            +DeltaFi[J+1]/DeltaFi[J]
            *(Tice[I+1,J]-Tice[I+1,J-1]-Tice[I-1,J]+Tice[I-1,J-1]))
            +2*C2/(DeltaFi[J+1]*DeltaFi[J])/(DeltaFi[J+1]+DeltaFi[J])*
            (Tice[I,J-1]*DeltaFi[J+1]+Tice[I,J+1]*DeltaFi[J])
            +D2*Q/2*(Tice[I+1,J]-Tice[I-1,J]);
            Residue:=Residue/(2*Stes/Ki/TimeStp/factor+A2*Q*Q*2
            +2*C2/DeltaFi[J]/DeltaFi[J+1])-Tice[I,J];
            Tice[I,J]:=Tice[I,J]+Alpha*Residue;
            SumRes:=SumRes+Abs(Residue);
            SumTemp:=SumTemp+Abs(Tice[I,J]);
      end; {for I, For J}

      {This completes the temperature calculations. The next line
       checks for convergence}
      If (SumRes/SumTemp)<0.001 then ReqExactness:=True;
writeln('ice ',k,' ',sumres/sumtemp);
end; {IceTemperature}

Procedure IceGrowth;
{Calculates the ice profile}
      Var Factor, Incr                                    :Real;
          Flast                                           :Vector;
          PipeFull                                        :Boolean;

      Function A(FRK:Real):real;
      begin
         Factor:=KI/(FRK-IntRad);
         A:=Factor*Q/4*(-4*Tice[Q-1,J]+Tice[Q-2,J])
         *(1+sqr(DF[j,k]/FRK/FI0));
      end; {Function A}

      Function B(FRK:real):Real;
      begin
         Factor:=1/(1-FRK);
         B:=Factor*N/4*(-4*Temp[N-1,J]+Temp[N-2,J])
         *(1+sqr(DF[j,k]/FRK/FI0));
      end; {Function B}

      Function Growth(FRK:real):real;
      begin
         Growth:=F[J,K]-FRK+TimeStp*(A(FRK)+B(FRK));
      end;

begin {IceGrowth}
      For J:=Marker[K] to M
      do FLast[J]:=0.50001;

      For J:=Marker[K] to M
      do begin
         If J>0
         then begin
            Incr:=0.02;
            PipeFull:=false;
            Repeat
               Repeat
```

```
            Flast[J]:=Flast[J]+Incr;
            If Flast[J]>=1 then Pipefull:=true;
         Until (((Growth(Flast[J])*Growth(Flast[J]-Incr))<0)
         or Pipefull);
         Flast[J]:=Flast[J]-Incr;
         Incr:=Incr/2;
         PipeFull:=false;
      until Incr<0.0001;
      if (Flast[J]>0.9) and (F[J,K]=0.5)
      then begin
         Flast[J]:=0.505;
         IceDoesNotGrow:=False;
      end; {then}
      writeln(k,'  ',j,'  ',Flast[j]);

      if abs(Flast[J]-Fiter[J])>0.0005 then IceDoesNotGrow:=False;
      Fiter[J]:=Flast[J];
   end; {then}
end; {For J}
end; {IceGrowth}

Procedure VelocityCalculation;
{This procedure iterates on the velocity distribution}
Begin
   Repeat
      IterateU;
      If not Optimum Then Sorbusiness;
   Until ReqExactness;
   Simpson;
   DimVelocity;
end;{VelocityCalculation}

Procedure WaterTemperature;
{This procedure calculates the temperature distribution in the liquid}
{and also in the metallic boundary}
begin
   CalculateMetal;
   PrintMetal;
   TempBoundary;
   Repeat
      IterateT;
   Until ReqExactness;
end; {WaterTemperature}

Procedure IceCalculations;
{This procedure calculates everything
 related to ice formation and ice growth}
begin
   for J:=0 to M
   do Newice[J,K]:=false;
   Repeat
      Repeat
         InitializeFiter;
         DerivativeF;
         VelocityCalculation;
```

```
         WaterTemperature;
         CheckTemp;
      Until IceMetal;
      IceDoesNotGrow:=true;
      If MarkerIter<>Marker[K] then icedoesnotgrow:=false;
      Marker[k]:=MarkerIter;
      Fiter[0]:=Flastit;
      InitializeFiter;
      AssumeF;
      Repeat
         IceTemperature;
      Until ReqExactness;
      IceGrowth;
      writeln(icedoesnotgrow);
   Until IceDoesnotGrow;
end; {IceCalculations}

Procedure Reversion;
{Simulates the reversion of the flow by switching the
direction of the Temperature, TempIce, F and Marker matrixes}
Var FSwap, Tswap: Real;
    MarkerSwap:Integer;

begin
   For K:=1 to ((P div 2)-1)
   do begin
      For J:=0 to M2pNpQ
      do begin
         Tswap:=Tmetal[J,K];
         Tmetal[J,K]:=Tmetal[J,P-K];
         Tmetal[J,P-K]:=Tswap;
      end; {for J}
      For J:=0 to M
      do begin
         FSwap:=F[J,K];
         F[J,K]:=F[J,P-K];
         F[J,P-K]:=FSwap;
         For I:=0 to N
         do begin
            Tswap:=Temperature[I,J,P-K];
            Temperature[I,J,P-K]:=Temperature[I,J,K];
            Temperature[I,J,K]:=Tswap;
         end; {for I}
         For I:=0 to Q
         do begin
            Tswap:=TempIce[I,J,P-K];
            TempIce[I,J,P-K]:=TempIce[I,J,K];
            TempIce[I,J,K]:=Tswap;
         end; {for I}
      end; {for J}
      Markerswap:=Marker[P-K];
      Marker[P-K]:=Marker[K];
      Marker[K]:=Markerswap;
      Markerswap:=UpMark[P-K];
      UpMark[P-K]:=UpMark[K];
```

```
      UpMark[K]:=Markerswap;
end; {for K}

   {The next lines start to zero the
   values at the old entrance of the pipe}
   Marker[P]:=M+1;
   UpMark[P]:=-1;
   For J:=0 to M2pNpQ
   do Tmetal[J,P]:=1;
   For J:=0 to M
   do begin
      F[J,P]:=IntRad;
      For I:=0 to N
      do Temperature[I,J,P]:=1;
   end; {for J}
end; {Reversion}

Procedure HeatCalculationWater;
{Calculates the overall heat transfer carried out by the water}
begin
   HeatWater:=HeatWater+DDensity*DspecHeatLiq*DFlow*Timestp/NonDimT
   *(1-Tmean)*(DTinit-DTfreeze);
   HeatIce:=VolumeIce*DDensity*(DLatHeat+(DTinit-DTfreeze)
   *DSpecHeatLiq);
   HeatExistIce:=ExistingIce*DDensity
   *(DLatHeat+(DTinit-DTfreeze)*DSpecHeatLiq);
   OverallHeat:=HeatWater+HeatIce;
   Power:=OverallHeat/(Chronograph/NonDimT);
   writeln(heatfile,Chronograph/NonDimT,'  ',HeatWater,'  ',HeatIce,
   '  ',HeatExistIce,'  ',OverallHeat,'  ',Power);
   writeln(Chronograph/NonDimT,'  ',HeatWater,'  ',HeatIce,
   '  ',HeatExistIce,'  ',OverallHeat,'  ',Power);
end; {HeatCalculationWater}

Procedure HeatCalculationIce;
{Calculates the heat stored by the ice in the pipe plus the
 heat carried out by the ice that pops out}
Var Increase     :Vector;
begin
   {the first lines calculate the amount of
   ice currently existing in the pipe}
   ExistingIce:=ExistingIce+IceArea*DAxialSt;
   writeln(ExistingIce);
   For J:=0 to M
   do begin
      Increase[J]:=Fiter[J]-F[J,K];
      if ((J=0) and (Increase[0]<0))
      then Increase[0]:=0;
      if ((J<Marker[K]) and (Increase[J]<0))
      then Increase[J]:=0;
   end; {for}
   GrowthVol:=0;
   For J:=1 to M-1
   do GrowthVol:=GrowthVol+Increase[J]*Fiter[J]
   *(DeltaFi[J+1]+DeltaFi[J])/2;
```

```
   GrowthVol:=GrowthVol+Increase[0]*Fiter[0]*DeltaFi[0]/2+
   Increase[M]*Fiter[M]*DeltaFi[M]/2;
   VolumeIce:=VolumeIce+GrowthVol*sqr(DExtRad)*Fi0*DAxialSt*2*Fins;
end; {HeatCalculationIce}



{The main program starts.}
{The required non-dimensional parameters and other
 constants are defined in the next lines}

begin {fins}
IntRad:=DIntRad/DExtRad;
FI0:=Pi/Fins;
Alpha:=1.4;
AlphaOld:=1.4;
SumResOld:=1000;
Ks:=DKSol/DKLiq;
KI:=DKIce/DKLiq;
T0:=DThickness/DExtRad;
Nu:=DHtCoef*DExtRad/DKLiq;
Tw:=(DTCool-DTFreeze)/(DTInit-DTFreeze);
SteL:=DSpecHeatLiq*(DTInit-DTFreeze)/DLatHeat;
SteS:=DSpecHeatLiq*(DTInit-DTFreeze)/DLatHeat;
DensSol:=DDensitySol/DDensity;
Cps:=DSpecHeatSol/DSpecHeatLiq;
NonDimT:=2*DKLiq*(DTinit-DTfreeze)/DDensity/DLatHeat/sqr(DExtRad);
TimeStp:=DTimeSt*NonDimT;
Umo:=DFlow/pi/(sqr(DExtRad)-sqr(DIntRad));
NonDimZ:=DKliq/DDensity/DSpecHeatLiq/sqr(DExtRad)/Umo;
ZStep:=DAxialSt*NonDimZ;
Chronograph:=0;
Optimum:=False;
VolumeIce:=0;
HeatWater:=0;

{The file assignments follow}

Assign(Profile,'PROFILE.DTA');
Rewrite(Profile);
Assign (tempfile,'temp.dta');
Rewrite(Tempfile);
Assign(Metalfile,'METAL.DTA');
Rewrite(Metalfile);
Assign(Heatfile,'Heat.dta');
Rewrite(HeatFile);

{The initializations follow}

InitializeF;
InitializeU;
```

```
InitializeDeltaFi;
InitializeMarker;
InitializeT;
InitializeTemperatureIce;
InitializeTmetal;
InitializeTvector;
JustReversion:=false;

{The loops start}

Repeat              {This repeat starts the time loop}
   SwitchChrono:=0;
   Repeat           {This repeat startsthe reversion loop}
      ExistingIce:=0;
      SwitchChrono:=SwitchChrono+TimeStp;
      Chronograph:=Chronograph+Timestp;
      SlowGrowth:=true;
      For K:=1 to P   {This is the beginning of the axial position loop}
      do begin
         writeln('the time is:',Chronograph/Nondimt);
         IceFormed:=False;
         InitializeTemp;
         InitializeTice;
         MarkerIter:=Marker[K];
         FlastIt:=Fiter[0];
         StartFiter;
         InitializeFiter;
         VelocityCalculation;
         WaterTemperature;
         CheckTemp;
         if not iceFormed
         then begin
            UpDateTemp;
            UpDatemetal;
         end
         else begin
            Icecalculations;
            UpDateTemp;
            UpDateMetal;
            UpDateTice;
            HeatCalculationIce;
            UpDateF;
            PrintProfile;
         end; {else}

      writeln('the ice-water nusselt number is:',Nus);
      writeln('marker',K,'  ',Marker[K]);

      {the loops close now}
      end; {for K}

   JustReversion:=false;
   HeatCalculationWater;
   if SlowGrowth then timestp:=2*timestp;
   if ((Switchchrono+timestp)/NonDimT)>DTimeRev
```

```
   then TimeStp:=DtimeRev*NonDimT-SwitchChrono;
   Until SwitchChrono/NonDimT+0.01>=DTimeRev;
Reversion;
timestp:=DTimeSt*nondimT;
JustReversion:=true;
Until Chronograph/NonDimT>=DTimeMax;

end.
```