MODELLING THE MICROCLIMATE IN A CONICAL BOTTOMED CIRCULAR GRAIN SILO

by

Ms. Yili Jin (B. Eng. Mechanical)

A thesis submitted for the degree
of *Master of Engineering*

Supervised by

Associate Professor G. R. Thorpe

VICTORIA UNIVERSITY OF TECHNOLOGY
LIBRARY

Department of Civil and Building Engineering
Faculty of Engineering
Victoria University of Technology, Australia

January, 1996

FTS THESIS
633.10468015118 JIN
30001004466803
Jin, Yili
Modelling the microclimate
in a conical bottomed
circular grain silo

DECLARATION

I hereby declare that this thesis is the result of my own research and has not been submitted for a degree to any other university.

Yili Jin

Department of Civil and Building Engineering

Faculty of Engineering

Victoria University of Technology

Melbourne, Australia

ABSTRACT

A three-dimensional mathematical model for predicting the behaviour of microclimate in the stored grain silo is presented. This analysis is implemented by calculating the air pressure field within an aerated conical bottomed circular silo, to continuing prediction the velocity field. The obtaining results are used to compute the simultaneous heat and mass transfer processes that occur in non-aerated and aerated grain silos. The conservation equations of momentum, mass and energy are employed in this study. The numerical scheme is based on applying a mapping technique with an algebraic grid generation method, to discretise the governing equations in the arbitrary geometries that describe the grain stores.

The finite different solution of the governing equations are obtained by using Alternating Direction Implicit method, Thomas algorithm and explicit method. The numerical experiments have been performed to investigate the effects of the non-uniform grid size, time step, duct shape and placement. Results in graphical form are presented for the pressure, velocity, temperature, moisture content, absolute humidity, wet bulb temperature, dry matter loss, seed viability and pesticide decay.

Significantly, a novel configuration of an annular aeration duct and a traditional linear aeration duct in farm silos have been investigated. It was found that the annular aeration system provides uniform airflow results in better protection of the stored grains than in the linear aeration system, in that the beneficial grain quality indices remain high, and the rate of growth of insect population is retarded.

An outcome of the research has been improved the designs of commercially available grain silos.

ACKNOWLEDGMENTS

I would like to express my sincere thanks to my supervisor Associate Professor G. R. Thorpe, for his invaluable encouragement and guidance during this research and writing of this thesis. Without his scrutiny of so many revised versions, my dream of completing this thesis would never become true.

I would also like to express my gratitude to Dr Mingchun Luo (Centre of Environmental Safety and Risk Engineering, VUT), for his constant help, advice and director, when I had many troubles with running my program.

Thanks to Mr. Clarence Perera, who gave me many suggestions during long discussions. Thanks to my department staff Mr. Jack Li and Mr. Doug Smith, for their assistance. Thanks to my colleagues for their useful help.

Finally, I am extremely grateful to my fiance Adrian for his love, and the shared sacrifices and happiness during the difficult times. Thanks to my parents, for they taught me the importance of a strong work ethic and perseverance. To them I dedicate this thesis.

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

a_1, a_2, a_3, a_4	- isosteric specific empirical constants ($1/m$)
aeration	- ventilation
b_1, b_2, b_3, b_4	- respiration specific empirical constants
b_5, b_6, b_7, b_8	- respiration specific empirical constants
B	- pesticide specific empirical constants ($1/s$)
c_{1v},	- empirical constants in latent heat equation (J/kg)
c_{2v}	- empirical constants in latent heat equation ($J/kg\ K$)
c_{c1}, c_{c2}	- species specific constants
c_a	- specific of dry air (J/kgK)
$(c_1)_\sigma$	- specific heat of liquid water (J/kgK)
$(c_2)_\sigma$	- specific heat of grain kernel (J/kgK)
$(c_1)_\gamma$	- specific heat of vapour water ($J/kg\ water\ K$)
C_i	- initial concentration of the pesticide (kg/m^3)
C_{pes}	- pesticide concentration (kg/m^3)
dtg	- time step during aeration cycle (s)
D_{eff}	- effective diffusivity (m^2/s)
Fo	- Fourier number
gn	- ratio of molecular weights of water and air
G_{sc}	- solar constant (W/m^2)
h_w	- differential heat of wetting (J/kg)
h_s	- heat of sorption (J/kg)

h_{vap}	- latent heat of evaporation of free water (J/kg)
$\frac{\partial h_{vap}}{\partial T}$	- differential of heat of vaporisation respect to the temperature (J/kgK)
H	- absolute humidity of the air (kg/kg)
H^*	- function of the saturation humidity of air
H_{duct}	- humidity of air entering aeration duct (kg/kg)
H_{life}	- half life of seeds (s)
H_W	- integral heat of wetting (J/kg)
H_{wall}	- height of the vertical wall of the silo (m)
$\bar{H}_d$	- average of daily diffuse radiation (J/m^2)
$\bar{H}_b$	- total average of daily beam radiation (J/m^2)
$\bar{H}_t$	- average of daily total radiation (J/m^2)
$ifreq$	- number of heat and mass transfer iterations
I_b	- total intensity of beam radiation (W/m^2)
I_t	- total intensity of radiation (W/m^2)
I_d	- total intensity of diffuse radiation (W/m^2)
Δl	- space distance (m)
k	- intrinsic permeability (m^2)
K_{eff}	- effective thermal dispersivity tensor ($J/kg/K$)
m	- mass of the substrate lost (kg)
$\frac{dm}{dt}$	- rate of dry matter loss (kg/s)
M_w	- grain moisture content wet basis ($\%$)
n	- day of the year
n_s	- average of sunshine hours per day
$n\xi$, $n\eta$, $n\phi$	- maximum nodes points in the three components directions
N	- length of the day ($hours$)
P	- pressure gradient (Pa/m)

ΔP	- pressure difference (Pa/m)
P_{atm}	- atmospheric pressure (Pa)
P_{sat}	- saturation vapour pressure (Pa)
P_{vap}	- vapour pressure (Pa)
P-centre	- pressure at centre of silo (Pa)
Pcon-min	- pressure at centre of silo with minimum value of iteration step (Pa)
$Psi\max$	- the largest value of $\psi_{\max}$,
Q_r	- cumulative consumption of oxygen, (kg)
rh	- relative humidity of air in equilibrium with the grain
r,ϕ,z	- cylindrical coordinates
r_s	- intrinsic rate of insect population growth ($1/s$)
$r_{duct\min}$, $r_{duct\max}$	- length bounds of the duct (m)
R	- radius of the grain silo (m)
R_b	- ratio of the total intensity of beam radiation
R_t	- ratio of the total intensity of radiation
R_d	- ratio of the total intensity of the diffuse radiation
R_P	- coefficient resistance to the airflow ($Pa\ s/m$)
R^P	- previous population of insect pests
S	- second Ergun (1952) coefficients ($Pa\ s^2/m^3$)
ΔS	- element of length (m)
Δt	- iteration time step (s)
t^*	- pesticide specific empirical constants (s)
t_{ime}	- total elapsed time of storage (s)
$time^*$	- total elapsed time of heat and mass transfer iteration (s)
T	- temperature ($°C$)
T_a	- ambient air temperature ($°C$)
T_g	- ground temperature ($°C$)

T_s	- surface temperature of the silo (K)
T_{sky}	- sky temperature (K)
T_{wetb}	- wet bulb temperature of the intergranular air ($°C$)
V	- velocity (m/s)
V_{wind}	- wind velocity across the surface of the silo (m/s)
V_{iab}	- fractional viability of seeds
V_r, V_ϕ, V_z	- three components of velocity in the cylindrical coordinates (m/s)
V_ξ, V_θ, V_η	- three components of velocity in the dimensionless coordinates ($1/s$)
$V_{T\xi}, V_{T\eta}, V_{T\theta}$	- effective velocities with respect to the temperature (m/s)
$V_{W\xi}, V_{W\theta}, V_{W\eta}$	- effective velocities with respect to the moisture content (m/s)
W	- moisture content of the grains
Z_{total}	- total height of the silo (m)

GREEK SYMBOLS

α_p	- false transient factor
β	- sloping angle (*radians*)
Γ	- second partial derivatives of the scalar
γ_1	- angle of silo floor to the horizontal plane (*radians*)
γ_2	- angle between upper surface of silo to horizontal plane (*radians*)
δ	- weighting factor of Crank-Nicholson method
δ_d	- angle of the declination (*radians*)
ε	- seed bulk porosity
ε_r	- initial void fraction of the bulk of grain

$\xi_{duct\,\min}, \xi_{duct\,\max}$	- length bounds of the duct in transformed coordinates
ξ, ϕ, η	- non-dimensional transformed coordinates
θ	- angle of beam radiation (*radians*)
θ_z	- zenith angle (*radians*)
Θ	- arbitrary variable
μ	- dynamic viscosity (Pa)
v_1, v_2, v_3, v_4	- seed specific constants
ρ_a	- density rate of the dry air (kg/m^3)
ρ_b	- bulk density (kg/m^3)
ρ	- the density of air (kg/m^3)
σ	- Stefan-Boltzman constant (W/m^2T^4)
ω_a	- solar hour angle (*radians*)
ω_s	- sunset hour angle (*radians*)
ϕ_l	- latitude of the geographical location (*radians*)
$\phi_{duct\,\min}, \phi_{duct\,\max}$	- width bounds of the duct (*radians*)

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

Humans have been storing grains and their products for a long time. At about 8000 BC, when humans began to cultivate plants and raise domesticated animals, they were concerned to build up reserve food supplies to alleviate hunger that might arise in the future. Over the last 10,000 years, the existence of hunger and malnutrition have became unacceptable morally and socially. Humans have recognised that the unavailability of a sufficient quantity and quality of food has been one of the major constraints in the socioeconomic progress throughout the world.

Grains contribute more than any other staple foods with regard to calorie, carbohydrate, protein, lipid, mineral and vitamin requirements of the world population (Godon, 1994). The particular regional ecologies and climates provide the world population with its basic food: rice in Asia, wheat in North American, Europe, the Middle East, and North Africa, maize in Latin American. According to F. A. O. analysis, that 90% of the world's food supplies come from the land, among which grains have occupied an incredible 68% of the total world harvested crops. The other 22% is accounted for fruits and vegetables. The remaining 10% of man's food comes from the world's oceans, seas and lakes (Bushuk and Rasper, 1994). This can be shown in a pie chart in figure 1.1.

Moreover, the grain supply in national and international trade is of great concern and influence in world markets.

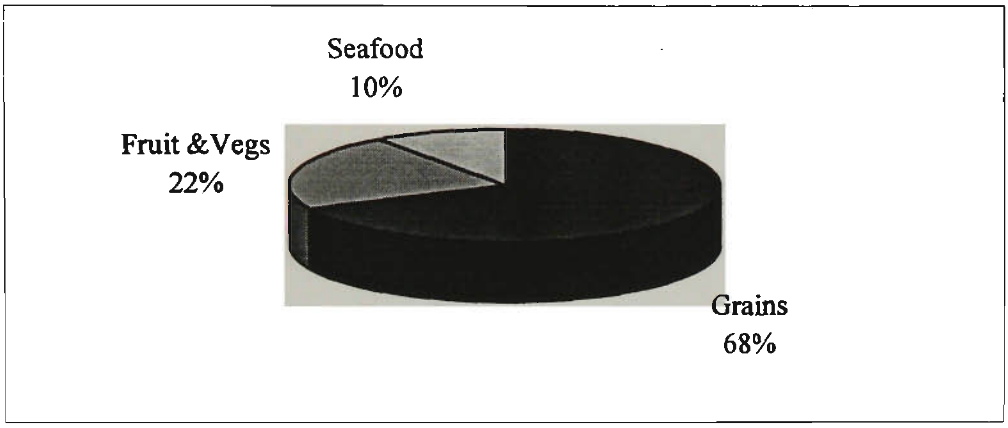


Figure 1.1 Food structure of world population

From 1982 to 1987, the world grain production exceeded 1500 million metric tonnes annually (table 1.1), about 360 kg per world inhabitant , that is therefore an important source of food in the world.

Table 1.1 The world production of grains

	Million tonnes	
Grain type	1982	1987
Wheat	453.2	515.4
Rye	24.3	34.8
Barley	160.4	182.0
Oats	46.0	48.0
Maize	438.0	451.6
Rice	410.9	457.5
Total	1630.8	1788.6

Source: Chelkowski (1991)

It is estimated that 10~30% of total world grain production is lost after harvest (Chelkowski, 1991), and it will vary from country to country and from year to year.

The reasons for this huge amount of losses are highly complex and governed by various natural factors: Harvesting, drying, transportation, storage, processing and marketing. Especially, the major cause of this losses in the quantity and quality of grains is during storage.

During storage, grains are liable to suffer changes in texture, colour, flavour, and nutritive value (Holdsworth, 1983). That is to say, stored grains form an ecological system in which deterioration arises from interactions among physical, chemical and biological variables. These can be described as: Moisture movement, temperature increasing, oxygen consumption, insects, mites, moulds and rodents causing damage.

Moisture is one of the basic physico-chemical variables. The moisture content of stored grain is defined as the amount of water that can be removed without alterations to its chemical structure. The variation of moisture content is dependent on the type of grain, the variety of grain, chemical composition, moisture at harvest, relative humidity of the atmosphere, and seasonal fluctuations (Salunkhe *et al.*, 1985).

Moreover, the maximum moisture content for safe storage of grains are normally based on the equilibrium moisture content at $25^{o}C$ and 75% relative humidity. These have summarized in table 1.2.

Temperature is a physical variable in grain storage, its value is strongly affected by the storage environment conditions, thus: the initial grain temperature, the intergranular air temperature, solar radiation which affects the surface temperature of the storage wall, metabolic heating due to the formation of hot spots and weather conditions.

The oxygen of atmospheric air is the most important single chemical variable influencing the growth and development of all harmful organisms in storage, because insects, mites, mould and rodents all require free oxygen for their development.

Table 1.2 Advised maximum moisture content for safe storage of grain at 25°C and 75% RH.

Grain type	Moisture (%)
Wheat	
durum	14.1 (25-28 Celsius)
red	14.7 "
white	15.0 "
Rye	14.9 "
Barley	14.3 "
Oats	13.4 "
Maize	14.3
Rice	14.0

Source: Kent and Evers (1994)

Besides, grain pests such as insects, mites, rodents and mould are the commonest biological variables which are associated with stored grains. Insects harbour microbes in grain stores, and they can act as vectors of organisms causing human and animal diseases. Moulds can serve as food for insects (Sinha, 1971). The degree of grain pests activity in grain store is dependent on grain moisture and temperature, grain history and grain genotype, source of pests, available food, cleaning, drying, chemical residues in the grain mass, pesticide resistance of the grain pests (Brooker *et al.*, 1992).

Mould damage to stored grain includes: Decrease in viability and discolouration of seeds, production of toxins, biochemical changes and loss in dry matter (Brooker *et al.*, 1992). Similarly, the insect pests damage stored grain also includes: Consumption of whole grains or germ, generation of heat which results in quantitative, qualitative and viability loss (Salunkhe *et al.*, 1985).

In addition, viability and respiration are the two important biological properties responsible for the death of seed and the spoilage of grain in storage. Seed as an index of grain soundness. The loss of seed viability is associated with respiration, which is a function of grain moisture content, temperature and light (Pomeranz, 1987). Respiration may contribute to a loss of dry matter, raise the level of carbon dioxide in the intergranular air, and also increase the temperature of the grain. The process of respiration is regulated by moisture and temperature, size and shape of grain, harvesting and postharvest ripening (Sinha and Mair, 1973).

A major proportion of the stored grain is damaged by insects, mites and rodents, thus infestation, growth and survival of insect pests in the stored grains are highly interrelated and need a careful appraisal to plan the strategy for their effective control.

A number of control methods have been developed, such as: Hygiene, pesticides, fumigants, controlled atmosphere, high temperature heating, ambient and refrigerated aeration. Amongst them, aeration control is one of the most powerful physical methods. Ambient aeration is the process of moving air at ambient temperature through stored grain, it lead to an improvement of the grain temperature and storage conditions, and can be accomplished in any regions of the world. Refrigerated aeration relies on mechanically cooling the air behaviour to continue the process of aeration.

Grain must be protected during all stages of handling, from the time of harvest through storage, transportation, and processing up to the time it is ready to be consumed. As the population increases and resources become more limited, the continuing challenge to feed the world's population is becoming more increasingly urgent. If that 10~30% of losses of grains are eliminated through improved postharvest biotechnology, the world's food supply can be increased by 10~30% without any additional cultivating, labouring and energy consuming.

In order to protect resources, to reduce hunger, ensure humanity's health and to have sufficient food supply, the study of preservation of stored grain for improvement the technology in the postharvest handling and processing is the important responsibility for scientists and engineers.

1.2 CONTRIBUTIONS

The main contributions of this thesis are addressed as follows:

- A numerical method of finite different mathematical model has been presented for an aerated conical bottomed circular silo .
- A mesh transformation technique that may be used to model grain stores of arbitrary shape has been applied.
- Energy of solar radiation which has a significant effect on the surface temperature of the silo has been accounted.
- The SILO3DFTN computer program for accommodating the specific characteristics of the silo has been modified.
- The numerical experimental results of simulating the different mesh sizes and time steps have been obtained for evaluating the accuracy of the solution .
- The two types of aeration duct layouts fitted with conical bottomed circular silos have been investigated, for determining the best controlling results of aeration system to improve the design and the manufacture of the farm silos.
- A prototype grain silo fitted with an annular duct has been designed and built.

1.3 ORGANIZATION

The contents of the remaining chapters are reviewed below:

Chapter 2 describes the background information necessary to understand the significant postharvest knowledge on controlling the quality and preservation of grain. These are considered in the two major parts of review: The relationship between air velocity to the pressure gradient for the physics of air flow through the silo are mentioned. The crucial importance of grain temperature and moisture content on grain quality are highlighted.

Chapter 3 provides the numerical method of finite different for analysing the conservation of momentum, mass and energy in an arbitrary geometry, and formulation the governing differential equations and the appropriate boundary conditions, by introducing the first step of mapping technique - the coordinate transformation scheme, which transforms the non-orthogonal mesh in the physical domain into a orthogonal mesh in the computational domain.

Chapter 4 provides the solution of finite different method by means of creating the second step of mapping technique - non-uniform grid mesh scheme, to discretise the governing differential equations and boundary conditions in the computational domain by using a central difference approximation for the space derivatives, and a forward difference approximation for the time derivative. The results are obtained by employing the Samarskii-Andreyev (1963) alternating direction implicit method, the Thomas algorithm and an explicit method for heat and mass conservation equations.

Chapter 5 covers the physical and biological properties of grain bulk ecosystems, which affect on the stored grain, such as: thermal capacity, conduction, absorption, diffusion, gemination, respiration and insect infestation. The major environmental variables affecting the microclimate in the grain bulk, namely radiation and ambient conditions.

These properties are associated with heat and mass transfer processes and physiological changes in the grain silo.

Chapter 6 presents the experiments of numerical investigation by using the computer program SILO3DFTN to simulate the performance of a conical bottomed circular silo, in which the effects on the time step and mesh size different have been investigated. Fourier number (Fo) is applied to examine the prospects of convergence and accuracy of the pressure and temperature distribution solutions. In addition, two types of aeration system layouts are also simulated.

Chapter 7 discusses the heat and mass transfer processes occurring in the three dimensional ecological systems by simulating with three different causes, namely the silo with non-aerated system, the silos fitted with linear aeration and annular aeration systems. The predictions are carried out to estimate the variation of pressure, velocity, temperature, moisture content, absolute humidity, wet bulb temperature, dry matter loss, seed viability and pesticide distributions in the grain silo.

Chapter 8 summarizes the contributions presented in this thesis, and gives some further suggestion for continuing this research.

CHAPTER 2

REVIEW OF RELATED WORK

This chapter reviews literature related to grain storage technology, which was used to develop novel silo design and insights into the ecology of grain storage.

2.1 AERATION

Aeration is a commonly used storage technique for cooling and drying grain reducing thermal gradients, avoiding the incidence of moisture migration, decreasing insect and mould activity, lowering the rate of pesticide decay and preserving the grain quality (Thorpe, 1986).

Aeration cooling is achieved by blowing cold night air through the stored grain by means of a fan, perforated air distribution and ducting on the silo floor (Elder, 1978). When grain is cooled by aeration, moisture content may be very slightly reduced. When grain is warmed by aeration through the movement of air at ambient temperature, moisture is added. Aeration systems can also be used for distributing fumigants, temporarily holding wet grains and clearing dust during grain movement operations.

The practice of aerating stored grain was established during the 1950s and is now an accepted quality maintenance measure (Foster, 1967). Aeration development was coincident with the build up of reserve stocks of grain following World War II, the larger flat storages were used to hold the reserve grains, aeration was for the first time applied for the control of grain quality. The experience with aeration in storage was favourable. Today, aeration has became a powerful tool for protecting the grain during storage.

There are two somewhat different types of aeration systems: Ambient aeration and refrigerated aeration. In temperate and subtropical climates, grain can be effectively cooled by means of ambient aeration. In tropical climates, the high relative humidity and temperature of ambient air make aeration less feasible, the cooling of grain must by refrigerated aeration.

Aeration systems consist of a fan, a supply duct leading from the fan to the perforated duct placed in the grain bulk. Some provision must be made for airflow into or out of the air space over the grain surface.

The operation of aeration systems depends on a number of factors: the rate of airflow, the moisture content of the product, the shape and size of the grain kernels, the density of the grain bulk, the type, number and location of the duct (Yang and Williams, 1990).

2.2 AIRFLOW RATE

Agricultural products are commonly stored in aerated stores for both short and long term storage. In order to maintain the temperature throughout the grain to prevent moisture migration, aeration generally uses an above floor duct system which creates non-uniformity and non-linearity in the airflow pattern. Therefore, the determination of pressure and airflow distributions is important to simulate the aeration system and storability, while ensuring that all areas of stored grain can receive sufficient airflow.

Early studies of airflow rate and pressure difference relationships have been presented by Ergun (1952) and Shedd (1953). Ergun (1952) predicted a formulation for incompressible airflow through a packed bed, and it can be expressed as:

$$\frac{\Delta P}{L} = RV + SV^2 \tag{2.1}$$

where ΔP is pressure difference along the length L, V is the velocity in the direction of airflow, R and S are empirical coefficients depending on the type of products.

Shedd (1953) measured the static pressure drop across common grains and seeds, and proposed the possibility of expressing this relationship by the mathematical equation, thus:

$$= A(\frac{\Delta P}{L})^B \tag{2.2}$$

where A and B are seed - specific constants, which are determined from Shedd's (1953) experimental data.

Hukill and Ives (1955) suggested that Shedd's (1953) equation can only apply approximately for narrow ranges of velocity fields, if the constant coefficients in equation can be modified, the accuracy throughout of the velocity ranges will be increase. The logarithmic formula which they proposed is:

$$\frac{\Delta P}{L} = \frac{aV^2}{\ln(1 + bV)} \tag{2.3}$$

where a and b are also the seed - specific constants.

Two major numerical methods have been developed to solve the pressure and velocity fields in storage, namely finite difference and finite element methods. (In this study, we shall exploit the finite difference method).

Brooker (1961, 1969) used the information given by Shedd (1953) to develop a finite difference computer simulation model, for establishing pressure patterns in a grain mass subjected to non-linear airflow resistance. Jindal and Thompson (1972) modified Brooker's model, and adapted it for predicting the pressure distribution and airflow paths through the long triangular shaped piles of grains. Pierce and Thompson (1975) modified Jindal and Thompson's (1972) simulation model for prediction airflow characteristics in conical shaped piles of grain. However, the mathematical models which describes the flow of air through stored grain assume that the steady state has been established and the flow of air is taken as incompressible.

By stating of the Ergun's (1952) equation in a vectorial form, Stanek and Szekely (1972, 1973, 1974) presented a formulation and measured the effect of non-uniform porosity on airflow maldistribution in isothermal and non-isothermal packed bed chemical reactors. They observed that the airflow maldistribution through packed bed results from a spatially non-uniform resistance to the airflow. This resistance is caused by variable porosity, particle size distribution and hot spot generation within the bed. Lai (1980) developed Ergun's (1952) equation into three vector components, and introduced the stream function for solving the non-linear partial differential equations that describe the airflow through porous media of grain bed. Hunter (1983) has given resistance coefficients for airflow through 28 different seeds, which are obtained from Ergun's (1952), by fitting them to Shedd's (1953) data. He analysed experimental data and compared them with Hukill and Ives' (1955) logarithmic expression for pressure gradient. Hunter's(1983) formula can be applied to most common duct store situations.

The application of the finite element method to solve the airflow problem was first used by Marchant (1976b) for calculating the airflow distribution and pressure patterns in several different storage structures. Since then, several finite element models have been developed by Segerlind (1982), Smith (1982), Rumsey and Fortis (1984) and Chapman *et al.* (1989) to analyse air velocity and pressure in grain store.

Thorpe and Hunter (1977) have presented a finite difference solution of Laplace equation to airflow distribution in a non-uniformed aeration circular silo. The linear algebraic technique is used to solve the banded form of the coefficient matrix. The explicit analytic expressions allow pressure and velocity distributions to be easily calculated. Jayas and Muir (1991) have developed a method of incorporating the effect of airflow direction on pressure drops across granular beds in the mathematical models, for predicting the static airflow pressure patterns. Goudie *et al.* (1995) presented a computer model for simulation the linear analysis in low velocities and the non-linear approach in larger velocities. They have shown that velocity in the non-linear analysis was lower than the velocity in the linear analysis; the effect of the non-linearity is to move airflow away from the higher velocity regions towards the lower velocity regions.

2.3 CHANGES IN GRAIN MOISTURE CONTENT DUE TO TEMPERATURE GRADIENTS

Grain temperature and moisture content are the two major factors affecting grain quality during storage. Seasonal and diurnal variations in ambient temperature and solar radiation can create temperature gradients in the grain store, and cause moisture to migrate from warmer to cooler regions of the grain mass. This migration occurs slowly but steadily in grain storage. In order to attain the local equilibrium with the surrounding air, the warmer region of grain loses moisture and the colder region of grain gains moisture. That is because the transport processes tend to reduce the water vapour gradient and at same time the sorption relationship tends to maintain gradients, and the problem of moisture migration becomes dynamic in nature.

Therefore, the moisture migration process can be primarily considered as the diffusive and convective transport of the water vapour through the intergranular spaces of the grain storage where grain moisture acts as a source for the water vapour. The flow diagram of moisture migration phenomena is shown in figure 2.1 (Khankari *et al.*, 1994).

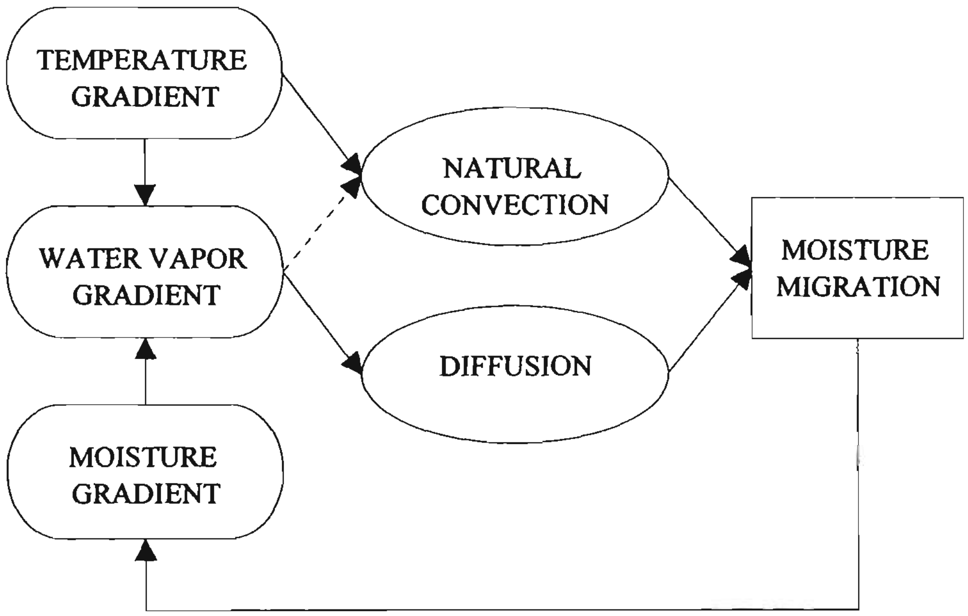


Figure 2.1 Moisture migration phenomena (after Khankari *et al.*, 1994)

However, the behaviour of migration of moisture is dependent on the amount of water present, structure of the storage, history of the grain temperature and atmospheric of relative humidity. Noyes *et al.* (1989) reported that moisture migration in stored grain is considered second to insect damage as a source of storage loss. The accumulation of moisture in a localised area of the grain mass stimulates the growth and development of fungi and insects (Hunt and Pixton, 1974), makes this region more susceptible, and leads to spoilage and microbial activity.

Several investigators have studied that moisture movement through the grain storage caused by water vapour diffusion or convection. Pixton and Griffiths (1971) described the diffusion of moisture through stored wheat by means of a diffusion coefficient which is an empirically determined function of grain moisture content and temperature. The relationship between the moisture content of grain and its equilibrium relative humidity is assumed to be essentially linear in the moisture content range considered. Stewart (1975) showed that the gradient of vapour pressure, which is responsible for the diffusion transport of moisture, can be developed from the gradients of moisture and temperature. His theoretical analysis stated that temperature gradient provided the driving force for moisture migration during storage. Thorpe (1981) has carried out an analysis of Pixton and Griffiths's (1971) work, and presented the mechanism of moisture migration is the equalisation of vapour pressure by molecular diffusion in the gas phase.

Grain and surrounding air are always in thermodynamic equilibrium, therefore grain moisture, relative humidity and temperature of the air can be determined from the sorption isotherm. Thorpe (1982) developed Smith's (1947) formula for sorption isotherm, and predicted the thermodynamic and physical properties of wheat, which may be used to calculated the rate of diffusion of moisture through wheat bulks subjected to a temperature gradient.

The relationship between equilibrium moisture content and equilibrium relative humidity of grain is used to predict the moisture content of the stored grain in equilibrium with intergranular air, to optimise storage conditions and quality of grain. Many theories have been forwarded to explain the phenomenon of hysteresis in the desorption and adsorption isotherms of porous grain, they are developed by: Henderson (1952), Chung and Pfost (1967a, 1967c), Foster (1967), Whitney and Porterfield (1968), Pixton and Warburton (1971), Versavel and Muir (1988), Mazza *et al.* (1990) and Sun and Woods (1993).

Most of work on predicting moisture movement in grain has been related to drying problems (Parry, 1985). Drying is the removal of moisture through grain and it is carried out by artificial means. It is likely to be a continuous process with changes in grain moisture content, temperature and absolute humidity all occurring simultaneously (Thompson *et al.*, 1968). In fact, these changes are related to different drying methods and different locations in the drying bed. Many tests have been made to describe the drying process, to determine drying time and safe drying conditions. Mathematical modelling and computer simulation are also widely used in research, in designing and in developing more efficient and safe grain drying. Normally, developing these models consists of two steps: Thin layer drying model and an in-bin drying model.

The application of thin layer drying models in deep beds has been given by Thompson *et al.* (1986), Henderson and Henderson (1968), Barre *et al.* (1971), Tang and Sokhansanj (1993), Parti (1993), Arinze *et al.* (1994) and Sun and Woods (1994). In-bin drying models have been developed by Spencer (1969), Bloome and Shove (1972), Ingram (1979), Smith *et al.* (1992) and Abawi (1993).

Drying has been the most common method for preserving grain in both cool and moist regions of the world and tropical regions. The proper drying is a precondition for safe storage and delivery. The climate, weather, airflow rate and depth of grain bed, the management of filling, fan and supplemental heat are all important factors affecting the performance of deep bed driers (Sharp, 1982).

2.4 INVESTIGATIONS IN CYLINDRICAL AND OTHER STORES

Numerical methods have served as useful tools for predicting the temperature and moisture content in grain storage. During the last 20 years, a number of models have been developed for modelling heat transfer in stored grain in cylindrical bins, neglecting the interaction with moisture transfer (Jayas *et al.*, 1992). In recent years, a few numerical models in cylindrical storage have been developed:

Obaldo *et al.* (1991) has presented a two dimensional finite difference model which is based on Fick's Law of diffusion, to predict moisture content in an unventilated steel bin. They concluded: (1). The change in moisture content of unventilated storage was more pronounced at the top grain surface than any region of the storage. (2). The moisture gradient tended to be higher in the vertical axis than along the radius of the storage.

Abbouda *et al.* (1992) applied finite difference methods to modify a two dimensional model by free convection and heat generation from respiration of grain to predict grain temperature in cylindrical bin. Experimental data showed: (1). The bin size and initial moisture content affected the grain temperature difference between the top surface and the bottom of the bin. (2). The average grain temperature followed the trend of the ambient air temperature with a time delay.

Jayas *et al.* (1992) used a three dimensional finite element model to simulate the influence on the grain temperature of bin diameter, grain bulk height, bin wall material, bin ship and time. They found that: (1). Large diameter bins maintained warmer centre temperatures than small diameter bins. (2). The influence of heat loss from the grain surface on the grain centre temperatures was negligible for tall grain bulks. (3). A small diameter tall grain bulk maintained lower grain temperature than a larger diameter short grain bulk. (4). A white painted steel bin maintained coolest temperature and a galvanised steel bin gave the highest temperature among the bin wall. (5). The shape of the bins had little influence on the grain temperature.

Khankari *et al.* (1993a, 1993b) has presented the two dimensional numerical simulation models to predict the patterns of temperature, moisture and natural convection flows in stored grain. The predictions showed: (1). Moisture migration during winter caused an increase near the top of the bin and decrease near the bottom of the bin. (2). During spring or summer grain moisture decreased at the outer boundaries of the bin. (3). The moisture migration occurs in all size of bins, and the moisture starts accumulating earlier in small bins. (4). The effect of natural convection is more evident in the moisture transfer process than in the heat transfer.

Casada and Young (1994) developed two dimensional finite difference model to predict heat and moisture transfer due to short term and long term natural convection and diffusion in arbitrarily shaped porous media. They reported: (1). The energy and moisture transport equation were best solved using a modified Crank-Nicolson method that was developed to control the tendency for instability caused by the source terms in the equations. (2). The differences between the solid and fluid temperatures in porous media were largest near the side boundaries. (3). The natural convection currents contribute the temperature solution at the upper corners of the porous media.

Chang *et al.* (1994) have presented a two dimensional model in cylindrical silo using finite difference method, for periodic aeration and daily variations in soil temperature, ambient air conditions, and solar radiation. They concluded: (1). Moisture content difference between grain near the silo wall and away from silo wall were small for grain 1.0 m or more blow the top surface. (2). During summer months, moisture content of the grain near the top of surface decrease, and were small for grain in the middle and bottom layers.

Fohr and Moussa (1994) carried out a experimental investigations of heat and mass transfer in a cylindrical grain silo submitted to a uniform and periodic wall heat flux, for understanding the mechanisms of the particular convection and to predict the amount of water displaced. They observed: (1). The atmospheric factors, thus: solar radiation, air temperature, and wind velocity do not create a uniform heat flux. (2). The moist airflow out of the bulk grain can condense on the colder walls and be transformed into water drops.

Moreover, the heat and mass transfer processes occurring in any enclosure storage have been developed by numerical method. They are Andales *et al.* (1979), Fortes and Okos (1981), Khanhari *et al.* (1990), Freer *et al.* (1990), Singh and Thorpe (1992), Singh *et al.* (1993) and Khankair *et al.* (1994).

2.5 RELATED TOPICS

The simulation and measurement of cooling performance of aeration systems have been carried out by many workers. Thompson (1972) developed a simulation model to evaluate the effects of harvest date, initial moisture content, initial grain temperature and weather conditions on the temporary storage of high moisture shelled corn, through continuous ambient aeration. Hunter and Taylor (1980) have designed and applied a refrigerated aeration system to produce conditions hostile to insects throughout the grain bulk and to determine the design factors of airflow rate and distribution, heat loads and sources. Sutherland *et al.* (1983) applied an equilibrium model to predict the formation of temperature and moisture fronts, due to variation of front velocity successive cooling and heating or drying and wetting interact during aeration.

Thorpe and Elder (1982) have presented a finite difference mathematical model of the processes of heat and moisture transfer and pesticide decay in aerated grain, to obtain the effects on the rate of pesticide decay of initial grain moisture content, initial grain temperature, fan operation, airflow rates and geographic location. Thorpe (1986) has outlined and quantified the advantages and benefits of aeration, he also described the fundamental processes of heat and mass transfer that govern the rate and amount of cooling during the aeration process, such as the dry matter loss due to respiration, insect population dynamics, seed viability, and the rate of pesticide decay. Hunter (1986) has presented the formula which are based on thermodynamic considerations to calculate the variation of aeration time fraction and airflow rate. Bridges *et al.* (1988) developed a computer model that determine duct sizes and duct location for aeration of grains in flat rectangular structures based on system design. Ismail *et al.* (1991) have carried out a experiment to predicate the performance of solar-regenerated aeration system. This solar cooling system is particularly effective in tropical climates.

Several researchers have proposed empirical models for the effect of physical damage, hybrid and weather conditions on estimated dry matter loss: Thompson (1972), Krishnamurthy *et al.* (1975), Freer *et al.* (1990), Stroshine and Yang (1990) and Singh *et al.* (1993).

In addition, moisture and temperature are dependent on transport properties which make the moisture migration and heat transfer problems coupled and nonlinear. Dutta *et al.* (1988) and Deshpande *et al.* (1993) have determined physical properties of grain. Kazarian and Hall (1965) and Dutta *et al.* (1988) have estimated the thermal properties of grain. An isostere equation for calculate the ratio of differential heat of sorption to latent heat is presented by Hunter (1987). Thorpe *et al.* (1990) have modified Hunter's isostere equation to derive an equation for the integral heat of wetting.

In this thesis, some of the above approaches and characterizations of grain and system properties will be drawn together to formulate a mathematical model of ventilated grain bulks stored in conical bottomed farm silos.

CHAPTER 3

ANALYSIS

This chapter marks the first major step in the development of the numerical method with the form and the meaning of the momentum, mass and thermal conservation equations, that apply in the physical domain, to be transformed into a computational domain. The boundaries conditions which apply to the silo walls, the conical base of silo, the upper surface of grains and the aeration duct are also implemented in each equation. The physical configuration of the system studied in this thesis is shown in figure 3.1.

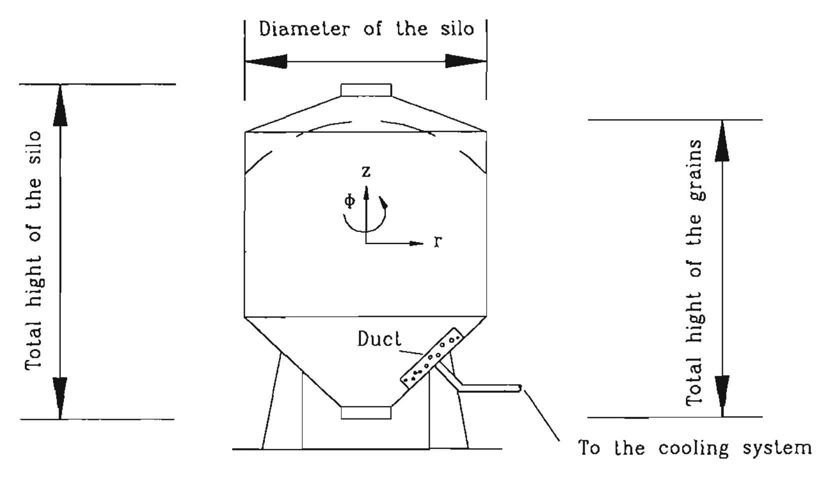


Figure 3.1 The physical configuration of the system

3.1 MOMENTUM EQUATION

The theory to predict air pressure and air velocity distributions through in aerated grain silo is based on Darcy's Law (1856). It is recognosed that variables such as velocity, pressure, temperature and grain moisture content vary on the length scales of the grain kernels and intergranular pores. In this thesis the relative length scales of these microscopic phenomena are very small (typically 10^{-3} times the height of the silo) and volume averaged quantities, *sensu* Thorpe *et al* (1991a, b), are used. In vectorial form of this momentum transfer is given by:

$$-grad\ P = \left(\frac{\mu}{k}\right) V \tag{3.1}$$

where μ is the dynamic viscosity, k is the intrinsic permeability, V is the air velocity vector, P is the air pressure pattern.

3.1.1 Pressures at interior nodes

A good approximation for calculation the air pressure distribution in aerated grain silo is to assume the pressure field satisfies the Laplace equation, which requires that the pressure can be take as constant with respect to time. In cylindrical coordinates (r,ϕ,z), the governing equation may be written as:

$$\frac{\partial^2 P}{\partial r^2}+\frac{1}{r}\frac{\partial P}{\partial r}+\frac{1}{r^2}\frac{\partial^2 P}{\partial \phi^2}+\frac{\partial^2 P}{\partial z^2}=0 \tag{3.2}$$

When the notional time tends to infinity (∞), and the pressure distribution become stable, a false transient term of the pressure flow can be effectively integrated to the steady state. This is expressed as:

$$\frac{1}{\alpha_P}\frac{\partial P}{\partial t}=0 \tag{3.3}$$

where α_P is a false transient factor and assigned that ensures a converged solution.

Combination equations (3.2) and (3.3) together, the pressure distribution in grain silo can be determined by solving the following equation:

$$\frac{1}{\alpha_P}\frac{\partial P}{\partial t}=\frac{\partial^2 P}{\partial r^2}+\frac{1}{r}\frac{\partial P}{\partial r}+\frac{1}{r^2}\frac{\partial^2 P}{\partial \phi^2}+\frac{\partial^2 P}{\partial z^2} \tag{3.4}$$

3.1.2 Pressures at boundary nodes

In finite difference approximation, the boundary conditions of the domain require special treatment. At the upper surface of the grain, the pressure field is zero relative to atmosphere, so we have:

$$P = 0 \tag{3.5}$$

when $$\begin{cases} 0 \le r \le R \\ 0 \le \phi \le 2\pi \\ z = H_{wall} + R\tan\gamma_1 + (R-r)\tan\gamma_2 \end{cases}$$

where H_{wall} is the height of the vertical wall of the silo, γ_1 is the angle of the silo floor to horizontal plane, and γ_2 is the angle between upper surface silo to horizontal plane, R is the radius of the silo.

At the internal vertical wall of the silo the normal velocity component is zero from which it follows that the pressure gradient is given by:

$$\frac{\partial P}{\partial r} = 0 \tag{3.6}$$

when $$\begin{cases} r = R \\ 0 \le \phi \le 2\pi \\ R\tan\gamma_1 \le z \le R\tan\gamma_1 + H_{wall} \end{cases}$$

At the bottomed cone base of the silo, the pressure field is determined by:

$$\frac{\partial P}{\partial z} = 0 \tag{3.7}$$

when $$\begin{cases} r = 1 \\ 0 \le \phi \le 2\pi \\ z = 1 \end{cases}$$

Along the sloping floor of the silo, the pressure gradient is obtained by:

$$\frac{\partial P}{\partial r}\frac{\partial r}{\partial n} + \frac{\partial P}{\partial \phi}\frac{\partial \phi}{\partial n} + \frac{\partial P}{\partial z}\frac{\partial z}{\partial n} = 0 \tag{3.8}$$

when
$$\begin{cases} 0 \le r \le R \\ 0 \le \phi \le 2\pi \\ z = 1 \end{cases}$$

At the aeration duct, the pressure distribution is deemed to consist of a line source of pressure pattern along the floor of the silo. We can write as:

$$P = P_{duct} \tag{3.9}$$

when
$$\begin{cases} r_{duct\,\min} \le r \le r_{duct\,\max} \\ \phi_{duct\,\min} \le \phi \le \phi_{duct\,\max} \\ z = r\tan\gamma_1 \end{cases}$$

where $r_{duct\,\min}$ and $r_{duct\,\max}$ are the length bounds of the duct, in the cylindrical coordinates (r,ϕ,z), they are along the r direction of the silo. $\phi_{duct\,\min}$ and $\phi_{duct\,\max}$ are the width bounds of the duct, they are along the ϕ direction of the silo.

3.1.3 Velocity components

Equation (3.1) describes the relationship between air velocity with air pressure difference, when the pressure field is substituted into the momentum equation, the resulting of velocity field will satisfy the continuity equation, namely the velocity field is equal to the negative gradient of the pressure. By means of equation (3.1), we obtain:

$$V = -\frac{k}{\mu}\cdot\frac{\Delta P}{\Delta S} \tag{3.10}$$

where ΔP is the pressure difference and ΔS is an element of length. So the three components of the velocity field can be expressed in cylindrical coordinates as:

The radial components velocity (V_r) is:

$$V_r = -\frac{1}{R_P}\frac{\partial P}{\partial r} \tag{3.11}$$

The angular components velocity (V_ϕ) is:

$$V_\phi = -\frac{1}{R_P}\frac{1}{r}\frac{\partial P}{\partial \phi} \tag{3.12}$$

The vertical components velocity (V_z) is:

$$V_z = -\frac{1}{R_P}\frac{\partial P}{\partial z} \tag{3.13}$$

where R_P is the coefficient resistance to the airflow given by Hunter (1983) for applying airflow distribution through 28 different seeds. For spherical particles, we can estimate values of R_P from:

$$R_P = \frac{49\mu\varepsilon^2}{\rho^2}S^2 \tag{3.14}$$

where S is the second Ergun (1952) coefficients, ρ and μ are the density and viscosity of air, and ε is the seed bulk porosity.

3.2 THERMAL ENERGY CONSERVATION EQUATION

3.2.1 Temperatures at interior nodes

Thorpe (1994, 1995) has presented a mathematical description of the heat and mass transfer phenomena that occur in ventilated beds of respiring grains. By following this work, the thermal energy conservation equation in cylindrical coordinates is written as:

$$\left[\rho_b\left\{(c_2)_\sigma + (c_1)_\sigma W + \frac{\partial H_W}{\partial T}\right\} + \varepsilon_\gamma \rho_a\left\{c_a + H((c_1)_\gamma + (c_{2v}))\right\}\right]\frac{\partial T}{\partial t}$$

$$-\rho_b h_s \frac{\partial W}{\partial t} - \rho_b \frac{dm}{dt}\int_0^W h_s dW + \rho_a(c_a + H((c_1)_\gamma + c_{2v}))\times$$

$$\left\{V_r\frac{\partial T}{\partial r} + \frac{V_\phi}{r}\frac{\partial T}{\partial \phi} + V_z\frac{\partial T}{\partial z}\right\} = K_{eff}\left\{\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{1}{r^2}\frac{\partial^2 T}{\partial \phi^2} + \frac{\partial^2 T}{\partial z^2}\right\}$$

$$+\rho_b\frac{dm}{dt}(Q_r - 0.6h_{vap}) \tag{3.15}$$

In order to solve equation (3.15) numerically, it is convenient to express the temperature transient ($\frac{\partial T}{\partial t}$) as the formula as below:

$$\frac{\partial T}{\partial t}=\frac{1}{denom}\{\ K_{eff}\left\{\frac{\partial^2 T}{\partial r^2}+\frac{1}{r}\frac{\partial T}{\partial r}+\frac{1}{r^2}\frac{\partial^2 T}{\partial \phi^2}+\frac{\partial^2 T}{\partial z^2}\right\}$$

$$+\rho_b\frac{dm}{dt}(Q_r-0.6h_{vap})+\rho_b h_s\frac{\partial W}{\partial t}+\rho_b\frac{dm}{dt}\int_0^w h_s dW$$

$$-\rho_a\{c_a+H((c_1)_\gamma+c_{2v})\}\times\left\{V_r\frac{\partial T}{\partial r}+\frac{V_\phi}{r}\frac{\partial T}{\partial \phi}+V_z\frac{\partial T}{\partial z}\right\}\} \qquad (3.16)$$

where $$denom=\rho_b\left\{(c_2)_\sigma+(c_1)_\sigma W+\frac{\partial H_W}{\partial T}\right\}+\varepsilon_\gamma\rho_a\{c_a+H((c_1)_\gamma+(c_{2v}))\} \qquad (3.17)$$

in which T is the temperature distribution of the stored grains, W is the grain moisture content field, H is the absolute humidity of the air flow, K_{eff} is the effective thermal dispersivity tensor, H_W is the integral heat of wetting of grains, and other physical properties have been given in the list of nomenclature. The further discussion of these physical properties will be given in Chapter 5.

3.2.2 Temperatures at boundary nodes

On the internal circular wall and floor of the grain silo, the temperature distribution is the function of solar radiation and ambient temperature. The best way for solving these boundary conditions of temperatures is to treat the wall and floor separately, because the intensity of solar radiation is clearly different on each of these components of the silo. At the circular wall of the silo the equation can be written as:

$$T=f_{wall}(R,\phi,z) \qquad (3.18)$$

when $$\begin{cases}0\le\phi\le 2\pi\\ R\tan\gamma_1\le z\le R\tan\gamma_1+H_{wall}\end{cases}$$

On the bottom of the silo floor, the function is:

$$T = f_{floor}(r,\phi,z) \tag{3.19}$$

when $\begin{cases} 0 \le r \le R \\ 0 \le \phi < \phi_{duct\,\min} \\ \phi_{duct\,\max} < \phi \le 2\pi \\ z = r\tan\gamma_1 \end{cases}$

At the aeration duct, the temperature field of the grain is set equal to the inlet air, thus:

$$T = T_{duct}(r,\phi,z) \tag{3.20}$$

when $\begin{cases} r_{duct\,\min} \le r \le r_{duct\,\max} \\ \phi_{duct\,\min} \le \phi \le \phi_{duct\,\max} \\ z = r\tan\gamma_1 \end{cases}$

At the upper surface of the grain, the temperature gradient is considered to have a zero gradient normal to the surface of grain, thus:

$$\frac{\partial T}{\partial r}\frac{\partial r}{\partial n} + \frac{\partial T}{\partial \phi}\frac{\partial \phi}{\partial n} + \frac{\partial T}{\partial z}\frac{\partial z}{\partial n} = 0 \tag{3.21}$$

when $\begin{cases} 0 \le r \le R \\ 0 \le \phi \le 2\pi \\ z = H_{wall} + R\tan\gamma_1 + (R-r)\tan\gamma_2 \end{cases}$

At the upper surface of the centre silo, the temperature field is expressed in equation as:

$$\frac{\partial T}{\partial z} = 0 \tag{3.22}$$

when $\begin{cases} r = 0 \\ 0 \le \phi \le 2\pi \\ z = H_{wall} + R\tan\gamma_1 + (R-r)\tan\gamma_2 \end{cases}$

At the bottom of the cone of the centre silo, the temperature distribution is given in the following equation:

$$\frac{\partial T}{\partial z} = 0 \tag{3.23}$$

when $\begin{cases} r = 0 \\ 0 \le \phi \le 2\pi \\ z = 0 \end{cases}$

3.3 MOISTURE CONSERVATION EQUATION

3.3.1 Moisture contents at interior nodes

Grain moisture is transported within the bulk from regions of high intergranular vapour pressure to regions of low vapour pressure by means of diffusion and convection, and lead the air entering regions of low temperature increase its relative humidity. This movement causes grain to adsorb moisture, and render grainself more susceptible to spoilage and microbial activity. According to Thorpe's (1994, 1995) analyses, the air and grain kernels are considered to be in thermodynamic equilibrium, and it does not need to consider a finite resistance to mass transfer. In this cause, the moisture conservation equation is expressed in cylindrical coordinates as:

$$\frac{\partial W}{\partial t}+\frac{\rho_a}{\rho_b}\left\{V_r\frac{\partial H}{\partial r}+\frac{V_\phi}{r}\frac{\partial H}{\partial \phi}+V_z\frac{\partial H}{\partial z}\right\}$$

$$=D_{eff}\frac{\rho_a}{\rho_b}\left\{\frac{\partial^2 H}{\partial r^2}+\frac{1}{r}\frac{\partial H}{\partial r}+\frac{1}{r^2}\frac{\partial^2 H}{\partial \phi^2}+\frac{\partial^2 H}{\partial z^2}\right\}$$

$$+\frac{dm}{dt}(0.6+W) \tag{3.24}$$

A small re-arrangement of equation (3.24) yields the equation governing the behaviour of moisture content of grains in the intergranular air, and it is conveniently expressed explicitly in term of $(\frac{\partial W}{\partial t})$, thus:

$$\frac{\partial W}{\partial t}=D_{eff}\frac{\rho_a}{\rho_b}\left\{\frac{\partial^2 H}{\partial r^2}+\frac{1}{r}\frac{\partial H}{\partial r}+\frac{1}{r^2}\frac{\partial^2 H}{\partial \phi^2}+\frac{\partial^2 H}{\partial z^2}\right\}$$

$$-\frac{\rho_a}{\rho_b}\left\{V_r\frac{\partial H}{\partial r}+\frac{V_\phi}{r}\frac{\partial H}{\partial \phi}+V_z\frac{\partial H}{\partial z}\right\}$$

$$+\frac{dm}{dt}(0.6+W) \tag{3.25}$$

where $\frac{dm}{dt}$ is the rate of dry matter loss, and this will be detailed in Chapter 5. D_{eff} is the effective diffusion coefficient of the moisture, this may be obtained from experimental relationships with the particle Reynolds number, and it has been calculated from first principle by Thorpe *et al.*, (1991a, b).

In order solving the moisture conservation equation, firstly the distribution of absolute humidities flow is must be calculated. At the interior nodes of the silo, it is obtained by solving equation (5.6) (see chapter 5). At the boundary conditions of the silo, the absolute humidities of the air can be solved by the equation as following.

3.3.2 Absolute humidities at boundary nodes

The floor of the silo is impermeable to moisture vapour, hence the distribution of absolute humidities of the air is expressed as the following equation:

$$\frac{\partial H}{\partial r}\frac{\partial r}{\partial n}+\frac{\partial H}{\partial \phi}\frac{\partial \phi}{\partial n}+\frac{\partial H}{\partial z}\frac{\partial z}{\partial n}=0 \tag{3.26}$$

when $\begin{cases} 0 \le r \le R \\ 0 \le \phi < \phi_{duct\,\min} \\ \phi_{duct\,\max} < \phi \le 2\pi \\ z = r\tan\gamma_1 \end{cases}$

Along the vertical wall of the silo, the absolute humidities of the air field is:

$$\frac{\partial H}{\partial r}=0 \tag{3.27}$$

when $\begin{cases} r = R \\ 0 \le \phi \le 2\pi \\ R\tan\gamma_1 \le z \le R\tan\gamma_1 + H_{wall} \end{cases}$

Along the upper surface of the grain, the flow of absolute humidities of the air is:

$$\frac{\partial H}{\partial z}=0 \tag{3.28}$$

when $$\begin{cases} 0 \le r \le R \\ 0 \le \phi \le 2\pi \\ z = R\tan\gamma_1 + H_{wall} + (R-r)\tan\gamma_2 \end{cases}$$

At the aeration duct, the distribution of absolute humidities of the air is the entering air through aeration duct to the grain silo, so we have:

$$H = H_{duct} \tag{3.29}$$

when $$\begin{cases} r_{duct\,\min} \le r \le r_{duct\,\max} \\ \phi_{duct\,\min} \le \phi \le \phi_{duct\,\max} \\ z = r\tan\gamma_1 \end{cases}$$

At the bottomed cone base of the silo, the absolute humidities of the air is given by:

$$\frac{\partial H}{\partial z} = 0 \tag{3.30}$$

when $$\begin{cases} r = 1 \\ 0 \le \phi \le 2\pi \\ z = 1 \end{cases}$$

At the peak of the grain, the absolute humidities of the air flow can be written as:

$$\frac{\partial H}{\partial z} = 0 \tag{3.31}$$

when $$\begin{cases} r = 1 \\ 0 \le \phi \le 2\pi \\ z = nz \end{cases}$$

3.4. COORDINATE TRANSFORMATION

In this study, the three dimensional physical domain of conical bottomed circular silo has a complicated geometry and boundary conditions. In the numerical solution procedure, it required that enables zero flux boundary conditions to be treated in a straight forward manner.

A useful technique for solving this problem is to employ the finite difference approximations to control the mapping, and using the non-orthogonal grid in the physical three-dimensional domain to transform into a simplistic three-dimensional computational domain which can be used to discretise the governing partial differential equations and the appropriate boundary conditions on an orthogonal grid. The easy way for implementing this technique is to apply the original circular cone in terms of the spatial variables (r,ϕ,z) in cylindrical space to map into a right circular cylinder in terms of the variables (ξ,ϕ,η) in cylindrical coordinates, this is illustrated in figure 3.2 as below.

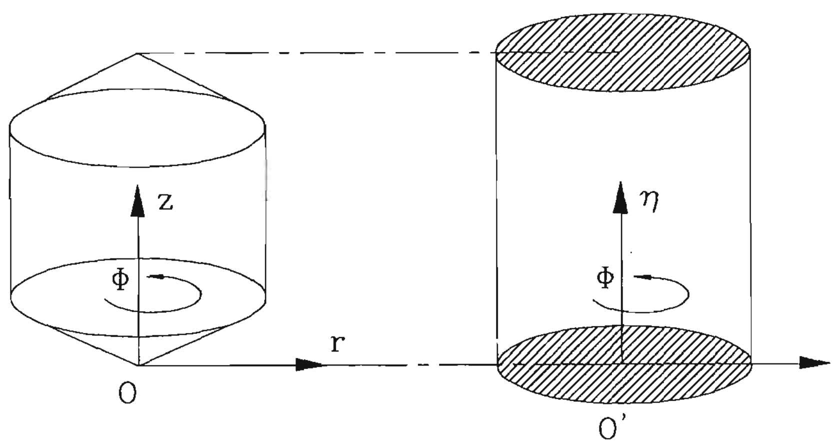


Figure 3.2 The original circular cone transformed to the right circular cylinder

Evidently, as figure 3.2 shows, the origins in both coordinates are $O(0,0,0)$ and $O'(0,0,0)$, the length of radius r in transformed coordinates is dimensionless radius ξ, the height of silo z in transformed coordinates is dimensionless height η, and the angles $\phi(0 \le \phi \le 2\pi)$ of both domains are equal.

In order to transform the original partial differential equations and boundary conditions from the circular conic domain (r,ϕ,z) to the right circular cylinder domain (ξ,ϕ,η), the relations of an arbitrary scalar Γ is found by the chain rule of partial differentiation in the first partial derivative, thus:

$$\frac{\partial \Gamma}{\partial r} = \frac{\partial \Gamma}{\partial \xi}\frac{\partial \xi}{\partial r} + \frac{\partial \Gamma}{\partial \eta}\frac{\partial \eta}{\partial r} + \frac{\partial \Gamma}{\partial \phi}\frac{\partial \phi}{\partial r} \tag{3.32}$$

$$\frac{\partial \Gamma}{\partial \phi} = \frac{\partial \Gamma}{\partial \xi}\frac{\partial \xi}{\partial \phi} + \frac{\partial \Gamma}{\partial \eta}\frac{\partial \eta}{\partial \phi} + \frac{\partial \Gamma}{\partial \phi}\frac{\partial \phi}{\partial \phi} \tag{3.33}$$

$$\frac{\partial \Gamma}{\partial z} = \frac{\partial \Gamma}{\partial \xi}\frac{\partial \xi}{\partial z} + \frac{\partial \Gamma}{\partial \eta}\frac{\partial \eta}{\partial z} + \frac{\partial \Gamma}{\partial \phi}\frac{\partial \phi}{\partial z} \tag{3.34}$$

where

$$\frac{\partial \phi}{\partial r} = \frac{\partial \phi}{\partial z} = \frac{\partial \xi}{\partial \phi} = \frac{\partial \eta}{\partial \phi} = \frac{\partial \phi}{\partial \xi} = 0 \tag{3.35}$$

and

$$\frac{\partial \phi}{\partial \phi} = 1 \tag{3.36}$$

where Γ can represent any function of ξ, ϕ and η.

Rearranging equations (3.32), (3.33) and (3.34), the formula can be written as:

$$\frac{\partial \Gamma}{\partial r} = \frac{\partial \Gamma}{\partial \xi}\frac{\partial \xi}{\partial r} + \frac{\partial \Gamma}{\partial \eta}\frac{\partial \eta}{\partial r} \tag{3.37}$$

$$\frac{\partial \Gamma}{\partial z} = \frac{\partial \Gamma}{\partial \xi}\frac{\partial \xi}{\partial z} + \frac{\partial \Gamma}{\partial \eta}\frac{\partial \eta}{\partial z} \tag{3.38}$$

From the cross-section of the cone, the half of the original circular conic silo is defined by four functions. These geometries are shown in the cylindrical coordinates at figure 3.3.

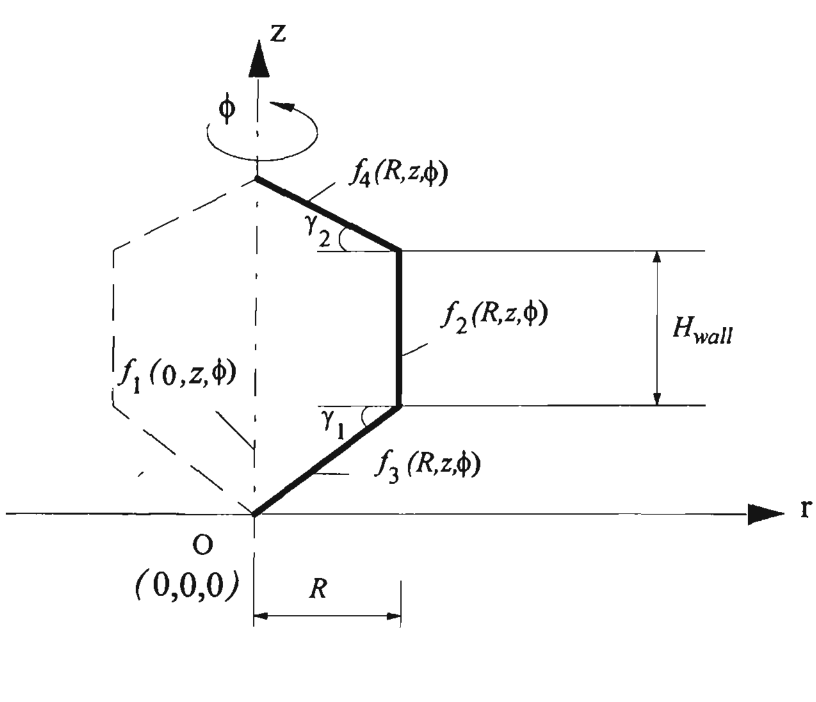


Figure 3.3 The four functions of original circular conic silo

In the above figure, R is the radius of the conical bottomed circular silo, H_{wall} is the wall height of the silo, O is the origin of the coordinate system, ϕ is the angular direction, and in the region of $0 \le \phi \le 2\pi$.

The four functions are expressed as:

$$f_1(0,z,\phi) = 0 \tag{3.39}$$

$$f_2(R,z,\phi) = R \tag{3.40}$$

$$f_3(r,z,\phi) = r\tan\gamma_1 \tag{3.41}$$

$$f_4(r,z,\phi) = H_{wall} + R\tan\gamma_1 + (R-r)\tan\gamma_2 \tag{3.42}$$

We define the relationship between the functions in cylindrical coordinates (r,ϕ,z) and the variables of ξ and η in transformed coordinates (ξ,ϕ,η), after the manner of Moretti and Abbett (1966), thus:

$$\xi = \frac{r - f_1(r,z,\phi)}{f_2(r,z,\phi) - f_1(r,z,\phi)} \tag{3.43}$$

$$\eta = \frac{z - f_3(r,z,\phi)}{f_4(r,z,\phi) - f_3(r,z,\phi)} \tag{3.44}$$

where the value of both ξ and η are assumed to be between 0 and 1. Substituting equations (3.39), (3.40), (3.41) and (3.42) into equations (3.43) and (3.44), results in the variables ξ and η being determined as:

$$\xi = \frac{r}{R} \qquad (0 \le r \le R) \tag{3.45}$$

$$\eta = \frac{z - r\tan\gamma_1}{H_{wall} + R\tan\gamma_1 + (R-r)\tan\gamma_2 - r\tan\gamma_1}$$

$$= \frac{z - r\tan\gamma_1}{Z_{total} - r(\tan\gamma_1 + \tan\gamma_2)} \tag{3.46}$$

where Z_{total} is the total height of the silo.

Therefore the first partial derivatives of the $\frac{\partial\xi}{\partial r}$, $\frac{\partial\eta}{\partial r}$, $\frac{\partial\xi}{\partial z}$ and $\frac{\partial\eta}{\partial z}$ are obtained:

$$\frac{\partial\xi}{\partial r}=\frac{1}{R} \tag{3.47}$$

$$\frac{\partial\eta}{\partial r}=\frac{-\tan\gamma_1(H_{wall}-r(\tan\gamma_1+\tan\gamma_2)-(z-r\tan\gamma_1)(-(\tan\gamma_1+\tan\gamma_2))}{(H_{wall}-r(\tan\gamma_1-\tan\gamma_2))^2}$$

$$=\frac{A^*-C^*\cdot D^*}{(B^*)^2} \tag{3.48}$$

we set

$$A^*=-\tan\gamma_1 \tag{3.49}$$

$$B^*=Z_{total}-r(\tan\gamma_1+\tan\gamma_2) \tag{3.50}$$

$$C^*=z-r\tan\gamma_1 \tag{3.51}$$

$$D^*=-(\tan\gamma_1+\tan\gamma_2) \tag{3.52}$$

and

$$\frac{\partial\xi}{\partial z}=0 \tag{3.53}$$

$$\frac{\partial\eta}{\partial z}=\left(\frac{z-r\tan\gamma_1}{Z_{total}-r(\tan\gamma_1+\tan\gamma_2)}\right)'$$

$$=\frac{1}{Z_{total}-r(\tan\gamma_1+\tan\gamma_2)} \tag{3.54}$$

A notation has been presented by Singh and Thorpe (1993a, b) for the first partial derivative by which we express:

$$\frac{\partial\Gamma}{\partial r}=\alpha_1\frac{\partial\Gamma}{\partial\xi}+\alpha_2\frac{\partial\Gamma}{\partial\eta} \tag{3.55}$$

$$\frac{\partial\Gamma}{\partial z}=\alpha_3\frac{\partial\Gamma}{\partial\xi}+\alpha_4\frac{\partial\Gamma}{\partial\eta} \tag{3.56}$$

we define

$$\alpha_1=\frac{\partial\xi}{\partial r} \tag{3.57}$$

$$\alpha_2=\frac{\partial\eta}{\partial r} \tag{3.58}$$

$$\alpha_3 = \frac{\partial \xi}{\partial z} \tag{3.59}$$

$$\alpha_4 = \frac{\partial \eta}{\partial z} \tag{3.60}$$

Singh and Thorpe (1993a) have presented an expression for the second partial derivatives of the scalar Γ, which we are able to write as:

$$\frac{\partial^2 \Gamma}{\partial r^2} + \frac{\partial^2 \Gamma}{\partial z^2} = \beta_1 \frac{\partial^2 \Gamma}{\partial \xi^2} + \beta_2 \frac{\partial^2 \Gamma}{\partial \eta^2} + \beta_3 \frac{\partial^2 \Gamma}{\partial \xi \partial \eta} + \beta_4 \frac{\partial \Gamma}{\partial \xi} + \beta_5 \frac{\partial \Gamma}{\partial \eta} \tag{3.61}$$

in which

$$\beta_1 = \alpha_1^2 + \alpha_3^2 \tag{3.62}$$

$$\beta_2 = \alpha_2^2 + \alpha_4^2 \tag{3.63}$$

$$\beta_3 = 2(\alpha_1 \alpha_2 + \alpha_3 \alpha_4) \tag{3.64}$$

$$\beta_4 = 2(\frac{\partial^2 \xi}{\partial r^2} + \frac{\partial^2 \xi}{\partial z^2}) \tag{3.65}$$

$$\beta_5 = 2(\frac{\partial^2 \eta}{\partial r^2} + \frac{\partial^2 \eta}{\partial z^2}) \tag{3.66}$$

From equations (3.47), (3.48), (3.53) and (3.54), we can determine the second partial derivatives $\frac{\partial^2 \xi}{\partial r^2}, \frac{\partial^2 \eta}{\partial r^2}, \frac{\partial^2 \xi}{\partial z^2}$ and $\frac{\partial^2 \eta}{\partial z^2}$, thus:

$$\frac{\partial^2 \xi}{\partial r^2} = (\frac{1}{R})' = 0 \tag{3.67}$$

$$\frac{\partial^2 \eta}{\partial r^2} = (\frac{B^* \cdot A^* - C^* \cdot D^*}{(B^*)^2})'$$

$$= 2\frac{(-B^* \cdot A^* + C^* \cdot D^*)D^*}{(B^*)^3} \tag{3.68}$$

$$\frac{\partial^2 \xi}{\partial z^2} = 0 \tag{3.69}$$

$$\frac{\partial^2 \eta}{\partial z^2} = (\frac{1}{Z_{total} - r(\tan\gamma_1 + \tan\gamma_2)})' = 0 \tag{3.70}$$

3.5 MOMENTUM EQUATION IN TRANSFORMED COORDINATES

3.5.1 Pressures at interior nodes in transformed coordinates

In transformed coordinates, equation (3.4) that governs the pressure field in the grain silo, it may be expressed as:

$$\frac{1}{\alpha_p}\frac{\partial P}{\partial t}=\beta_1\frac{\partial^2 P}{\partial \xi^2}+\beta_2\frac{\partial^2 P}{\partial \eta^2}+\beta_3\frac{\partial^2 P}{\partial \xi \partial \eta}+\beta_6\frac{\partial P}{\partial \xi}+\beta_7\frac{\partial P}{\partial \eta}+\frac{1}{R^2\xi^2}\frac{\partial^2 P}{\partial \phi^2} \tag{3.71}$$

where we define

$$\beta_6=\beta_4+\frac{\alpha_1}{R\xi} \tag{3.72}$$

$$\beta_7=\beta_5+\frac{\alpha_2}{R\xi} \tag{3.73}$$

and $$r=R\xi \tag{3.74}$$

3.5.2 Pressures at boundary nodes in transformed coordinates

Equation (3.6) that governs pressure boundary conditions at the vertical wall of the grain silo, it can be expressed in transformed coordinates as:

$$\alpha_1\frac{\partial P}{\partial \xi}+\alpha_2\frac{\partial P}{\partial \eta}=0 \tag{3.75}$$

when $$\begin{cases}\xi=1\\ 0\le\phi\le 2\pi\\ 0\le\eta\le 1\end{cases}$$

In transformed coordinates, the conical floor is transformed into horizontal floor, and the normal distance is independent of ϕ. So equation (3.8) can be written in cylindrical coordinates as:

$$\frac{\partial P}{\partial r}\frac{\partial r}{\partial n}+\frac{\partial P}{\partial z}\frac{\partial z}{\partial n}=0 \tag{3.76}$$

we observe that

$$\frac{\partial r}{\partial n}=-\sin\gamma_1 \tag{3.77}$$

$$\frac{\partial z}{\partial n}=\cos\gamma_1 \tag{3.78}$$

Rearranging equation (3.76), enables it to be written in the transformed coordinates as following:

$$(\alpha_1\frac{\partial P}{\partial \xi}+\alpha_2\frac{\partial P}{\partial \eta})(-\sin\gamma_1)+(\alpha_3\frac{\partial P}{\partial \xi}+\alpha_4\frac{\partial P}{\partial \eta})\cos\gamma_1=0 \tag{3.79}$$

when $\begin{cases} 0\le\xi\le1 \\ 0\le\phi\le2\pi \\ \eta=0 \end{cases}$

Equation (3.5) governs the pressure field at the upper surface of the grain. In transformed coordinates it is:

$$P=0 \tag{3.80}$$

when $\begin{cases} 0\le\xi\le1 \\ 0\le\phi\le2\pi \\ \eta=1 \end{cases}$

Equation (3.7) is expressed the pressure pattern at the bottomed conical base of the silo, in transformed coordinates is:

$$\frac{\partial P}{\partial \eta}=0 \tag{3.81}$$

when $\begin{cases} \xi=0 \\ 0\le\phi\le2\pi \\ \eta=0 \end{cases}$

Equation (3.9) shows the pressure flow at aeration duct, in transformed coordinates is:

$$P=P_{duct} \tag{3.82}$$

$$\text{when} \quad \begin{cases} \xi_{duct\,\min} \le \xi \le \xi_{duct\,\max} \\ \phi_{duct\,\min} \le \phi \le \phi_{duct\,\max} \\ \eta = 0 \end{cases}$$

Where $\xi_{duct\,\min}, \xi_{duct\,\max}$ are the length bounds of the duct in the transformed coordinates of ξ direction of the silo.

3.5.3 Velocity components in transformed coordinates

The three components of the velocity field in transformed coordinates is obtained by modifying equations (3.11), (3.12) and (3.13), which govern the velocity field in cylindrical coordinates. The expressions are given as following:

The components of the radial velocity is presented in transformed coordinates as:

$$V_r = -\frac{1}{R_p}(\alpha_1 \frac{\partial P}{\partial \xi} + \alpha_2 \frac{\partial P}{\partial \eta}) \tag{3.83}$$

The components of the angular velocity is obtained in transformed coordinates as:

$$V_\phi = -\frac{1}{R_p}\frac{1}{R\xi}\frac{\partial P}{\partial \phi} \tag{3.84}$$

The components of the vertical velocity is written in transformed coordinates as:

$$V_z = -\frac{1}{R_p}(\alpha_3 \frac{\partial P}{\partial \xi} + \alpha_4 \frac{\partial P}{\partial \eta}) \tag{3.85}$$

3.6 THERMAL ENERGY CONSERVATION EQUATION IN TRANSFORMED COORDINATES

3.6.1 Temperatures at interior nodes in transformed coordinates

Equation (3.16) is governing the temperature field which can be written in the transformed coordinates system as:

$$\frac{\partial T}{\partial t}=\frac{1}{denom}\{\ K_{eff}\left\{\beta_1\frac{\partial^2 T}{\partial \xi^2}+\beta_2\frac{\partial^2 T}{\partial \eta^2}+\beta_3\frac{\partial^2 T}{\partial \xi \partial \eta}+\beta_6\frac{\partial T}{\partial \xi}+\beta_7\frac{\partial T}{\partial \eta}+\beta_8\frac{\partial^2 T}{\partial \phi^2}\right\}$$

$$+\rho_b\frac{dm}{dt}(Q_r-0.6h_{vap})+\rho_b h_s\frac{\partial W}{\partial t}+\rho_b\frac{dm}{dt}\int_0^W h_s dH-\rho_a\{c_a+H((c_1)_\gamma+c_{2v})\}$$

$$\times\left\{V_r(\alpha_1\frac{\partial T}{\partial \xi}+\alpha_2\frac{\partial T}{\partial \eta})+\frac{V_\phi}{R\xi}\frac{\partial T}{\partial \phi}+V_z(\alpha_3\frac{\partial T}{\partial \xi}+\alpha_4\frac{\partial T}{\partial \eta})\right\}\} \qquad (3.86)$$

where $denom=\rho_b\left\{(c_2)_\sigma+(c_1)_\sigma W+\frac{\partial H_W}{\partial T}\right\}+\varepsilon_\gamma\rho_a\{c_a+H((c_1)_\gamma+(c_{2v}))\}$

3.6.2 Temperatures at boundary nodes in transformed coordinates

Equation (3.18) governs the temperature field in the circular wall of the silo. In the transformed coordinates, it may expressed as:

$$T=f_{wall}(1,\phi,\eta) \qquad (3.87)$$

when $\begin{cases}0\le\phi\le 2\pi\\ 0\le\eta\le 1\end{cases}$

At the floor of the silo, equation (3.19) can be given by transformed coordinates as:

$$T=f_{floor}(\xi,\phi,0) \qquad (3.88)$$

when $\begin{cases}0\le\xi\le 1\\ 0<\phi\le\phi_{duct\min}\\ \phi_{duct\min}<\phi\le 2\pi\end{cases}$

At the aeration duct, equation (3.20) becomes:

$$T=T_{duct} \qquad (3.89)$$

when $\begin{cases}\xi_{duct\min}\le\xi\le\xi_{duct\max}\\ \phi_{duct\min}\le\phi\le\phi_{duct\max}\\ \eta=0\end{cases}$

Equation (3.21) is zero flux boundary condition. It is written in transformed coordinates as (the value of ϕ is constant):

$$(\alpha_1 \frac{\partial T}{\partial \xi} + \alpha_2 \frac{\partial T}{\partial \eta})\frac{\partial r}{\partial n} + (\alpha_3 \frac{\partial T}{\partial \xi} + \alpha_4 \frac{\partial T}{\partial \eta})\frac{\partial z}{\partial n} = 0 \tag{3.90}$$

when $\begin{cases} 0 \le \xi \le 1 \\ 0 \le \phi \le 2\pi \\ \eta = 1 \end{cases}$

Substitution of equations (3.77) and (3.78) into equation (3.90), yields:

$$(\alpha_1 \frac{\partial T}{\partial \xi} + \alpha_2 \frac{\partial T}{\partial \eta})(-\sin\gamma_2) + (\alpha_3 \frac{\partial T}{\partial \xi} + \alpha_4 \frac{\partial T}{\partial \eta})\cos\gamma_2 = 0 \tag{3.91}$$

Equation (3.22) presents the temperatures at the upper surface of the centre silo, it is written in transformed coordinates as:

$$\frac{\partial T}{\partial \eta} = 0 \tag{3.92}$$

when $\begin{cases} \xi = 0 \\ 0 \le \phi \le 2\pi \\ \eta = 1 \end{cases}$

Equation (3.23) is expressed the temperature field at the bottom of the cone base of the silo, it can be written in transformed coordinates as:

$$\frac{\partial T}{\partial \eta} = 0 \tag{3.93}$$

when $\begin{cases} \xi = 0 \\ 0 \le \phi \le 2\pi \\ \eta = 0 \end{cases}$

3.7 MOISTURE CONSERVATION EQUATION IN TRANSFORMED COORDINATES

3.7.1 Moisture contents at interior nodes in transformed coordinates

Equation (3.24) governs the moisture content field in the silo. In transformed coordinates it can be expressed as:

$$\frac{\partial W}{\partial t} = D_{eff}\frac{\rho_a}{\rho_b}\left\{\beta_1\frac{\partial^2 H}{\partial \xi^2}+\beta_2\frac{\partial^2 H}{\partial \eta^2}+\beta_3\frac{\partial^2 H}{\partial \xi \partial \eta}+\beta_6\frac{\partial H}{\partial \xi}+\beta_7\frac{\partial H}{\partial \eta}+\beta_8\frac{\partial^2 H}{\partial \phi^2}\right\}$$

$$-\frac{\rho_a}{\rho_b}\left\{V_r\left(\alpha_1\frac{\partial H}{\partial \xi}+\alpha_2\frac{\partial H}{\partial \eta}\right)+\frac{V_\phi}{R\xi}\frac{\partial H}{\partial \phi}+V_z\left(\alpha_3\frac{\partial H}{\partial \xi}+\alpha_4\frac{\partial H}{\partial \eta}\right)\right\}$$

$$+\frac{dm}{dt}(0.6+W) \tag{3.94}$$

3.7.2 Absolute humidities at boundary nodes in transformed coordinates

Along the floor of the silo, equation (3.26) expresses the absolute humidities of the air in the grain silo. Because in the transformed horizontal floor along ϕ direction is constant, so in transformed coordinates we will have:

$$\frac{\partial H}{\partial \xi}\frac{\partial \xi}{\partial n}+\frac{\partial H}{\partial \eta}\frac{\partial \eta}{\partial n}=0 \tag{3.95}$$

when $\begin{cases} 0 \le \xi \le 1 \\ 0 \le \phi < \phi_{duct\,min} \\ \phi_{duct\,max} < \phi < 2\pi \\ \eta = 0 \end{cases}$

Substituting the transformed coordinates equations (3.55), (3.56), (3.77) and (3.78) into equation (3.95), results in:

$$(\alpha_1\frac{\partial H}{\partial \xi}+\alpha_2\frac{\partial H}{\partial \eta})(-\sin\gamma_1)+(\alpha_3\frac{\partial H}{\partial \xi}+\alpha_4\frac{\partial H}{\partial \eta})\cos\gamma_1=0 \tag{3.96}$$

Equation (3.27) governs the absolute humidities of the air in the silo along the wall. Substituting the transformed coordinates equation (3.55) into equation (3.27), we get:

$$\frac{\partial H}{\partial r} = \alpha_1 \frac{\partial H}{\partial \xi} + \alpha_2 \frac{\partial H}{\partial \eta} = 0 \tag{3.97}$$

when
$$\begin{cases} \xi = 1 \\ 0 \le \phi \le 2\pi \\ R\tan\gamma_1 \le \eta \le H_{wall} + R\tan\gamma_1 \end{cases}$$

Equation (3.28) governs the absolute humidities of the air along the upper surface of the grain. In the transformed coordinates it is:

$$\frac{\partial H}{\partial \eta} = 0 \tag{3.98}$$

when
$$\begin{cases} 0 \le \xi \le 1 \\ 0 \le \phi \le 2\pi \\ \eta = 1 \end{cases}$$

Equation (3.29) expresses the absolute humidities of the air at the aeration duct, it can be written in transformed coordinates as:

$$H = H_{duct} \tag{3.99}$$

when
$$\begin{cases} \xi_{duct\min} \le \xi \le \xi_{duct\max} \\ \phi_{duct\min} \le \phi \le \phi_{duct\max} \\ \eta = 0 \end{cases}$$

Equation (3.30) shows the formula of the absolute humidities of the air at the bottomed cone base of the silo, it has been given in the transformed coordinates as:

$$\frac{\partial H}{\partial \eta} = 0 \tag{3.100}$$

when
$$\begin{cases} \xi = 0 \\ 0 \le \phi \le 2\pi \\ \eta = 0 \end{cases}$$

Equation (3.31) governs the absolute humidities of the air at the peak of the grain, in the transformed coordinates it can be written as:

$$\frac{\partial H}{\partial \eta} = 0 \tag{3.101}$$

when $\begin{cases} \xi = 0 \\ 0 \le \phi \le 2\pi \\ \eta = 1 \end{cases}$

CHAPTER 4

NUMERICAL METHOD

The goals of this chapter are to complete the description of the numerical solution of finite difference method for solving the partial differential equations in transformed coordinates, which will include: Alternating Direction Implicit method, explicit method for heat and mass transfer equations and Thomas algorithm.

4.1 FINITE DIFFERENCE SCHEME

Numerical approximation methods that employ the finite difference methods have more advantageous than the others for solving partial differential equations. This is because, the finite difference equations are obtained by using the local expansions for the variables, the derivatives at a point are approximated by difference quotients over a small grid mesh. The solution produce is relatively simple, straightforward and flexible. When the results of solutions can be satisfactorily calculated from the analytical formulas, the accuracy of the approximate solution can usually also be improved by increasing the number of mesh points and iteration time step in the approximations.

In this thesis, the finite difference method will be used for deriving the approximate computational solutions in an aerated conical bottomed circular silo.

4.2 APPROXIMATION GRID MESH

The basic objective of the finite difference method is to represent the time-space continuum by a set of discretely spaced points. A separate algebraic approximation of the partial differential equation is derived for every one of these points. The solution of the partial differential equation is found by solving these algebraic equations. Therefore, the network of grid mesh throughout the region of interest is first required to be established. A meshed section of the three dimensional cylindrical domain is shown in figure 4.1. In the ξ direction, the row spacings are non-uniform, and the mesh spacing $h\xi(i)$ is the distance between $(i-1)th$ point to ith point along the ξ axis. In the η direction, the columns are uniform, the mesh spacing $h\eta(k)$ is the distance between $(k-1)th$ point to kth point and respectively in η axis. In the ϕ direction, the mesh spacing $h\phi(j)$ is equal divided.

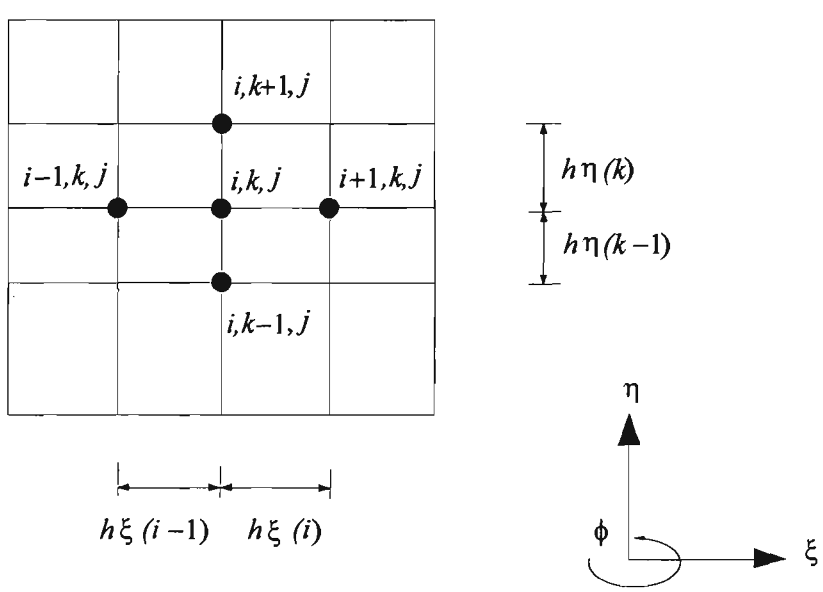


Figure 4.1 Three points grid mesh in three dimensional space.

In order to get the exact solution of the finite difference equation which is near to the true solution of the governing partial differential equation, it is important to set up the grid mesh spacing for increasing the accuracy of the solution. The respective mesh spacings $h\xi(i)$, $h\eta(k)$ and $h\phi(j)$ are defined by the following equations:

When $1 \le i \le 0.7n\xi$

$$h\xi(i) = \frac{1 - 0.15/R}{n\xi - 7} \tag{4.1}$$

When $i = 0.75n\xi$

$$h\xi(i) = \frac{0.05}{R} \tag{4.2}$$

When $0.8n\xi \le i \le n\xi$

$$h\xi(i) = \frac{0.02}{R} \tag{4.3}$$

and

$$h\eta(k) = \frac{1}{n\eta - 1} \tag{4.4}$$

$$h\phi(j) = \frac{2\pi}{n\phi} \tag{4.5}$$

where $n\xi$, $m\eta$ and $n\phi$ are the maximum node points in the three components directions (ξ, ϕ, η). R is the radius of the grain silo as indicated in Chapter 3.

In the above, we obtain the fine mesh close to the wall of the silo where there are significant diurnal variations, and the wide mesh in the centre.

4.3 FINITE DIFFERENCE OPERATORS: INTERIOR NODES

The three point finite difference operators $c\xi1(i)$, $c\xi2(i)$, $c\xi3(i)$, $c\xi4(i)$, $c\xi5(i)$, $c\xi6(i)$, $c\eta1(k)$, $c\eta2(k)$,$c\eta3(k)$, $c\eta4(k)$, $c\eta5(k)$, $c\eta6(k)$, $c\phi4(j)$, $c\phi5(j)$ and $c\phi6(j)$ can be achieved by applying Taylor's series expansion for the arbitrary scalar forward to $\Gamma_{i+1,j,k}$ and backward to $\Gamma_{i-1,j,k}$ about the central value $\Gamma_{i,j,k}$, to describe the first and second partial derivatives. These are presented a couple information at the neighbouring points of a grid as following:

Taylor's expansion about the $(i,j,k)th$ node in the direction of ξ is:

$$\Gamma_{i+1,j,k} = \Gamma_{i,j,k} + h\xi(i)\frac{\partial\Gamma}{\partial\xi} + \frac{h\xi(i)^2}{2!}\frac{\partial^2\Gamma}{\partial\xi^2} + O(h\xi(i)^2) \tag{4.6}$$

$$\Gamma_{i-1,j,k} = \Gamma_{i,j,k} - h\xi(i-1)\frac{\partial\Gamma}{\partial\xi} + \frac{h\xi(i-1)^2}{2!}\frac{\partial^2\Gamma}{\partial\xi^2} + O(h\xi(i-1)^2) \tag{4.7}$$

Omitting terms of order $O(h\xi(i)^2)$, $O(h\xi(i-1)^2)$ and higher, eliminating the second-order derivative $\frac{h\xi(i)^2}{2!}\frac{\partial^2\Gamma}{\partial\xi^2}$ and $\frac{h\xi(i-1)^2}{2!}\frac{\partial^2\Gamma}{\partial\xi^2}$ by means of equation (4.7) multiplies term $h\xi(i)^2$, and equation (4.6) multiplies term $h\xi(i-1)^2$, then subtracting equation (4.6) from equation (4.7). Therefore the first partial derivative is obtained as:

$$\frac{\partial\Gamma}{\partial\xi} = \frac{-h\xi(i)}{h\xi(i-1)(h\xi(i-1)+h\xi(i))}\Gamma_{i-1,j,k} + \frac{h\xi(i)-h\xi(i-1)}{h\xi(i-1)h\xi(i)}\Gamma_{i,j,k} + \frac{h\xi(i-1)}{h\xi(i)(h\xi(i-1)+h\xi(i))}\Gamma_{i+1,j,k} \tag{4.8}$$

we define

$$c\xi 1(i) = \frac{-h\xi(i)}{h\xi(i-1)(h\xi(i-1)+h\xi(i))} \tag{4.9}$$

$$c\xi 2(i) = \frac{h\xi(i)-h\xi(i-1)}{h\xi(i-1)h\xi(i)} \tag{4.10}$$

$$c\xi 3(i) = \frac{h\xi(i-1)}{h\xi(i)(h\xi(i-1)+h\xi(i))} \tag{4.11}$$

So the simplified equation (4.8) becomes:

$$\frac{\partial\Gamma}{\partial\xi} = c\xi 1(i)\Gamma_{i-1,j,k} + c\xi 2(i)\Gamma_{i,j,k} + c\xi 3(i)\Gamma_{i+1,j,k} \tag{4.12}$$

In an analogous manner, the second partial derivative is:

$$\frac{\partial^2\Gamma}{\partial\xi^2} = c\xi 4(i)\Gamma_{i-1,j,k} + c\xi 5(i)\Gamma_{i,j,k} + c\xi 6(i)\Gamma_{i+1,j,k} \tag{4.13}$$

where

$$c\xi 4(i) = \frac{2}{h\xi(i-1)(h\xi(i)+h\xi(i-1))} \tag{4.14}$$

$$c\xi5(i) = \frac{-2}{h\xi(i-1)h\xi(i)} \tag{4.15}$$

$$c\xi6(i) = \frac{2}{h\xi(i)(h\xi(i)+h\xi(i-1))} \tag{4.16}$$

Similarly, in the η direction we have:

$$\frac{\partial \Gamma}{\partial \eta} = c\eta1(k)\Gamma_{i,j,k-1} + c\eta2(k)\Gamma_{i,j,k} + c\eta3(k)\Gamma_{i,j,k+1} \tag{4.17}$$

where

$$c\eta1(k) = \frac{-h\eta(k)}{h\eta(k-1)(h\eta(k-1)+h\eta(k))} \tag{4.18}$$

$$c\eta2(k) = \frac{h\eta(k)-h\eta(k-1)}{h\eta(k-1)h\eta(k)} \tag{4.19}$$

$$c\eta3(k) = \frac{h\eta(k-1)}{h\eta(k)(h\eta(k-1)+h\eta(k))} \tag{4.20}$$

and

$$\frac{\partial^2 \Gamma}{\partial \eta^2} = c\eta4(k)\Gamma_{i,j,k-1} + c\eta5(k)\Gamma_{i,j,k} + c\eta6(k)\Gamma_{i,j,k+1} \tag{4.21}$$

where

$$c\eta4(k) = \frac{2}{h\eta(k-1)(h\eta(k)+h\eta(k-1))} \tag{4.22}$$

$$c\eta5(k) = \frac{-2}{h\eta(k-1)h\eta(k)} \tag{4.23}$$

$$c\eta6(k) = \frac{2}{h\eta(k)(h\eta(k)+h\eta(i-1))} \tag{4.24}$$

Again in the ϕ direction, we define the first partial derivative :

$$\frac{\partial \Gamma}{\partial \phi} = c\phi1(j)\Gamma_{i,j,k-1} + c\phi2(j)\Gamma_{i,j,k} + c\phi3(j)\Gamma_{i,j,k+1} \tag{4.25}$$

where

$$c\phi1(j) = -\frac{1}{2\phi} \tag{4.26}$$

$$c\phi2(j) = 0 \tag{4.27}$$

$$c\phi 3(j) = \frac{1}{2\phi} \tag{4.28}$$

and the second partial derivative is:

$$\frac{\partial^2 \Gamma}{\partial \phi^2} = c\phi 4(j)\Gamma_{i,j-1,k} + c\phi 5(j)\Gamma_{i,j,k} + c\phi 6(j)\Gamma_{i,j+1,k} \tag{4.29}$$

where

$$c\phi 4(j) = \frac{1}{\Delta\phi^2} \tag{4.30}$$

$$c\phi 5(j) = \frac{-2}{\Delta\phi^2} \tag{4.31}$$

$$c\phi 6(j) = \frac{1}{\Delta\phi^2} \tag{4.32}$$

Moreover, the finite difference approximation of the mixed derivative $\frac{\partial^2 \Gamma}{\partial \xi \partial \eta}$ is also derived by applying Taylor's expansion. We have obtained the differential expression as following:

$$\frac{\partial^2 \Gamma}{\partial \xi \partial \eta} = c\eta 1(k)(c\xi 1(i)\Gamma_{i-1,j,k-1} + c\xi 2(i)\Gamma_{i,j,k-1} + c\xi 3(i)\Gamma_{i+1,j,k-1}) +$$

$$c\eta 2(k)(c\xi 1(i)\Gamma_{i-1,j,k} + c\xi 2(i)\Gamma_{i,j,k} + c\xi 3(i)\Gamma_{i+1,j,k}) + c\eta 3(k)$$

$$(c\xi 1(i)\Gamma_{i-1,j,k+1} + c\xi 2(i)\Gamma_{i,j,k+1} + c\xi 3\Gamma_{i+1,j,k+1}) \tag{4.33}$$

4.4 FINITE DIFFERENCE OPERATORS: BOUNDARY NODES

In the boundary conditions, the three point finite difference operators are also derived by applying Taylor's expansion. Figure 4.2 shows the boundary nodes at the ξ direction of the silo boundaries, such as:

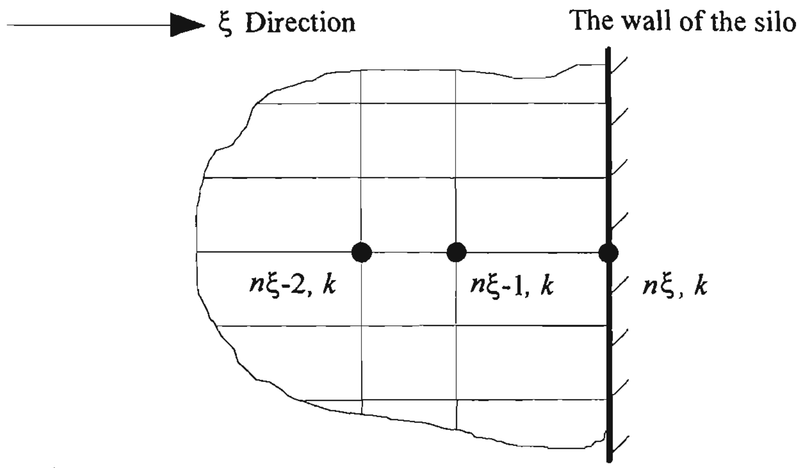


Figure 4.2 The boundary nodes in the ξ direction of the silo.

we have:

$$\frac{\partial \Gamma}{\partial \xi} = c\xi b4\Gamma_{n\xi,j,k} + c\xi b5\Gamma_{n\xi-1,j,k} + c\xi b6\Gamma_{n\xi-2,j,k} \tag{4.34}$$

where

$$c\xi b4 = \frac{2h\xi(n\xi-1)+h\xi(n\xi-2)}{h\xi(n\xi-1)(h\xi(n\xi-1)+h\xi(n\xi-2))} \tag{4.35}$$

$$c\xi b5 = \frac{-(h\xi(n\xi-1)+h\xi(n\xi-2))}{h\xi(n\xi-1)h\xi(n\xi-2)} \tag{4.36}$$

$$c\xi b6 = \frac{h\xi(n\xi-1)}{h\xi(n\xi-2)(h\xi(n\xi-1)+h\xi(n\xi-2))} \tag{4.37}$$

In the η direction at the bottom of the cylinder, the boundary condition is depicted in figure 4.3 as:

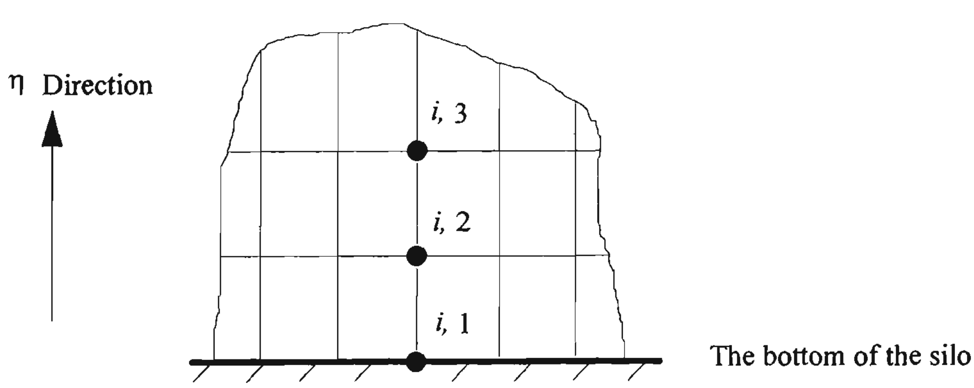


Figure 4.3 The boundary nodes in the η direction of the bottom of the silo

we obtain:

$$\frac{\partial\Gamma}{\partial\eta} = c\eta b1\Gamma_{i,j,1} + c\eta b2\Gamma_{i,j,2} + c\eta b3\Gamma_{i,j,3} \tag{4.38}$$

where

$$c\eta b1 = \frac{-2h\eta(1) + h\eta(2)}{h\eta(1)(h\eta(1) + h\eta(2))} \tag{4.39}$$

$$c\eta b2 = \frac{h\eta(1) + h\eta(2)}{h\eta(1)h\eta(2)} \tag{4.40}$$

$$c\eta b3 = \frac{-h\eta(1)}{h\eta(2)(h\eta(1) + h\eta(2))} \tag{4.41}$$

In the η direction at the upper surface of the grains, the boundary condition as shown in figure 4.4:

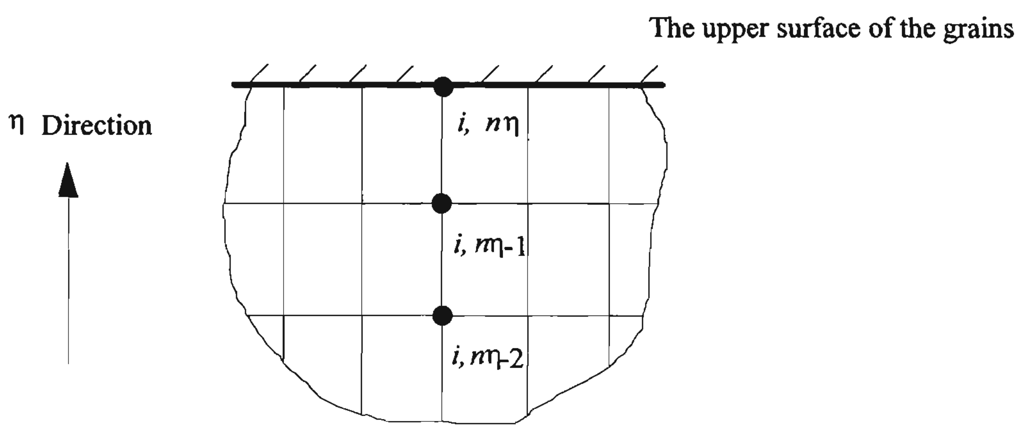


Figure 4.4 The boundary nodes in the η direction of the upper surface of the grains

we get:

$$\frac{\partial\Gamma}{\partial\eta} = c\eta b4\Gamma_{i,j,n\eta} + c\eta b5\Gamma_{i,j,n\eta-1} + c\eta b6\Gamma_{i,j,n\eta-2} \tag{4.42}$$

where

$$c\eta b4 = \frac{2h\eta(n\eta-1) + h\eta(n\eta-2)}{h\eta(n\eta-1)(h\eta(n\eta-1) + h\eta(n\eta-2))} \tag{4.43}$$

$$c\eta b5 = \frac{-(h\eta(n\eta-1) + h\eta(n\eta-2))}{h\eta(n\eta-1)h\eta(n\eta-2)} \tag{4.44}$$

$$c\eta b6 = \frac{h\eta(n\eta - 1)}{h\eta(n\eta - 2)(h\eta(n\eta - 1) + h\eta(n\eta - 2))} \qquad (4.45)$$

4.5 ANDREYEV - SAMARSKII ALGORITHM

In the numerical solution procedure, the finite difference equation governing the pressure distribution is discretised by replacing spatial derivatives and applying the three point finite difference scheme in a non-uniform grid mesh. In order to better derive the appropriate difference equation, to eliminate some of the difficulties and to associate with solving the equation directly, a simplified method - the Alternating Direction Implicit method is developed as proposed by Peaceman and Rachford (1955) and modified by Samarskii and Andreyev (1963). In this study, the governing equation of pressure distribution in transformed coordinates will be solved by this method. In the Andreyev-Samarskii algorithm, a new variable ω is defined, thus:

$$\omega = \frac{1}{\alpha_P}\frac{\partial P}{\partial t} = \frac{P^{n+1} - P^n}{\alpha_P \Delta t} \qquad (4.46)$$

where P^{n+1} is the pressure value at the $(i, j, k)th$ node after $(n+1)th$ iteration. During the iteration time step Δt is within the integration interval, and the intermediate value of the pressure P is given by:

$$P = \alpha_P \cdot \delta \cdot \omega \cdot \Delta t + P^n \qquad (4.47)$$

where δ is a weighting factor which value is between 0 and 1. If $\delta = 0$, the scheme is fully explicit. If $\delta = 1$, the scheme is fully implicit. If $\delta = 0.5$, the scheme yields Crank-Nicholson method (1947), it is splitting the solution of equation (4.47) into a series of locally one dimensional problems.

4.5.1 Solution of the pressures at interior nodes

In order to solve Laplace equation, as equation (3.71) in transformed coordinates by using Andreyev-Samarskii (1963) method, the solution process is obtained by substitution equation (4.47) into the right hand side of equation (3.71), and equation (4.46) is replaced at the left hand side of equation (3.71). The new rearrangement of the expression is written as follows:

$$\omega-\alpha_P\delta\Delta t(\beta_1\frac{\partial^2\omega}{\partial\xi^2}+\beta_2\frac{\partial^2\omega}{\partial\eta}+\beta_3\frac{\partial^2\omega}{\partial\xi\partial\eta}+\beta_6\frac{\partial\omega}{\partial\xi}+\beta_7\frac{\partial\omega}{\partial\eta}+\frac{1}{R^2\xi^2}\frac{\partial^2\omega}{\partial\phi^2})$$

$$=\beta_1\frac{\partial^2 P}{\partial\xi^2}+\beta_2\frac{\partial^2 P}{\partial\eta^2}+\beta_3\frac{\partial^2 P}{\partial\xi\partial\eta}+\beta_6\frac{\partial P}{\partial\xi}+\beta_7\frac{\partial P}{\partial\eta}+\frac{1}{R^2\xi^2}\frac{\partial^2 P}{\partial\phi^2} \tag{4.48}$$

In equation (4.48), the right hand side of the pressures have been omitted, and the mixed derivative is treated explicitly.

Firstly, the ω^* is solved by applying implicit equation (4.48) in the ξ direction, thus:

$$\omega^*-\alpha_P\cdot\delta\cdot\Delta t(\beta_1\frac{\partial^2\omega^*}{\partial\xi^2}+\beta_6\frac{\partial\omega^*}{\partial\xi})=RHS1 \tag{4.49}$$

where

$$RHS1=\beta_1\frac{\partial^2 P}{\partial\xi^2}+\beta_2\frac{\partial^2 P}{\partial\eta^2}+\beta_3\frac{\partial^2 P}{\partial\xi\partial\eta}+\beta_6\frac{\partial P}{\partial\xi}+\beta_7\frac{\partial P}{\partial\eta}+\beta_8\frac{\partial^2 P}{\partial\phi^2} \tag{4.50}$$

and

$$\beta_8=\frac{1}{R^2\xi^2} \tag{4.51}$$

The first and second order partial derivatives are represented by the three point backward and forward difference scheme with the finite difference operators, so substituting equations (4.12), (4.13), (4.17), (4.21) and (4.29) into equation (4.49). The discretization of the linear algebraic equation is became as $(1<i<n\xi)$:

$$\omega^*_{i,j,k}-\alpha_P\cdot\delta\cdot\Delta t(\beta_1(c\xi4(i)\omega^*_{i-1,j,k}+c\xi5(i)\omega^*_{i,j,k}+c\xi6(i)\omega^*_{i+1,j,k})+$$

$$(\beta_6(c\xi1(i)\omega^*_{i-1,j,k}+c\xi2(i)\omega^*_{i,j,k}+c\xi3(i)\omega^*_{i+1,j,k})=RHS2 \tag{4.52}$$

in which

$$RHS2=\beta_1(c\xi4(i)P_{i-1,j,k}+c\xi5(i)P_{i,j,k}+c\xi6(i)P_{i+1,j,k})+\beta_2(c\eta4(k)P_{i,j,k-1}+$$

$$c\eta 5(k)P_{i,j,k}+c\eta 6(k)P_{i,j,k+1})+\beta_3(c\eta 1(k)(c\xi 1(i)P_{i-1,j,k-1}+$$
$$c\xi 2(i)P_{i,j,k-1}+c\xi 3(i)P_{i+1,j,k-1})+c\eta 2(k)(c\xi 1(i)P_{i-1,j,k}+c\xi 2(i)P_{i,j,k}+$$
$$c\xi 3(i)P_{i+1,j,k})+c\eta 3(k)(c\xi 1(i)P_{i-1,j,k+1}+c\xi 2(i)P_{i,j,k+1}+$$
$$c\xi 3P_{i+1,j,k+1})+\beta_6(c\xi 1(i)P_{i-1,j,k}+c\xi 2(i)P_{i,j,k}+c\xi 3(i)P_{i+1,j,k})+$$
$$\beta_7(c\eta 1(k)P_{i,j,k-1}+c\eta 2(k)P_{i,j,k}+c\eta 3(k)P_{i,j,k+1})+$$
$$\beta_8(c\phi 4(j)P_{i,j-1,k}+c\phi 5(j)P_{i,j,k}+c\phi 6(j)P_{i,j+1,k}) \qquad (4.53)$$

we define again

$$a_{ii}=-\Delta t\cdot\alpha_P\cdot\delta(c\xi 1(i)\beta_6+c\xi 4(i)\beta_1)$$
$$=-atd(c\xi 1(i)\beta_6+c\xi 4(i)\beta_1) \qquad (4.54)$$

where $$atd=\alpha_P\cdot\delta\cdot\Delta t \qquad (4.55)$$

$$b_{ii}=1-atd(c\xi 2(i)\beta_6+c\xi 5(i)\beta_1) \qquad (4.56)$$

$$c_{ii}=-atd(c\xi 3(i)\beta_6+c\xi 6(i)\beta_1) \qquad (4.57)$$

$$d_{ii}=RHS2 \qquad (4.58)$$

in which $ii=i-1$, $ii=1,2...n\xi-2$, $i=2,3...n\xi-1$.

The simplified expression of equation (4.52) is written as:

$$a_{ii}\omega^*_{i-1,j,k}+b_{ii}\omega^*_{i,j,k}+c_{ii}\omega^*_{i+1,j,k}=d_{ii} \qquad (4.59)$$

From equation (4.59), a set of simultaneous linear equations may be represented algebraically by:

$$b_1\omega^*_{2,j,k}+c_1\omega^*_{3,j,k}=d_1-a_1\omega^*_{1,j,k}$$
$$a_2\omega^*_{2,j,k}+b_2\omega^*_{3,j,k}+c_2\omega^*_{4,j,k}=d_2$$
$$a_3\omega^*_{3,j,k}+b_3\omega^*_{4,j,k}+c_3\omega^*_{5,j,k}=d_3$$

....

$$a_{n\xi-3}\omega^*_{n\xi-3,j,k}+b_{n\xi-3}\omega^*_{n\xi-2,j,k}+c_{n\xi-3}\omega^*_{n\xi-1,j,k}=d_{n\xi-3}$$
$$a_{n\xi-2}\omega^*_{n\xi-2,j,k}+b_{n\xi-2}\omega^*_{n\xi-1,j,k}=d_{n\xi-2}-c_{n\xi-2}\omega^*_{n\xi,j,k} \qquad (4.60)$$

where

if $ii=1$ $$d_1=d_1-a_1\omega^*_{1,j,k}\ ; \qquad (4.61)$$

if $ii=n\xi-2$ $$d_{n\xi-2}=d_{n\xi-2}-c_{n\xi-2}\omega^*_{n\xi,j,k} \qquad (4.62)$$

The coefficients a,b,c,d of the linear system can be expressed as a tri-diagonal matrix and solved by means of the Thomas (1949) algorithm which is a special case of the Gaussian Elimination method (see Appendix I).

The second step for solving ω^{**} is implicit equation (4.48) in the ϕ direction, thus:

$$\omega^{**} - atd\cdot\beta_8 \frac{\partial^2\omega}{\partial\phi^2} = \omega^* \tag{4.63}$$

Replacing the second partial derivative in equation (4.63), with the finite difference operators and three point difference scheme, we obtain $(1 < j < n\phi)$:

$$\omega^{**} - atd\cdot\beta_8(c\phi4(j)\omega^{**}_{i,j-1,k} + c\phi5(j)\omega^{**}_{i,j,k} + c\phi6(j)\omega^{**}_{i,j+1,k} = \omega^* \tag{4.64}$$

we define again

$$a_{jj} = -atd\cdot\beta_8\cdot c\phi4(j) \tag{4.65}$$

$$b_{jj} = 1 - atd\cdot\beta_8\cdot c\phi5(j) \tag{4.66}$$

$$c_{jj} = -atd\cdot\beta_8\cdot c\phi6(j) \tag{4.67}$$

$$d_{jj} = \omega^* \tag{4.68}$$

where

$jj = j-1$,

$jj = 1,2\ldots n\phi-2$, and $j = 2,3\ldots n\phi-1$.

we get:

$$a_{jj}\omega^{**}_{i,j-1,k} + b_{jj}\omega^{**}_{i,j,k} + c_{jj}\omega^{**}_{i,j+1,k} = d_{jj} \tag{4.69}$$

This tri-diagonal matrix can be again solved by the Thomas (1949) algorithm.

The third step for solving ω^{***} is implicit equation (4.48) in the η direction, we will obtain:

$$\omega^{***} - atd(\beta_2 \frac{\partial^2\omega}{\partial\eta^2} + \beta_7 \frac{\partial\omega}{\partial\eta}) = \omega^{**} \tag{4.70}$$

Replacing the first and second order partial derivatives in equation (4.70), with the finite difference operators and three point difference scheme, we get $(1 < k < n\eta)$:

$$\omega^{***} - atd(\beta_2(c\eta 4(k)\omega^{***}_{i,j,k-1} + c\eta 5(k)\omega^{***}_{i,j,k} + c\eta 6(k)\omega^{***}_{i,j,k+1}) +$$
$$\beta_7(c\eta 1(k)\omega^{***}_{i,j,k-1} + c\eta 2(k)\omega^{***}_{i,j,k} + c\eta 3(k)\omega^{***}_{i,j,k+1}) \quad (4.71)$$

let

$$a_{kk} = -atd(\beta_2 \cdot c\eta 4(k) + \beta_7 \cdot c\eta 1(k)) \quad (4.72)$$

$$b_{kk} = 1 - atd(\beta_2 \cdot c\eta 5(k) + \beta_7 \cdot c\eta 2(k)) \quad (4.73)$$

$$c_{kk} = -atd(\beta_2 \cdot c\eta 6(k) + \beta_7 \cdot c\eta 3(k)) \quad (4.74)$$

$$d_{kk} = \omega^{**} \quad (4.75)$$

where $kk = k-1$, $kk = 1,2...n\eta - 2$, and $k = 2,3...n\eta - 1$. Therefore we have the tri-diagonal matrix:

$$a_{kk}\omega^{***}_{i,j,k-1} + b_{kk}\omega^{***}_{i,j,k} + c_{kk}\omega^{***}_{i,j,k+1} = d_{kk} \quad (4.76)$$

By applying the Thomas (1949) algorithm, the final intermediate values of ω^{***} are obtained, then the updated final values of the pressures P_{new} are determined from the following expression:

$$P_{new} = P + atd \cdot \omega^{***} \quad (4.77)$$

4.5.2 Solution of the pressures at boundaries

Equation (3.75) governs the pressure field at the vertical wall of the silo and it is expressed in the transformed coordinates. Substituting equations (4.34) and (4.17) into equation (3.75) for $\frac{\partial P}{\partial \xi}$ and $\frac{\partial P}{\partial \eta}$, the discretised form of equation (3.75) can be written as below:

$$\alpha_1(c\xi b4\, P_{n\xi,j,k} + c\xi b5\, P_{n\xi-1,j,k} + c\xi b6\, P_{n\xi-2,j,k})$$
$$+\alpha_2(c\eta 1(k)\, P_{n\xi,j,k-1} + c\eta 2(k)\, P_{n\xi,j,k} + c\eta 3(k)\, P_{n\xi,j,k+1}) = 0 \quad (4.78)$$

we define

$$a_{kk} = \alpha_2 c\eta 1(k) \quad (4.79)$$

$$b_{kk} = \alpha_2 c\eta 2(k) + \alpha_1 c\xi b4 \quad (4.80)$$

$$c_{kk} = \alpha_2 c\eta 3(k) \tag{4.81}$$

$$d_{kk} = -\alpha_1 c\xi b5 P_{n\xi-1,j,k} - \alpha_1 c\xi b6 P_{n\xi-2,j,k} \tag{4.82}$$

where

If $kk = 1,$ $\qquad d_{kk} = d_{kk} - c_{kk} P_{n\xi,j,1}$ (4.83)

If $kk = n\eta - 2,$ $\qquad d_{kk} = d_{kk} - c_{kk} P_{n\xi,j,n\eta}$ (4.84)

So equation (4.78) becomes:

$$a_{kk} P_{n\xi,j,k-1} + b_{kk} P_{n\xi,j,k} + c_{kk} P_{n\xi,j,k+1} = d_{kk} \tag{4.85}$$

Equation (3.76) governs the pressure field along the sloping floor of the grain silo and it is expressed in transformed coordinates. The discretised equation is formed by employing the three point finite difference scheme, the first partial derivatives $\frac{\partial P}{\partial \xi}$ and $\frac{\partial P}{\partial \eta}$ are given by equations (4.12) and (4.38), so we will obtain ($\alpha_3 = 0$ and $0 < i < n\xi - 1$):

$$\begin{aligned} &-\sin\gamma_1(\alpha_1(c\xi 1(i) P_{i-1,j,1} + c\xi 2(i) P_{i,j,1} + c\xi 3(i) P_{i+1,j,1}) \\ &+\alpha_2(c\eta b1 P_{i,j,1} + c\eta b2 P_{i,j,2} + c\eta b3 P_{i,j,3})) + \cos\gamma_1(\alpha_4(\\ &c\eta b1 P_{i,j,1} + c\eta b2 P_{i,j,2} + c\eta b3 P_{i,j,3})) \end{aligned} \tag{4.86}$$

we define

$$a_{ii} = -\alpha_1 \sin\gamma_1 c\xi 1(i) \tag{4.87}$$

$$b_{ii} = -\sin\gamma_1 \alpha_1 c\xi 2(i) - \sin\gamma_1 \alpha_2 c\eta b1 + \cos\gamma_1 \alpha_4 c\eta b1 \tag{4.88}$$

$$c_{ii} = -\sin\gamma_1 \alpha_1 c\xi 3(i) \tag{4.89}$$

$$\begin{aligned} d_{ii} = &(\sin\gamma_1 \alpha_2 - \cos\gamma_1 \alpha_4) c\eta b2 P_{i,j,2} + \\ &(\sin\gamma_1 \alpha_2 - \cos\gamma_1 \alpha_4) c\eta b3 P_{i,j,3} \end{aligned} \tag{4.90}$$

where

if $ii = 1,$ $\qquad d_{ii} = d_{ii} - c_{ii} P_{1,1,1}$ (4.91)

if $ii = n\xi - 2,$ $\qquad d_{ii} = d_{ii} - c_{ii} P_{n\xi,j,1}$ (4.92)

Equation (4.86) may be written as:

$$a_{ii}P_{i-1,j,1}+b_{ii}P_{i,j,1}+c_{ii}P_{i+1,j,1}=d_{ii} \tag{4.93}$$

Equation (3.81) expresses the pressure field at the bottomed conical base of the silo in transformed coordinates. Using the three point finite difference scheme to replace equation (4.38) into equation (3.81), we get:

$$\frac{\partial P}{\partial \eta}=c\eta b1P_{1,j,1}+c\eta b2P_{1,j,2}+c\eta b3P_{1,j,3}=0 \tag{4.94}$$

so we will have:

$$P_{1,j,1}=\frac{c\eta b2\cdot P_{1,j,2}+c\eta b3\cdot P_{1,j,3}}{c\eta b1} \tag{4.95}$$

At the intersection of the vertical wall and the conical bottom of the silo, the pressure field is calculated by means of linear interpolation, and the expressed form takes as:

$$P_{coner}=P_{n\xi,j,1}=P_{n\xi-1,j,1}+\frac{h\xi(n\xi-1)}{h\xi(n\xi-1)+h\eta(1)}(P_{n\xi,j,2}-P_{n\xi-1,j,1}) \tag{4.96}$$

we define

$$a_{ii}=\frac{h\xi(n\xi-1)}{h\xi(n\xi-1)+h\eta(1)}-1 \tag{4.97}$$

$$b_{ii}=1 \tag{4.98}$$

$$c_{ii}=-\frac{h\xi(n\xi-1)}{h\xi(n\xi-1)+h\eta(1)} \tag{4.99}$$

$$d_{ii}=0 \tag{4.100}$$

Equation (4.96) becomes:

$$a_{ii}P_{n\xi-1,j,1}+b_{ii}P_{n\xi,j,1}+c_{ii}P_{n\xi,j,2}=0 \tag{4.101}$$

So the set of equations which comprise equations (4.85), (4.93) and (4.101) are solved simultaneously to yield the pressure distribution along the floor and wall of the silo.

At the centre of the silo, the pressure field is obtained by averaging the sum of the values which occur at the second centre nodes, and the equation may be written as:

$$P_{1,j,k} = \frac{sigmaP(k)}{n\phi} \tag{4.102}$$

where

$$sigmaP(k) = \sum_{j=1}^{n\phi} P_{2,j,k} \tag{4.103}$$

4.5.3 Solution of the velocity components

According to Darcy's law, when the pressure field is given, the solution of the momentum equation can be obtained by employing equations (3.83), (3.84) and (3.85), which are governing the components of the velocity field in transformed coordinates. In these equations, the first partial derivatives can be replaced by using finite difference operators as equations (4.12), (4.17) and (4.29), so we have:

The components of the radial velocity is discretised in the transformed coordinate as:

$$V_\xi = -\frac{1}{R_p}\{\alpha_1(c\xi 1(i)P_{i-1,j,k} + c\xi 2(i)P_{i,j,k} + c\xi 3(i)P_{i+1,j,k})$$

$$+\alpha_2(c\eta 1(k)\Gamma_{i,j,k-1} + c\eta 2(k)\Gamma_{i,j,k} + c\eta 3(k)\Gamma_{i,j,k+1})\} \tag{4.104}$$

The components of the angular velocity is discretised in the transformed coordinate as:

$$V_\theta = -\frac{1}{R_p}\frac{1}{R\xi}(c\phi 1(j)P_{i,j-1,k} + c\phi 2(j)P_{i,j,k} + c\phi 3(j)P_{i,j+1,k}) \tag{4.105}$$

The components of the vertical velocity is discretised in the transformed coordinate as:

$$V_\eta = -\frac{1}{R_p}\{\alpha_3(c\xi 1(i)P_{i-1,j,k} + c\xi 2(i)P_{i,j,k} + c\xi 3(i)P_{i+1,j,k})$$

$$+\alpha_4(c\eta 1(k)\Gamma_{i,j,k-1} + c\eta 2(k)\Gamma_{i,j,k} + c\eta 3(k)\Gamma_{i,j,k+1})\} \tag{4.106}$$

4.6 SOLUTION OF THE THERMAL ENERGY EQUATION

The transient thermal energy conservation equation is discretised by applying upwind finite difference scheme and solved by an explicit method. Equation (3.86) governing the temperature field in transformed coordinates is rearranged as following:

$$\frac{\partial T}{\partial t}=\frac{1}{denom}\{\ K_{eff}\left\{\beta_1\frac{\partial^2 T}{\partial\xi^2}+\beta_2\frac{\partial^2 T}{\partial\eta^2}+\beta_3\frac{\partial^2 T}{\partial\xi\partial\eta}+\beta_8\frac{\partial^2 T}{\partial\phi^2}\right\}$$

$$+\rho_b\frac{dm}{dt}(Q_r-0.6h_{vap})+\rho_b h_s\frac{\partial W}{\partial t}+\rho_b\frac{dm}{dt}\int_0^w h_s dw-\rho_a\left\{c_a+w((c_1)_\gamma+c_{2v})\right\}$$

$$\times\left\{V_{T\xi}\frac{\partial T}{\partial\xi}+V_{T\theta}\frac{\partial T}{\partial\phi}+V_{T\eta}\frac{\partial T}{\partial\eta}\right\}\ \} \tag{4.107}$$

where

$$denom=\rho_b\left\{(c_2)_\sigma+(c_1)_\sigma W+\frac{\partial H_W}{\partial T}\right\}+\varepsilon_\gamma\rho_a\left\{c_a+w((c_1)_\gamma+(c_{2v}))\right\}$$

$$V_{T\xi}=V_r\alpha_1+V_z\alpha_3-\beta_6\cdot K_{eff} \tag{4.108}$$

$$V_{T\eta}=V_r\alpha_2+V_z\alpha_4-\beta_7\cdot K_{eff} \tag{4.109}$$

$$_{T\theta}=\frac{V_\phi}{R\xi} \tag{4.110}$$

in which $V_{T\xi}$, $V_{T\eta}$ and $V_{T\theta}$ are effective velocities in the thermal energy balance.

Simplifying equation (4.107) again, we obtain:

$$\frac{\partial T}{\partial t}=\Pi_t+S_t-C_c(V_{T\xi}\frac{\partial T}{\partial\xi}+V_{T\eta}\frac{\partial T}{\partial\eta}+V_{T\theta}\frac{\partial T}{\partial\phi}) \tag{4.111}$$

where

$$\Pi_t=K_{eff}\left\{\beta_1\frac{\partial^2 T}{\partial\xi^2}+\beta_2\frac{\partial^2 T}{\partial\eta^2}+\beta_3\frac{\partial^2 T}{\partial\xi\partial\eta}+\beta_8\frac{\partial^2 T}{\partial\phi^2}\right\}\cdot denom \tag{4.112}$$

$$S_t=(\rho_b\frac{dm}{dt}(Q_r-0.6h_{vap})+\rho_b h_s\frac{\partial W}{\partial t}+\rho_b\frac{dm}{dt}\int_0^w h_s dw)\cdot denom \tag{4.113}$$

$$C_c=\rho_a\left\{c_a+w((c_1)_\gamma+c_{2v})\right\}\cdot denom \tag{4.114}$$

4.6.1 Upwind finite difference operators

The upwind finite difference operators $a\xi1u$, $a\xi2u$, $a\xi3u$, $a\eta1u$, $a\eta2u$, $a\eta3u$, $a\phi1u$, $a\phi2u$ and $a\phi3u$ are obtained by employing the centre difference scheme, it is shown in figure 4.5 for the ξ direction:

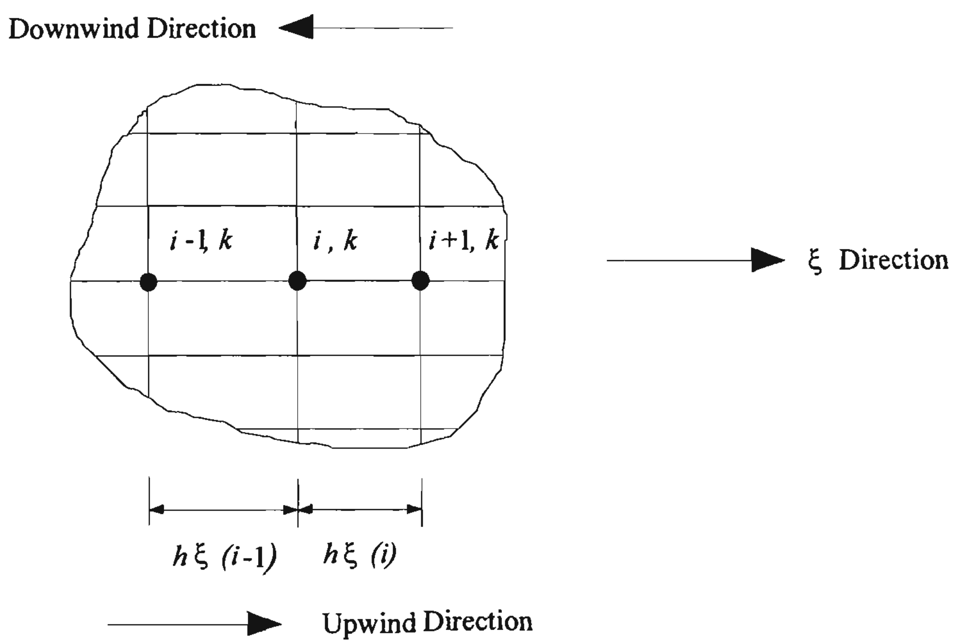


Figure 4.5 The three points used in determining the upwind operators

In the ξ direction, the first order derivative $\frac{\partial\Theta}{\partial\xi}$ is solved by the centre point (i) upwind to the point $(i-1)$, such as:

$$\frac{\overrightarrow{\partial\Theta}}{\partial\xi} = \frac{\Theta_{i,j,k} - \Theta_{i-1,j,k}}{h\xi(i-1)} \tag{4.115}$$

where Θ is an arbitrary variable.

we define

$$a\xi1u = -\frac{1}{h\xi(i-1)} \tag{4.116}$$

$$a\xi2u = \frac{1}{h\xi(i-1)} \tag{4.117}$$

$$a\xi 3u = 0 \tag{4.118}$$

So equation (4.115) becomes:

$$\frac{\partial \Theta}{\partial \xi} = a\xi 1u\Theta_{i-1,j,k} + a\xi 2u\Theta_{i,j,k} + a\xi 3u\Theta_{i+1,j,k} \tag{4.119}$$

When the flow is in the opposite direction of ξ, we have:

$$\frac{\overleftarrow{\partial \Theta}}{\partial \xi} = \frac{\Theta_{i+1,j,k} - \Theta_{i,j,k}}{h\xi(i)} \tag{4.120}$$

In this case, the upwind finite difference operators in equation (4.119) assume the value as:

$$a\xi 1u = 0 \tag{4.121}$$

$$a\xi 2u = -\frac{1}{h\xi(i)} \tag{4.122}$$

$$a\xi 3u = \frac{1}{h\xi(i)} \tag{4.123}$$

Similarly, in the η direction, the first order derivative $\frac{\partial \Theta}{\partial \eta}$ is also solved by the centre point (k) upwind to the point $(k-1)$, we have:

$$\frac{\overrightarrow{\partial \Theta}}{\partial \eta} = \frac{\Theta_{i,j,k} - \Theta_{i,j,k-1}}{h\eta(k-1)} \tag{4.124}$$

we define again

$$a\eta 1u = -\frac{1}{h\eta(k-1)} \tag{4.125}$$

$$a\eta 2u = \frac{1}{h\eta(k-1)} \tag{4.126}$$

$$a\eta 3u = 0 \tag{4.127}$$

So equation (4.124) becomes:

$$\frac{\partial \Theta}{\partial \eta} = a\eta 1u\Theta_{i,j,k-1} + a\eta 2u\Theta_{i,j,k} + a\eta 3u\Theta_{i,j,k+1} \tag{4.128}$$

When the flow is in the opposite direction of η, we have:

$$\frac{\overleftarrow{\partial\Theta}}{\partial\eta}=\frac{\Theta_{i,j,k+1}-\Theta_{i,j,k}}{h\eta(k)} \tag{4.129}$$

The upwind finite difference operators in equation (4.128) assume the value as:

$$a\eta 1u=0 \tag{4.130}$$

$$a\eta 2u=-\frac{1}{h\eta(k)} \tag{4.131}$$

$$a\eta 3u=\frac{1}{h\eta(k)} \tag{4.132}$$

In the ϕ direction, the first order derivative $\frac{\partial\Theta}{\partial\phi}$ is determined by the centre point (j) to the upwind point $(j-1)$, we get:

$$\frac{\overrightarrow{\partial\Theta}}{\partial\phi}=\frac{\Theta_{i,j,k}-\Theta_{i,j-1,k}}{h\phi(j-1)} \tag{4.133}$$

we define

$$a\phi 1u=-\frac{1}{h\phi(j-1)} \tag{4.134}$$

$$a\phi 2u=\frac{1}{h\phi(j-1)} \tag{4.135}$$

$$a\phi 3u=0 \tag{4.136}$$

So equation (4.133) becomes:

$$\frac{\partial\Theta}{\partial\phi}=a\phi 1u\Theta_{i,j-1,k}+a\phi 2u\Theta_{i,j,k}+a\phi 3u\Theta_{i,j+1,k} \tag{4.137}$$

When the flow is in the opposite direction of ϕ , we also have:

$$\frac{\overleftarrow{\partial\Theta}}{\partial\phi}=\frac{\Theta_{i,j+1,k}-\Theta_{i,j,k}}{h\phi(j)} \tag{4.138}$$

The upwind finite difference operators in equation (4.128) assume the value as:

$$a\phi 1u = 0 \tag{4.139}$$

$$a\phi 2u = -\frac{1}{h\phi(j)} \tag{4.140}$$

$$a\phi 3u = \frac{1}{h\phi(j)} \tag{4.141}$$

4.6.2 Explicit method for the thermal energy equation

The solution procedure of explicit method for solving thermal energy equation is presented by the following. In equation (4.111), the first derivative with respect to temperatures $\frac{\partial T}{\partial \xi}$ is calculated by the upwind finite difference scheme as equation (4.119). The upwind finite difference operators are given by equations (4.116), (4.117) and (4.118). So we will have:

$$\frac{\partial T}{\partial \xi} = c\xi 1uT_{i-1,j,k} + c\xi 2uT_{i,j,k} + c\xi 3uT_{i+1,j,k} \tag{4.142}$$

Similarly, the first derivatives with respect to temperatures $\frac{\partial T}{\partial \eta}$ and $\frac{\partial T}{\partial \phi}$ are also given by the upwind finite difference scheme as equations (4.128) and (4.137). The upwind finite difference operators are determined by equations (4.125), (4.126), (4.127), (4.134), (4.135) and (4.136), we will obtain:

$$\frac{\partial T}{\partial \eta} = a\eta 1uT_{i,j,k-1} + a\eta 2uT_{i,j,k} + a\eta 3uT_{i,j,k+1} \tag{4.143}$$

and

$$\frac{\partial T}{\partial \phi} = a\phi 1uT_{i,j-1,k} + a\phi 2uT_{i,j,k} + a\phi 3uT_{i,j+1,k} \tag{4.144}$$

In equation (4.111), the second order derivatives $\frac{\partial^2 T}{\partial \xi^2}, \frac{\partial^2 T}{\partial \phi^2}, \frac{\partial^2 T}{\partial \xi^2}$ and the mixed derivative $\frac{\partial^2 T}{\partial \xi \partial \eta}$ are determined by the three point finite difference scheme, which are defined in equations (4.13), (4.21) (4.29) and (4.33). So we are able to write:

$$\beta_1\frac{\partial^2 T}{\partial\xi^2}+\beta_2\frac{\partial^2 T}{\partial\eta^2}+\beta_3\frac{\partial^2 T}{\partial\xi\partial\eta}+\beta_8\frac{\partial^2 T}{\partial\phi^2}=\beta_1(c\xi 4(i)T_{i-1,j,k}+c\xi 5(i)T_{i,j,k}+c\xi 6(i)T_{i+1,j,k})+$$
$$\beta_2(c\eta 4(k)T_{i,j,k-1}+c\eta 5(k)T_{i,j,k}+c\eta 6(k)T_{i,j,k+1})+$$
$$\beta_3(c\eta 1(k)(c\xi 1(i)T_{i-1,j,k-1}+c\xi 2(i)T_{i,j,k-1}+$$
$$c\xi 3(i)T_{i+1,j,k-1})+c\eta 2(k)(c\xi 1(i)T_{i-1,j,k}+c\xi 2(i)T_{i,j,k}$$
$$+c\xi 3(i)T_{i+1,j,k})+c\eta 3(k)(c\xi 1(i)T_{i-1,j,k+1}+$$
$$c\xi 2(i)T_{i,j,k+1}+c\xi 3T_{i+1,j,k+1}))+$$
$$\beta_8(c\phi 4(j)T_{i,j-1,k}+c\phi 5(j)T_{i,j,k}+$$
$$c\phi 6(j)T_{i,j+1,k}) \tag{4.145}$$

In the left hand side of equation (4.111), the partial derivative with respect to time $\frac{\partial T}{\partial t}$ is discretised as a forward finite difference, thus:

$$\frac{\partial T}{\partial t}=\frac{T_{i,j,k}^{P+1}-T_{i,j,k}^{P}}{\Delta t} \tag{4.146}$$

where $T_{i,j,k}^{P+1}$ is expressed in terms of temperatures pertaining to the previous iteration time step P. Substituting equations (4.145) and (4.146) into equation (4.111), the updated values of the temperatures $T_{i,j,k}^{P+1}$ in grain silo can be obtained by:

$$T_{i,j,k}^{P+1}=T_{i,j,k}^{P}+\Delta t\{\Pi_t+S_t-C_c(V_{T\xi}\frac{\partial T}{\partial\xi}+V_{T\eta}\frac{\partial T}{\partial\eta}+V_{T\theta}\frac{\partial T}{\partial\phi})\} \tag{4.147}$$

4.6.3 Solution of the temperatures at boundaries

Equation (3.90) governs the temperature field at the upper surface of the grain silo in transformed coordinates. This equation is found by applying the three point differencing scheme, and discretization the first partial derivatives $\frac{\partial T}{\partial\xi}$ and $\frac{\partial T}{\partial\eta}$, which occur pertained in equation (4.12) and equation (4.38). So we will obtain ($\alpha_3=0$ and $0<i<n\xi-1$):

$$\sin\gamma_2(\alpha_1(c\xi 1(i)T_{i-1,j,n\eta}+c\xi 2(i)T_{i,j,n\eta}+c\xi 3(i)T_{i+1,j,n\eta})$$

$$+\alpha_2(c\eta b4T_{i,j,n\eta}+c\eta b5T_{i,j,n\eta-1}+c\eta b6T_{i,j,n\eta-2}))+\cos\gamma_2(\alpha_4(c\eta b4T_{i,j,n\eta}+c\eta b5T_{i,j,n\eta-1}+c\eta b6T_{i,j,n\eta-2}))=0 \tag{4.148}$$

we define

$$a_{ii}=\alpha_1\sin\gamma_2 c\xi 1(i) \tag{4.149}$$

$$b_{ii}=\sin\gamma_2\alpha_1 c\xi 2(i)+\sin\gamma_2\alpha_2 c\eta b4+\cos\gamma_2\alpha_4 c\eta b4 \tag{4.150}$$

$$c_{ii}=\sin\gamma_2\alpha_1 c\xi 3(i) \tag{4.151}$$

$$d_{ii}=-(\sin\gamma_2\alpha_2+\cos\gamma_2\alpha_4)c\eta b5T_{i,j,n\eta-1}-(\sin\gamma_2\alpha_2+\cos\gamma_2\alpha_4)c\eta b6T_{i,j,n\eta-2} \tag{4.152}$$

where

if $ii=1$, $$d_{ii}=d_{ii}-c_{ii}T_{1,j,n\eta} \tag{4.153}$$

if $ii=n\xi-2$, $$d_{ii}=d_{ii}-c_{ii}T_{n\xi,j,n\eta} \tag{4.154}$$

Equation (4.148) can be written as:

$$a_{ii}T_{i-1,j,n\eta}+b_{ii}T_{i,j,n\eta}+c_{ii}T_{i+1,j,n\eta}=d_{ii} \tag{4.155}$$

The tri-diagonal matrix form of equation (4.155) is readily solved by applying the Thomas (1949) algorithm (see Appendix I).

Along the vertical line of the centre silo, the temperature field is calculated by averaging of second nodes at radial direction, that is:

$$T_{1,j,k}=\frac{sigmaT(k)}{n\phi} \tag{4.156}$$

where

$$sigmaT(k)=\sum_{j=1}^{n\phi}T_{2,j,k} \tag{4.157}$$

Equation (3.87) expresses the temperature field at the vertical wall of the silo in the transformed coordinates, this temperature field ($T_{n\xi,j,k}$) is related with solar radiation and ambient temperature flopping on the vertical wall of the silo ($ts1$, see figure 5.1), thus:

$$T_{n\xi,j,k}=ts1 \tag{4.158}$$

Similarly, in transformed coordinates equation (3.88) governs the temperature field ($T_{i,j,1}$) along the sloping floor of the silo, which is depending on solar radiation and ambient air temperature flopping on the silo surface ($ts2$, see figure 5.1), such as:

$$T_{i,j,1} = ts2 \tag{4.159}$$

Equation (3.92) and equation (3.93) are determined the temperature field at the upper surface and the bottom cone base of the silo in transformed coordinates. Employing the three point finite difference scheme at the boundaries, which is expressed by equation (4.42), we get:

$$c\eta b4 T_{1,j,n\eta} + c\eta b5 T_{1,j,n\eta-1} + c\eta b6 T_{1,j,n\eta-2} = 0 \tag{4.160}$$

let

$$T_{1,j,n\eta} = \frac{c\eta b5 T_{1,j,n\eta-1} + c\eta b6 T_{1,j,n\eta-2}}{c\eta b4} \tag{4.161}$$

Again the temperatures at the bottomed cone base of the silo ($T_{1,j,1}$) is also dependent on solar radiation and ambient air temperature ($ts2*$, see figue 5.1) which is striking on the same point of the silo surface, namely,

$$T_{1,j,1} = ts2* \tag{4.162}$$

4.7 SOLUTION OF THE MOISTURE CONSERVATION EQUATION

The moisture conservation equation is discretised by using the upwind finite difference scheme and solved by a explicit method. Equation (3.94) that governs moisture contents in transformed coordinates may be written in a compact form as:

$$\frac{\partial W}{\partial t} = D_{eff} \frac{\rho_a}{\rho_b} \left\{ \beta_1 \frac{\partial^2 H}{\partial \xi^2} + \beta_2 \frac{\partial^2 H}{\partial \eta^2} + \beta_3 \frac{\partial^2 H}{\partial \xi \partial \eta} + \beta_8 \frac{\partial^2 H}{\partial \phi^2} \right\}$$

$$-\left\{V_{W\xi}\frac{\partial H}{\partial \xi}+V_{W\theta}\frac{\partial H}{\partial \phi}+V_{W\eta}\frac{\partial H}{\partial \eta}\right\}+\frac{dm}{dt}(0.6+W) \tag{4.163}$$

where

$$V_{W\xi}=\frac{\rho_a}{\rho_b}(V_r\alpha_1+V_z\alpha_3-\beta_6\cdot D_{eff}) \tag{4.164}$$

$$V_{W\eta}=\frac{\rho_a}{\rho_b}(V_r\alpha_2+V_z\alpha_4-\beta_7\cdot D_{eff}) \tag{4.165}$$

$$V_{W\theta}=\frac{\rho_a}{\rho_b}\frac{V_\phi}{R\xi} \tag{4.166}$$

where $V_{W\xi}$, $V_{W\theta}$ and $V_{W\eta}$ are the effective velocities with respect to the moisture contents.

If we define again

$$\Pi_W=D_{eff}\left\{\beta_1\frac{\partial^2 H}{\partial \xi^2}+\beta_2\frac{\partial^2 H}{\partial \eta^2}+\beta_3\frac{\partial^2 H}{\partial \xi\partial \eta}+\beta_8\frac{\partial^2 H}{\partial \phi^2}\right\} \tag{4.167}$$

$$S_W=(0.6+W)\frac{dm}{dt} \tag{4.168}$$

Equation (4.163) becomes:

$$\frac{\partial W}{\partial t}=\Pi_W-\left\{V_{W\xi}\frac{\partial H}{\partial \xi}+V_{W\theta}\frac{\partial H}{\partial \phi}+V_{W\eta}\frac{\partial H}{\partial \eta}\right\}+S_W \tag{4.169}$$

4.7.1 Explicit method for the moisture conservation equation

The solution procedure of explicit method for solving the moisture content equation is given as follows. In equation (4.169), the first derivative with respect to the absolute humidities of the air $\frac{\partial H}{\partial \xi}$ is obtained by the upwind finite difference scheme given by equation (4.119) with the upwind finite difference operators which are determined by equations (4.116), (4.117) and (4.118), so we will find that:

$$\frac{\partial H}{\partial \xi}=c\xi 1uH_{i-1,j,k}+c\xi 2uH_{i,j,k}+c\xi 3uH_{i+1,j,k} \tag{4.170}$$

In the same way, the first derivatives with respect to the absolute humidities of the air $\frac{\partial H}{\partial \eta}$ and $\frac{\partial H}{\partial \phi}$ are given by upwind finite difference scheme in equations (4.128) and

(4.137), and the upwind finite difference operators which are calculated by equations (4.125), (4.126), (4.127), (4.134), (4.135) and (4.136), so the expressions are:

$$\frac{\partial H}{\partial \eta} = a\eta 1uH_{i,j,k-1} + a\eta 2uH_{i,j,k} + a\eta 3uH_{i,j,k+1} \tag{4.171}$$

and

$$\frac{\partial H}{\partial \phi} = a\phi 1uH_{i,j-1,k} + a\phi 2uH_{i,j,k} + a\phi 3uH_{i,j+1,k} \tag{4.172}$$

In equation (4.169), the second derivatives $\frac{\partial^2 H}{\partial \xi^2}$, $\frac{\partial^2 H}{\partial \eta^2}$ and $\frac{\partial^2 H}{\partial \phi^2}$ and the mixed derivative $\frac{\partial^2 H}{\partial \xi \partial \eta}$ are determined by the three point finite difference scheme, it is reflected in equations (4.13), (4.21) (4.29) and (4.33). We have:

$$\begin{aligned}\beta_1\frac{\partial^2 H}{\partial \xi^2} + \beta_2\frac{\partial^2 H}{\partial \eta^2} + \beta_3\frac{\partial^2 H}{\partial \xi \partial \eta} + \beta_8\frac{\partial^2 H}{\partial \phi^2} = {} & \beta_1(c\xi 4(i)H_{i-1,j,k} + c\xi 5(i)H_{i,j,k} + c\xi 6(i)H_{i+1,j,k}) + \\ & \beta_2(c\eta 4(k)H_{i,j,k-1} + c\eta 5(k)H_{i,j,k} + c\eta 6(k)H_{i,j,k+1}) + \\ & \beta_3(c\eta 1(k)(c\xi 1(i)H_{i-1,j,k-1} + c\xi 2(i)H_{i,j,k-1} + \\ & c\xi 3(i)H_{i+1,j,k-1}) + c\eta 2(k)(c\xi 1(i)H_{i-1,j,k} + \\ & c\xi 2(i)H_{i,j,k} + c\xi 3(i)H_{i+1,j,k}) + c\eta 3(k) \\ & (c\xi 1(i)H_{i-1,j,k+1} + c\xi 2(i)H_{i,j,k+1} + c\xi 3H_{i+1,j,k+1})) + \\ & \beta_8(c\phi 4(j)H_{i,j-1,k} + c\phi 5(j)H_{i,j,k} + \\ & c\phi 6(j)H_{i,j+1,k})\end{aligned} \tag{4.173}$$

The partial derivative with respect to time $\frac{\partial W}{\partial t}$ which is in the left hand side of equation (4.169) is discretised by forward finite difference scheme, such as:

$$\frac{\partial W}{\partial t} = \frac{W_{i,j,k}{}^{P+1} - W_{i,j,k}{}^{P}}{\Delta t} \tag{4.174}$$

where $W_{i,j,k}^{P+1}$ is the term of moisture contents pertaining to the previous iteration time step P of the moisture contents. So updated value of the grain moisture content distribution $W_{i,j,k}^{P+1}$ is found from:

$$W^{P+1}_{i,j,k} = W^{P}_{i,j,k} + \Delta t\{\Pi_t + S_t - C_c(V_{T\xi}\frac{\partial H}{\partial \xi} + V_{T\eta}\frac{\partial H}{\partial \eta} + V_{T\theta}\frac{\partial H}{\partial \phi})\} \quad (4.175)$$

4.7.2 Solution of the absolute humidities at boundaries

The boundary conditions along the floor of the silo in equation (3.95) is expressed the absolute humidities of the air at the transformed coordinates. In order to discretise the partial derivatives $\frac{\partial H}{\partial \xi}$ and $\frac{\partial H}{\partial \eta}$, the three point finite difference scheme will be employed by equations (4.12) and (4.38), we will get ($\alpha_3 = 0$ and $0 < i < n\xi - 1$):

$$-\sin\gamma_1(\alpha_1(c\xi 1(i)H_{i-1,j,1} + c\xi 2(i)H_{i,j,1} + c\xi 3(i)H_{i+1,j,1})$$
$$+\alpha_2(c\eta b1H_{i,j,1} + c\eta b2H_{i,j,2} + c\eta b3H_{i,j,3})) + \cos\gamma_1(\alpha_4($$
$$c\eta b1H_{i,j,1} + c\eta b2H_{i,j,2} + c\eta b3H_{i,j,3})) \quad (4.176)$$

we define

$$a_{ii} = -\alpha_1\sin\gamma_1 c\xi 1(i) \quad (4.177)$$
$$b_{ii} = -\sin\gamma_1\alpha_1 c\xi 2(i) - \sin\gamma_1\alpha_2 c\eta b1 + \cos\gamma_1\alpha_4 c\eta b1 \quad (4.178)$$
$$c_{ii} = -\sin\gamma_1\alpha_1 c\xi 3(i) \quad (4.179)$$
$$d_{ii} = (\sin\gamma_1\alpha_2 - \cos\gamma_1\alpha_4)c\eta b2H_{i,j,2} +$$
$$(\sin\gamma_1\alpha_2 - \cos\gamma_1\alpha_4)c\eta b3H_{i,j,3} \quad (4.180)$$

where

if $ii = 1$, $\quad d_{ii} = d_{ii} - c_{ii}H_{1,1,1}$ (4.181)

if $ii = n\xi - 2$, $\quad d_{ii} = d_{ii} - c_{ii}H_{n\xi,j,1}$ (4.182)

Equation (4.176) can be rewritten as:

$$a_{ii}H_{i-1,j,1} + b_{ii}H_{i,j,1} + c_{ii}H_{i+1,j,1} = d_{ii} \quad (4.183)$$

Equation (3.97) is expressed the absolute humidities of the air along the vertical wall of the silo and written in transformed coordinates, the first partial derivatives $\frac{\partial H}{\partial \xi}$ and

$\frac{\partial H}{\partial \eta}$ are discretised from the finite difference scheme as equations (4.17) and (4.34), it can be presented as following:

$$\alpha_1(c\xi b4 H_{n\xi,j,k} + c\xi b5 H_{n\xi-1,j,k} + c\xi b6 H_{n\xi-2,j,k})$$
$$+\alpha_2(c\eta 1(k) H_{n\xi,j,k-1} + c\eta 2(k) H_{n\xi,j,k} + c\eta 3(k) H_{n\xi,j,k+1}) = 0 \quad (4.184)$$

we define

$$a_{kk} = \alpha_2 c\eta 1(k) \quad (4.185)$$
$$b_{kk} = \alpha_2 c\eta 2(k) + \alpha_1 c\xi b4 \quad (4.186)$$
$$c_{kk} = \alpha_2 c\eta 3(k) \quad (4.187)$$
$$d_{kk} = -\alpha_1 c\xi b5 H_{n\xi-1,j,k} - \alpha_1 c\xi b6 H_{n\xi-2,j,k} \quad (4.188)$$

where

If $kk = 1$, $\quad d_{kk} = d_{kk} - c_{kk} H_{n\xi,j,1} \quad (4.189)$

If $kk = n\eta - 2$, $\quad d_{kk} = d_{kk} - c_{kk} H_{n\xi,j,n\eta} \quad (4.190)$

So equation (4.184) becomes:

$$a_{kk} H_{n\xi,j,k-1} + b_{kk} H_{n\xi,j,k} + c_{kk} H_{n\xi,j,k+1} = d_{kk} \quad (4.191)$$

The absolute humidities of the air at the intersection of the floor and wall of the silo is determined by linear interpolation, it can be written as:

$$H_{coner} = H_{n\xi,j,1} = H_{n\xi-1,j,1} + \frac{h\xi(n\xi-1)}{h\xi(n\xi-1) + h\eta(1)}(H_{n\xi,j,2} - H_{n\xi-1,j,1}) \quad (4.192)$$

we define

$$a_{ii} = \frac{h\xi(n\xi-1)}{h\xi(n\xi-1) + h\eta(1)} - 1 \quad (4.193)$$

$$b_{ii} = 1 \quad (4.194)$$

$$c_{ii} = -\frac{h\xi(n\xi-1)}{h\xi(n\xi-1) + h\eta(1)} \quad (4.195)$$

$$d_{ii} = 0 \quad (4.196)$$

Equation (4.192) becomes:

$$a_{ii}H_{n\xi-1,j,1}+b_{ii}H_{n\xi,j,1}+c_{ii}H_{n\xi,j,2}=0 \tag{4.197}$$

Similar, the set of equations (4.183), (4.191) and (4.197) can be solved simultaneously by means of Thomas (1949) algorithm (see Appendix I) for obtaining the absolute humidity field along the floor and the wall of the silo.

At the middle of the silo, the absolute humidities of the air is obtained by averaging of second nodes in the radial direction, thus:

$$H_{1,j,k}=\frac{sigmaH(k)}{n\phi} \tag{4.198}$$

where

$$sigmaH(k)=\sum_{j=1}^{n\phi}H_{2,j,k} \tag{4.199}$$

At the upper surface of the grains, equation (3.98) govern the absolute humidities of the air in transformed coordinates, the first partial derivative $\frac{\partial H}{\partial \eta}$ can be solved by the three point finite difference scheme from equations (4.42) and (3.98), we will obtain:

$$c\eta b4H_{n\xi,j,n\eta}+c\eta b5H_{n\xi,j,n\eta-1}+c\eta b6H_{n\xi,j,n\eta-2}=0 \tag{4.200}$$

so we have

$$H_{n\xi,j,n\eta}=-\frac{c\eta b5H_{n\xi,j,n\eta-1}+c\eta b6H_{n\xi,j,n\eta-2}}{c\eta b4} \tag{4.201}$$

Equation (3.100) yields the absolute humidities of the air at the bottomed cone base of the silo in transformed coordinates, the first partial derivative $\frac{\partial H}{\partial \eta}$ is discretised by using the three point finite difference scheme which is given by equation (4.38), we get:

$$\frac{\partial H}{\partial \eta}=c\eta b1H_{1,j,1}+c\eta b2H_{1,j,2}+c\eta b3H_{1,j,3}=0 \tag{4.202}$$

so we have

$$H_{1,j,1} = \frac{c\eta b2 \cdot H_{1,j,2} + c\eta b3 \cdot H_{1,j,3}}{c\eta b1} \tag{4.203}$$

Equation (3.101) is used to calculate the absolute humidities of the air at the peak of the grain in transformed coordinates, discretising it with the three point finite difference scheme given by equation (4.42), we will obtain:

$$c\eta b4 H_{1,j,m\eta} + c\eta b5 H_{1,j,m\eta-1} + c\eta b6 H_{1,j,m\eta-2} = 0 \tag{4.204}$$

let

$$H_{1,j,m\eta} = \frac{c\eta b5 H_{1,j,m\eta-1} + c\eta b6 H_{1,j,m\eta-2}}{c\eta b4} \tag{4.205}$$

4.7.3 Solution of the moisture contents at boundaries

According to the numerical solution of the grain moisture content field at the interior nodes, the grain moisture contents at boundary conditions can easily solve:

At the centre of the silo, the grain moisture content distribution is obtained by averaging the sum of the grain moisture contents which were given from the second centre nodes, the formula is written as:

$$W_{1,j,k} = \frac{sigmaW(k)}{n\phi} \tag{4.206}$$

where

$$sigmaW(k) = \sum_{j=1}^{n\phi} W_{2,j,k} \tag{4.207}$$

Along the floor of the silo, the grain moisture contents are determined by employing the Taylor expansion, thus:

$$W_{i,j,1} = W_{i,j,2} - h\eta(1)\frac{\partial W}{\partial \eta} \tag{4.208}$$

and

$$W_{i,j,1} = W_{i,j,3} - (h\eta(1) + h\eta(2))\frac{\partial W}{\partial \eta} \tag{4.209}$$

Re-arranging both of equations (4.208) and (4.209), we get

$$W_{i,j,1} = \frac{(h\eta(1) + h\eta(2))W_{i,j,2} - h\eta(1)W_{i,j,3}}{h\eta(2)} \tag{4.210}$$

At the upper surface of the grain, the moisture contents of grain are also obtained by applying the Taylor expansion, the results are from the formula as following:

$$W_{i,j,n\eta} = \frac{(h\eta(n\eta - 1) + h\eta(n\eta - 2))W_{i,j,n\eta-1} - h\eta(n\eta - 1)W_{i,j,n\eta-2}}{h\eta(n\eta - 2)} \tag{4.211}$$

Along the wall of the silo, Taylor expansion is employed, the update rule of the moisture contents are:

$$W_{n\xi,j,k} = \frac{(h\xi(n\xi - 1) + h\xi(n\xi - 2))W_{n\xi-1,j,k} - h\xi(n\xi - 1)W_{n\xi-2,j,k}}{h\xi(n\xi - 2)} \tag{4.212}$$

CHAPTER 5

SYSTEM PROPERTIES

The heat and mass transfer conservation equations that govern the behaviour of forced convective processes occurrence in ventilated grain silos have been discussed in chapter 3 and 4. In this chapter we shall consider some of the thermodynamic properties of grain that are the essential constitution for requiring to model heat and mass transfer phenomenon, and other relevant biological properties required for modelling the stored grain ecosystem. In this work typical values of the physical and biological parameters that govern the rate processes have been used. It should be borne in mind, however, that stored grains and their spoilage agents are biological in nature, and therefore extremely variable. The results presented in this thesis therefore must, by necessity, indicate trends rather than absolute values.

5.1 INTERRELATIONS

Stored grains have several physical and biological properties, these are interrelated with one another and are affected by other variables. The interrelationship among physical and biological properties will be divided into two categories:

(1). Physical - Thermal properties which are related to the air / grain / water system are responsible for transfer and exchange of heat and moisture through stored grain.

They include heat conductivities, thermal capacitance, moisture diffusivity, absorption and desorption characteristics of the grain, and boundary conditions. The latter include the impermeable nature of the silo wall, and the absorptivity and emissivity of the silo surfaces which are the system properties.

(2). Biological properties are the external sources which invade the quality of grain. They are included: the response of insect population to their local environments, the rates at which moulds respire and cause damage to the grain by loss in the dry weight, and the viability of seed to germinate and develop into strong and healthy seedlings. The degree of deterioration is affected by the interaction of temperature, moisture content and stored grains.

5.2 SORPTION ISOTHERMS

Grains are hygroscopic materials, because they adsorb and desorb water. When water is absorbed by grain, it can be take up to about half of the dry weight in water. When the water content is about 30-40% of the air-dry basis of the grain, germination occurs. In general, grain can contents moisture and be considered in three forms: (a) Water of composition or bound water, it is an integral part of organic molecules, such as carbohydrate, protein, vitamin, mineral, fat and fibre. (b) Absorbed water which is more closely linked to the sorbing surface. (c) Free water or absorbed water, it is held in the intergranular spaces and within the pores of the material. These hygroscopic properties of the grain and the water exchange relationships do not cut clearly for demarcation the three forms. The water content of grain alone is insufficient to assess its keeping qualities. When the relative humidity of the air surrounding the grain is in the range of 65-75%, the critical moisture of stored grains will be arise, and heat is liberated as fungi becomes active. When the vapour pressure of the water is decreased, the heat of sorption exceeds the latent heat of vaporisation of free water.

In this section, the behaviour of hygroscopic porous media will be considered as the relationship between grain temperature, grain moisture content, and the humidity of the intergranular air. The mathematical descriptions that related to these variables are called sorption isotherms and are formulated as following.

In order to use Smith's (1947) equation that relates the relative humidity of the intergranular air with grain temperature and moisture content, it is necessary to determine the value of the wet basis grain moisture content (M_w) corresponding to a given the fractional dry basis grain moisture content ($W_{i,j,k}$). The expression is:

$$M_w = \frac{100 \cdot W_{i,j,k}}{1 + W_{i,j,k}} \tag{5.1}$$

According to Smith's (1947) isosteric relationship, the wet basis grain moisture content is function of grain temperature and the relative humidity of the intergranular air *(rh)* in equilibrium with the grain, and it can be written as:

$$M_w = a_1 + a_2 T_{i,j,k} - (a_3 + a_4 T_{i,j,k}) \ln(1 - rh) \tag{5.2}$$

so we obtain

$$rh = 1 - e^{\frac{(a_1 + a_2 T_{i,j,k}) - M_w}{a_3 + a_4 T_{i,j,k}}} \tag{5.3}$$

where a_1, a_2, a_3 and a_4 are the isosteric specific empirical constants for wheat given by Sutherland (1983) and presented in table 5.1.

Table 5.1 Isosteric specific empirical constants

a_1	a_2	a_3	a_4
9.304	-0.052076	4.1483	0.0215

The equation that relates the saturation vapour pressure of free water (P_{sat}) and temperature ($T_{i,j,k}$) is proposed by Hunter (1987) to be:

$$P_{sat} = \frac{6\times10^{25}}{(T_{i,j,k}+273)^5}\cdot e^{\frac{-6800}{T_{i,j,k}+273}} \tag{5.4}$$

Therefore the vapour pressure of moisture (P_{vap}) in the intergranular air is defined as:

$$P_{vap} = rh\cdot P_{sat} \tag{5.5}$$

and since the absolute humidity of air ($H_{i,j,k}$) at the interior nodes is found from:

$$H_{i,j,k} = \frac{gn\cdot P_{vap}}{P_{atm}-P_{vap}} \tag{5.6}$$

where P_{atm} is atmospheric pressure, and gn is ratio of molecular weights of water and air.

The heat of sorption is obtained by Gallaher's (1951) expression, that is reasonably thermodynamically consistent with the above sorption isotherm. The formula can be written as:

$$h_s = h_{vap}(1+23\cdot e^{(-40\cdot W_{i,j,k})}) \tag{5.7}$$

where h_{vap} is the latent heat of evaporation of free water, it is accurately approximated by a linear function of temperature, hence we have:

$$h_{vap} = c_{1v} + c_{2v}\cdot T_{i,j,k} \tag{5.8}$$

where c_{1v} and c_{2v} are the empirical constants, and in which:

$$c_{2v} = \frac{\partial h_{vap}}{\partial T} \tag{5.9}$$

where $\frac{\partial h_{vap}}{\partial T}$ is the differential of latent heat of vaporisation of water with respect to the temperature, and it is obtainable from steam tables.

So that the integral of the heat of sorption may be found by the following equation, thus:

$$\int_0^W h_s dW = h_s\{0.575(1-e^{(-40\cdot W)})+W\} \tag{5.10}$$

By definition, the integral heat of wetting that arises from the hygroscopic nature of the grain is written in formula as:

$$H_W = \int_0^W h_w dW \tag{5.11}$$

where W is the moisture content of the grain on a dry basis, h_w is the isothermal differential heat of wetting of grain, and it is given by:

$$h_w = -23h_{vap} \cdot e^{(-40 \cdot W)} \tag{5.12}$$

Substituting equation (5.12) into equation (5.11), we get:

$$H_W = \int_0^W -23h_{vap} \cdot e^{(-40 \cdot W)} dW$$

$$= -23h_{vap}(-\frac{1}{40})e^{(-40 \cdot W)}\Big|_0^W$$

$$= 0.575h_{vap}(e^{(-40 \cdot W)} - 1) \tag{5.13}$$

By the following, the differential heat of wetting which is obtained from equation (5.13) may be written as:

$$\frac{\partial H_W}{\partial T} = 0.575\frac{\partial h_{vap}}{\partial T}(e^{(-40 \cdot W)} - 1) \tag{5.14}$$

5.3 RATE OF RESPIRATION

When grains respire, the solid substrate is oxidised, carbon dioxide and water are produced along with the heat of chemical reaction. This signifies that the oxidation of carbohydrate liberates thermal energy and together with water vapour, the results may be translated into dry matter loss. The loss of dry matter of the grain is associated with a deterioration in quality.

The formula that enables to calculate the cumulative consumption of oxygen (Q_r) by respiring wheat has been presented by Lacey *et al.* (1994), thus:

$$Q_r = \frac{b_1 + b_2 t_{ime}}{Y(1 + e^{\{(b_5 + b_6 t + b_7 T_{i,j,k})(b_8 + M_w)\}})} \tag{5.15}$$

where t_{ime} is the total elapsed time of storage.

$$Y = 1 + e^{b_3(b_4 - T_{i,j,k})} \tag{5.16}$$

where b_1, b_2, b_3, b_4, b_5, b_6, b_7 and b_8 are the grain specific empirical constants. They are reported by Lacey *et al.* (1994), and have the values given in table 5.2.

Table 5.2 Respiration specific empirical constants

b_1	b_2	b_3	b_4
3.4583E-04	3.478E-08	0.1737	20.33
b_5	b_6	b_7	b_8
0.9143	-2.878E-07	-0.013634	24.38

Therefore the differential equation of the rate of oxygen production is the function of grain moisture content, temperature and time, and is obtained by:

$$\frac{\partial Q_r}{\partial t_{ime}} = \left(\frac{b_1 + b_2 t_{ime}}{Y(1 + e^{\{(b_5 + b_6 t_{ime} + b_7 T_{i,j,k})(b_8 - M_w)\}})}\right)'$$

$$= \frac{(1 + e^{\{(b_5 + b_6 t_{ime} + b_7 T_{i,j,k})(b_8 - M_w)\}})b_2 - (b_1 + b_2 t_{ime})e^{\{(b_5 + b_6 t_{ime} + b_7 T_{i,j,k})(b_8 - M_w)\}}(b_6(b_8 - M_w))}{Y(1 + e^{\{(b_5 + b_6 t_{ime} + b_7 T_{i,j,k})(b_8 - M_w)\}})^2} \tag{5.17}$$

we define

$$\arg = e^{\{(b_5 + b_6 t_{ime} + b_7 T_{i,j,k})(b_8 - M_w)\}} \tag{5.18}$$

$$z_1 = 1 + \arg \tag{5.19}$$

$$z_2 = b_2 \tag{5.20}$$

$$z_3 = b_1 + b_2 t_{ime} \tag{5.21}$$

$$z_4 = b_6 (b_8 - M_w) \tag{5.22}$$

So equation (5.17) becomes:

$$\frac{\partial Q_r}{\partial t_{ime}} = \frac{(z_1 z_2 - z_3 z_4)}{Y z_1^2} \tag{5.23}$$

According to Lacey *et al.*'s (1994) paper, the respiration quotient of the respiring wheat is taken to the unity. From the stoichiometric equation that is governing the oxidation of hexose, it follows that the rate of dry matter loss ($\frac{dm}{dt}$) is determined by:

$$\frac{dm}{dt} = \frac{20}{21} \frac{\partial Q_r}{\partial t_{ime}} \tag{5.24}$$

where $\frac{20}{21}$ is the specific constant for equation (5.24).

5.4 SURFACE TEMPERATURE OF SILO RESULTING FROM SOLAR RADIATION

5.4.1 Introduction

Radiation is emitted from the sun with an energy distribution as if it were a 'black body', at the temperature of $6000K$. Radiation travels with a velocity of $3 \times 10^8 \, m/s$, taking approximately 8 minutes to reach the earth's atmosphere. Some of the radiation entering the atmosphere of earth is scattered by dust, molecules of air and water vapour, and some of them is absorbed by atmosphere ozone, water vapour and carbon dioxide. As the value of solar constant is $1353 W/m^2$, the energy of solar radiation striking the top of the earth's atmosphere, only about 85% can reach the earth's surface.

Solar radiation can be regarded as the only original energy source for living creatures on the earth. From the viewpoint of the aerated grain silo, solar radiation makes an important contribution to reduce the high humid air during the four seasons, and gives rise to a considerable amount of cooling load in the hot weather, especially in tropical regions. The variation of interior temperatures in the aerated grain silos not only very much depend on the amount of incident solar radiation and ambient temperatures on the exterior surface of the silo during the day, but also depends on the heat loss during the night as well.

The estimation of the boundary conditions as the temperatures along the vertical wall (*ts*1) and the sloping floor (*ts*2), and at the bottom cone (*ts*2*) of the silo (see figure 5.1) will be carried out as following:

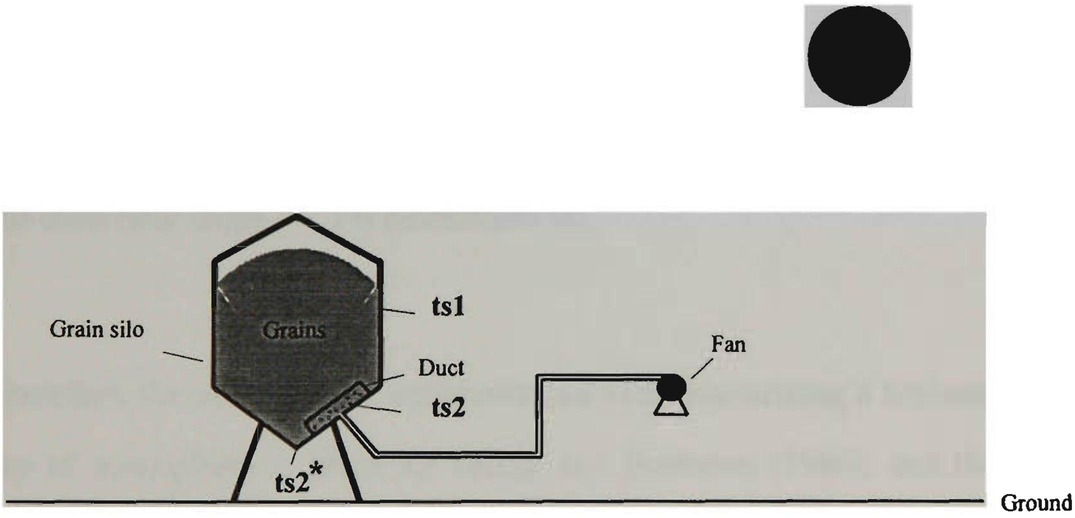


Figure 5.1 Solar radiation and ambient air around the surface of a grain silo

5.4.2 Extraterrestrial solar radiation

Solar radiation is received outside the earth's atmosphere. The earth follows an elliptical path round the sun, and takes one year for each cycle. The earth's axis is tilted at a constant angle of 23°27' relative to the plane of rotation at all times. The daily motion of the sun across the sky viewed from any particular location on earth varies cyclically throughout the year, it is defined by the angle of declination. Cooper (1969) has derived a approximate equation for determining the angle of declination (δ_d) as:

$$\delta_d = 23.45\sin(\frac{360(284+n)}{n}) \tag{5.25}$$

where n is the day of the year.

The variation of the angle of declination will affect the angle of incidence of the solar radiation on the earth's surface, and cause seasonal variations in the length of the day. Duffie and Beckman (1980) have presented the equation for calculation the length of the day (N), it can be written as:

$$N = \frac{2}{15}\cos^{-1}(-\tan\phi_l \tan\delta_d) \tag{5.26}$$

where ϕ_l is the latitude of the geographical location under consideration.

The sunset hour angle (ω_s) is also given by:

$$\omega_s = \cos^{-1}(-\tan\phi_l \tan\delta_d) \tag{5.27}$$

and the solar hour angle (ω_a) is determined as:

$$\omega_a = 15t_h \tag{5.28}$$

Therefore, the average daily extraterrestrial radiation striking a horizontal surface at the top of atmosphere is given by Duffie and Beckman (1980), and the equation is written as:

$$\bar{H}_0 = \frac{0.0864G_{sc}}{\pi}\{1+0.033\cos(\frac{360n}{365})\}\{\cos\phi_l \cos\delta \sin\omega_s + \frac{\pi\omega_s}{180}\sin\phi_l \sin\delta_d\} \tag{5.29}$$

where G_{sc} is the solar constant.

5.4.3 Mean intensities of solar radiation

The total solar radiation reaching the ground consists of both beam and diffuse radiation components. A high accuracy method for calculating the average of daily diffuse radiation ($\bar{H}_d$) is reported by Page (1961) and recommend by Ma and Iqbal (1984). It may be written as:

$$\bar{H}_d = \bar{H}_t\{(1-1.13)\frac{\bar{H}_t}{\bar{H}_0}\} \tag{5.30}$$

where $\bar{H}_t$ is the average of daily total radiation falling on a horizontal surface given by Rietveld (1978), thus:

$$\bar{H}_t = H_0\left\{0.18+\bar{0}.62(\frac{n_s}{N})\right\} \tag{5.31}$$

where n_s is the average of sunshine hours per day.

So the total average of daily beam radiation ($\bar{H}_b$) is readily obtained by the equation as following:

$$\bar{H}_b = \bar{H}_t - \bar{H}_d \tag{5.32}$$

5.4.4 Intensities of solar radiation

According to Collares-Pereira and Rabl's (1979) presentation, the total intensity of radiation (I_t) on a horizontal surface is given as:

$$I_t = \frac{\pi}{24}\{a+b\cos\omega_a\}\left\{\frac{\cos\omega_a-\cos\omega_s}{\sin\omega_s-\omega_s\cos\omega_s}\right\}\frac{\bar{H}_t}{3600} \tag{5.33}$$

where

$$a = 0.409+0.5016\sin(\omega_s-60t_s) \tag{5.34}$$

$$b = 0.6609-0.4767\sin(\omega_s-60t_s) \tag{5.35}$$

and

$$t_s = \frac{\pi}{180} \tag{5.36}$$

Liu and Jordan (1960) have presented the expression of the total intensity of diffuse radiation (I_d) falling on a horizontal surface, thus:

$$I_d = \frac{\pi}{24}\left\{\frac{\cos\omega_a - \cos\omega_s}{24(\sin\omega_s - \omega_s\cos\omega_s)}\right\}\frac{\bar{H}_d}{3600} \tag{5.37}$$

By following, the total intensity of beam radiation (I_b) is obtained by:

$$I_b = I_t - I_d \tag{5.38}$$

5.4.5 Ratios of solar radiation

In practice, on the top, bottom and wall of the silo surfaces are usually at angle other than horizontal. The method for definition the ratio (R_t) of the total intensity of radiation falling on a tilted surface (I_{tT}) to the total intensity of radiation falling on a horizontal surface (I_t) is written as:

$$R_t = \frac{I_{tT}}{I_t} \tag{5.39}$$

Similarly, the ratio (R_d) of the total intensity of the diffuse radiation (I_{dT}) falling on a sloping surface to the total intensity of diffuse radiation falling on a horizontal surface (I_d) is given by:

$$R_d = \frac{I_{dT}}{I_d} \tag{5.40}$$

The ratio (R_b) of the total intensity of beam radiation falling on a inclined surface to the total intensity of beam radiation falling on a horizontal surface is obtained as:

$$R_b = \frac{I_{bT}}{I_b} \tag{5.41}$$

Because the total intensity of radiation is sum of the total intensity of diffuse radiation and the total intensity of beam radiation, so we have:

$$I_{tT} = I_{bT} + I_{dT} \tag{5.42}$$

Replacing equations (5.39), (5.40) and (5.41) into equation(5.42), we get:

$$R_t I_t = R_b I_b + R_d I_d = I_{tT} \tag{5.43}$$

we obtain

$$R_t = \frac{R_b I_b + R_d I_d}{I_t} \tag{5.44}$$

Duffie and Beckman (1980) have recommend a procedure for calculating ratio R_t, as following:

When the surface is inclined at an angle to the horizontal, some of the diffuse radiation comes from sky and some of it reaches the surface by reflection from the ground. The value of ratio R_d is:

$$R_d = \frac{1+\cos\beta}{2} \tag{5.45}$$

where β is the angle between a flat surface and the horizontal plane. The value of β is obtained by:

$$\beta = \begin{cases} \dfrac{\pi}{2} & \textit{on the vertical wall} \\ \pi - \gamma_1 & \textit{on the sloping floor} \end{cases} \tag{5.46}$$

A view factor of the ground of the tilted surface is:

$$V_e = \frac{1-\cos\beta}{2} \tag{5.47}$$

The fraction of the total radiation hitting the ground then the total solar radiation hitting the sloping surface is:

$$g_s = (I_b + I_d)\rho \cdot V_e \tag{5.48}$$

where ρ is the reflected fraction and equation (5.44) becomes:

$$R_t = \frac{I_b R_b + I_d(\frac{1+\cos\beta}{2}) + g_s}{I_t}$$

$$= \frac{I_b R_b + I_d(\frac{1+\cos\beta}{2})}{I_t} + \rho(\frac{1-\cos\beta}{2}) \tag{5.49}$$

The total intensity of radiation on the tilted surface is expressed as:

$$I_{tT} = I_b R_b + I_d(\frac{1+\cos\beta}{2}) + (I_b + I_d)\rho(\frac{1-\cos\beta}{2}) \qquad (5.50)$$

The ratio of beam radiation (R_b) on a tilted surface to a horizontal surface is:

$$R_b = \frac{\cos\theta}{\cos\theta_z} \qquad (5.51)$$

where θ_z is the zenith angle, it is between the sun and a vertical wall of the silo through the point of interest, θ is the angle of incidence of the beam radiation, and between the beam and the normal to the plane. It is given by the expression as:

$$\cos\theta = \sin\delta\sin\phi_l\cos\beta - \sin\delta\cos\phi_l\sin\beta\cos\gamma + \cos\delta\cos\phi_l\cos\beta\cos\omega_a$$
$$+\cos\delta\sin\phi_l\sin\beta\cos\gamma\cos\omega_a + \cos\delta\sin\beta\sin\gamma\sin\omega_a \qquad (5.52)$$

where γ is the surface azimuth angle, it is the angler deviation from a line running north and south through the location of the silo, and the value of γ is calculated by:

$$\gamma = -\frac{\pi}{2} + 2\pi\frac{j-1}{nphi} \qquad (5.53)$$

The zenith angle (θ_z) is also determined by :

$$\cos\theta_z = \cos\delta\cos\phi_l\cos\omega_a + \sin\delta\sin\phi_l \qquad (5.54)$$

5.4.6 Surface temperatures of the silo

The temperatures of the surface of the silo are obtained by carrying out a numerical heat balance scheme and assuming the total heat fluxes into the surface of the silo is zero. This can be written in words as:

Beam Radiation Flux + Diffuse Radiation Flux + Solar Radiation Flux Reflected from the Ground + Radiation Flux to the Sky + Radiation Flux to the Ground + Convective Heat Loss through Convection = 0

(5.55)

Swinbank (1963) has presented the following relationship between the ambient temperature (T_a) to the sky temperature (T_{sky}), thus:

$$T_{sky} = 0.05522 T_a^{1.5} \tag{5.56}$$

The heat exchange between the surface of the silo to the sky is given by:

$$q_s = \sigma \cdot \varepsilon (T_{sky}^4 - T_s^4)(\frac{1+\cos\beta}{2}) \tag{5.57}$$

where σ is the Stefan-Boltzman constant. ε is the emissivity of the external surface of the grain silo.

The exchange of radiation between the ground temperature (T_g) to the surface temperature of the silo (T_s) is written as:

$$q_g = \sigma \cdot \varepsilon (T_g^4 - T_s^4)(\frac{1-\cos\beta}{2}) \tag{5.58}$$

The convective heat transfer coefficient is given by:

$$h_0 = 2.8 + 3V_{wind} \tag{5.59}$$

where V_{wind} is the wind velocity across the surface of the silo.

So equation (5.55) in algebraic form is written as:

$$\alpha I_b R_b + \alpha I_d (\frac{1+\cos\beta}{2}) + \alpha (I_b + I_d)\rho(\frac{1-\cos\beta}{2})$$

$$+q_g + q_s + h_0 (T_a - T_s) = 0 \tag{5.60}$$

Re-arranging equation (5.60), we get:

$$\alpha I_b R_b + \alpha I_d (\frac{1+\cos\beta}{2}) + \alpha (I_b + I_d)\rho(\frac{1-\cos\beta}{2}) + \sigma \cdot \varepsilon (T_{sky}^4 - T_s^4)$$

$$(\frac{1+\cos\beta}{2}) + \sigma \cdot \varepsilon (T_g^4 - T_s^4)(\frac{1-\cos\beta}{2}) + (2.8 + 3V_{wind})(T_a - T_s) = 0 \tag{5.61}$$

Equation (5.61) is solved by applying Newton-Raphson method (see Appendix II), we define:

$$f(T_s) = \alpha I_b R_b + \alpha I_d (\frac{1+\cos\beta}{2}) + \alpha (I_b + I_d)\rho(\frac{1-\cos\beta}{2})$$

$$+\sigma\cdot\varepsilon(T_{sky}^4 - T_s^4)(\frac{1+\cos\beta}{2}) + \sigma\cdot\varepsilon(T_g^4 - T_s^4)(\frac{1-\cos\beta}{2})$$

$$+(2.8 + 3V_{wind})(T_a - T_s) \quad (5.62)$$

and

$$T_s^{P+1} = T_s^P - \frac{f(T_s^P)}{f'(T_s^P)} \quad (5.63)$$

where T_s^P is the previous surface temperature at the commencement of the iteration step P, hence we have:

$$f'(T_s^P) = -4\sigma\varepsilon T_s^3 (\frac{1+\cos\beta}{2}) - 4\sigma\varepsilon T_s^3 (\frac{1-\cos\beta}{2}) - h_0 \quad (5.64)$$

So equation (5.63) sets the stage for using the Newton-Raphson method to solved the updated value of the surface temperatures of the silo (T_s), such as:

$$T_s = T_s^{P+1} \quad (5.65)$$

We note that the temperatures varies with radial position, may cause the temperatures on the vertical wall and sloping flow of the silo different, so the surface temperature (T_s) can be defined as following:

At the vertical wall of the silo, T_s is in the three components space $(n\xi, j, k)$, we obtain:

$$T_{s-(n\xi,j,k)} = ts1 \quad (5.66)$$

Along the sloping floor of the silo, T_s is in the space $(i, j, 1)$, we have:

$$T_{s-(i,j,1)} = ts2 \quad (5.67)$$

At the bottom of the cone of the silo, T_s is in the space $(1, j, 1)$, we get:

$$T_{s-(1,j,1)} = ts2^* \quad (5.68)$$

5.5 VIABILITY OF STORED GRAINS

Seeds of different plant species vary widely in their life spans under identical storage conditions. If seed is alive or dead it dose not influence its food value, but it is inevitable that grains spoilage will always be accompanied by loss of germination. A fundamental difference exists between seeds stored as seeds, and seeds stored as food, but the essential requirement for successful storage is to maintain a high viability of stored grains.

The loss of viability of grain seeds is related to the temperature and wet basis moisture content within the grain store, and both of them vary with time and space. Roberts (1960) has presented an expression for determining the half-life of the seeds, and it may be written as:

$$H_{life} = 10^{(v_1+v_2M_w+v_3T_{i,j,k})} \tag{5.69}$$

where v_1, v_2 and v_3 are seed-specific constants, M_w is wet basis of moisture content, it is defined by equation (5.1).

The standard deviation of the mortality of grain seeds is given by:

$$\sigma = v_4 H_{life} \tag{5.70}$$

where v_4 is also a seed specific constant.

Moreover, the fraction of the seeds dead is determined by:

$$d_{ead} = 1 - V_{iab} \tag{5.71}$$

where V_{iab} is the fractional viability of seeds.

In general, we consider that the initial viability of grain seeds is 99%, that is:

$$d_{ead} = 1 - 0.99 = 0.01 \tag{5.72}$$

So the time required the half of the seeds to be dead (when viability of the grain is 99%) is determined as following:

If $d_{ead} \geq 0.5$, then

$$d_{ead1} = 0.5 + 0.5 \cdot erf(x) \tag{5.73}$$

where the $erf(x)$ is the error function, it is given by:

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{(-x^2)} d(x) \tag{5.74}$$

in which, if $x \leq 1$, the error function is solved by series expansion, thus:

$$erf(x) = x + \frac{2}{\sqrt{\pi}} \left(\sum_{n=2}^{\infty} \frac{(-1)^{n-1} (x)^{2n-1}}{(2n-1)n!} \right) \tag{5.75}$$

and if $x > 1$, the error function is evaluated by Simpson's rule, namely:

$$erf(x) = \int_0^x e^{(-x^2)} d(x)$$

$$= \int_0^1 e^{(-x^2)} d(x) + \int_1^x e^{(-x^2)} d(x)$$

$$= 0.8427008 + \int_1^x e^{(-x^2)} d(x) \tag{5.76}$$

and the first term $\int_0^1 e^{(-x^2)} d(x) = 0.8427008$ is found from a table of the error function, and the second term $\int_1^x e^{(-x^2)} d(x)$ is solved by Simpson's rule.

So we have (when $d_{ead} \geq 0.5$):

$$func_1 = d_{ead1} - \{0.5 + 0.5 \cdot erf(x)\} \tag{5.77}$$

and the Newton-Raphson method (see Appendix II) is used for determining x, thus:

$$x^{P+1} = x^P + \frac{func_1}{0.5 \cdot derf(x)} \tag{5.78}$$

where x^P is the previous error function at the commencement of the iteration step P, and $derf(x)$ is the first derivative of $erf(x)$ from equation (5.74), we get:

$$derf(x) = \frac{2}{\sqrt{\pi}} e^{(-x^2)} d(x) \tag{5.79}$$

Update the value x from equation (5.78), we have:

$$x_1 = x^{P+1} \tag{5.80}$$

and we know

$$x = \frac{|t_v - H_{life}|}{\sigma\sqrt{2}} \tag{5.81}$$

So when viability of seed is 99% and $d_{ead} \geq 0.5$, the value of t_v can be obtained from equation (5.81) that is:

$$t_{v1} = x_1 \cdot \sigma\sqrt{2} + H_{life} \tag{5.82}$$

If $d_{ead} \leq 0.5$, then

$$d_{ead2} = 0.5 - 0.5 \cdot erf(x) \tag{5.83}$$

$$func_2 = d_{ead2} - \{0.5 - 0.5 \cdot erf(x)\} \tag{5.84}$$

The Newton-Raphson method (see Appendix II) for determining x is:

$$x^{P+1} = x^P - \frac{func_2}{0.5 \cdot derf(x)} \tag{5.85}$$

Update the value x from equation (5.85), it is:

$$x_2 = x^{P+1} \tag{5.86}$$

Re-arrange equation (5.81) again, we have:

$$t_{v2} = H_{life} - x_2 \cdot \sigma\sqrt{2} \tag{5.87}$$

where t_{v2} is the value of when $d_{ead} \leq 0.5$ and the viability of grain seeds is 99%.

Then the total elapsed time ($time^*$) of heat and mass transfer iteration required the time of half seeds to be dead is calculated as:

$$t_{v1-new} = t_{v1} + time^* \tag{5.88}$$

$$t_{v2-new} = t_{v2} + time^* \tag{5.89}$$

where

$$time^* = \frac{dtg \cdot ifreq}{3600 \times 24} \tag{5.90}$$

in which *ifreq* is the number of heat and mass transfer iterations, because the viability of seed changes very slowly compared with other changes, and since the computation of seed viability is numerically intensive, so it is calculated infrequently. The number of the seeds dead are obtained by:

When $t_{v1-new} > H_{life}$, that:

$$d_{ead1-new} = 0.5 + 0.5 \cdot erf(x_{new1}) \tag{5.91}$$

where

$$x_{new1} = \frac{\left| t_{v1-new} - H_{life} \right|}{\sigma\sqrt{2}} \tag{5.92}$$

so the viability of grain seeds is:

$$V_{iab1} = 1 - d_{ead1-new} \tag{5.93}$$

When $t_{v2-new} < H_{life}$, that:

$$d_{ead2-new} = 0.5 - 0.5 \cdot erf(x_{new2}) \tag{5.94}$$

which

$$x_{new2} = \frac{\left| t_{v2-new} - H_{life} \right|}{\sigma\sqrt{2}} \tag{5.95}$$

Again the viability of grain seeds is:

$$V_{iab2} = 1 - d_{ead2-new} \tag{5.96}$$

5.6 PESTICIDE DECAY

Pesticides may be effective against the insect pests for many months, and they are often admixed with grains to resist the attack by insect pests. When pesticides are applied to warm and damp grains, they can decompose to ineffectively low levels only in few weeks. The pesticides studied in this thesis are not labile, hence we need not study their movement within the grain store. The kinetics of pesticide decay has been presented by Desmarchelier and Bengston (1979) and Thorpe (1989), and it can be shown that the concentration of pesticide on the grain as a function of the relative humidity of the intergranular air (rh), grain temperature ($T_{i,j,k}$) and total elapsed time (t_{ime}) of storage. It can be written in algebraic form as:

$$C_{pes} = C_i e^{(-1.3863(rh)t_{ime} \times 10^{B(T_{i,j,k}-30)/t^{*}})} \tag{5.97}$$

where C_i is the initial concentration of the pesticide, B and t^{*} are the pesticide specific empirical constants.

The relative humidity of the intergranular air is obtained by:

$$rh = \frac{P_{vap}}{P_{sat}} \tag{5.98}$$

where

$$P_{vap} = \frac{P_{atm} H_{i,j,k}}{gn + H_{i,j,k}} \tag{5.99}$$

and P_{sat} is calculated by Hunter's (1987) equation as same as equation (5.4), namely:

$$P_{sat} = \frac{6 \times 10^{25}}{(T_{i,j,k} + 273)^5} \cdot e^{\frac{-6800}{T_{i,j,k}+273}} \tag{5.100}$$

Desmarchelier and Bengston (1979) have determined that the values of B and t^{*} for fenitrothion are those given in table 5.3, thus:

Table 5.3 Pesticide specific empirical constants

B	t*
0.036	8.467E06

5.7 INSECT POPULATION GROWTH

Stored products insect pests are ubiquitous and they are detected in storage or during transport. They are spread throughout the world. They easily adapt themselves to a new environment, and readily become established. Stored grain damage by insect pests attack will cause enormous quantitative and qualitative losses.

The growth of insect pests are dependent on the moisture content and temperature of the grains. Desmarchelier (1988) reported the intrinsic weekly multiplication rate of the insect population growth is calculated by equation as:

$$r_s = c_{c1}(T_{wetb} - c_{c2}) \qquad (5.101)$$

where c_{c1} and c_{c2} are species specific constants, T_{wetb} is the wet bulb temperature of the intergranular air, which is a function of the saturation humidity of air (H^*), absolute humidity ($H_{i,j,k}$) and dry bulb temperature (T_{dryb}), and it expressed as:

$$(T_{dryb} - T_{wetb}) + H_{i,j,k}(2502 + 1.809T_{dryb} - 4.186T_{wetb})$$
$$- H^*(2502 - 2.377T_{wetb}) = 0 \qquad (5.102)$$

in which the dry bulb temperature (T_{dryb}) is determined by Smith's (1947) isosteric, it is expressed as:

$$T_{dryb} = \frac{M_w - a_1 + a_3 \cdot \ln(1 - rh)}{a_2 - a_4 \cdot \ln(1 - rh)} \qquad (5.103)$$

and the saturation humidity of air is function of wet bulb temperature. it is obtained as:

$$H^* = \frac{gn \cdot p_s}{P_{atm} - p_s} \qquad (5.104)$$

where

$$p_s = \frac{6 \times 10^{25}}{(T_{wetb} + 273)^5} \cdot e^{\frac{-6800}{T_{wetb} + 273}} \qquad (5.105)$$

The wet bulb temperature T_{wetb} is obtained by solving equation (5.102), by employing the Newton-Raphson method (see Appendix II), we define:

$$f(T_{wetb}) = (T_{dryb} - T_{wetb}) + H_{i,j,k}(2502 + 1.809T_{dryb} - 4.186T_{wetb})$$
$$-H^*(2502 - 2.377T_{wetb}) \tag{5.106}$$

and

$$T_{wetb}^{P+1} = T_{wetb}^{P} - \frac{f(T_{wetb}^{P})}{f'(T_{wetb}^{P})} \tag{5.107}$$

where T_{wetb}^{P} is the previous wet bulb temperature at the commencement of the iteration step P, and the first derivative of T_{wetb} is:

$$f'(T_{wetb}) = -1 - 4.186H_{i,j,k} + 2.377H^* - 2502\frac{dH^*}{dT_{wetb}}$$
$$+2.377T_{wetb} \cdot \frac{dH^*}{dT_{wetb}} \tag{5.108}$$

By the chain rule of the differentiation, we get:

$$\frac{dH^*}{dT_{wetb}} = \frac{dH^*}{dp_s}\frac{dp_s}{dT_{wetb}} \tag{5.109}$$

So from equation (5.104), we have:

$$\frac{dH^*}{dp_s} = \left(\frac{gn \cdot p_s}{P_{atm} - p_s}\right)'$$
$$= \frac{gn \cdot P_{atm}}{(P_{atm} - p_s)^2} \tag{5.110}$$

and from equation (5.105), we get:

$$\frac{dp_s}{dT_{wetb}} = \left(\frac{6 \times 10^{25}}{(T_{wetb} + 273)^5} \cdot e^{\frac{-6800}{T_{wetb}+273}}\right)'$$

$$= \frac{6 \times 10^{25}}{(T_{wetb} + 273)^5} \cdot e^{\frac{-6800}{T_{wetb}+273}} \cdot \left(\frac{6800}{T_{wetb} + 273} - 5\right) \tag{5.111}$$

Substituting equations (5.110) and (5.111) into equation (5.109), we will obtain:

$$\frac{dH^*}{dT_{wetb}} = \frac{gn \cdot P_{atm}}{(P_{atm} - p_s)^2} \cdot \frac{6 \times 10^{25}}{(T_{wetb} + 273)^5} \cdot e^{\frac{-6800}{T_{wetb}+273}} \cdot \left(\frac{6800}{T_{wetb} + 273} - 5 \right) \quad (5.112)$$

So equation (5.107) sets the stage for the Newton-Raphson method (see Appendix II) is to easily find the updated value of the wet bulb temperature (T_{wetb}), thus:

$$T_{wetb} = T_{wetb}^{P+1} \quad (5.113)$$

Then the population of insect pests after *dtg* weeks may be obtained by :

$$R^{P+1} = R^P \cdot e^{r_s \cdot dt_{week}} \quad (5.114)$$

where R^P is the previous population of insect pests at the commencement of the time step *P*. dt_{week} is the end of time step that is calculated by weeks, it is written as:

$$dt_{week} = \frac{dtg}{3600 \times 7 \times 24} \quad (5.115)$$

where *dtg* is the time step used during the aeration cycle.

CHAPTER 6

NUMERICAL EXPERIMENTS

Since computational methods have become an important tool for scientific investigations, the solution of finite difference scheme has been realized to numerically solve the problems in an efficient manner. In this chapter, the numerical experiments implicate a study of factors that have significant influence on the effect of convergence and accuracy of the numerical results, they include grid mesh size, iteration time step and aeration duct layouts (The discussion is included in chapter 7).

6.1 PROGRAM EXPLANATION

The program SILO3DFTN is largely based on the software CBHMT developed, to calculate the response of bulk respiring grains contained in a conical bottomed circular silo to aeration with ambient on conditioned air, written by Thorpe (1994). The main modifications were in aspects of program control and in the inclusion of variation of duct positioning. The program was also modified to accommodate specific characteristic of the conical bottomed circular silo, such as located at the Academy of Grain Technology at Werribee.

Program SILO3DFTN has provided three major categories of outputs, namely: (a). pressure and velocity fields, (b). temperature and moisture content fields, and (c) fields of biological activity and the rate of pesticide decay. The procedures of this FORTRAN program is illustrated in flow charts as shown in figures 6.1 and 6.2, and the program instruction is explained as following.

To start, subroutines INPUT, SETCON, PROPS and INITIAL are all input for setting up the initial conditions, system properties and parameters. Subroutine SURTEMP calculates the surface temperature of grain silo subjected to heating and cooling by radiation and convection. Subroutines PRESSURE and VELOCITY predict the pressure and velocity distributions by replacing the spatial derivative for the governing equations. Subroutine ITERP estimates the approach to responsible the convergence of the pressure distribution. Subroutine FLOWRATE calculates the air flow rate through the upper surface of the grain silo.

The inner loop of HMTR is a system control of aeration and the second major loop for heat and mass transfer calculations. Subroutine HUMSM obtains the absolute humidity of air from the temperature and moisture content of the grain. Subroutine RESPL determines the rate of respiration of wheat as the dry matter loss of grain. Subroutine MOISTU and TEMPERA solve moisture conservation and thermal energy equations. Subroutine TBC, HUMBC and MCBC calculate the boundary conditions of temperature, humidity and moisture content in stored grain. Subroutine WETBULB uses the dry bulb temperature of air to calculate the wet bulb temperature of air. Subroutine INSECTS obtains the insect population growth in the silo, and subroutine VIABLE calculates the updated viability of the seeds after a time, and the subroutine PESTI calculates the concentrations of pesticide on the stored grain.

When the inner loop HMTR is equal to the maximum iteration time, the inner loop will be returned to the main program, and the final results will be obtained by calling subroutine RESULTS, then program stop.

In addition, program SILO3DFTN is shown in Appendix III.

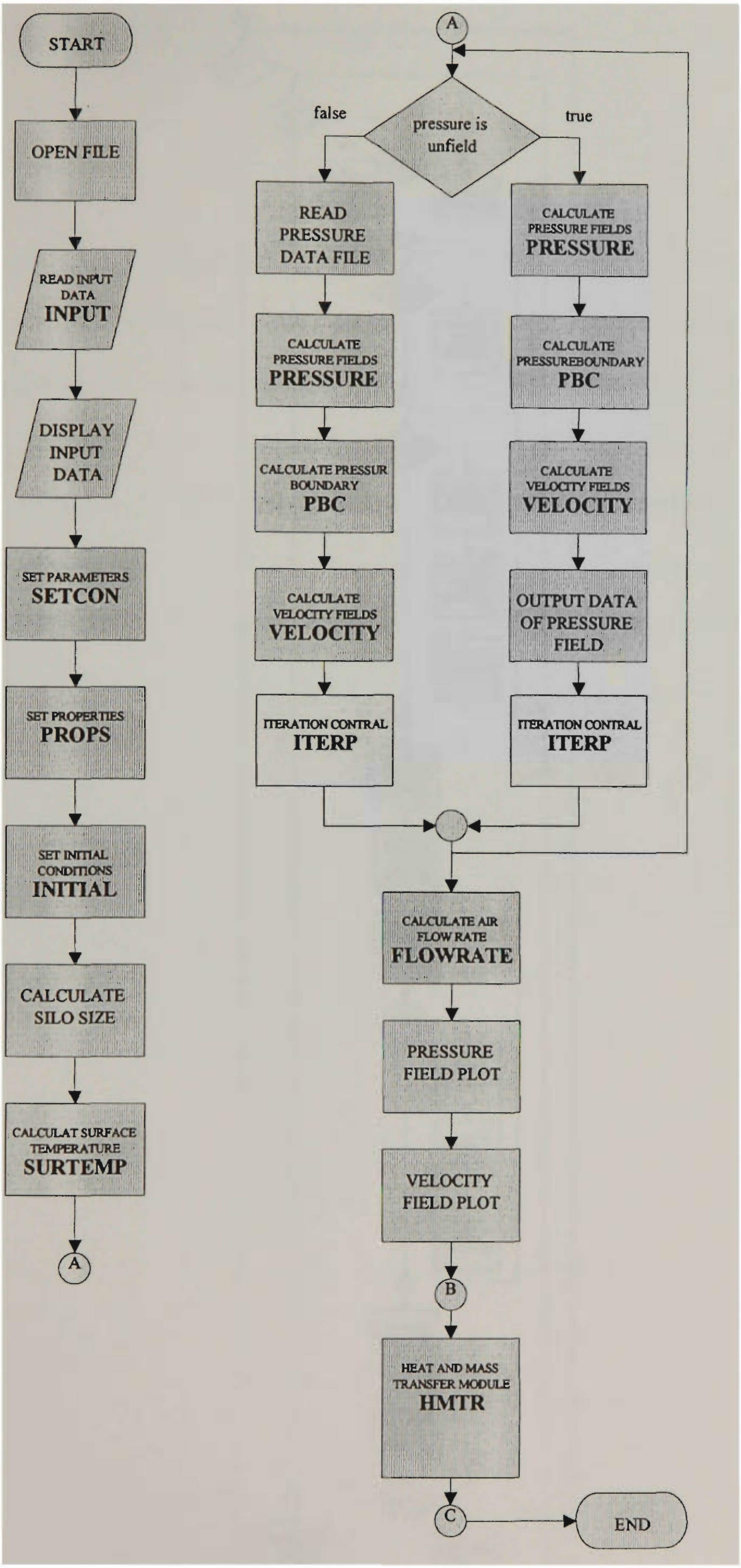


Figure 6.1 The main loop of program SILO3DFTN

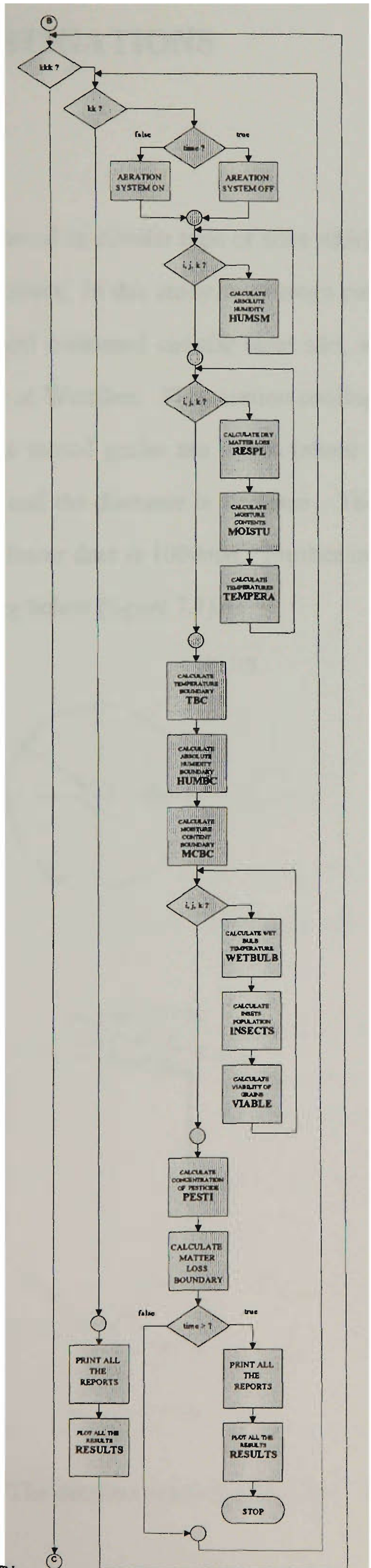


Figure 6.2 The inner loop of program SILO3DFTN

6.2 SYSTEM INVESTIGATIONS

6.2.1 The silo

Traditionally most grains are stored in circular type of silos which provide a convenient mean for handling and management. In this study, the design parameters of the silo are measured on a prototype conical bottomed circular steel silo, which is located at the Academy of Grain Technology at Werribee. The aeration cooling system is consist of a fan and a duct operation. The stored grains are wheat (about 10 tonnes). The total height of the silo is $4220mm$, and the diameter is $1230mm$. The volume of the silo is $11.9675m^3$. The length of the linear duct is $1000mm$. Further information of this grain silo can be found in the drawing below (figure 7.1).

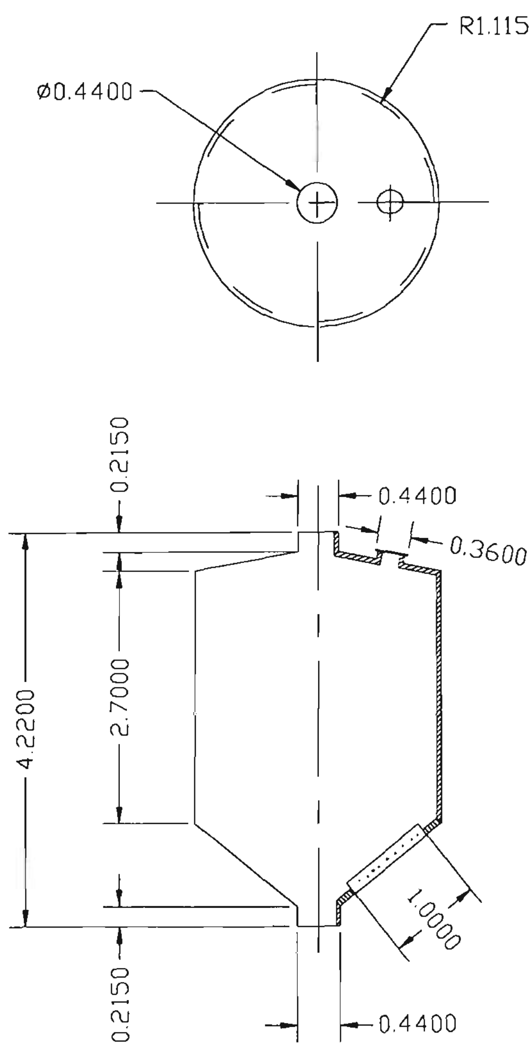
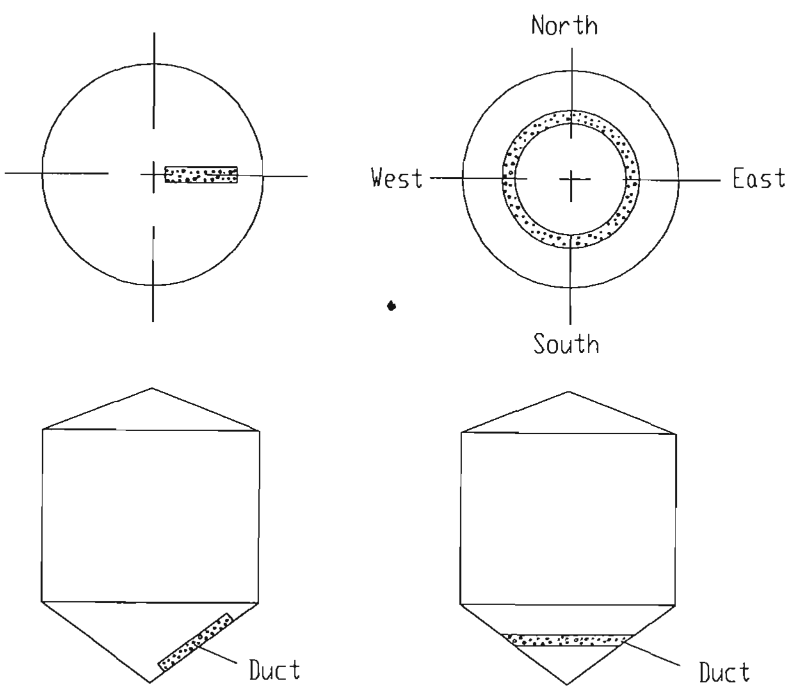


Figure 6.3 The detailed drawing of the silo

6.2.2 The ducts

In the numerical experiments, two type of aeration systems were simulated. In one case, a conventional linear aeration duct was used, and in the other case, the effect on cooling processes of an annular duct was investigated (see figure 6.4). The details of this information are included in table 6.1.



(a) linear duct (b) annular duct

Figure 6.4 The two types of ducts: linear and annular ducts

Table 6.1 The character of linear and annular ducts

Type	Length of duct	Configuration
Linear duct	1000 mm	Laid on the sloping base of the silo
Annular duct	2170 mm (Perimeter)	Placing on the conical base of the silo

The simulation results of linear and annular aeration systems are implemented by applying program SILO3DFTN, as shown in chapter 7.

6.2.3 Further investigations

Aeration system must be carefully controlled to achieve minimum grain temperature with a minimum of moisture increase. Excessive fan operation with humid air will cause moisture uptake by the grain near the perforated duct, and may result in bulging on the silo walls. The following variables were used in simulating aeration systems of linear and annular ducts. Consideration low capital costs of the electric power, the aeration fan is operated between midnight to 6am each day, and the airflow rate through the silo of grain is $1.47 l/s/tonne$. The duration of aeration cooling are 60 days. Further details are listed in table 6.2.

Table 6.2 Physical data of aeration system

Iinital grain temperature	30 °C
Initial grain moisture content (wet basis)	12%
Initial grain moisture content (dry basis)	13.64%
Airflow rate through aeration	1.47 *l/s/tonne*
Aeration fan turned on	0 hours
Aeration fan turned off	6 hours
Duration of aeration	60 days
Mean minimum daily temperature	14.2 °C
Mean maximum daily temperature	24.3 °C
Mean daily sunshine hours	8.1 hours / per day
Mean absolute ambient humidity	0.008 kg / kg
Latitute	-37.67
Absorptivity of silo material	0.8
Emissivity of silo material	0.9
Silo of volum	11.9675 cubic meter
Angle of conical corn to horizontal	38.67 °
Angle of silo roof to horizontal	10.17 °
Mean ambient temperature	15.973 °C
Mean ambient wet-bulb temperature	12.997 °C
Humidity of air entering grain	0.005 kg/kg
Months	January-February

6.3 CONVERGENCE OF ITERATION SCHEME

Convergence criterion is the expression used to indicate that the approximate computed solution of partial differential equation approaches to the exact solution of the finite difference problem as the grid spacing in time and distance tend to zero (Carnahan *et al.*, 1969). In this study, equation (4.48) of Laplace equation in transformed coordinates is solved by adding a false transient term $(\frac{1}{\alpha_P})$ as proposed by Mallinson and Vahl Davis (1973), the solution procedure corresponding to equation (4.48) was iterated to a steady state, until the following convergence criterion which is called $Psi\max$ was satisfied as:

$$\frac{\sum_i \sum_j \sum_k \left|\psi^{n+1} - \psi^n\right|}{\sum_i \sum_j \sum_k \left|\psi^n\right|} \leq Psi\max \tag{6.1}$$

Here the superscripts $(n+1)$ and n represent the $(n+1)^{th}$ and $(n)^{th}$ iterations respectively. The indices *i, j,* and *k* represent grid location in the space (ξ, ϕ, η).

Table 6.3 shows the first inner loop of program SILO3DFTN calculating the pressure distribution at the centre of the silo (as the values of *Pcon-min*), to represent the finite difference scheme consistent with the given partial differential equation. The *Pcon-min* computational results are obtained by employing Implicit method with the three different types of grid mesh. Moreover, the best advice for decision the right size of grid mesh in the computer simulation will be provide.

Observing has been made, as iter-number increase, the values of *Pcon-min* can be tended to similarity, that means the convergence of the solution is appeared. The worthy of note is: when a non-uniform mesh of 11x10x11 grid is used, and the values of *Pcon-min* are changed from *Psimax*=0.00001 to *Psimax*=0.000001, the accuracy of the pressure distribution can be improved by 1%. So *Psimax*=0.00001 is small enough to obtain the reasonable accurate results for the pressure distribution.

When non-uniform grid mesh 21x20x21 is used, from *Psimax*=0.000001 to *Psimx*=0.0000001, the difference between values *Pcon-min* is 0.3% increase, but the iteration time step is 1.5 times greats, so it can be seen when *Psimax* is less than 0.000001, the solution of pressure will converge. It is also clear when the mesh size is 31x30x31, from iter-number=90159, to iter-number=160000, *Psimax* is still in the region of $<10^{-7}$ and $>10^{-8}$, the solution for the pressure field improves very slowly.

Table 6.3 Convergence effect of grid mesh

Mesh size	Time step	*Psimax*	Iter number	*Pcon-min*
11x10x11	0.5	0.01	47	37.6242
11x10x11	0.5	0.001	552	181.0662
11x10x11	0.5	0.0001	2315	343.1566
11x10x11	0.5	0.00001	4726	377.9526
11x10x11	0.5	0.000001	7206	381.7996
11x10x11	0.5	0.0000001	9633	382.1920
11x10x11	0.5	0.00000001	10724	382.2184
21x20x21	0.5	0.01	56	15.7051
21x20x21	0.5	0.001	612	87.9188
21x20x21	0.5	0.0001	3918	260.3447
21x20x21	0.5	0.00001	9947	331.7149
21x20x21	0.5	0.000001	17081	339.9291
21x20x21	0.5	0.0000001	25369	340.9936
21x20x21	0.5	0.00000001	33346	341.1255
31x30x31	0.5	0.01	72	2.7542
31x30x31	0.5	0.001	559	24.7522
31x30x31	0.5	0.0001	5160	103.8654
31x30x31	0.5	0.00001	22373	184.1912
31x30x31	0.5	0.000001	48908	204.1521
31x30x31	0.5	0.0000001	90159	206.6867
31x30x31	0.5	0.0000001	160000	207.1201
				It hasn't converged yet

Theoretically, the small mesh size, such as 31x30x31 can increase the accuracy of the results, but it will cause the program to take very long time to run. (In Apollo workstation, I use 12 days to run the results, but still can not get the solution of convergence). From the view of economy, this is unsuitable choice. Oppositely, the mesh size 11x10x11 can rapidly reach convergence, (in Apollo workstation, I can get the results of convergence in 10 hours), but the accuracy of the results is lower.

Clearly, mesh size 21x20x21 can achieve the good balance on the accuracy and economic time of the computational results to reach the convergence of the numerical solution. (I can get the results of convergence in 3 days at Apollo workstation).

At the following chapter 7, the simulation results are employed at the mesh size of 21x20x21 in the cause of non-aerated silo, and the silos fitted with linear and annular ducts.

Figure 6.5 shows the convergence of pressure distribution at the centre of the silo by grid mesh 21x20x21, on the accurate of numerical solution to reach the steady state with different iteration time step. It is clear when iteration number is over 15000, the pressure distribution is convergent.

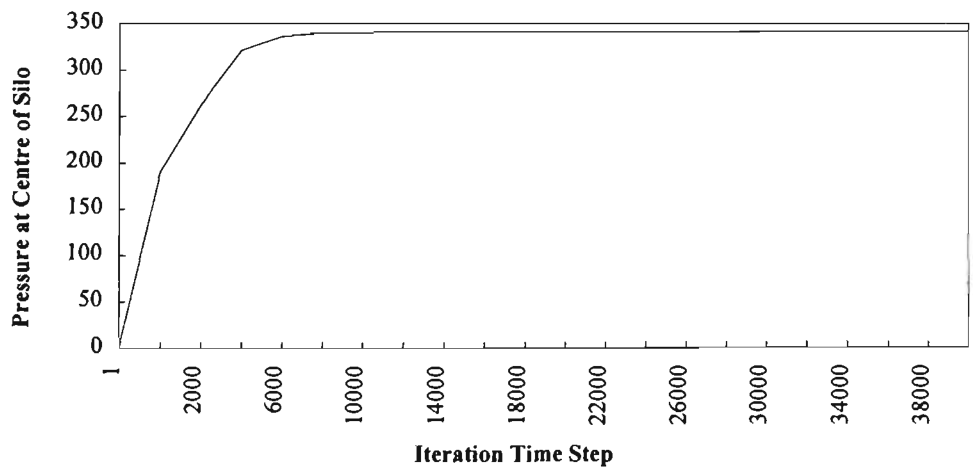


Figure 6.5 Convergence of pressure distribution

As Patankar (1980) noted, an iterative process is said to have converged when further iterations will not produce any change in the values of the dependent variables.

In practice, the iterative process is terminated when some arbitrary convergence criterion is satisfied. An appropriate convergence criterion depends on the nature of the problem and on the objectives of the computation.

6.3 EFFECT OF TIME STEP AND GRID MESH

Accuracy of a numerical solution depends on two major classes of error: Roundoff and truncation errors. Roundoff error is a property of the computer and the program. Truncation error is a property of the method. Often the real danger as far as roundoff error are concerned results from the fact that solution obtained in one cycle of calculations. Therefore, the danger of error propagation and error growth can be generated as solution proceeds over a larger number of steps.

As Croft and Lilley (1977) discussed, roundoff error is caused by finite significant figure restriction and associated with the Fourier number (Fo) to given space - time grid. Such instability phenomenon is inherent in finite difference representation, and obviously yields an unsatisfactory computed solution. However, roundoff error can have tendency to grow with time in unstable scheme. Croft and Lilley (1977) also recommend that the Fourier number (Fo) is the well-known stability criterion to analyse the oscillation of the solution, and to limit the maximum value of Δt for a given mesh size subdivision. This can be expressed by the formula as :

$$Fo = \alpha_P \frac{\Delta t}{(\Delta l)^2} \tag{6.2}$$

where Δt is the iteration time step, Δl is the space distance, and α_P is the false transient factor.

Sometimes, refining the time step Δt and distance step Δl can improve the accuracy of the solution. Table 6.4 presents the second inner loop of program as the numerical solution of temperature distribution at the centre of the silo by simulated with 6 groups of different time step. In this explicit scheme, the large value of time step is employed for handling the multiple iterations.

Table 6.4 Solution effect by different time step at centre of silo

	Temperatures at centre of the silo						
Time step	Day 10	Day20	Day30	Day40	Day50	Day60	Mesh size
dt=45	23.0992	19.5796	19.7011	19.8235	19.7937	19.8158	21x20x21
dt=90	23.1325	19.6750	19.8481	19.9260	19.8827	19.9108	21x20x21
dt=180	23.1783	19.8284	20.0178	20.0380	19.9937	20.0452	21x20x21
dt=270	23.2124	19.9017	20.0745	20.0805	20.0376	20.0963	21x20x21
dt=315	23.6912	20.1885	20.2524	20.2321	20.2187	20.2808	21x20x21
dt=360	Floating point appearing						21x20x21

Evidently, with time step increments and grid mesh size stable, the numerical results show (see table 6.4), from time step dt=45 to dt=270, the temperatures difference between each neighbouring step is about 0.2% increase. According to equation 6.2, as Fourier number (Fo) is limited, the time step increase is real dangerous for the very large time step, such as dt=270 and dt=315, it is 2% increase between both of them, but when dt=360, the solution of temperature field can rapidly diverge, tend to infinity and cause floating point error appearing, this is indicated in table 6.4.

Conversely, the relatively small time step leads to a better accuracy in predicting the solution to be converge, such as dt=45, but the program running time (in Apollo station I use 7 days to get this results) is two times large than that dt=90. Even through dt=180 and dt=270 can diminish the CPU time, but the accuracy of the results are still not good enough, so the best choice for simulation program SILO3DFTN is dt=90.

However as equation (6.2) of Fourier number (Fo) shown, the gain in accuracy of numerical solution is associated with an necessary small time step in stability reason, the infinite small time step only can increase in labour without specially increasing the desired accuracy.

In this study, the principle arrangement of non-uniform grid mesh has been mentioned in Chapter 4 of section 4.2, and it is calculated by equations (4.1), (4.2), (4.3), (4.4) and (4.5). In order to select a suitable mesh size that yields the accurate results, the effect of grid points on the numerical solution is tested by a series running of the program. A $21 \times 20 \times 21$ non-uniform grid mesh was constructed in table 6.5.

Table 6.5 Non-uniform grid mesh: 21x20x21

	Space from the centre of silo to the silo wall
ξ direction	6.1819x6.1819x6.1819x6.1819x6.1819x6.1819x6.1819x6.1819 x6.1819x6.1819x6.1819x6.1819x6.1819x6.1819x4.4843x 1.7937x1.7937x1.7937x1.7937x1.7937

A $31 \times 30 \times 31$ non-uniform grid mesh was designed in table 6.6 as follows:

Table 6.6 Non-uniform grid mesh: $31 \times 30 \times 31$

	Space from the centre of silo to the silo wall
ξ direction	0.036x0.036x0.036x0.036x0.036x0.036x0.036x0.036x0.036x 0.036x0.036x0.036x0.036x0.036x0.036x0.036x0.036x0.036x 0.036x0.036x0.036x0.036x00036x0.036x0.045x0.018x0.018x 0.018x0.018x0.018

Table 6.7 shows the numerical solution effect by trial different grid mesh. In the pressure distribution calculation, the governing equation of Laplace formula is solved by a false transient method. The time step is set quite small for the suitable number of iterations. Obviously, the decrease grid mesh size can achieve the solution of the desired accuracy, but can lead the CPU time limit increase as same percentage as grid mesh decrease. Table 6.7 indicates, at the same iter-number, the *P-centre* can given the different results by the different grid mesh. As we discussed before, grid mesh 21x20x21 is the best choice for improve the accuracy of the results.

Table 6.7 Effect of grid mesh on the solution

Non-uniformed mesh	Time step	Iter number	*P-centre*
11x10x11	1	4500	349.0681
	1	6000	366.2688
	1	7040	372.6066
	1	8040	376.3117
21x20x21	1	4500	200.3025
	1	6000	232.2351
	1	7040	249.0050
	1	8040	261.8310
31x30x31	1	4500	192.4070
	1	6000	200.8912
	1	7040	203.7110
	1	8040	205.1922

* *P-centre* is the pressure fields at the centre of the silo

Practice has been proved, as observed from Fourier number (Fo), reducing time step and space distance can increase in accuracy of the solution, but it is also important in adjusting a reasonable balance between accuracy and efficiency.

CHAPTER 7

COMPARISONS AND DISCUSSION

The main objective of this chapter is to compare the ecosystems in a non-aerated silo, and silos fitted with linear and annular aeration ducts. These simulations are carried out using climatic data obtained for Melbourne, Australia. Each of the simulations assumes the same initial conditions, the same physical properties, the same silo dimensions and the same period of storage.

7.1 DEFINITIONS

A non-aerated silo is by definition not ventilated, and cooling occurs in practice by conduction and free convection processes. In this thesis, only thermal conduction is considered. Work conducted by Jayas *et al* (1992) has shown that cooling in silos is dominated by thermal conduction, but free convection processes become important when studying moisture migration and the distribution of fumigant gases.

The linear aeration system is defined as a silo fitted with a linear aeration duct which is single straight and placed on the conical base of the silo, as illustrated in figure 6.3 (a).

The annular aeration system is also defined as a silo fitted with an annular aeration duct which is seat on the conical base of the silo, as shown in figure 6.3 (b).

The orientation of the silo is defined by means of the points of the compass, that is to say the top is north, the bottom is south, the left is west, and the right is east, as indicated in figure 6.4 (b).

7.2 PRESSURE AND VELOCITY DISTRIBUTIONS

Airflow is the major factor affecting the performance of the aeration system. Airflow resistance is overcome by providing the pressure in the aeration duct. When air is forced through grain mass, air pressure and air velocity fields develop. The simulations of the silos fitted with linear and annular ducts have been implemented. The predictions have shown that the different type of ducts can significantly effect the air pressure and air velocity distributions.

Figures 7.3 and 7.4 show, the contour maps of pressures and velocities in the silos fitted with the two type of ducts investigated. It is observed, that the pressure gradients in the silo with the annular duct are more uniform than that the silo fitted with the linear duct. However, it can be observed that in both of the aeration systems considered the pressure patterns become very similar, particularly near the grain surface. One reason for their similarity in this region is that the air flow rate are the same in each system.

The component of vertical velocity in the silo fitted with the annular duct is generally more uniform in the conical base than in the silo fitted with a linear duct, as can be seen from figure 7.3 (c) and figure 7.4 (c). As would be expected, the radial velocity in the silo fitted with the annular duct is symmetricai about the centre line of the silo. However, in the cause of the silo fitted with the linear duct, air flow from one side of the silo to the other. These phenomena can be observed in figures 7.3 (b) and 7.4 (b).

The vertical component of velocity along the walls in both causes of aeration systems are reflected by the pressure distributions, they are indicated in figures 7.1 and

7.2. It can be inferred that the vertical velocity is generally higher on the eastern side of the silo where a linear duct is placed, and lowest on the opposite side. Interestingly, the vertical pressure gradients on the north and south, and the west facing portions of the wall are very similar.

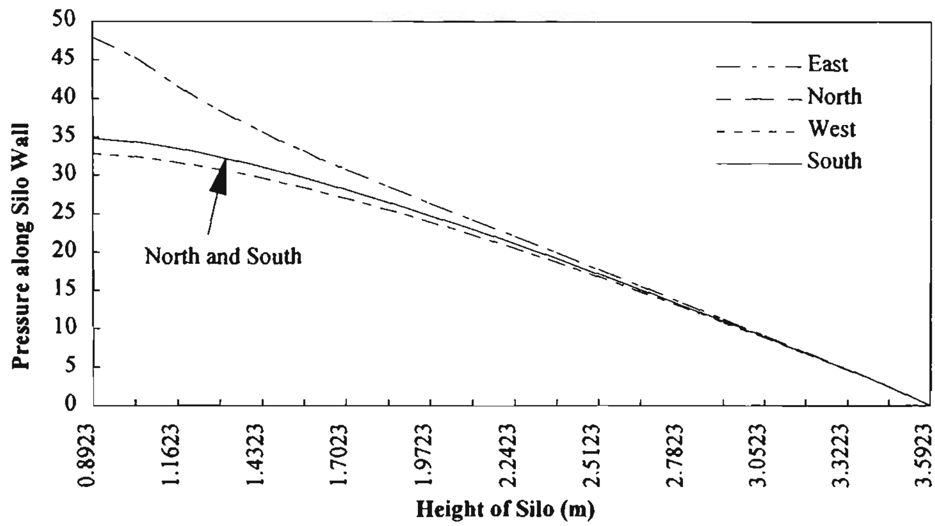


Figure 7.1 Pressure distributions along the wall of silo in the linear duct case

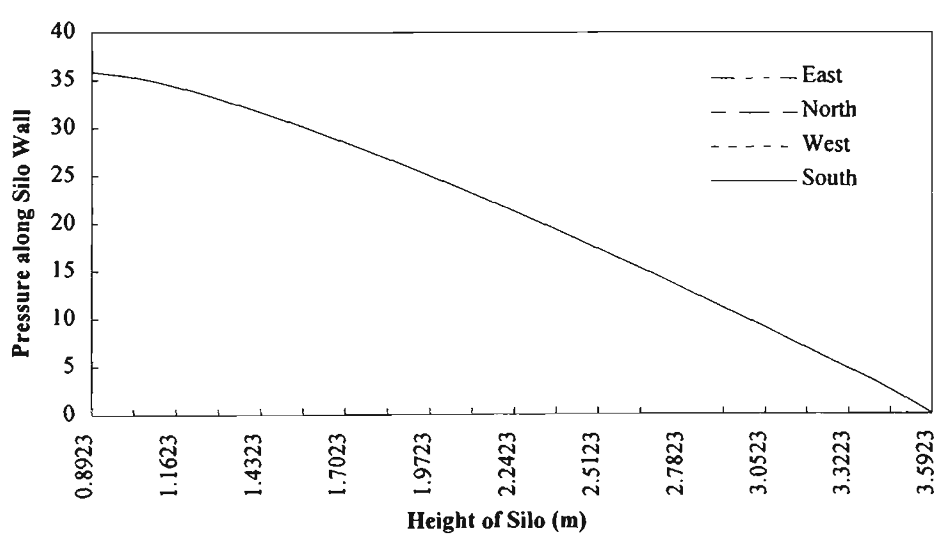
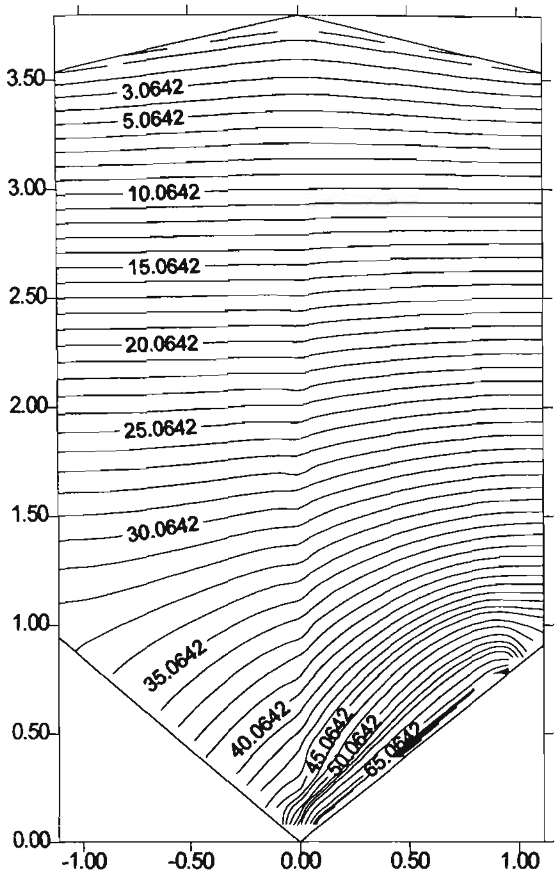
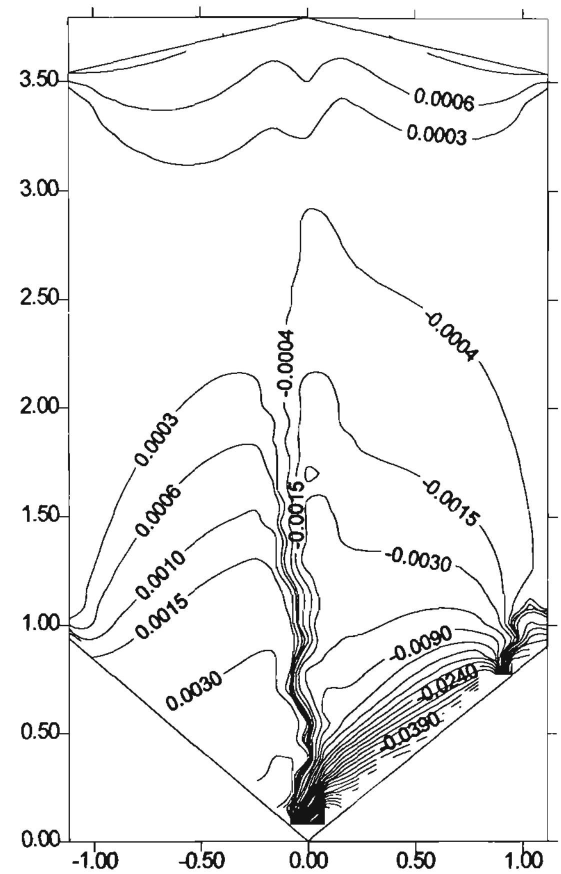


Figure 7.2 Pressure distributions along the wall of silo in the annular duct case

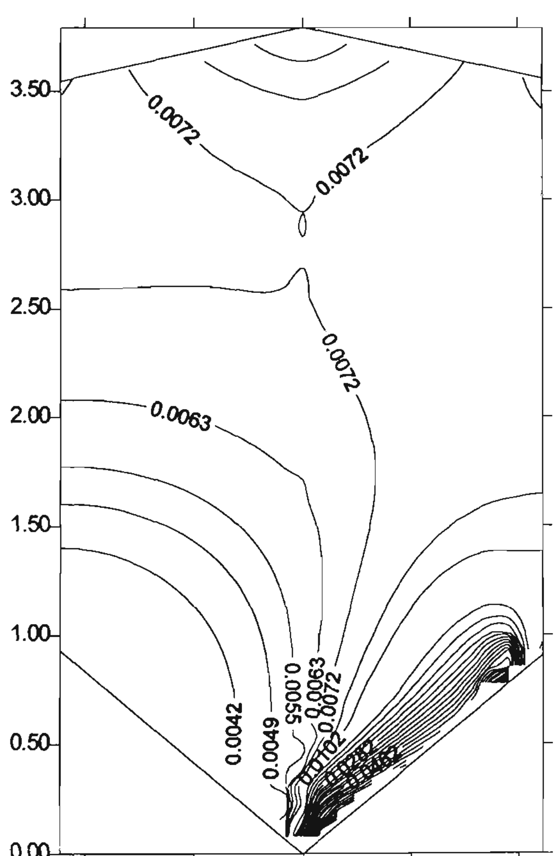
The pressure distributions on the wall of the silo fitted with an annular duct (as shown in figure 7.2) are symmetrical, as would be expected. Hence, the components of velocity distributions are also symmetrical in this case.



(a) Pressure distributions (linear duct)

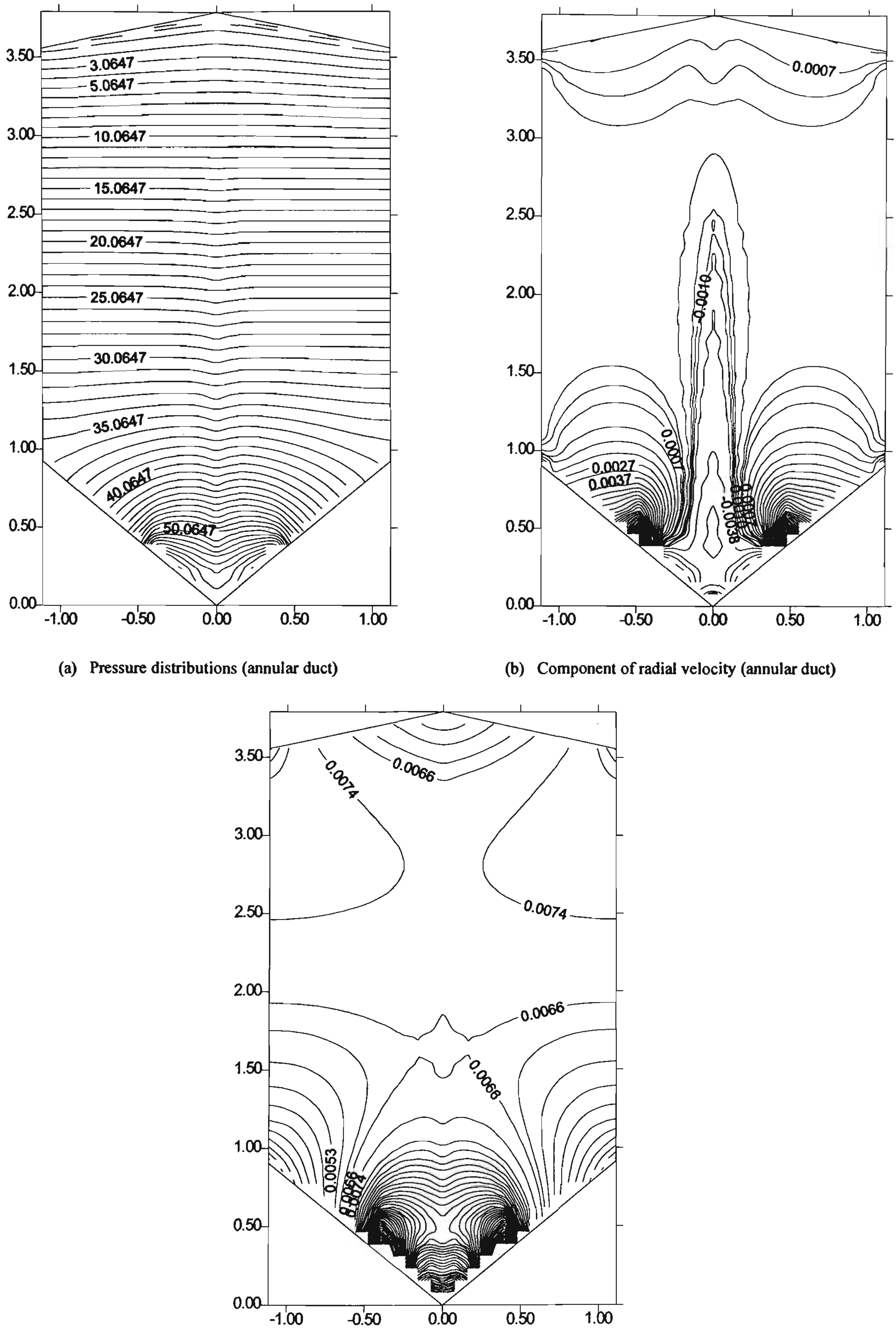


(b) Component of radial velocity (linear duct)



(c) Component of vertical velocity (linear duct)

Figure 7.3 Pressure and velocity distributions in a silo fitted with linear aeration duct



(a) Pressure distributions (annular duct)

(b) Component of radial velocity (annular duct)

(c) Component of vertical velocity (annular duct)

Figure 7.4 Pressure and velocity distributions in a silo fitted with annular aeration duct

7.3 TEMPERATURE DISTRIBUTIONS

Temperature changes in stored grain may come from two heat sources: Internal and external sources. Internal sources are the heat, generated by the respiration of grains, moulds and insects. External sources largely come from the atmospheric environment around the silo, especially solar energy which is transferred from the wall and the roof of the silo to create uncomfortably hot conditions.

The temperatures in the unaerated silo have been developed as shown in figure 7.8. The temperature distributions in the silo fitted with linear and annular aeration ducts are shown in figures 7.9 and 7.10 respectively. Comparison of three figures shows that for a same initial temperature of 30°C, and after 60 days storage, the centre temperature of the non-aerated silo remain at 26°C, in the silo fitted with the linear duct, the temperatures after 60 days has fallen to about 20°C, and in the silo fitted with the annular duct, the corresponding temperature is 16°C.

In figure 7.8, it can be seen that the centre temperature reduces typically average 1~2°C by every 20 days. This is because ambient temperatures are generally lower in Melbourne than that the initial grain temperatures in the centre of the silo.

In figure 7.9, it is clear that the grain temperature in a silo fitted with a linear duct falls quickly, and after 20 days storage, much of the grain on the side of the silo with the linear aeration duct is about 20°C. After 40 days aeration, the grain temperature is 20°C in the most of the centre region of the silo. After 60 days aeration, the temperature in the central region has not changed much. However, the volume of 20°C grain is slightly reduced, because the climate of Melbourne in February is warmer than that of January.

As figure 7.10 presented, after 20 days aeration, most of the grain in the silo fitted with an annular duct has a temperature about 19°C, as can be seen in figure 7.10. This is because the annular aeration system has a symmetric layout, and the air flow distribution is uniform in the silo. After 40 days aeration, the grain in the centre region of the silo has fallen to around 16°C, After two months aeration, the temperature in the central region within the silo has increased somewhat to 18°C. This change is effected by the climatic conditions.

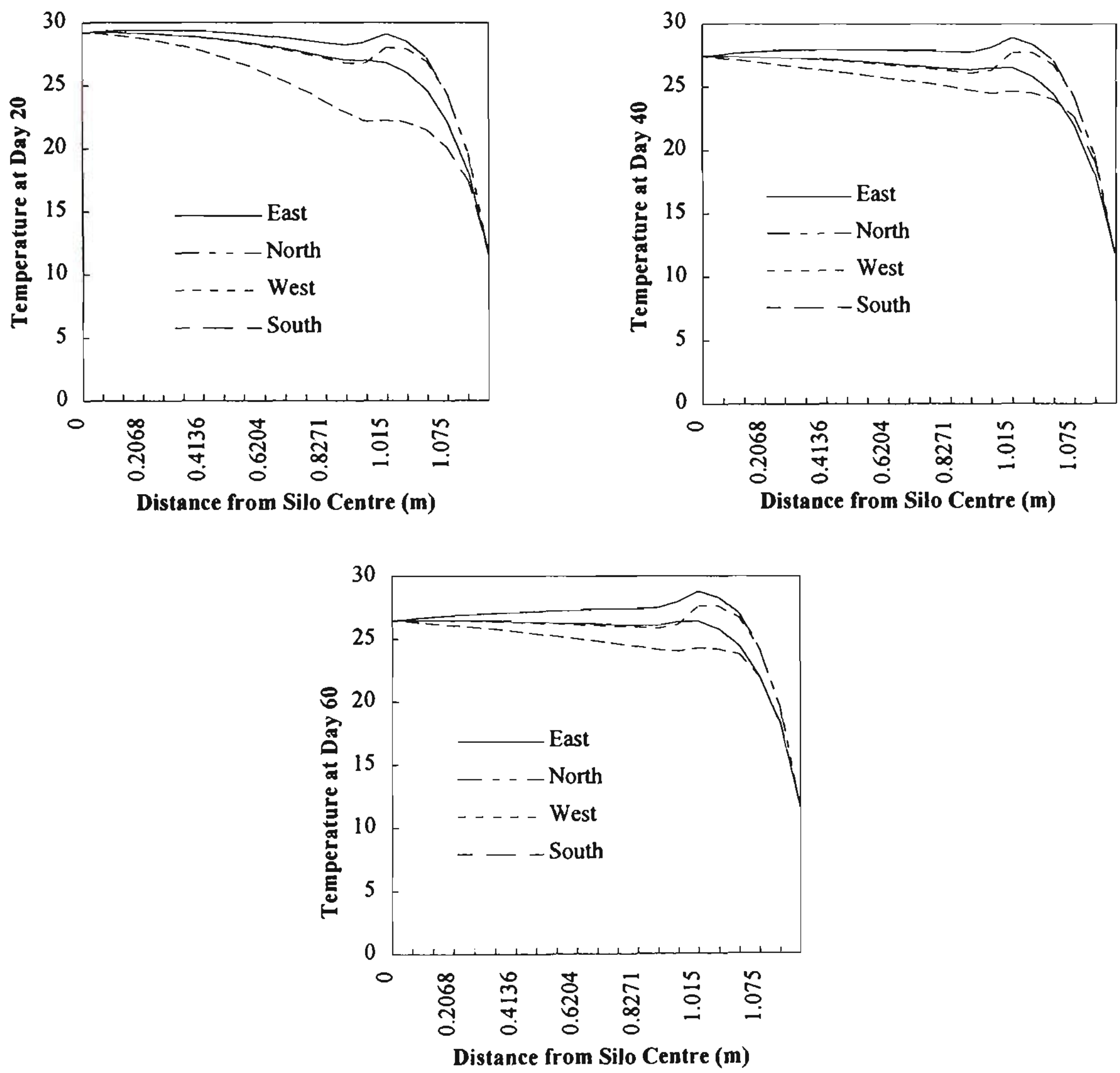


Figure 7.5 Radial profiles of the temperature in the non-aerated case

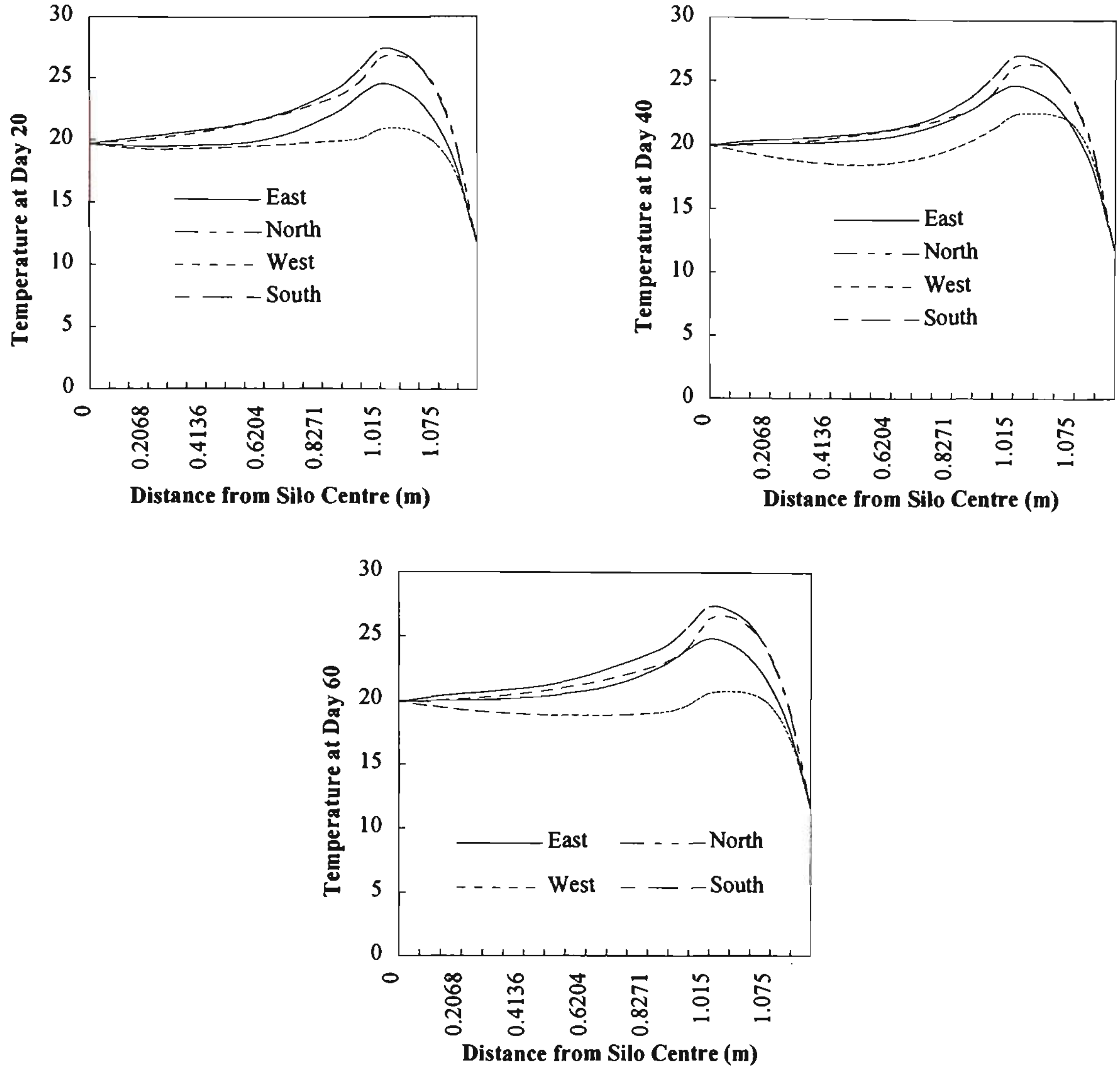
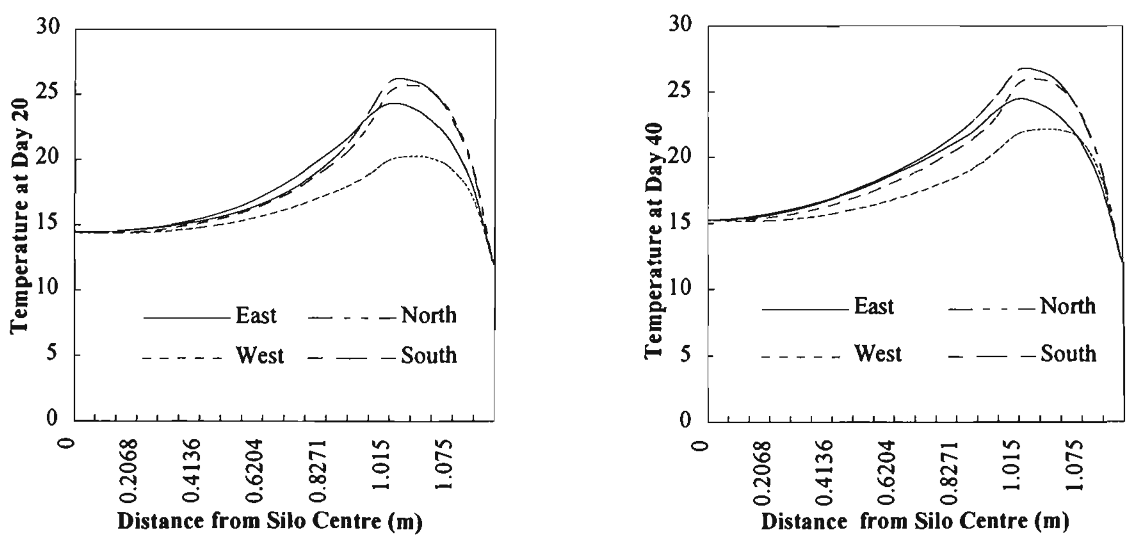


Figure 7.6 Radial profiles of the temperature in the linear duct case



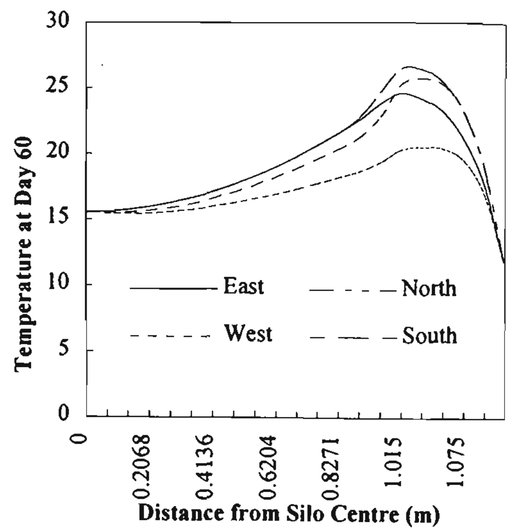


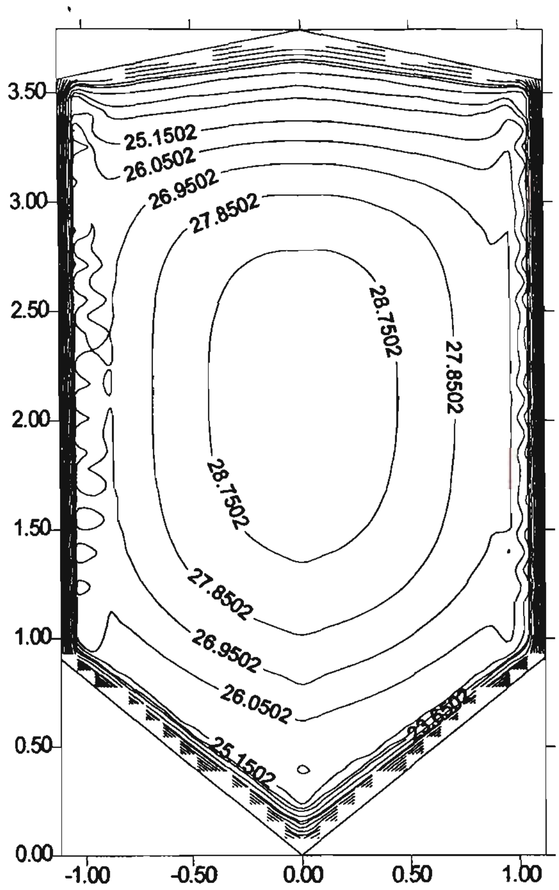
Figure 7.7 Radial profiles of the temperature in the annular duct case

Generally, the temperature of the stored grain does not change rapidly unless it is aerated. Figures 7.5, 7.6 and 7.7 reflect the temperature distributions show in figures 7.8, 7.9 and 7.10, that results from the imposition of varying temperatures in the east, north, west and south facing portions of the wall. It can be seen that the temperature of the grain in the centre of the non-aerated silo is much slower to cool than that in either of the two aerated silos, and it rises towards to the centre of silo (around 26-28 °C).

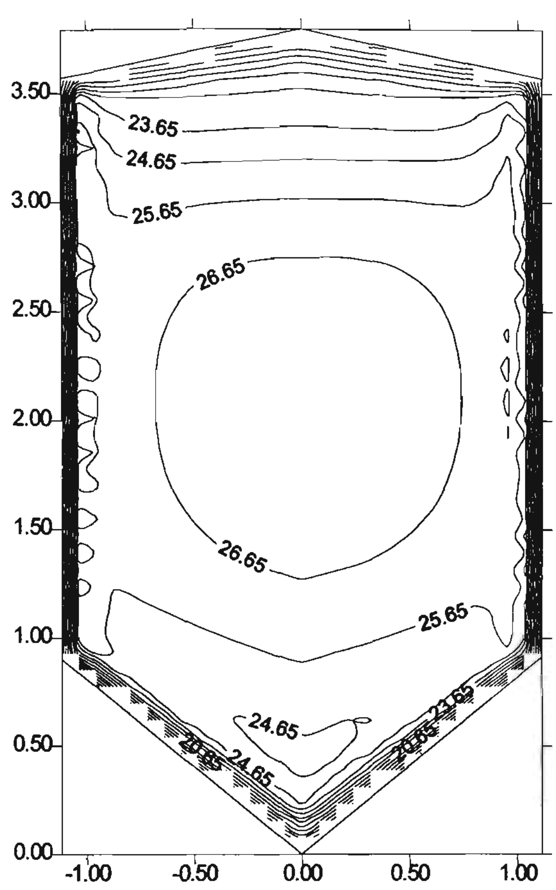
Figures 7.6 and 7.7 show that the aerated silo fitted with an annular duct performs much more effectually than the one fitted with a linear duct. The silo with the linear duct is cooler towards to the centre of the silo, around 19-20 °C, and the performance of annular aeration system is better towards to the centre of the silo, when grain temperatures in the region of 14-17 °C are observed.

Importantly, it can be seen that the temperature near the walls of all three cases considered are quite similar. Figures 7.5, 7.6 and 7.7 capture the situation during the night, when the walls of the silo are cold. At this time, the maximum temperature occurs about 10cm distant from those portions of the wall that are exposed to solar radiation.

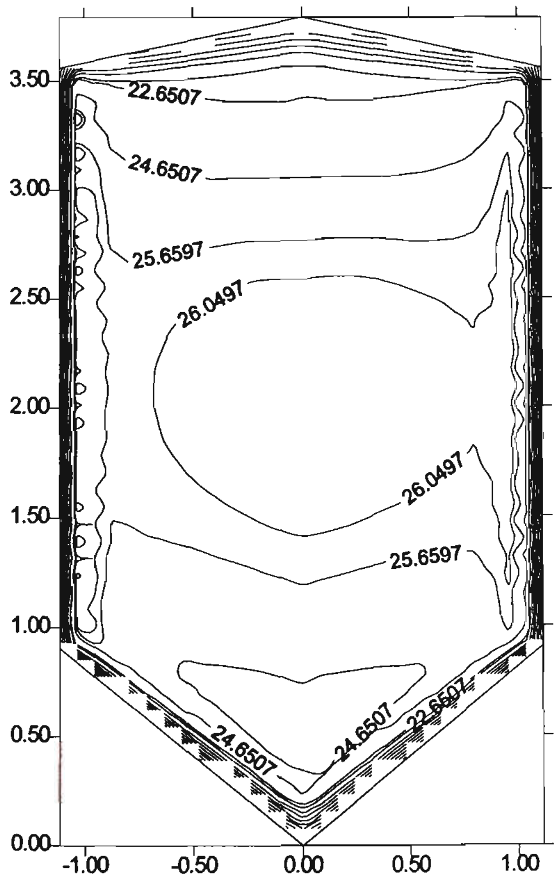
However, the grain near the southern wall, which receives no sunlight, has a minimum temperature similar to the mean ambient temperature. It is noteworthy that the maximum temperature occur on the north and west facing portions of the wall, where there is a combination of high ambient temperature, and a high intensity of solar radiation which is insolated particularly around noon.



Temperature at day 20 (non-aerated)

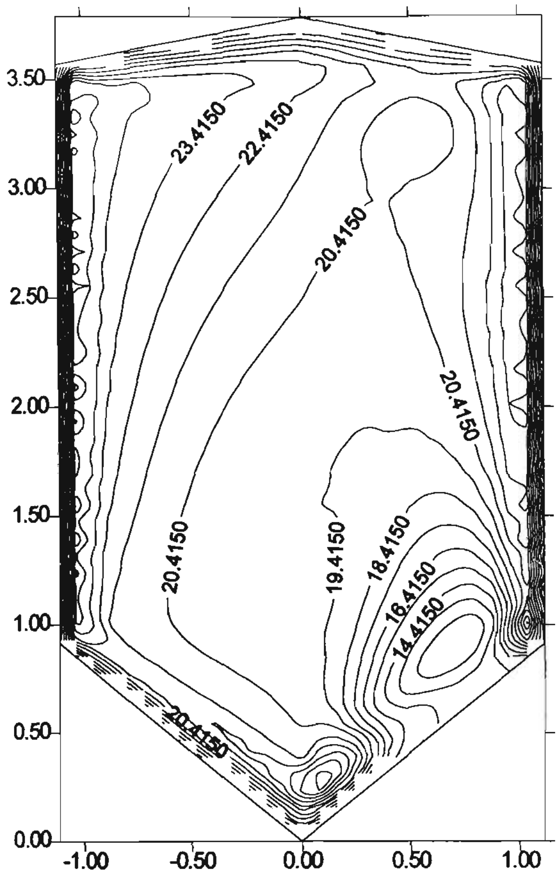


Temperature at day 40 (non-aerated)

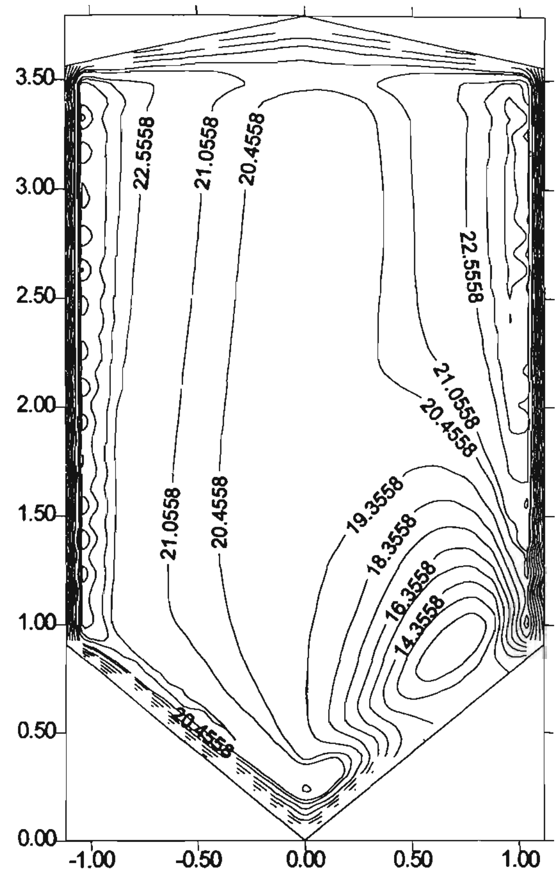


Temperature at day 60 (non-aerated)

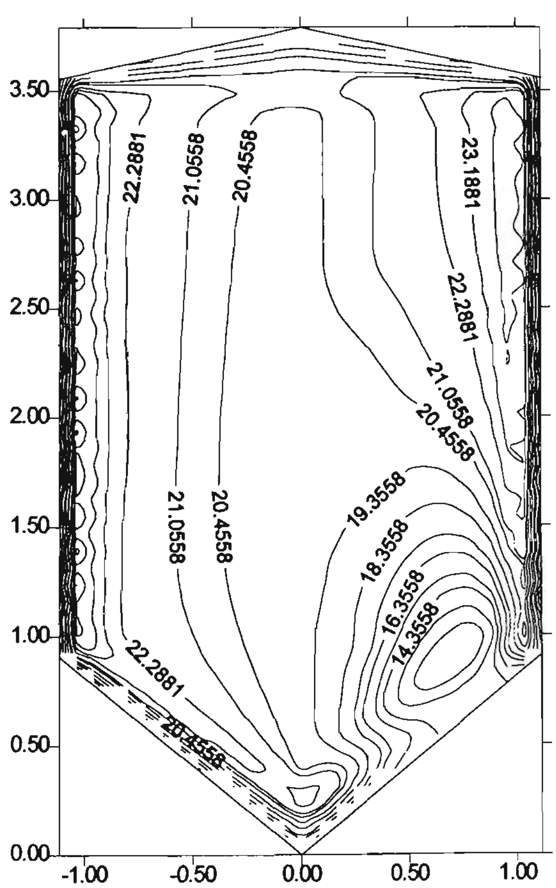
Figure 7.8 Temperature distributions in the non-aerated silo



Temperature at day 20 (linear duct)

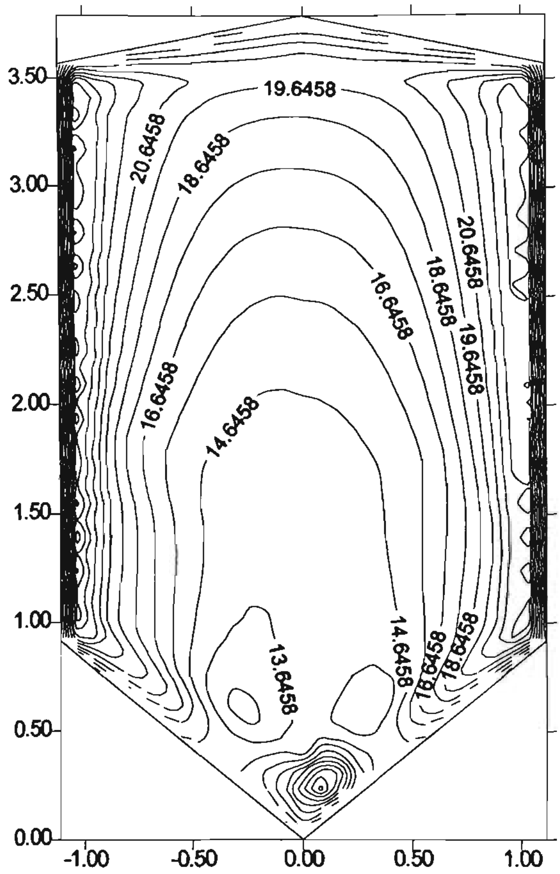


Temperature at day 40 (linear duct)

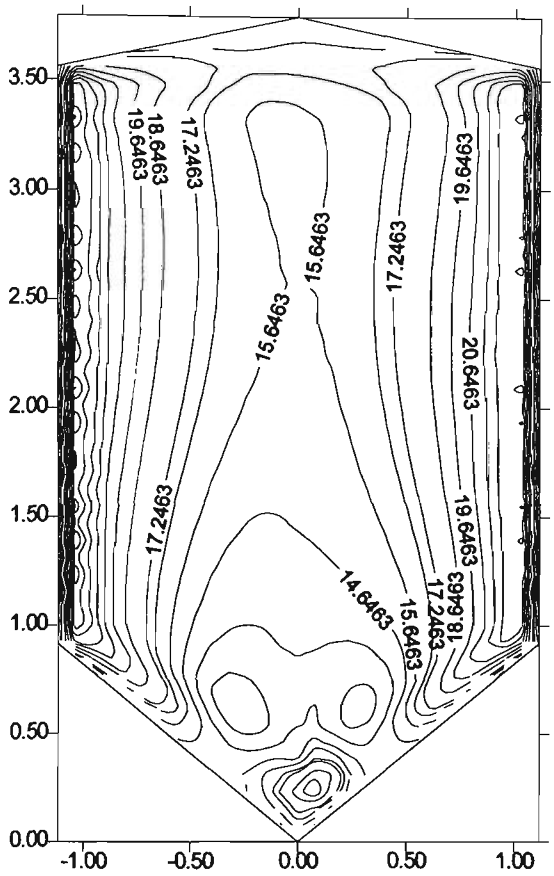


Temperature at day 60 (linear duct)

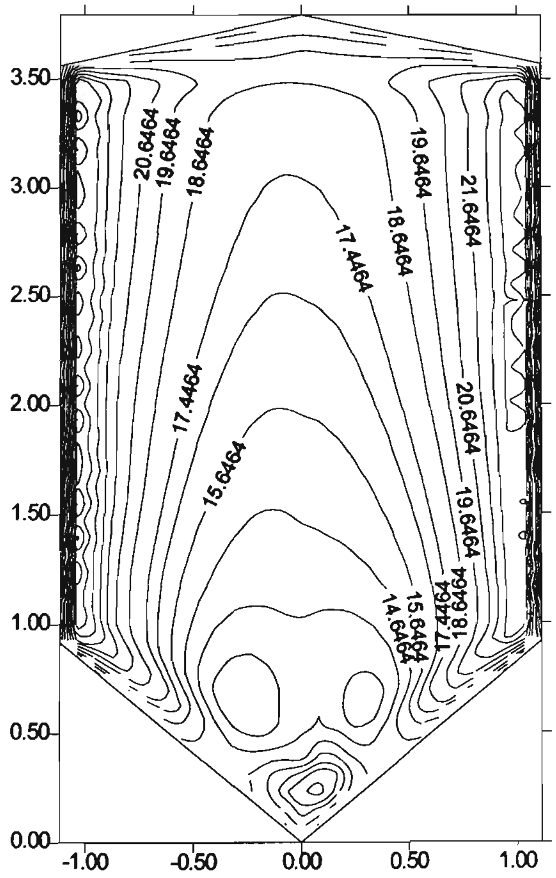
Figure 7.9 Temperature distributions in the silo fitted with a linear duct



Temperature at day 20 (annular duct)



Temperature at day 40 (annular duct)



Temperature at day 60 (annular duct)

Figure 7.10 Temperature distributions in the silo fitted with an annular duct

7.4 MOISTURE CONTENT DISTRIBUTIONS

Moisture content of grain denotes the quantity of water per unit mass of either wet or dry grain, and it is usually expressed on a percentage basis. Figures 7.14, 7.15 and 7.16 show the moisture content distributions in the non-aerated and the two aerated grain silos fitted with linear and annular ducts.

After 60 days simulation, compared with an initial condition of 13.64% (dry basis), we obtain: In the non-aerated silo, the centre region of grain moisture content is 13.63%, it is 0.07% (relative different, and as follows in this Chapter) lower than initial condition. In the silo fitted with the linear duct, the centre volume of the moisture content of grain has a 12.8% dry basis, it is 7% lower than the initial condition, and in the silo fitted with an annular duct, it is 12.36%, this is about 10% less than the initial moisture content.

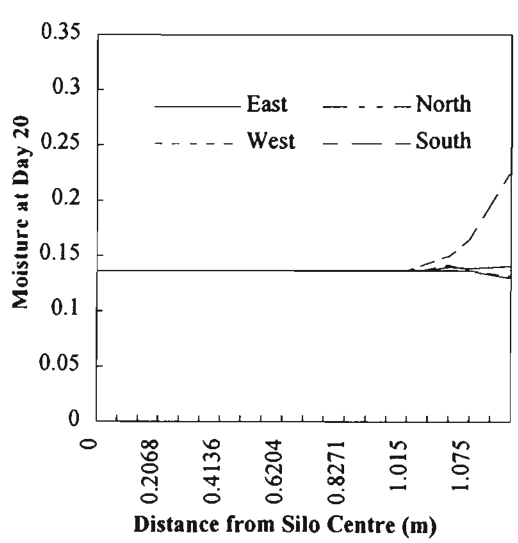
Figure 7.14 that indicates the non-aerated silo during first 20 days storage, the moisture diffuse causes a high moisture content to develop at the bottom and the walls of the silo which have low temperatures, and the maximum moisture content is about 13.75%. However, molecular diffusion alone can not be the cause of the apparently rapid increase of grain moisture content near the boundaries of the silo. Such process would requires months, not days. It is suspected that respiration is the cause of the build up in moisture content, and whether or not this reflects reality is discussed below.

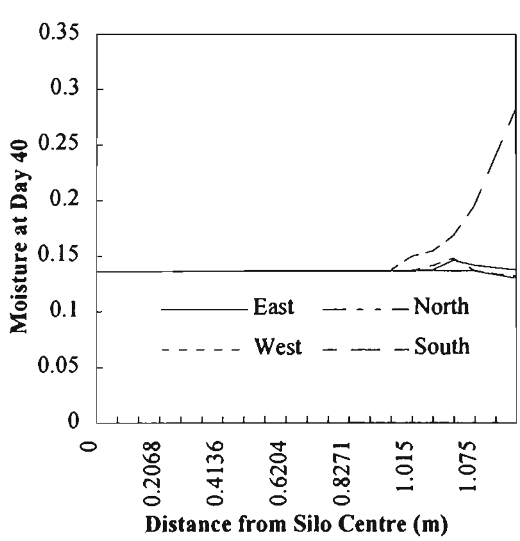
In figure 7.15, after 20 days aeration, the low temperature and the high relative humidity in the silo with linear aeration system is occurred around the duct, and the moisture content of the grain is therefore high. At the centre of the silo, the moisture content decreases to about average 12.8% dry basis. After 40 days aeration, the moisture content increases near the duct zone, and reduces in the centre of the silo. After 60 days aeration, the moisture content at upper surface of silo is about 4% lower than the initial condition, and around duct area is 22% higher than the initial condition which is 30% higher than the moisture content at centre area.

It seems that the grain moisture content does not increase so much near an annular duct, possibly because such duct has a large area than a linear duct. As a result, the moisture is distributed throughout a large volume of grain, as indicated in figure 7.16.

It is also clear that figure 7.16 shows, the grain near the conical base of the silo aerated with an annular duct is increasing in moisture content. This may not reflect reality, but the increase in moisture content suggested by the model arised from respiration. The respiration model of Lacey *et al.* (1994) appears to be somewhat sensitive, and it is quick to predict respiration at low grain moisture contents. However, grain storage technologists would do well to monitor possible mould activity near the base of conical bottomed silos, if they were to be fitted with annular ducts.

On the other hand, this increase in respiration is not observed in the silo fitted with a linear duct as shown in figure 7.15, because this region is aerated by air flowing from the duct to the opposite side of the silo.





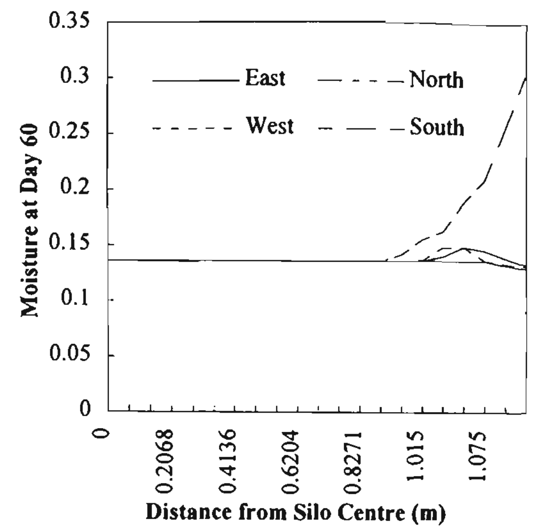


Figure 7.11 Radial profiles of the moisture content in the non-aerated case

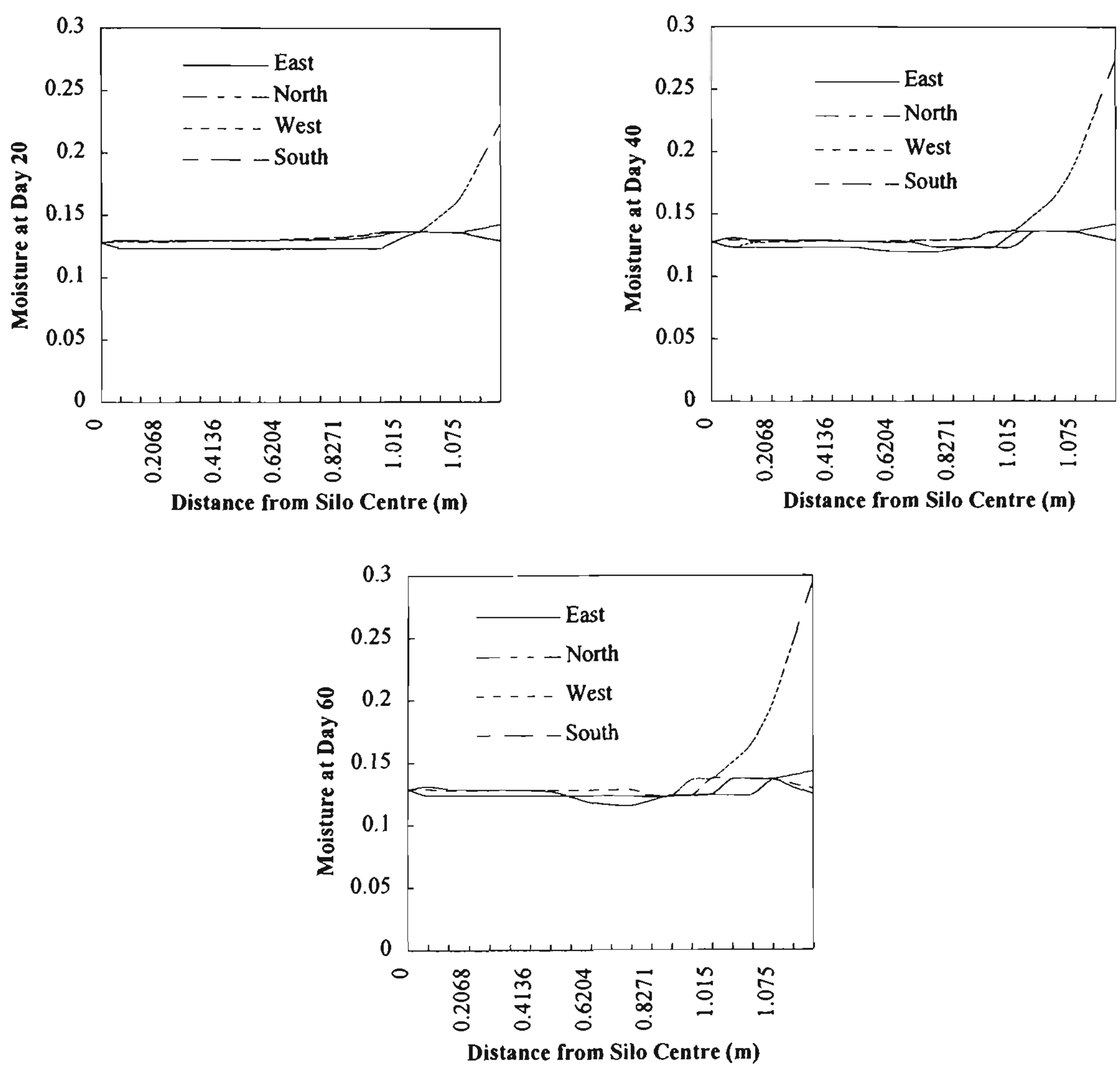


Figure 7.12 Radial profiles of the moisture content in the linear duct case

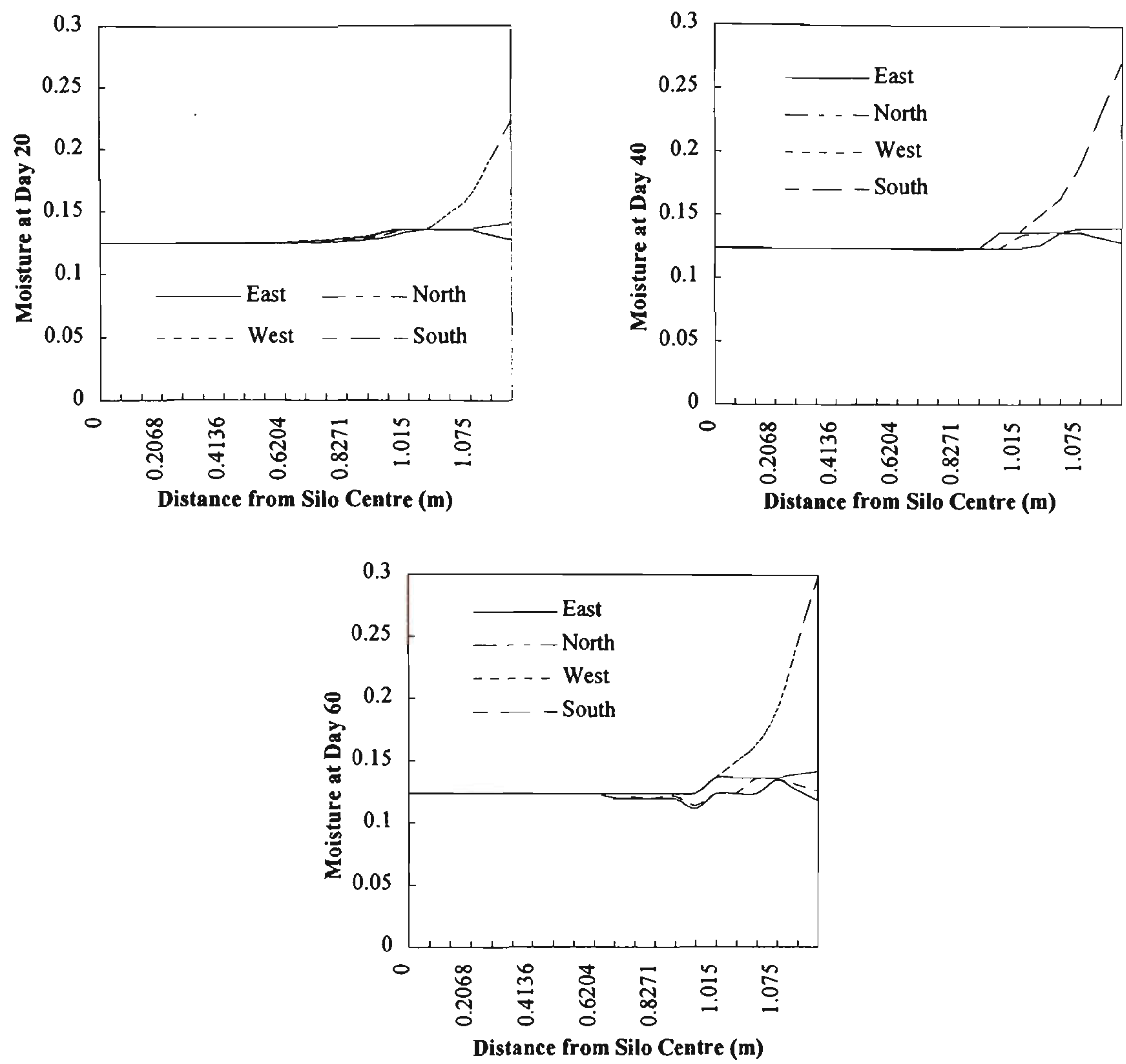


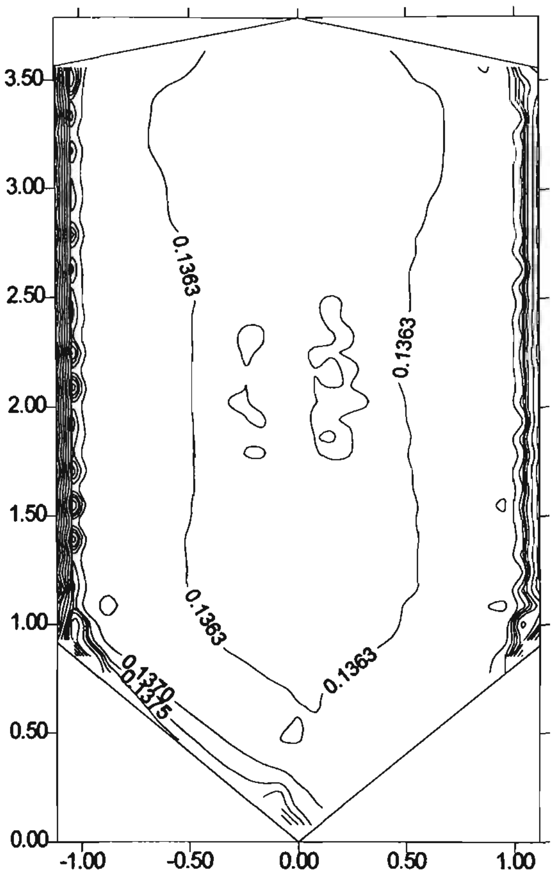
Figure 7.13 Radial profiles of the moisture content in the annular duct case

Figures 7.11, 7.12 and 7.13 show that the variation of moisture content from the centre of the silo to the facing of east, north, west and south sides of the silo in the cases of non-aerated and the two aeration systems considered. As would be expected, the moisture content of the grain in the centre of the non-aerated silo changes imperceptibly. However, the grain moisture content does decrease slightly in the centre of the silos fitted with linear and annular aeration ducts, this is expected from the work of Sutherland *et al.* (1971), who show that the passage of a cooling wave is associated with a fall in grain moisture content.

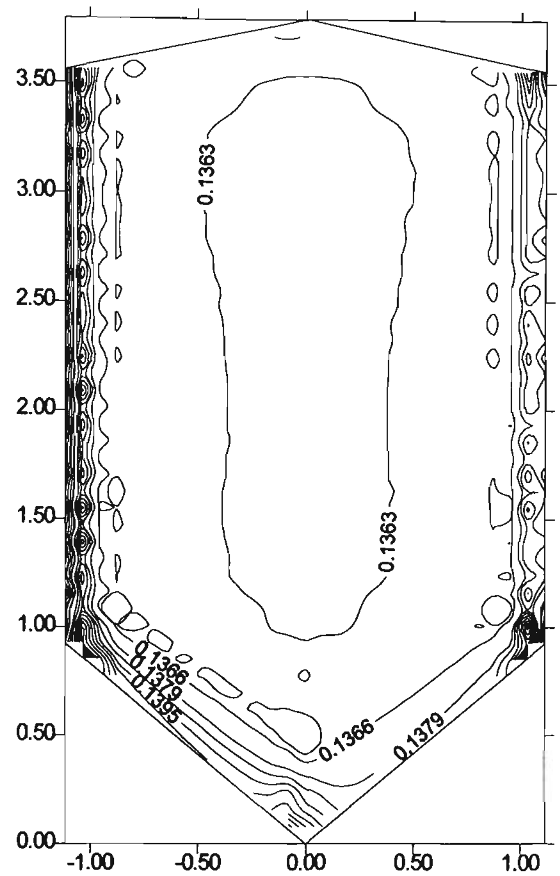
It has observed that the moisture content changes occur in the vicinity of the walls of both non-aerated and aerated silos. It is particularly noteworthy that on the coldest southern sides of all three causes are considered, the grain moisture content is observed to increase considerably. Figure 7.11 also indicates that the moisture migration is important up to about 10 cm away from the silo wall.

However, the apparent moisture migration to the southern of the wall occurs both non-aerated and aerated silos, as suggested above, the increase in moisture content may have resulted from respiration.

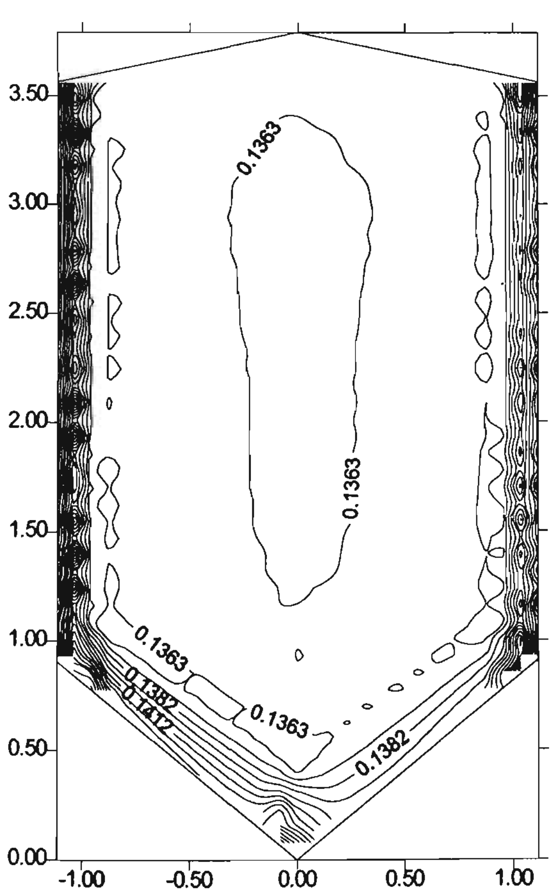
Furthermore, the grain moisture content is generally lower throughout the aerated silo than through the unaerated silo, this is consistent with the physics of one-dimensional heat and mass transfer processes that occur in grain (Thorpe, 1986).



Moisture content at day 20 (non-aerated)

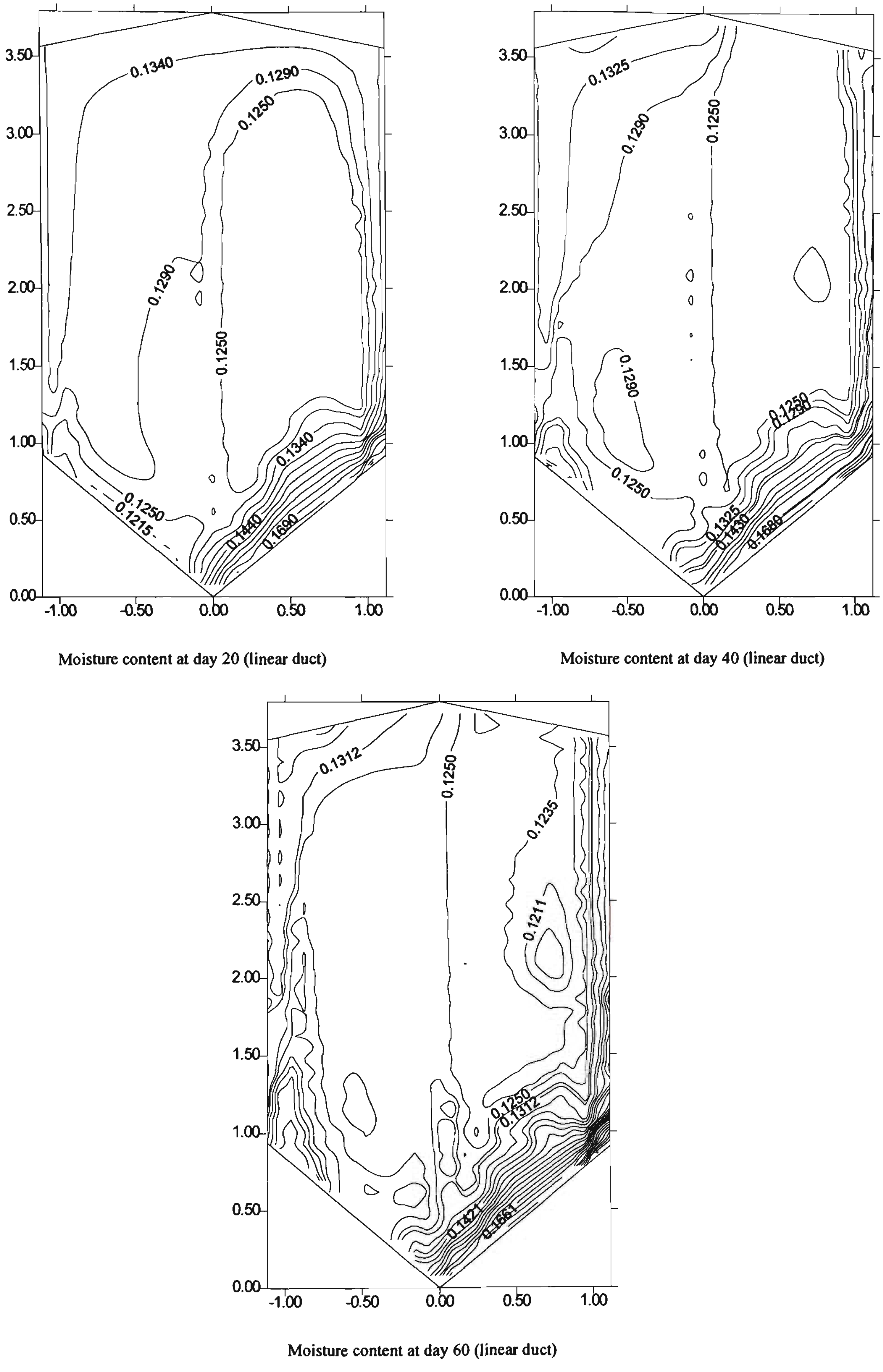


Moisture content at day 40 (non-aerated)



Moisture content at day 60 (non-aerated)

Figure 7.14 Moisture content distributions in the non-aerated silo

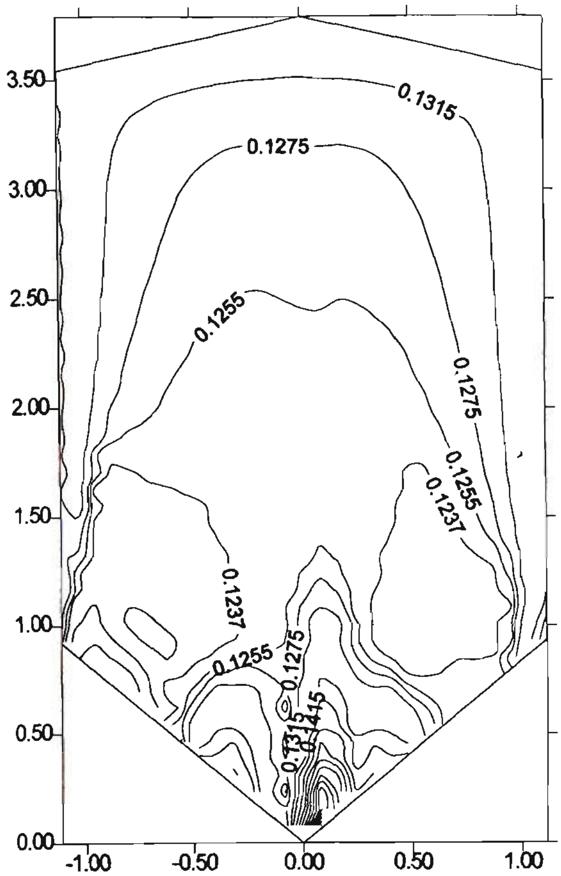


Moisture content at day 20 (linear duct)

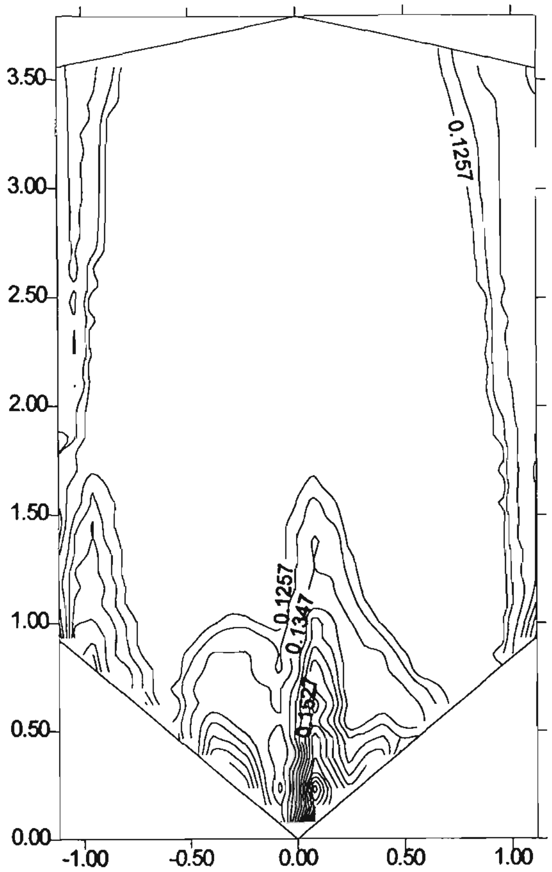
Moisture content at day 40 (linear duct)

Moisture content at day 60 (linear duct)

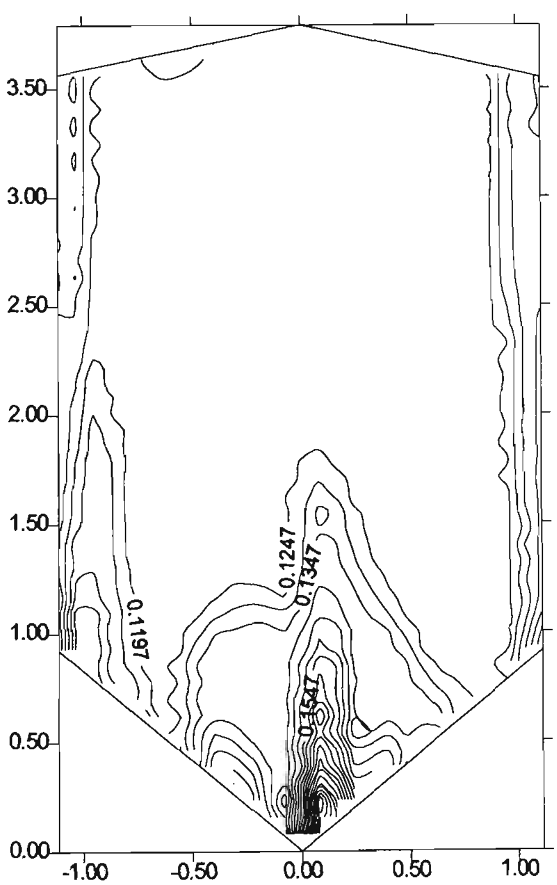
Figure 7.15 Moisture content distributions in the silo fitted with a linear duct



Moisture content at day 20 (annular duct)



Moisture content at day 40 (annular duct)



Moisture content at day 60 (annular duct)

Figure 7.16 Moisture content distributions in the silo fitted with an annular duct

7.5 ABSOLUTE HUMIDITY DISTRIBUTIONS

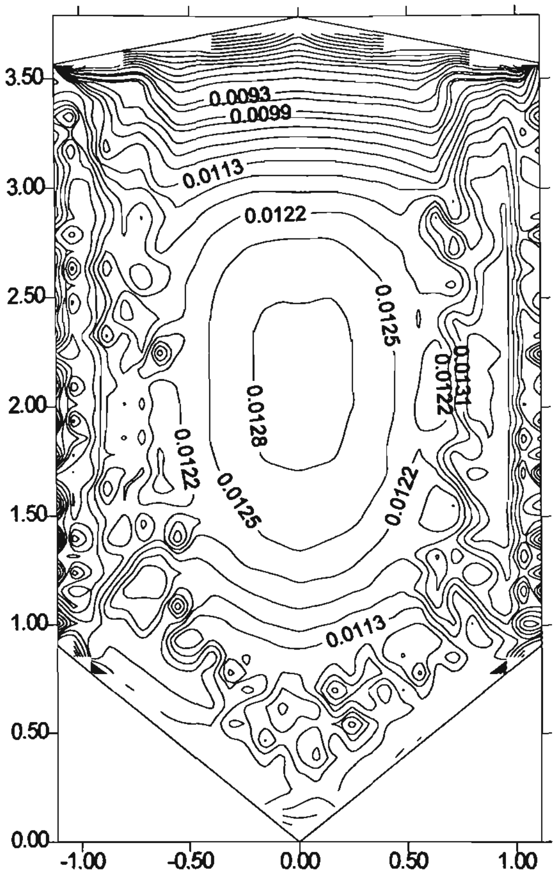
Absolute humidity of air is the mass of water vapour present in one kilogram grain of dry air. The absolute humidity of air is dependant on the temperature and grain moisture content, and these three variables are related by the sorption isotherm. The contour maps of absolute humidity of air in the non-aerated silo, and the silos fitted with linear and annular ducts are shown in figures 7.17, 7.18 and 7.19.

It is clear that after 60 days storage, with an initial humidity of 0.0136kg/kg, the absolute humidity of air in the non-aerated silo of the central region falls to 0.0106kg/kg, which is 28% less than initial condition. In the aerated silo fitted with a linear duct, the humidity is 0.0069kg/kg, this is 97% lower than the initial humidity condition. In the annular duct case, the absolute humidity of air at centre region is 0.0048kg/kg, about 183% lower than the initial condition.

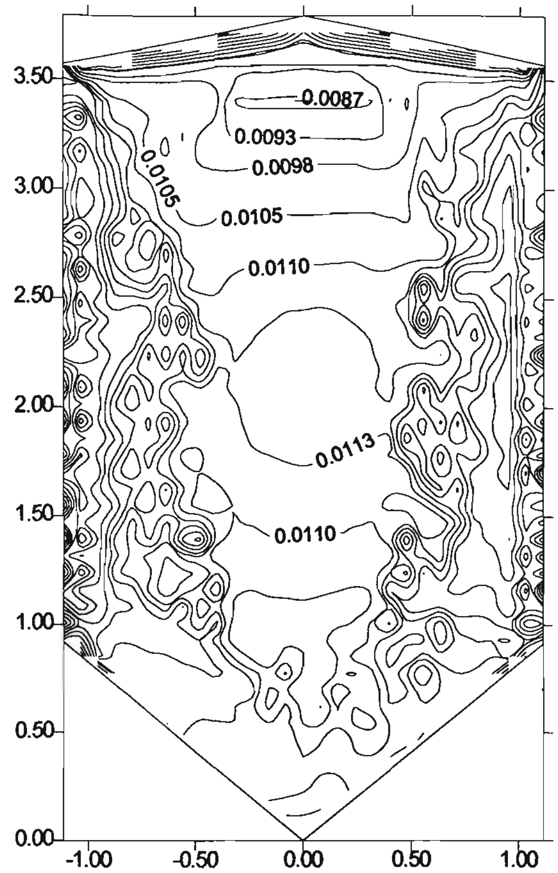
The main reason for the lowering of the humidities is the decrease in grain temperature, although drying of the grain contributes to the decrease in humidity.

The contour plots of the humidity of the intergranular air appear to be very noisy, and this is clearly shown in figure 7.17. One reason for this is that the model predicts very small humidity gradients in the grain bulk, but their presence is amplified by the plotting routine.

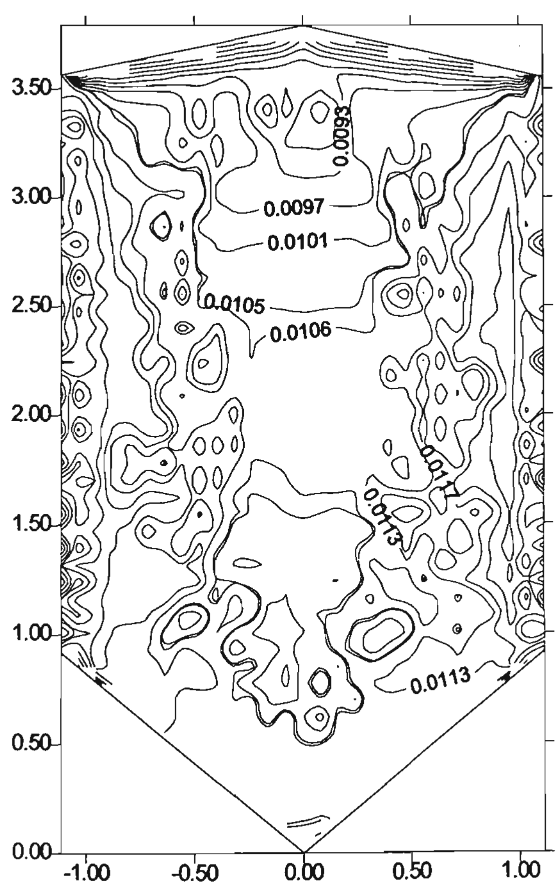
As might be expected, the higher absolute humidities appear near the linear aeration duct (as shown in figure 7.18). In the case of the silo fitted with an annular duct (it is indicated in figure 7.19), the high absolute humidity of air near the base of the cone reflects the respiration (actual or simply as a result of an artifice of the model) alluded to above.



Humidity at day 20 (non-aerated)



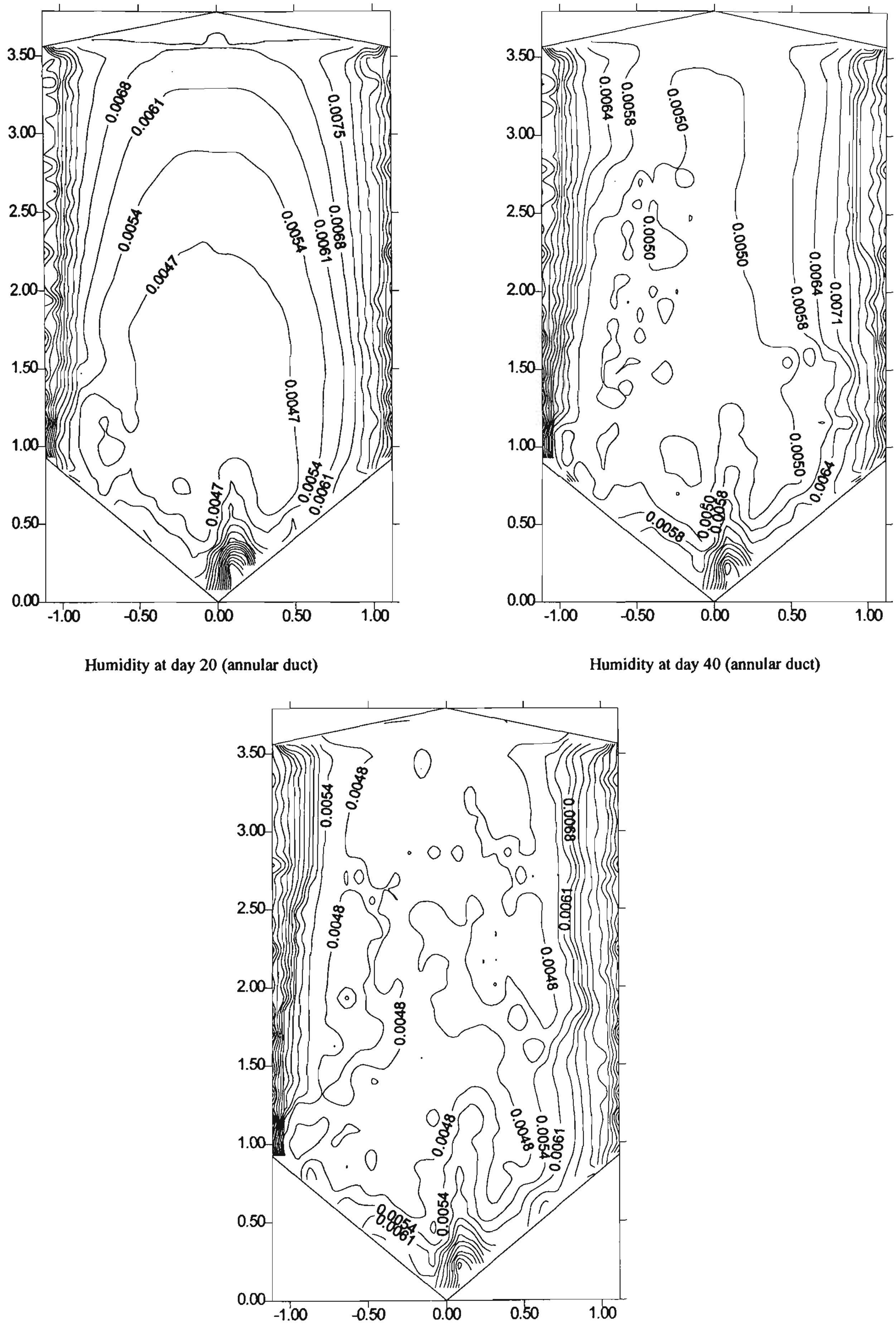
Humidity at day 40 (non-aerated)



Humidity at day 60 (non-aerated)

Figure 7.17 Humidity distributions in the non-aerated silo

Figure 7.18 Humidity distributions in the silo fitted with a linear duct



Humidity at day 20 (annular duct)

Humidity at day 40 (annular duct)

Humidity at day 60 (annular duct)

Figure 7.19 Humidity distributions in the silo fitted with an annular duct

7.6 WET BULB TEMPERATURE DISTRIBUTIONS

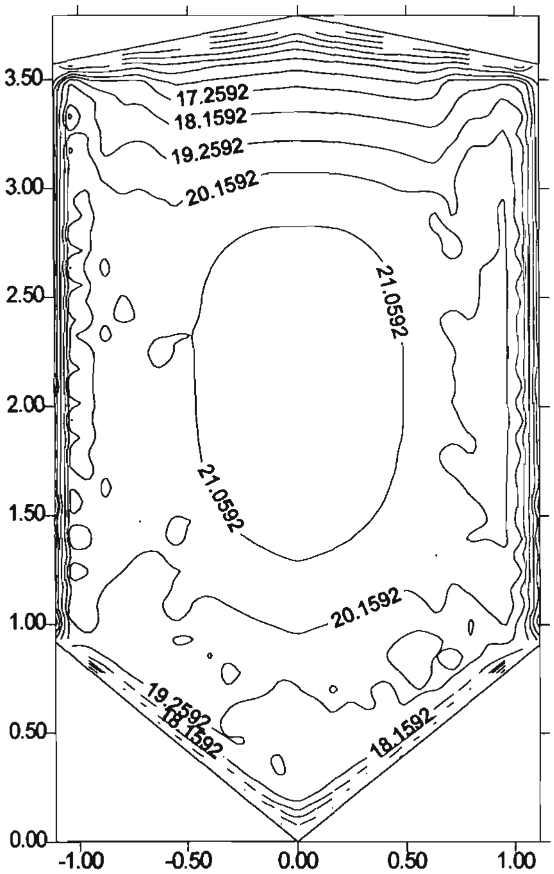
Wet bulb temperature is the temperature reached by moist air and water, if the air is adiabatically saturated by evaporating the water. Wet bulb temperature in many way dictates the performance of aeration system, and it may be described as the joint effect of temperature and moisture content on the rate of increase of insect population. The contour maps of wet bulb temperature are presented in figures 7.20, 7.21 and 7.22.

After 60 days storage with an initial wet bulb temperature of 22.3°C, in the non-aerated silo, the average of wet bulb temperature at centre region is 19.1°C, and in the linear duct aeration system is 14.1°C. In the silo fitted with annular aeration duct, the wet bulb temperature at centre area is 10.5°C.

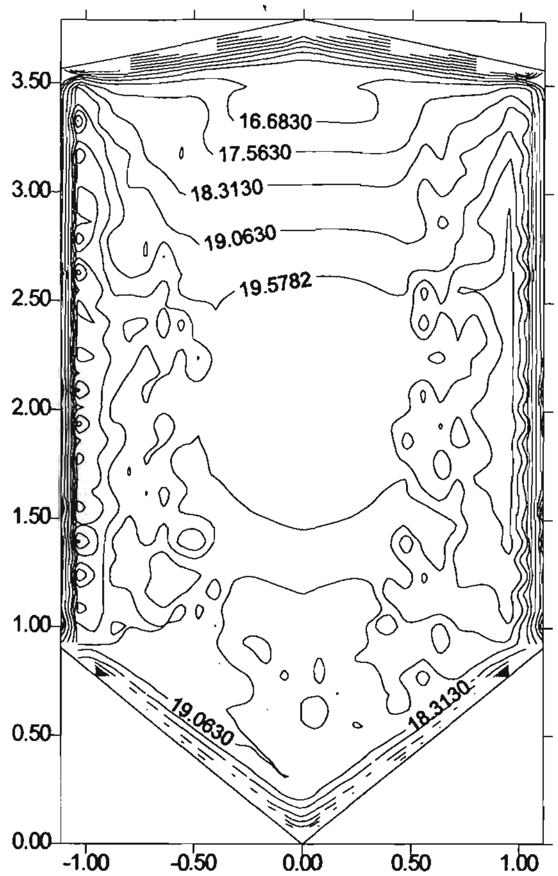
In figure 7.20, it can be seen that in the non-aerated silo, after 20 days storage, the highest wet bulb temperature in the central region of the silo is about 21°C. After 40 days storage, the wet bulb temperature is continues to reduce due principally to conduction cooling. After 60 days storage, the wet bulb temperature at upper surface of grain has fallen to 17.3°C.

In the silo fitted with a linear aeration duct (as shown in figure 7.21), the wet bulb temperature falls most quickly in the region near the duct. It can be observed that a cooling front passes from the duct, and that the region of grain furthest from the duct is the slowest to cool. It is seen that little cooling occurs after day 40, when there appears to be a dynamic equilibrium set up by continued cooling being offset by conduction heating.

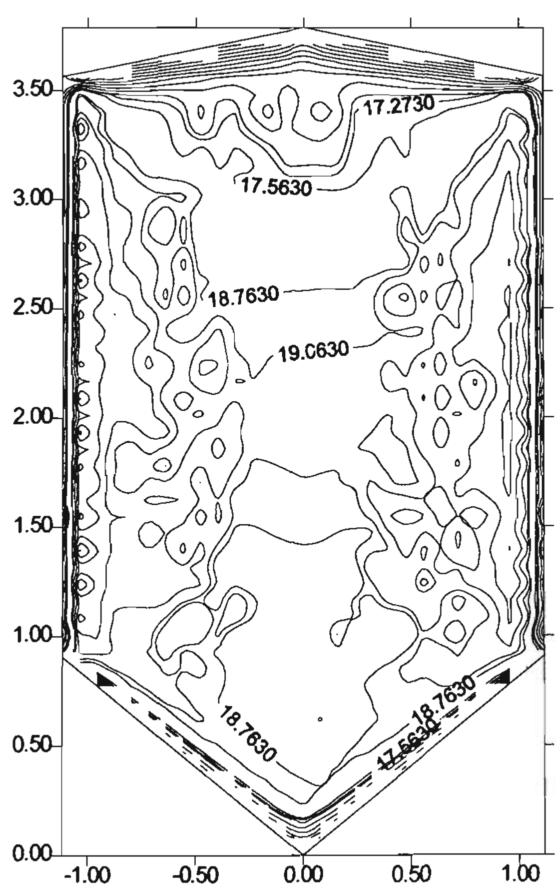
Figure 7.22 again shows, the passage of a cooling front from the annular duct and some dynamic equilibrium also appear to be approached. Must of the grains have attained wet bulb temperature around 10°C, which is inimical to insect population growth. It is noted that the distribution of the intergranular wet bulb temperature is asymmetrical as a result of uneven heating of the silo wall caused by the combined effects of solar radiation and ambient temperature.



Wet bulb temperature at day 20 (non-aerated)

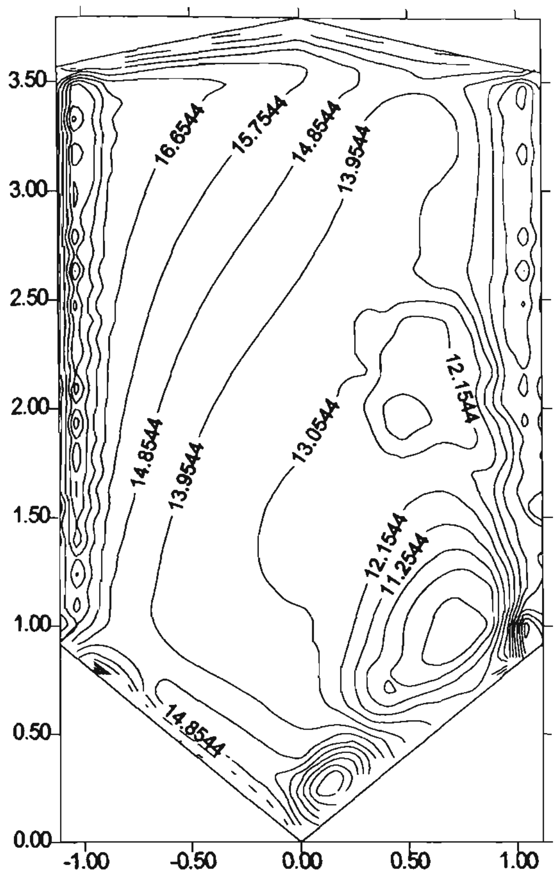


Wet bulb temperature at day 40 (non-aerated)

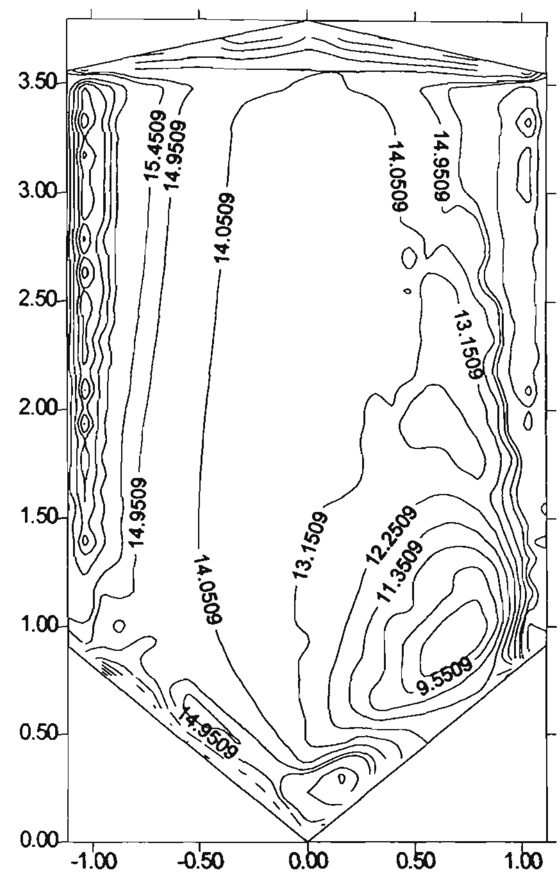


Wet bulb temperature at day 60 (non-aerated)

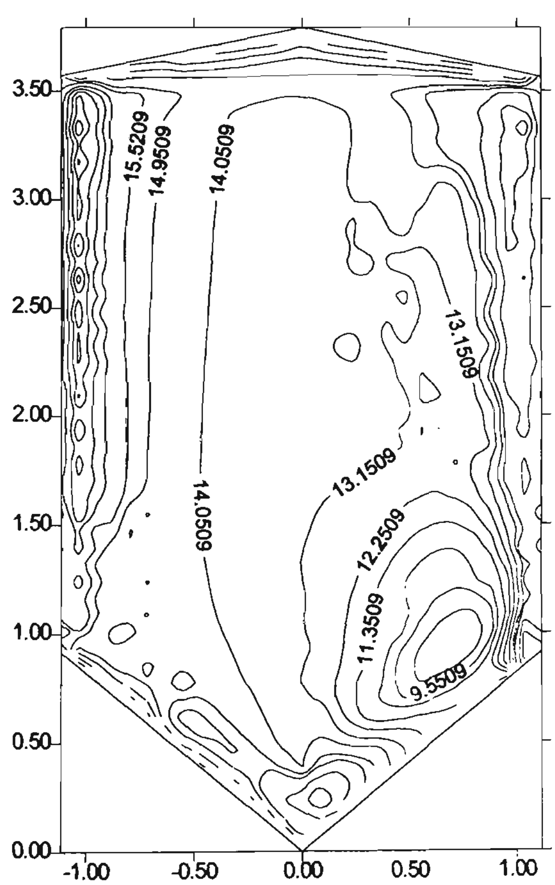
Figure 7.20 Wet bulb temperature distributions in the non-aerated silo



Wet bulb temperature at day 20 (linear duct)

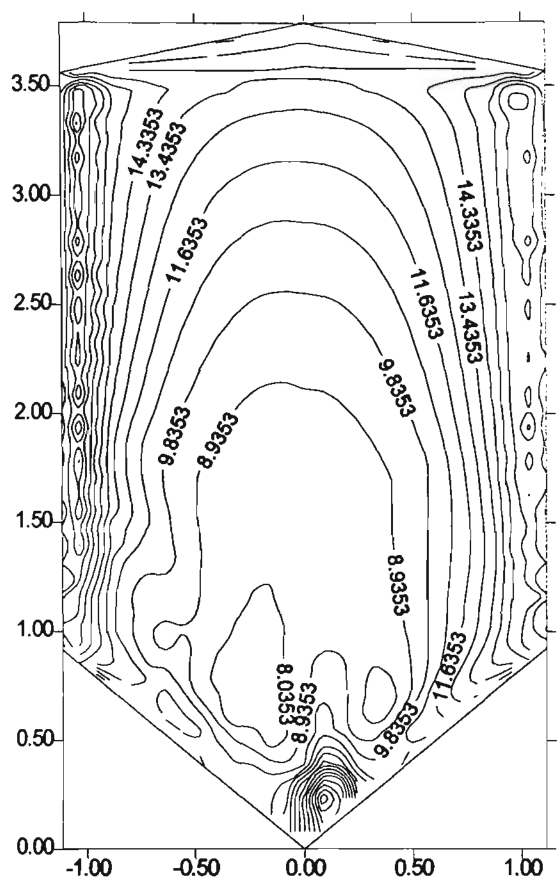


Wet bulb temperature at day 40 (linear duct)

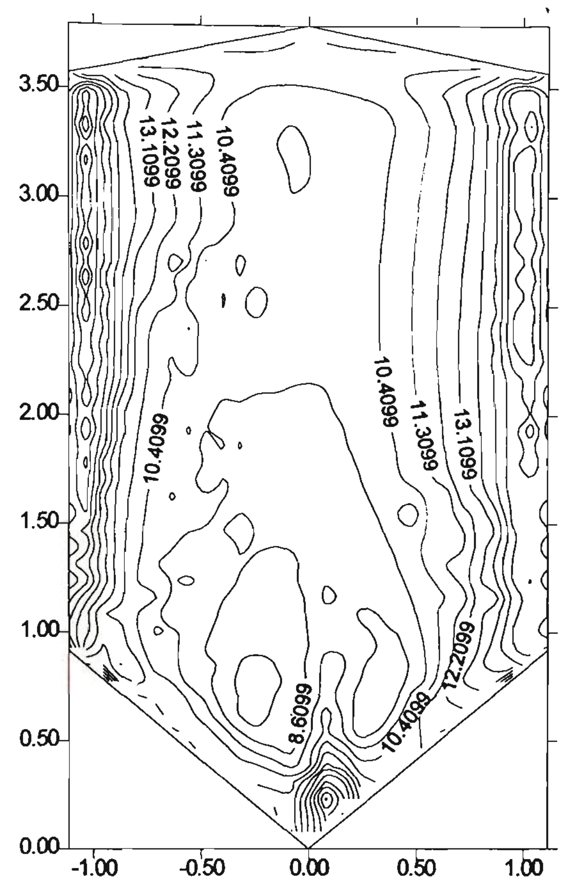


Wet bulb temperature at day 60 (linear duct)

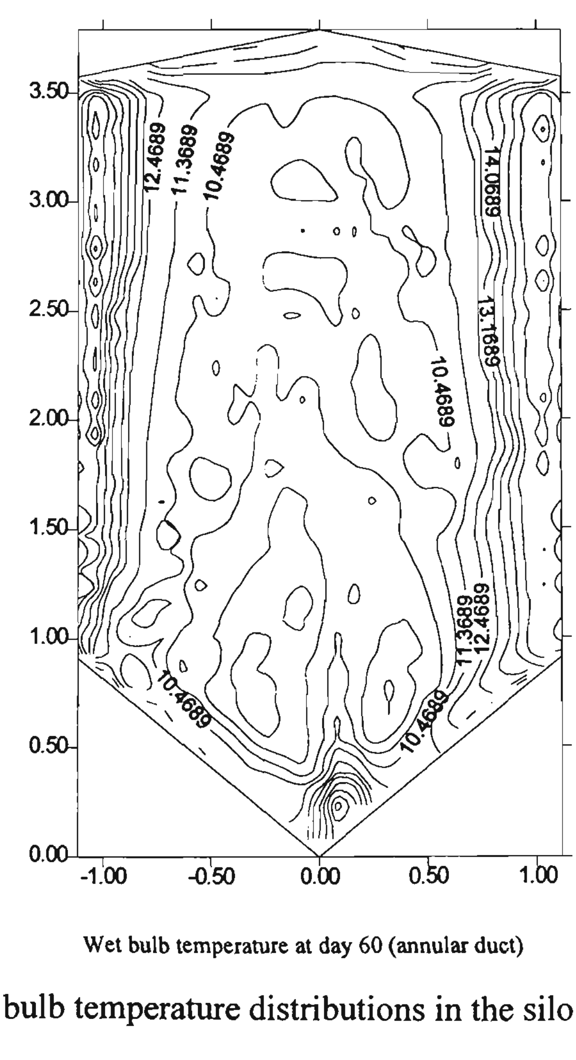
Figure 7.21 Wet bulb temperature distributions in the silo fitted with a linear duct



Wet bulb temperature at day 20 (annular duct)



Wet bulb temperature at day 40 (annular duct)



Wet bulb temperature at day 60 (annular duct)

Figure 7.22 Wet bulb temperature distributions in the silo fitted with an annular duct

7.7 DRY MATTER LOSS DISTRIBUTIONS

Dry matter loss of grain results from an oxidative process accompanied by the release of carbon dioxide, water and heat. In the system considered in this thesis, the amount of dry matter loss occurs as a result of respiration, which occurs in the vicinity of the aeration duct, and the south facing wall. The rate of dry matter loss is a function of grain temperature, grain moisture content and time.

In the non-aerated simulation, no dry matter loss is evident on the east-west section of the grain store, because the grain stays very dry.

Dry matter loss in the silo fitted with the linear duct is observed near the aeration duct, as can be seen from figure 7.26. When the air enters the grain with a high relative humidity, the mass flow rate of air per unit volume of grain is very high in this region, and the grain becomes moist. From the initial dry matter loss of zero, after 20 days aeration, the maximum value of dry matter loss is 0.0136. After 40 days aeration, the maximum value is become to 0.0555, which is 308% higher than after 20 days aeration. After 60 days aeration, the maximum dry matter loss has reached 0.0701, this is 415% higher than after 20 days aeration.

Compared with whole the system, the total loss of dry matter is very small. In addition, the grain moisture content may increase as a results of respiration of moulds.

Moisture content is not observed in the cases of the silo aerated with an annular duct, as shown in figure 7.27, perhaps because the flow rate of air per unit volume of grain is somewhat lower. Hence the moisture in the air is adsorbed by a large volume of grain, so the dry matter loss near the annular duct does not appear to be a problem. Only at the base cone of the silo, the dry matter loss is seen to increase slightly. This is because after the storage period, the respiration of grain appears to occur, thus giving rise to the production of moisture and dry matter loss.

Figures 7.23, 7.24 and 7.25 reflect moisture and temperature changes in the grain adjacent to the south facing portion of the wall. It is clear that aeration appear to have negligible effect on the rate of dry matter loss in these south facing regions.

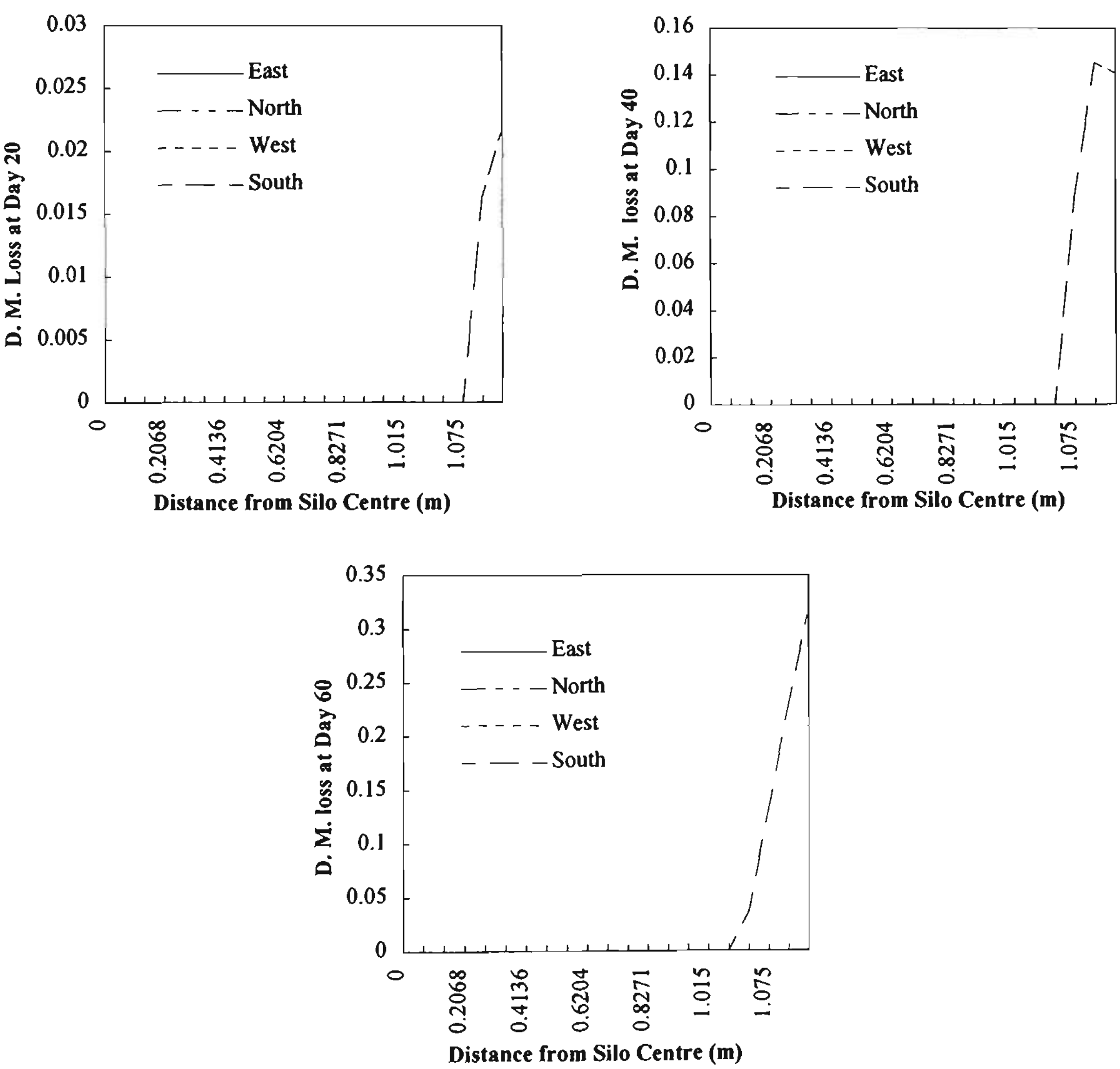


Figure 7.23 Radial profiles of the dry matter loss in the non-aerated case

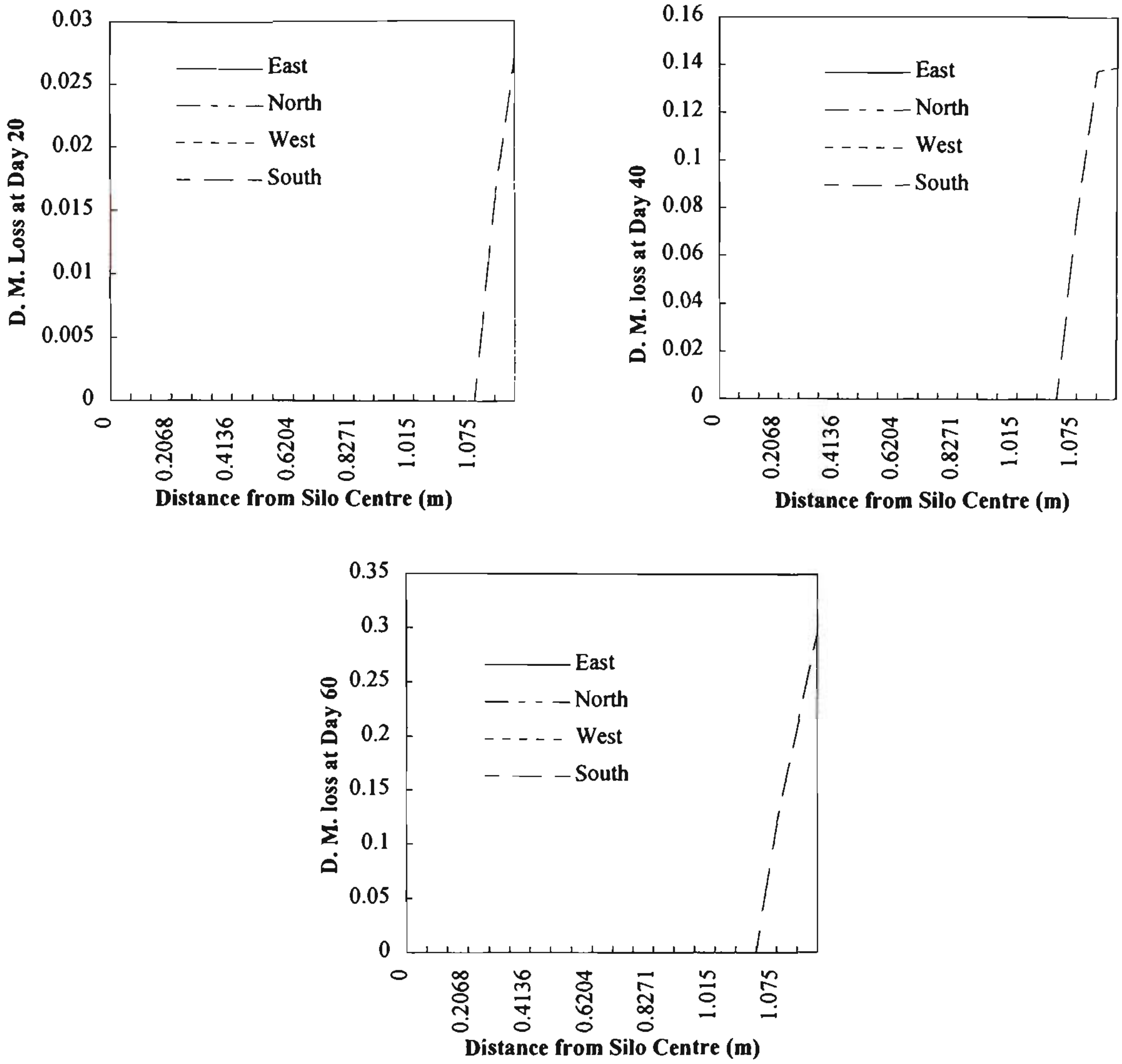
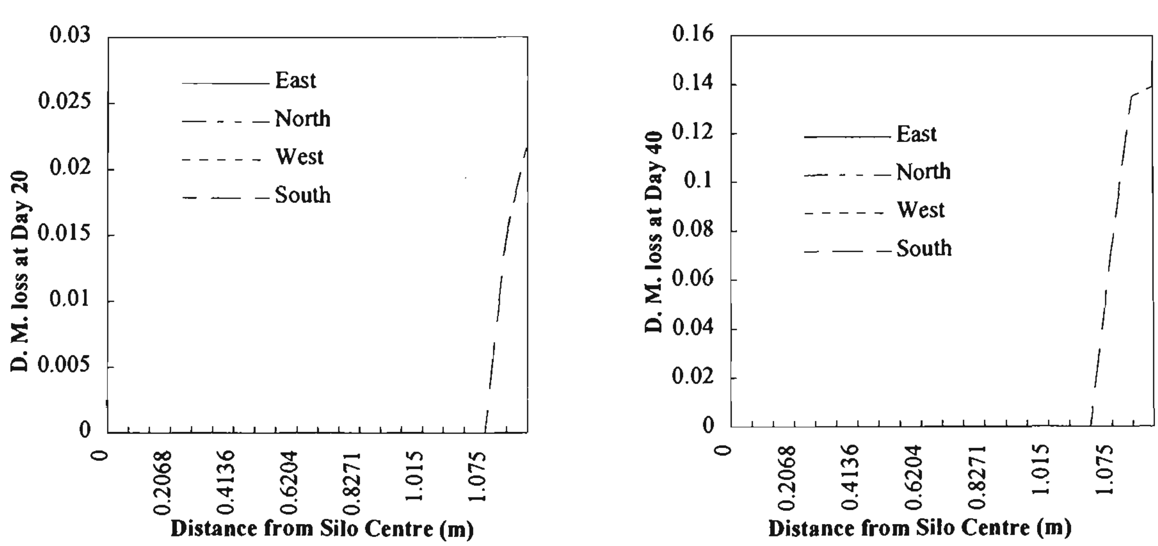


Figure 7.24 Radial profiles of the dry matter loss in the linear duct case



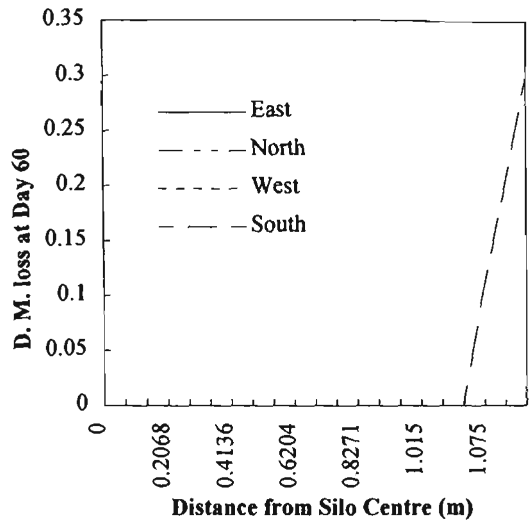
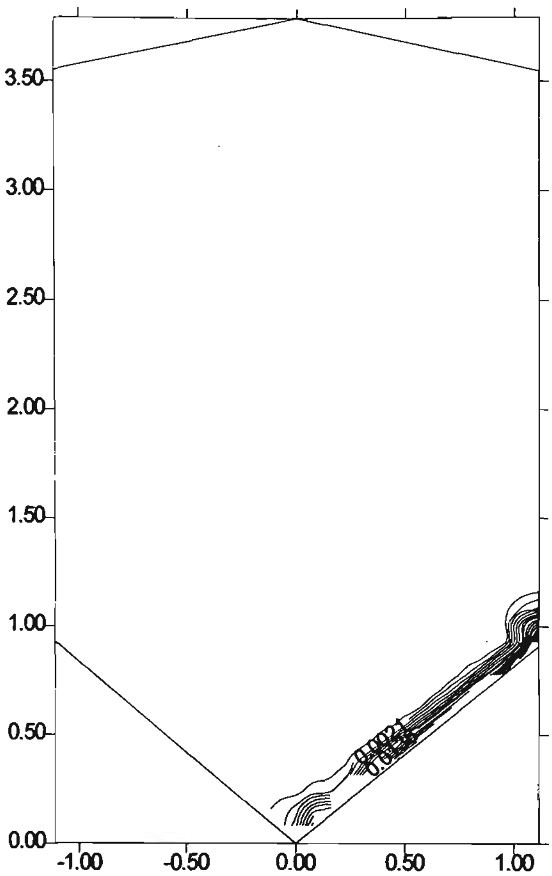
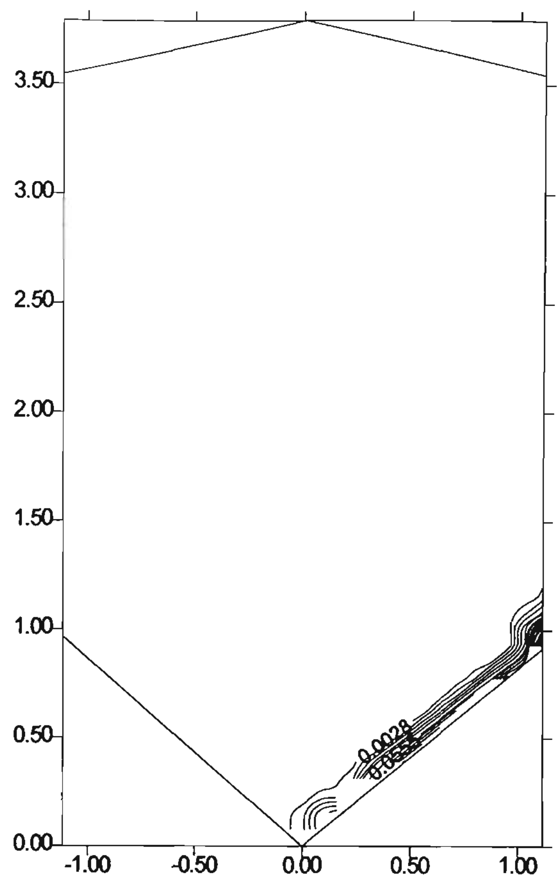


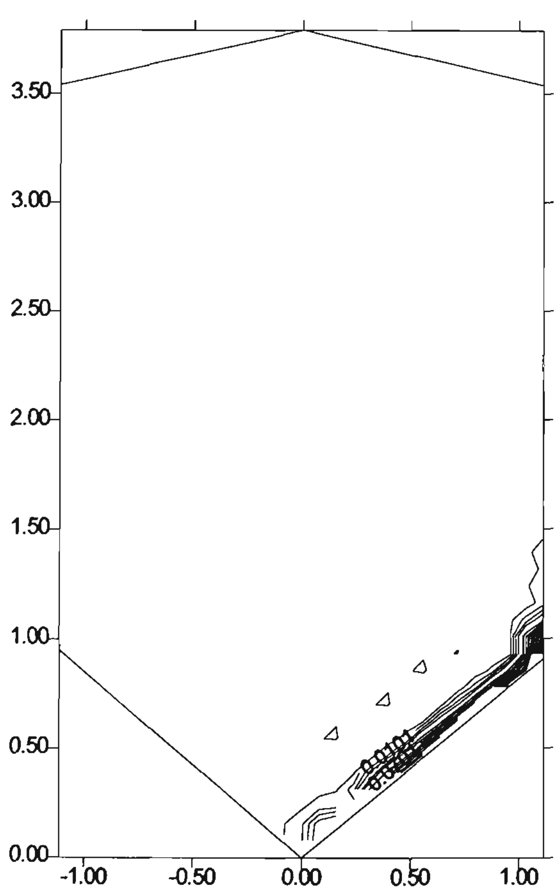
Figure 7.25 Radial profiles of the dry matter loss in the annular duct case



Dry matter loss at day 20 (linear duct)

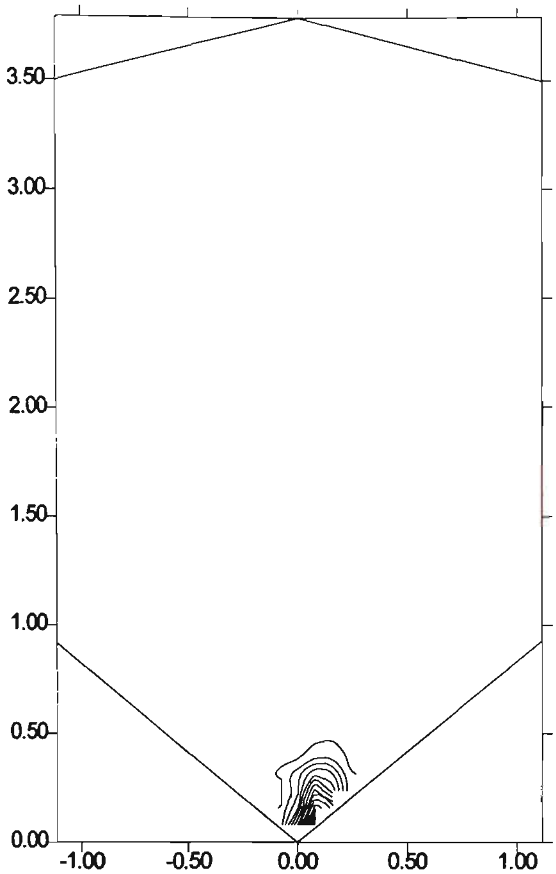


Dry matter loss at day 40 (linear duct)

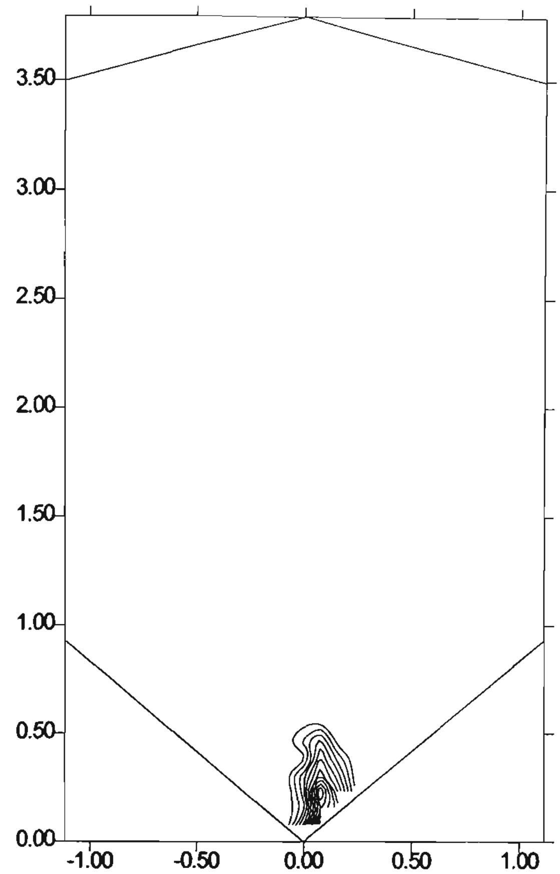


Dry matter loss at day 60 (linear duct)

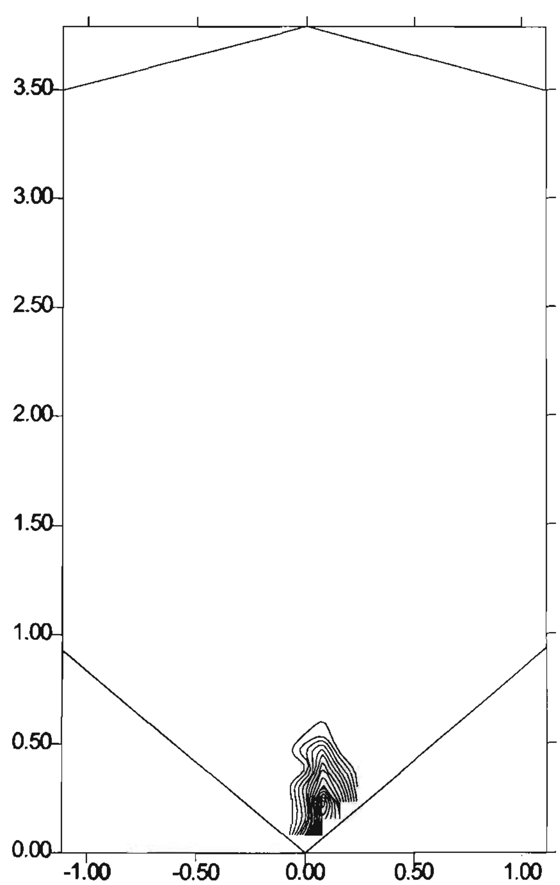
Figure 7.26 Dry matter loss distributions in the silo fitted with a linear duct



Dry matter loss at day 20 (annular duct)



Dry matter loss at day 40 (annular duct)



Dry matter loss at day 60 (annular duct)

Figure 7.27 Dry matter loss distributions in the silo fitted with an annular duct

7.8 INSECT POPULATION DISTRIBUTIONS

Insects such as *Sitophilus oryzae, Rhyzopotha dominica* and *Sitophilas zeamais* are likely to infest stored grain, and spoil its quality. Insect population growth has been linearly related to the wet bulb temperature of the intergranular air. As a result, insect population often grow rapidly in unaerated bulk of grain.

The potential for insect population growth is shown in figures 7.31 to 7.42. After two months storage in a non-aerated silo, the population of the strain *S. oryzae 1* has increased 180 fold. Compared with the initial concentration in the central region, it can be seen that aeration system is quite effective in controlling insect population growth, in a silo fitted with a linear aeration duct, the population of insects is 15 fold, and in the silo fitted with an annular duct, the population growth is only 3 fold.

Figure 7.31 shows in the non-aerated silo, the distribution of the more vigorous strain of *S. oryzae 1*, the population increases by about 6 fold in the centre, and typically 4 fold in the upper surface of the grain and bottom region of the bulk, this is because in the central region of the silo, the wet bulb temperature of the air is higher than at the boundaries. After 40 days storage, the insect population increase 36 fold in the central region. After 60 days storage, the upper surface of the grain has 40 insects, and at the bottom of the silo has around 80 fold increase.

Figure 7.32 indicates the effect on insect population growth in a silo fitted with a linear aeration duct. After 20 days aeration, the population of *S. oryzae1* at the centre of the silo has increases by 3.5 fold, and at the warm region of the upper surface of grain, the maximum numbers of insects are about 5.15 fold. After 40 days aeration, the centre area of silo is maintained cool and the population growth is 7.5 fold, and 13.5 fold near the upper surface of the silo. After 60 days aeration, the rate of insects grown is lower at the region is close to the duct, and higher distant from the duct. At the upper surface of grain, the maximum potential for insect growth is 28 times the initial population.

Figure 7.33 shows that after 20 days aeration of the silo fitted with an annular duct, the centre region of the silo has around 2.5 insects more that initially, at the upper surface of the grain, there is about 4.28 fold increase on *S. oryzae 1*. After 40 days of cooling, the rate of insect growth at centre area of the silo is small, this is because cold air provides insects a difficult environment for living, and the population increase is only 3.5 fold. After 60 days aeration, the insect growth at upper surface of the grain has become 12.67 fold.

In a similar way to *S. oryzae1*, which is we discussed above , figure 7.34 shows insect population in an non-aerated silo after 60 days storage. In the centre area of the silo, insect *S. oryzae2* grown to 87 times its initial population. From this point of view, we can see: Storage insect grown rapidly without aeration control, and causing the stored grain completely lost in the short period of time under storage environment. With a linear aeration duct, after 60 days aeration, *S. oryzae2* only has increase 9 fold around the centre of the silo. Similarly, after two months aeration with the annular aeration duct, the population of the loss vigorous strain of *S. oryzae2* has increased 2.77 times its initial value.

Figure 7.37 shows, the distributions of *R. dominica* in the non-aeration silo. After 60 days storage, the number of *R. dominica* at the centre region of the silo increases 16.5 fold. In the silo fitted with a linear duct (as shown in figure 7.38), after two months aeration cooling control, in the centre area of the silo the population of *R. dominica* has increased by 2.1 fold. In the silo fitted with an annular duct (as shown in figure 7.39), after 60 days period storage, the centre area of insect *R. dominica* is only 1.5 fold. It is clear that this species is not particular cold tolerant.

The final insect species which are discussed in this thesis is *S. zeamais*. Figure 7.40 shows that the simulation results of *S. zeamais* distributions after two months storage without aeration. It is clear that at the centre area of the silo, the *S. zeamais* growth is 2.7 fold. In a silo fitted with a linear duct (see figure 7.41), the storage management keep the insect population under good control, only about 1.3 fold of insects *S. zeamais* in the centre region of the silo. When a silo is fitted with an annular duct (see figure 7.42), the population of *S. zeamais* appear to be very tightly controlled, and after 60 days storage, the population of *S. zeamais* has increased by about only 1.2 fold.

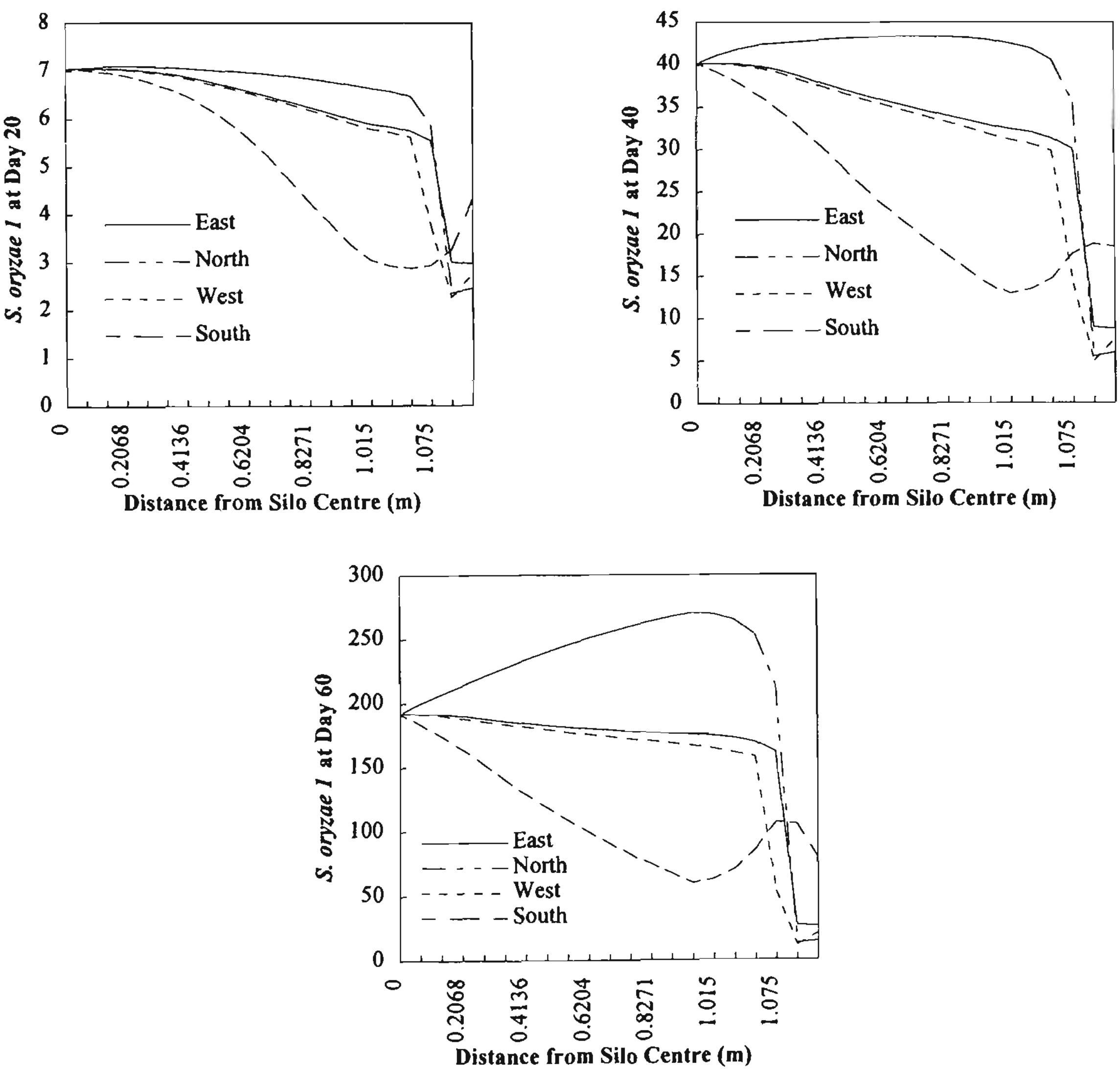


Figure 7.28 Radial profiles of the population of *S. oryzae1* in the non-aerated case

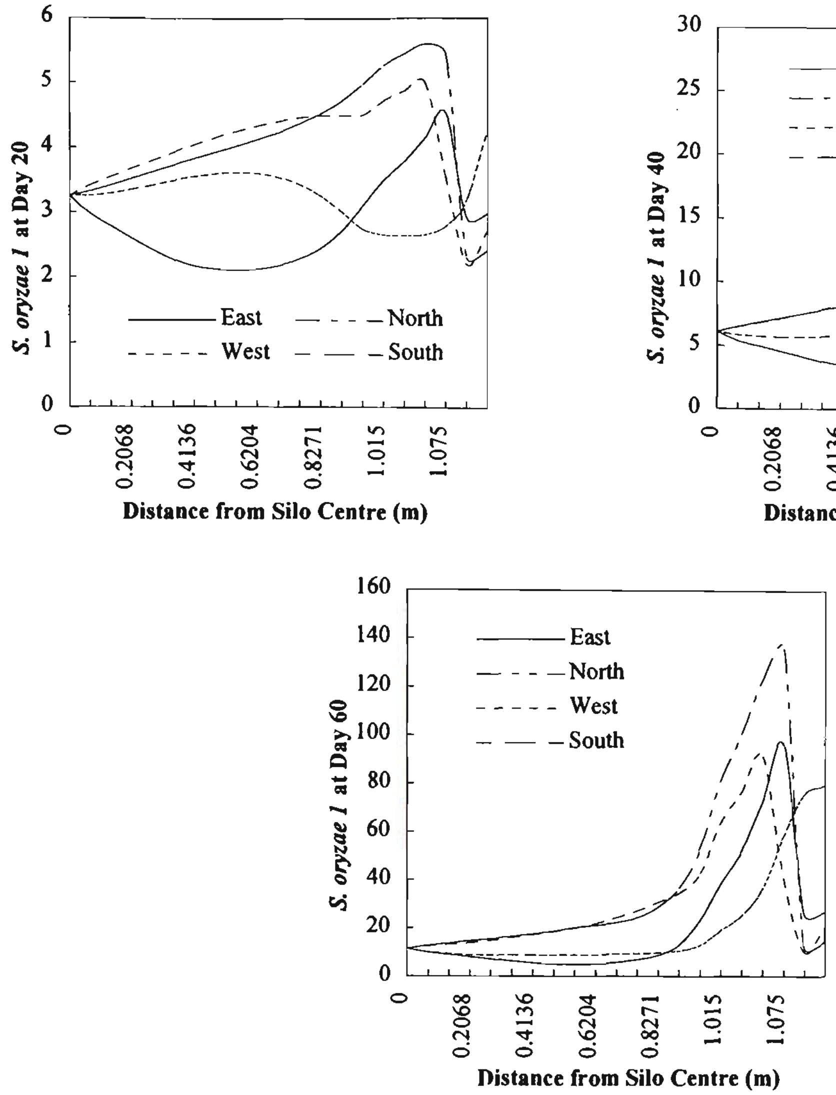
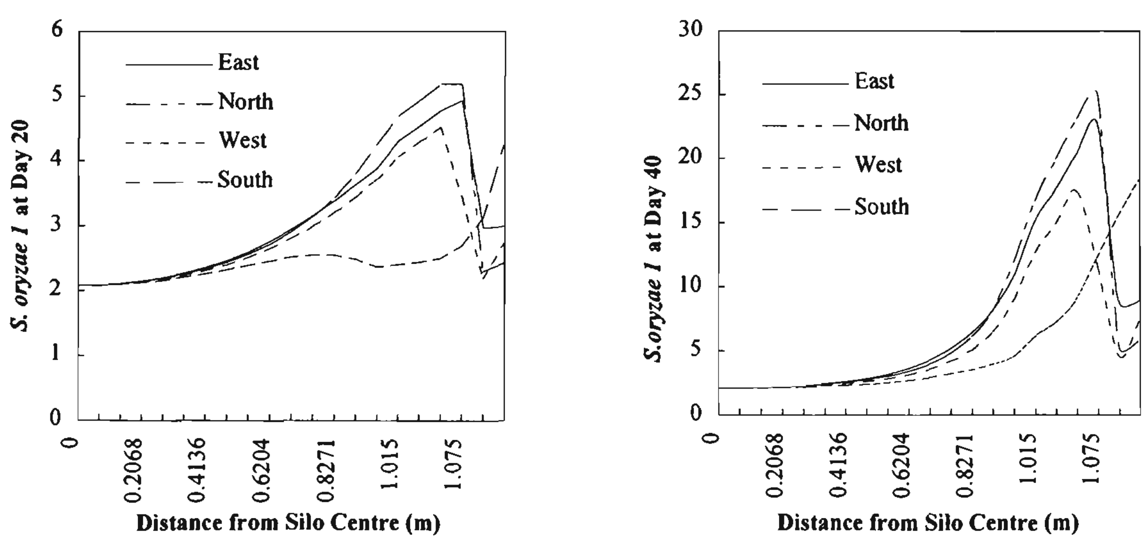


Figure 7.29 Radial profiles of the population of *S. oryzae1* in the linear duct case



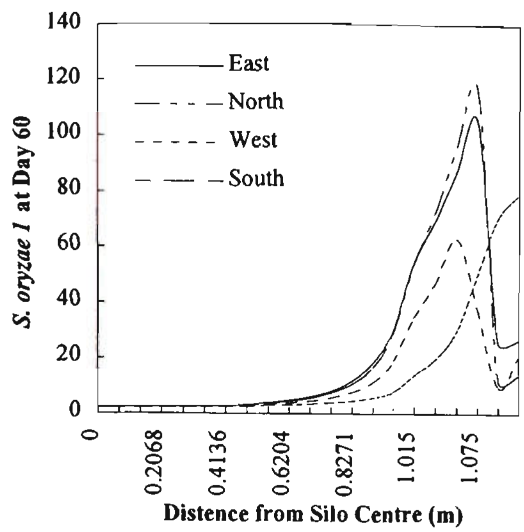
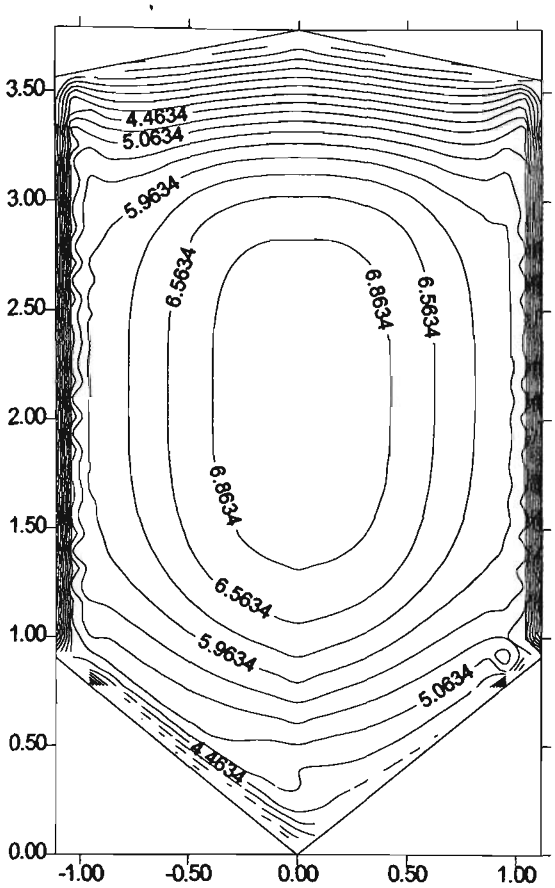


Figure 7.30 Radial profiles of the population of *S. oryzae1* in the annular duct case

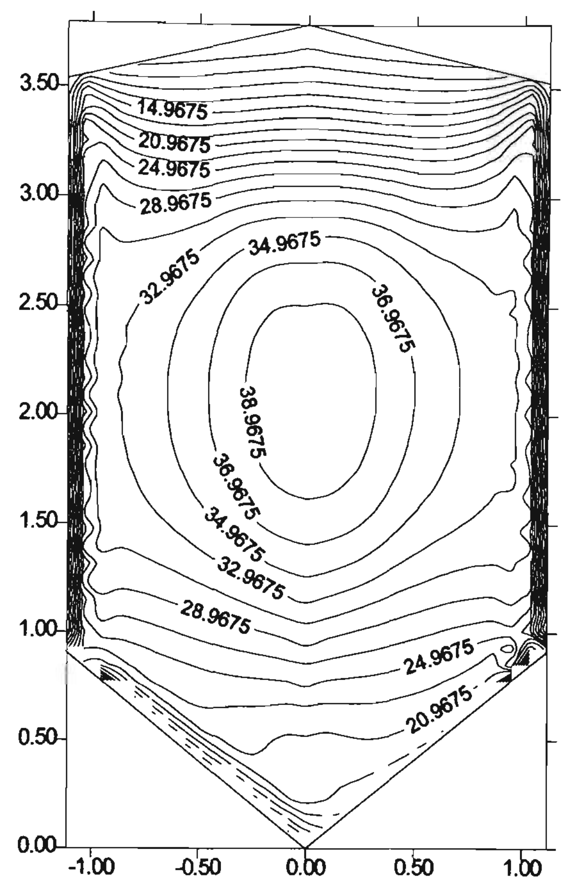
Figure 7.28 shows the insects of *S. oryzae1* along east, south, west and south radii of a non-aerated silo. It is clear to see that population of *S. oryzae1* continue to grow rapidly in the centre of the silo throughout the storage period. However, the population growth near the east, north and west facing regions of the silo is low, because of the high temperatures of the insolation during the noon. Population growth on the south side of the silo is generally lower, because of the lower grain temperatures that result from heat transfer process occurring the central region of the silo.

In the aerated silo fitted with a linear duct (as shown in figure 7.29), the population of *S. oryzae1* growth is lower near the centre of the silo, and higher within a 10 centimetre of the wall, this is because that the linear aeration system has controlled the central region of the silo, and the temperature along the wall of the silo is heated by solar radiation and ambient air during the day. The density of population *S. oryzae1* growth on the east, north and west facing of the silo is very low due to the high temperature occurring during the day.

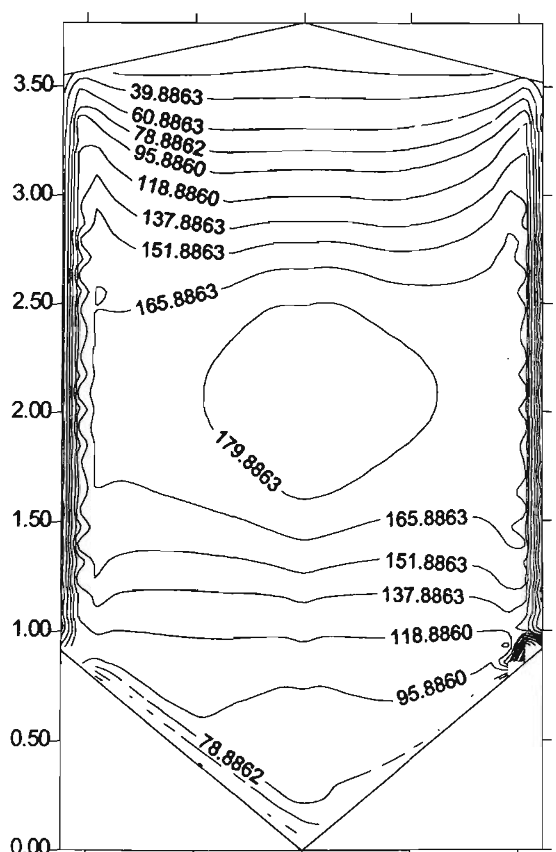
Similar with an annular duct (as shown in figure 7.30), it can be seen that the controlling of population growth of *S. oryzae1* is better than that linear aeration system in the centra region of the silo.



S. oryzae 1 at day 20 (non-aerated)

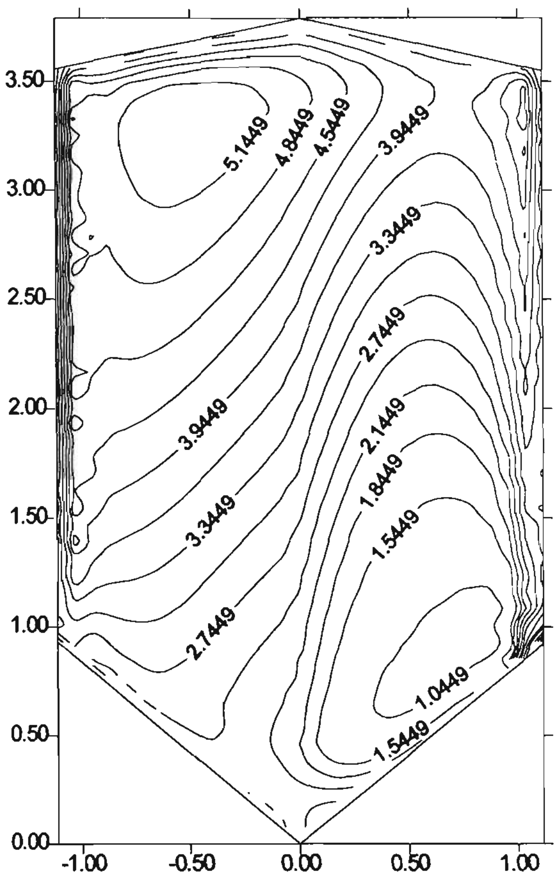


S. oryzae 1 at day 40 (non-aerated)

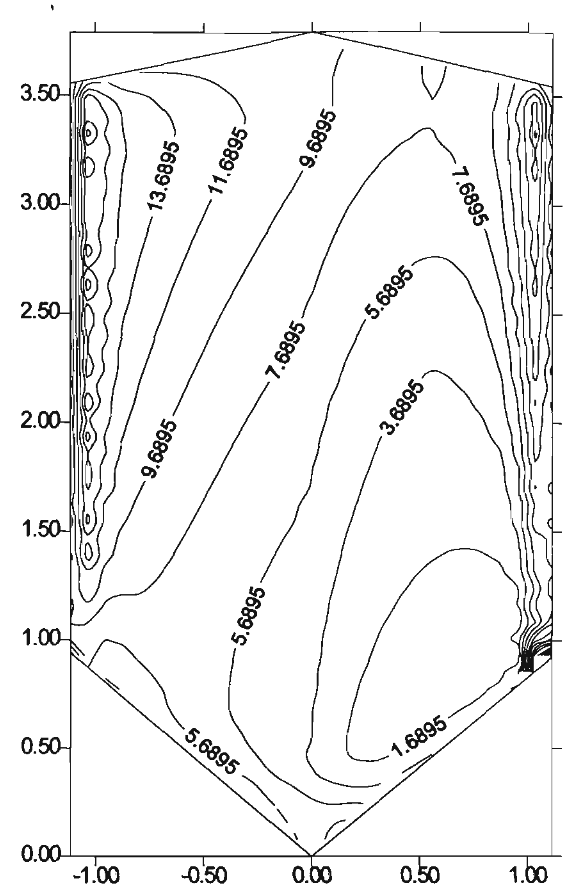


S. oryzae 1 at day 60 (non-aerated)

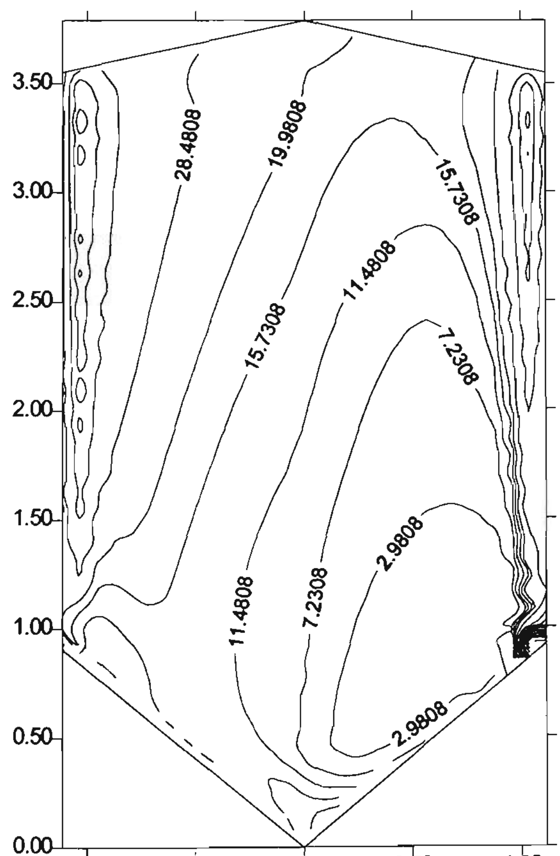
Figure 7.31 *S. oryzae 1* distributions in the non-aerated silo



S. oryzae 1 at day 20 (linear duct)

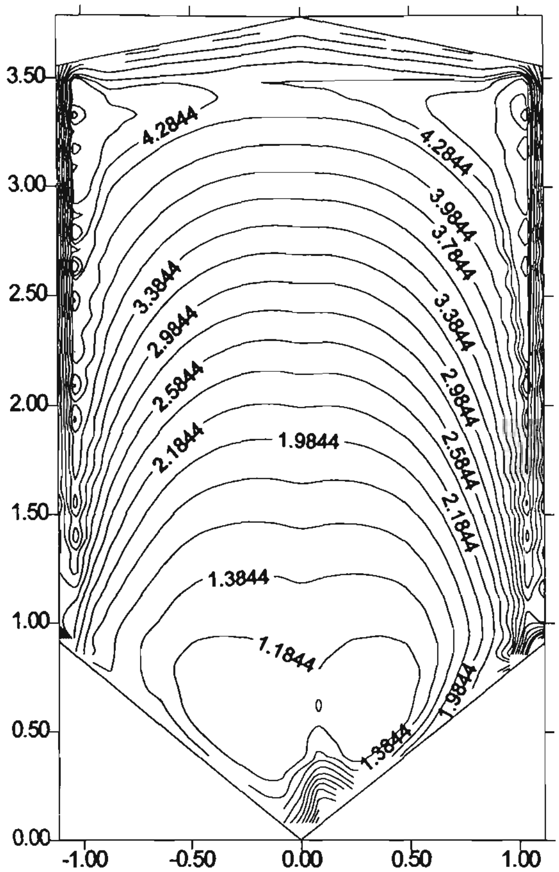


S. oryzae 1 at day 40 (linear duct)

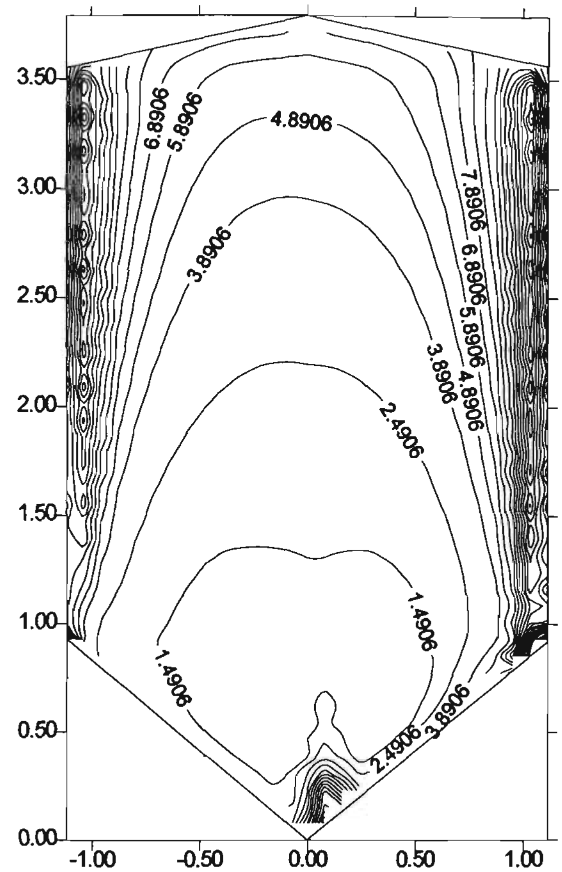


S. oryzae 1 at day 60 (linear duct)

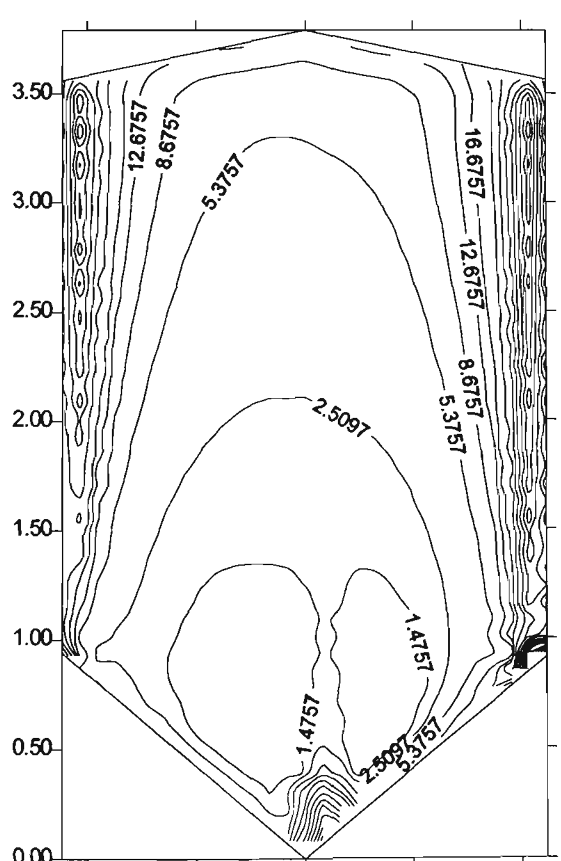
Figure 7.32 *S. oryzae 1* distributions in the silo fitted with a linear duct



S. oryzae 1 at day 20 (annular duct)

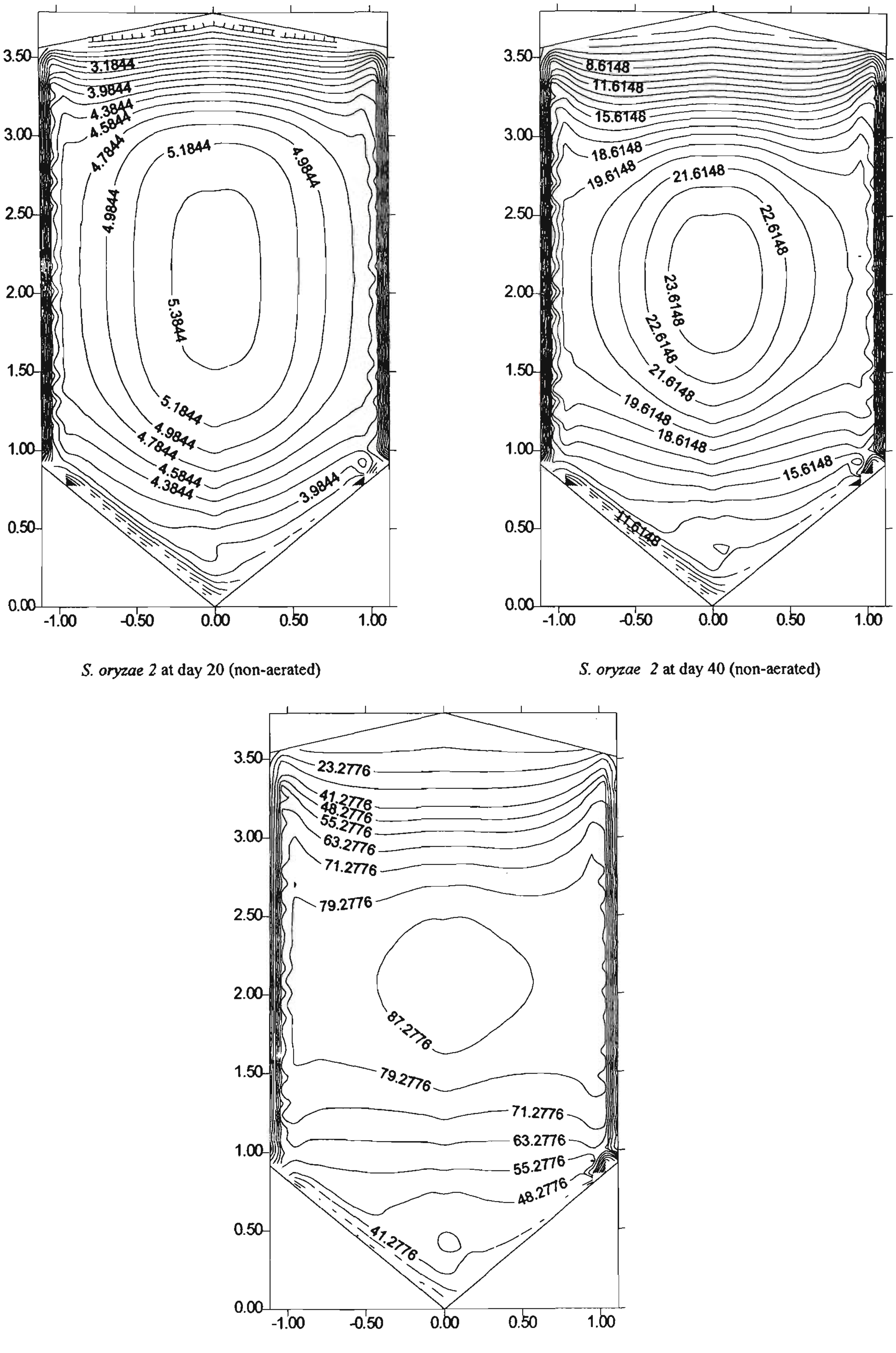


S. oryzae 1 at day 40 (annular duct)



S. oryzae 1 at day 60 (annular duct)

Figure 7.33 *S. oryzae 1* distributions in the silo fitted with an annular duct

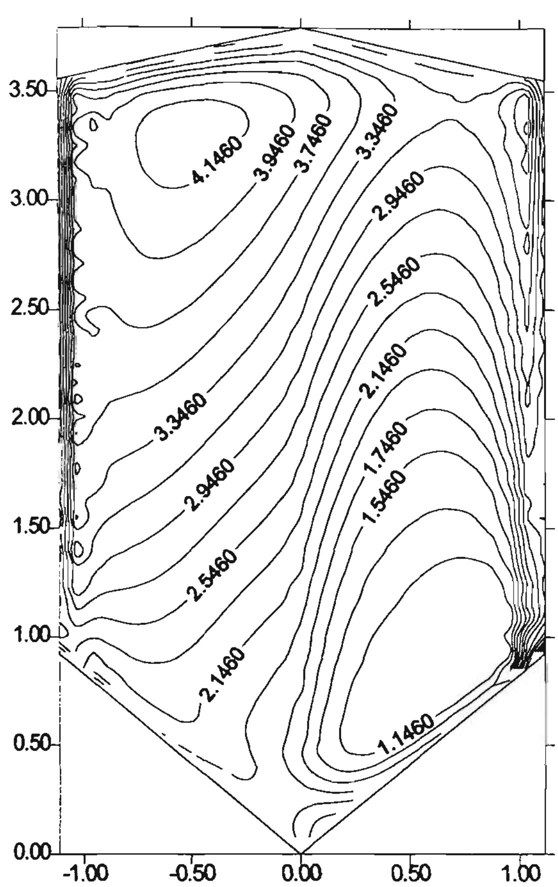


S. oryzae 2 at day 20 (non-aerated)

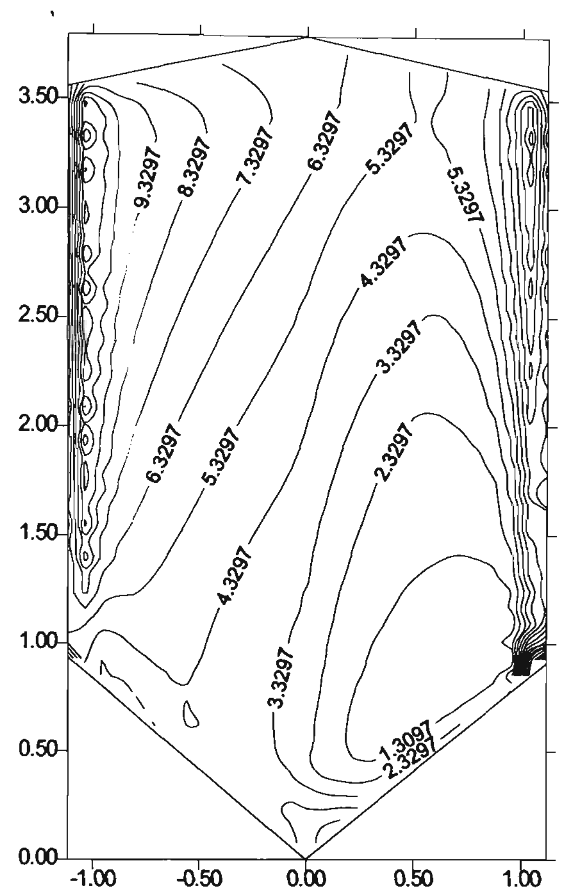
S. oryzae 2 at day 40 (non-aerated)

S. oryzae 2 at day 60 (non-aerated)

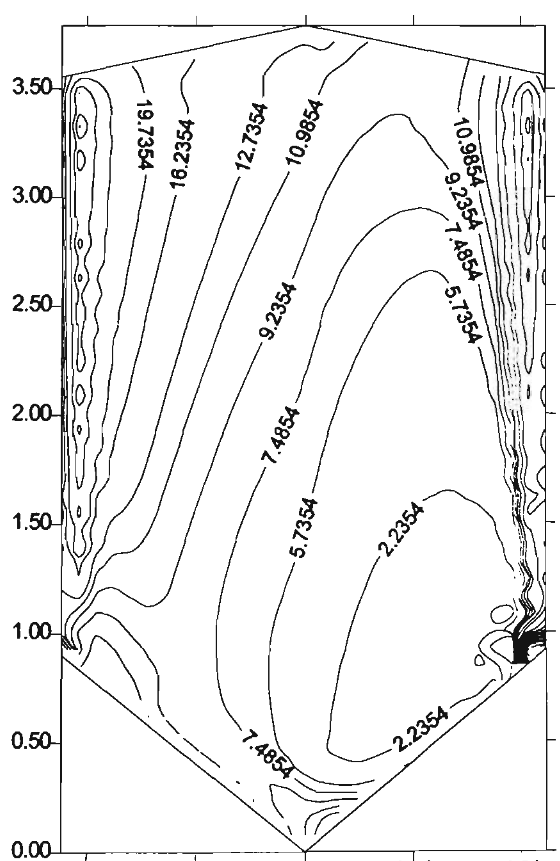
Figure 7.34 *S. oryzae 2* distributions in the non-aerated silo



S. oryzae 2 at day 20 (linear duct)

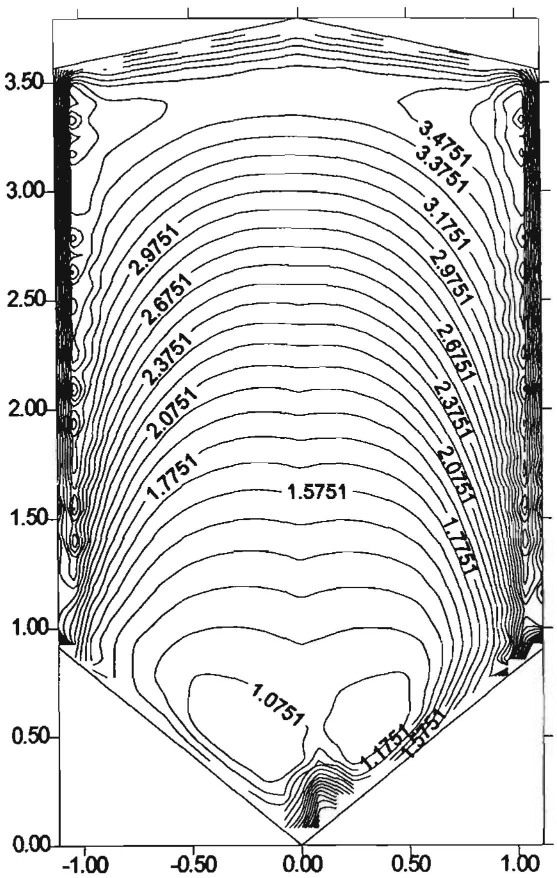


S. oryzae 2 at day 40 (linear duct)

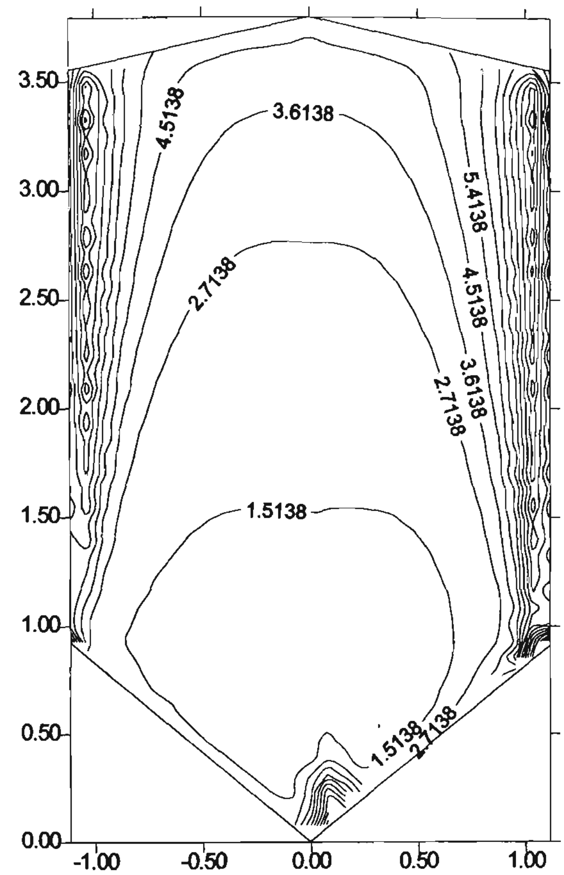


S. oryzae 2 at day 60 (linear duct)

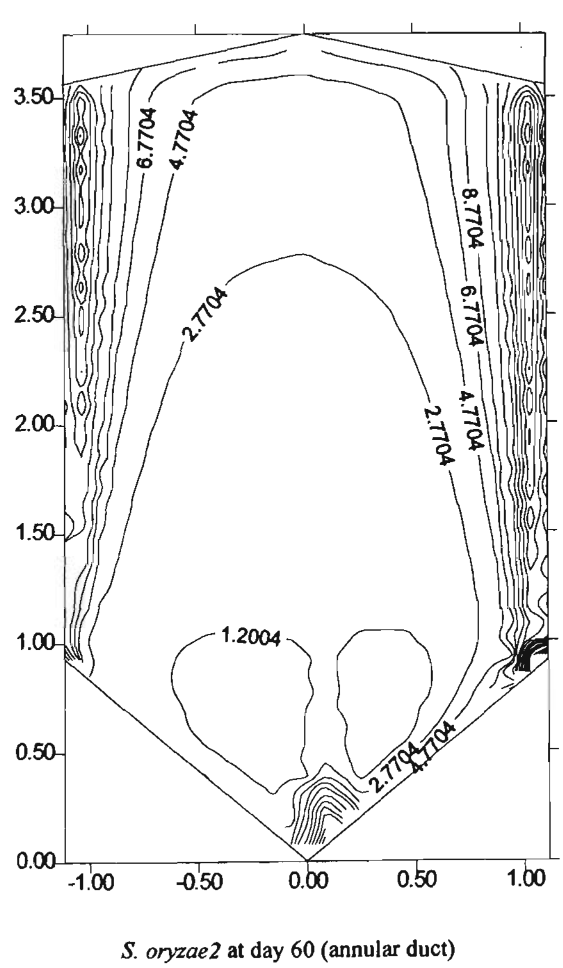
Figure 7.35 *S. oryzae2* distributions in the silo fitted with a linear duct



S. oryzae 2 at day 20 (annular duct)

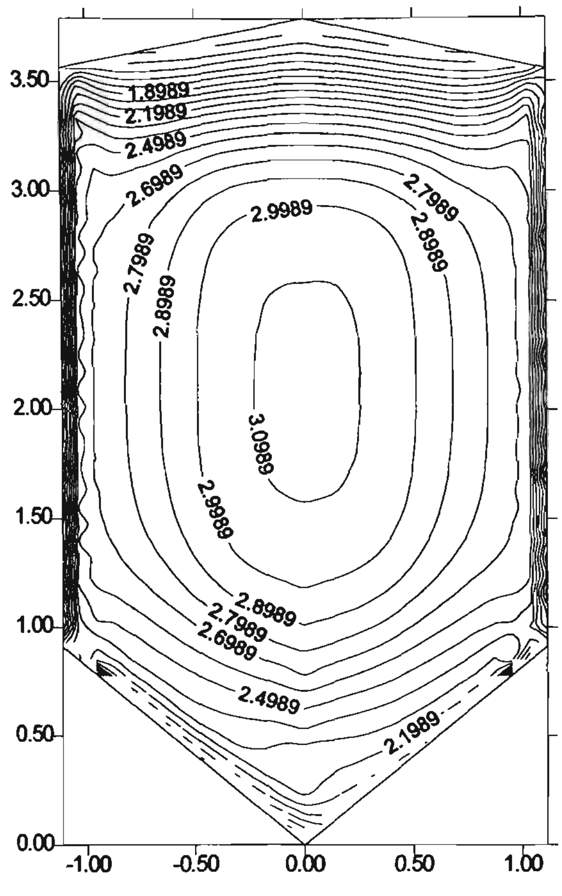


S. oryzae 2 at day 40 (annular duct)

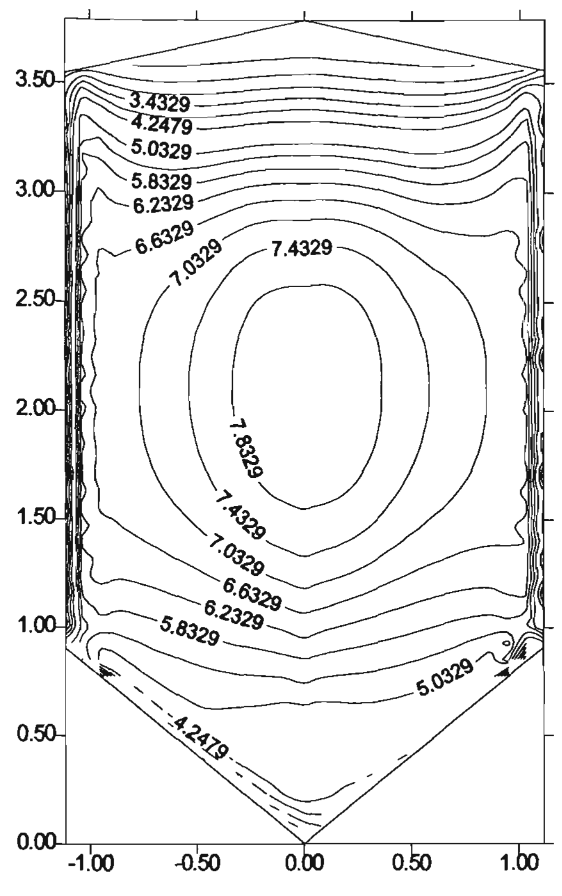


S. oryzae2 at day 60 (annular duct)

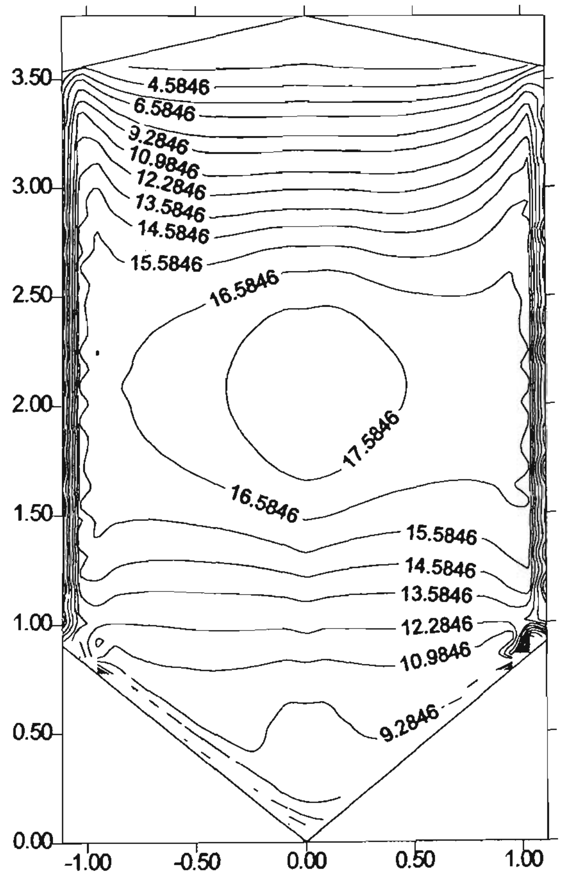
Figure 7.36 *S. oryzae 2* distributions in the silo fitted with an annular duct



R. dominica at day 20 (non-aerated)

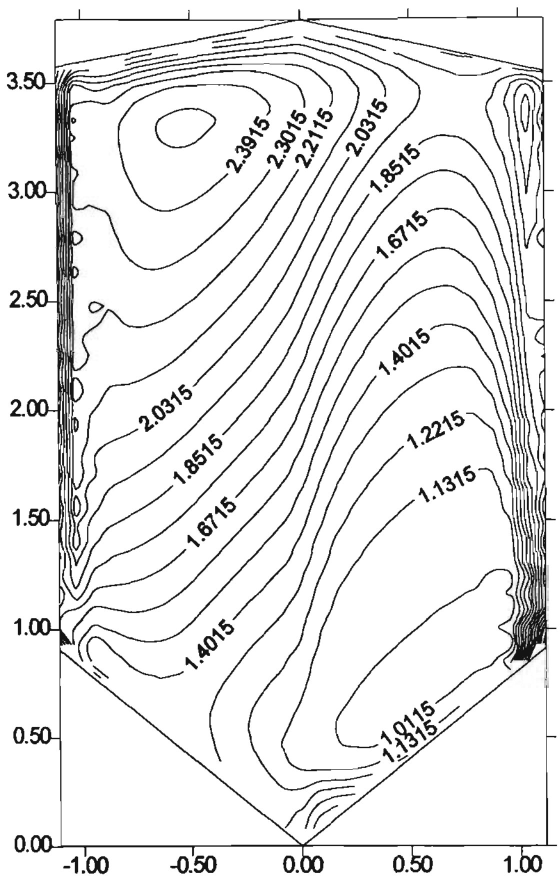


R. dominica at day 40 (non-aerated)

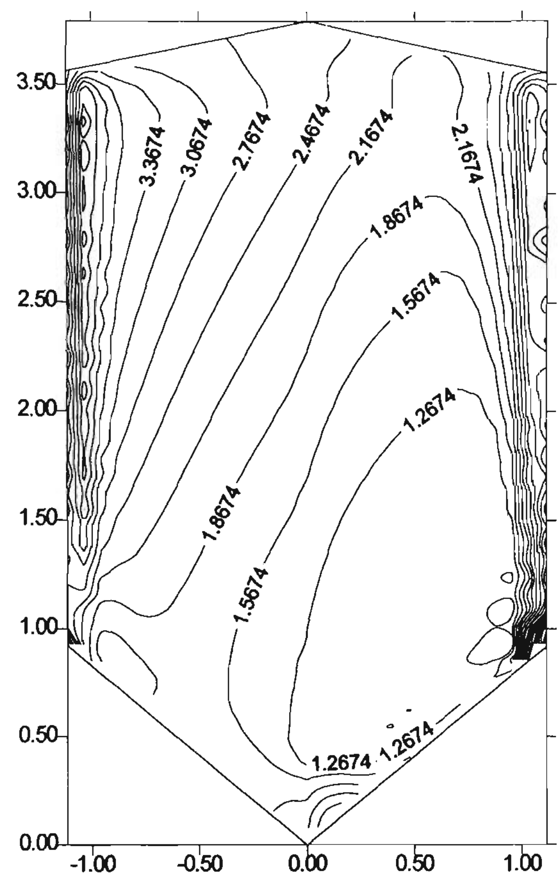


R. dominica at day 60 (non-aerated)

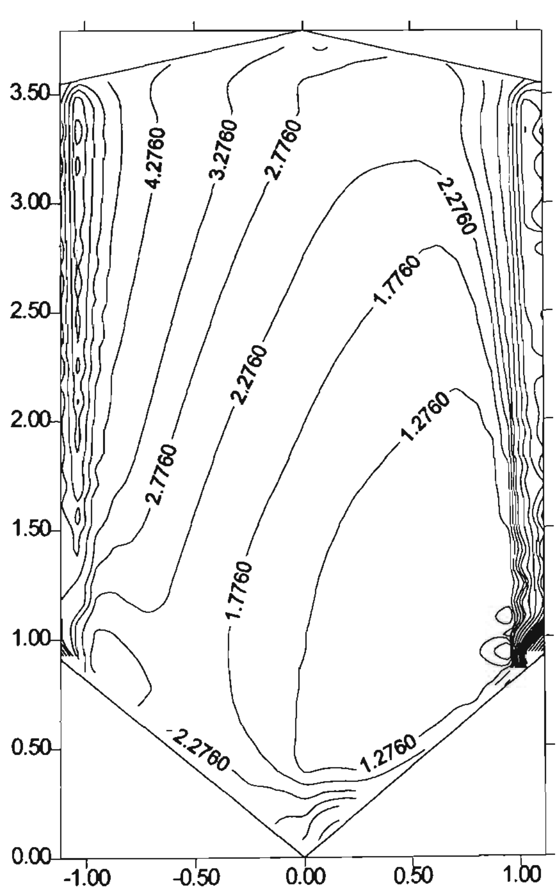
Figure 7.37 *R. dominica* distributions in the non-aerated silo



R. dominica at day 20 (linear duct)

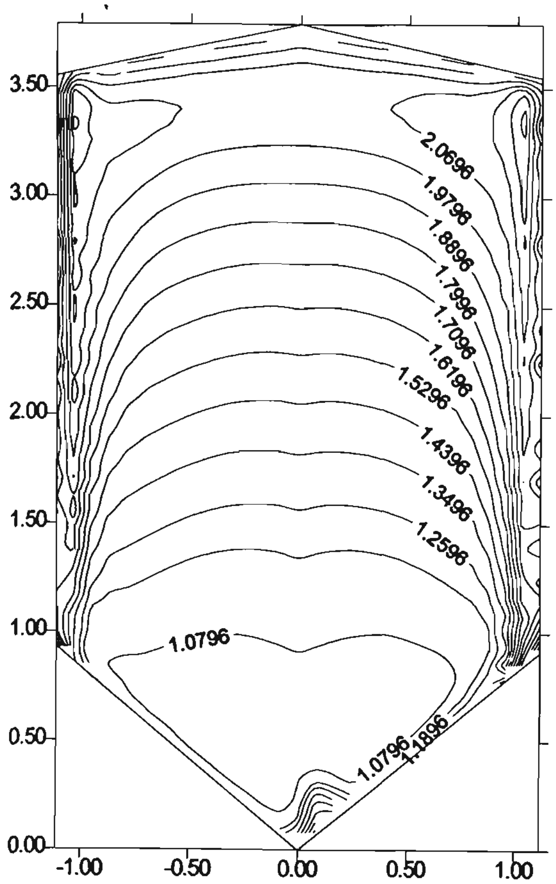


R. dominica at day 40 (linear duct)

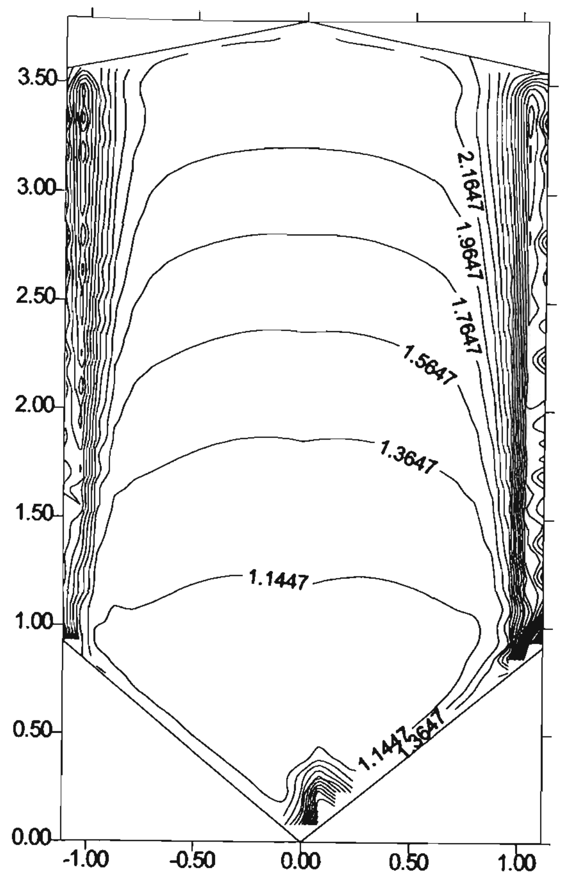


R. dominica at day 60 (linear duct)

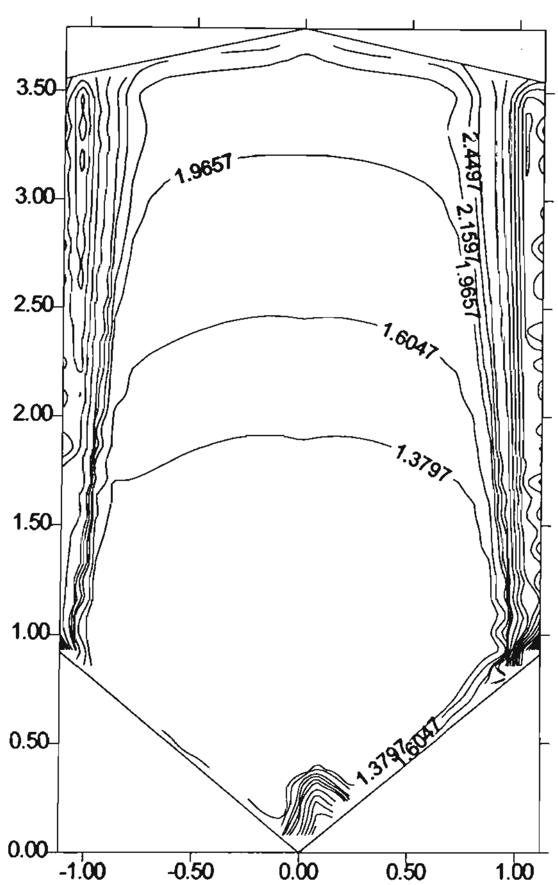
Figure 7.38 *R. dominica* distributions in the silo fitted with a linear duct



R. dominica at day 20 (annular duct)

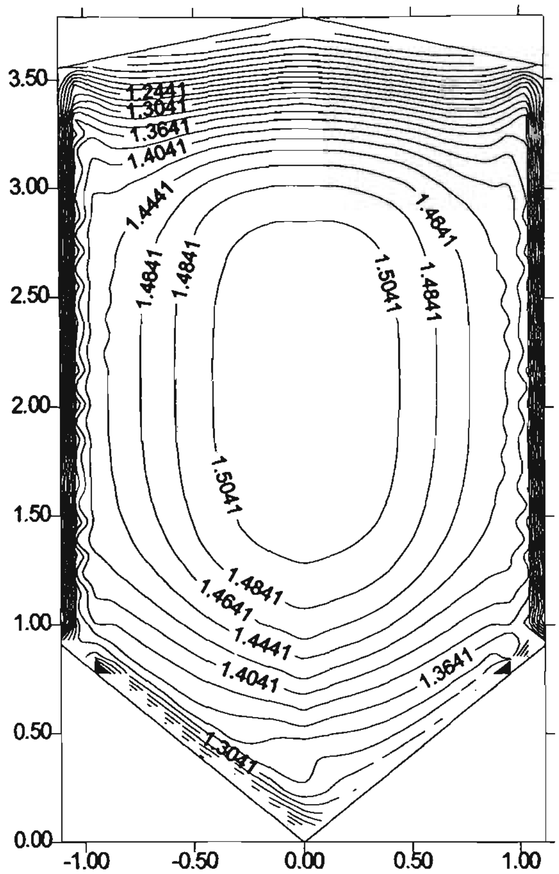


R. dominica at day 40 (annular duct)

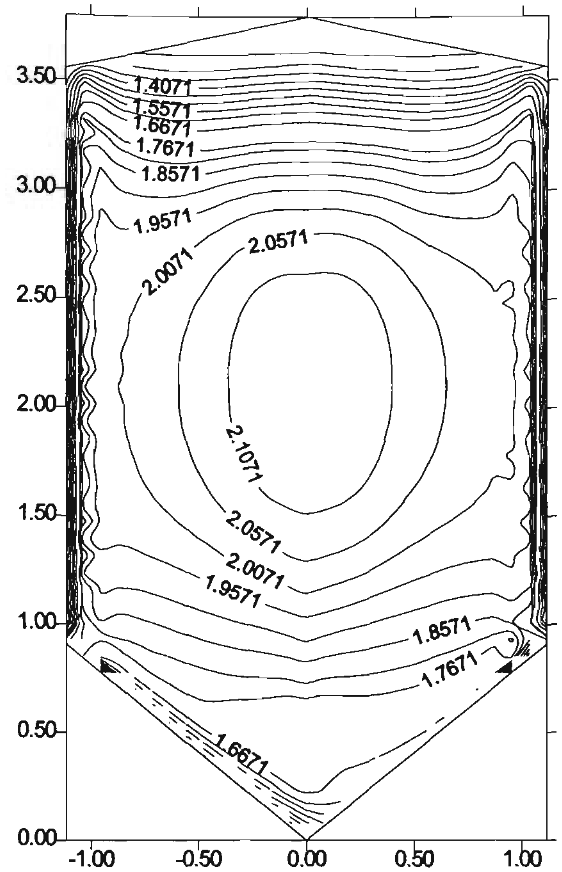


R. dominica at day 60 (annular duct)

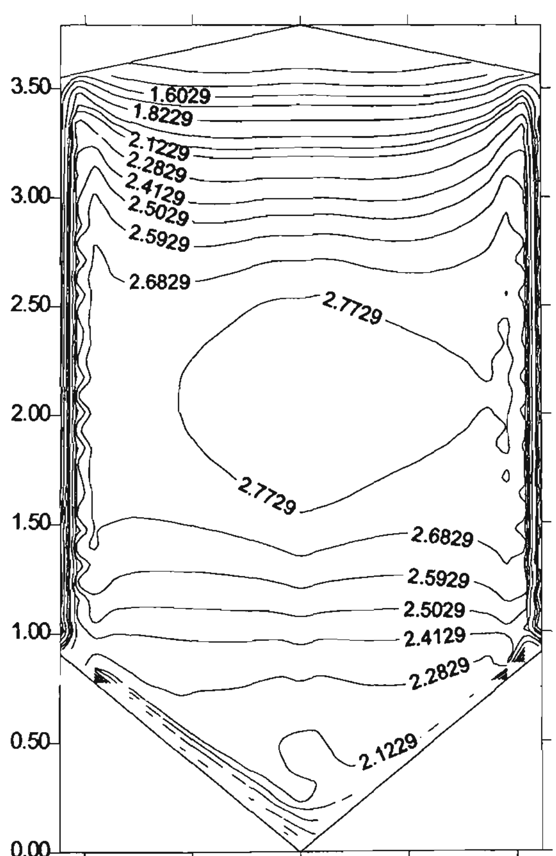
Figure 7.39 *R. dominica* distributions in the silo fitted with an annular duct



S. zeamais at day 20 (non-aerated)

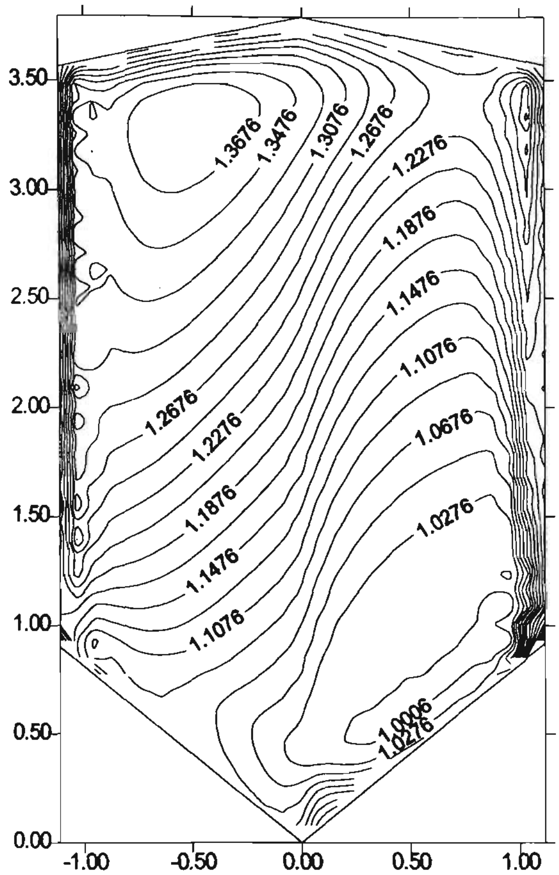


S. zeamais at day 40 (non-aerated)

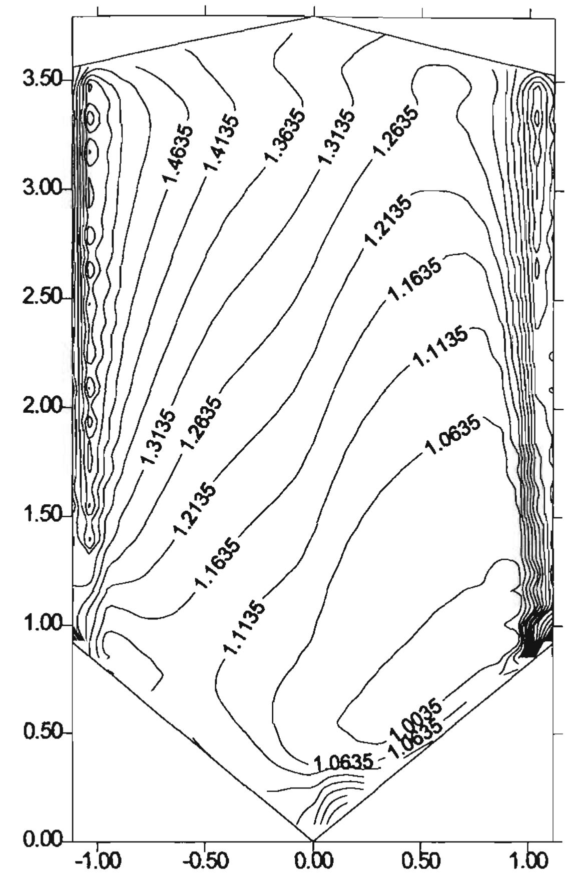


S. zeamais at day 60 (non-aerated)

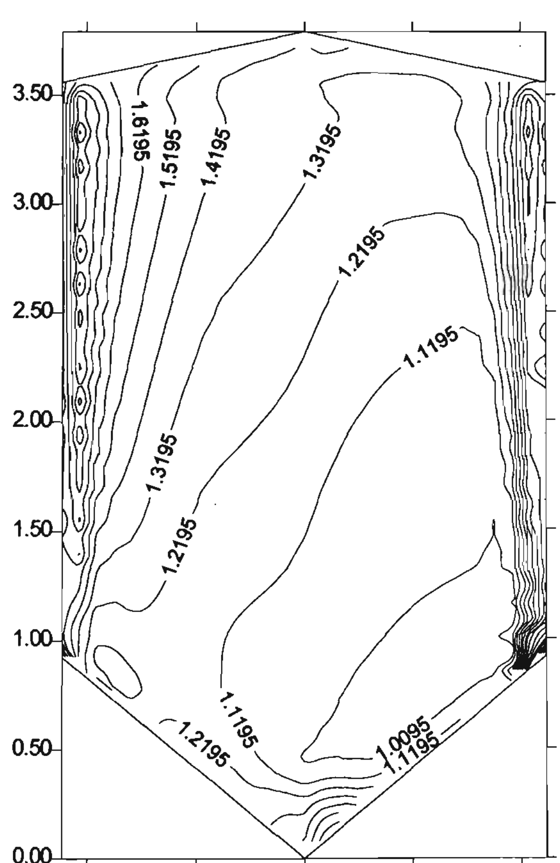
Figure 7.40 *S. zeamais* distributions in the non-aerated silo



S. zeamais at day 60 (linear duct)

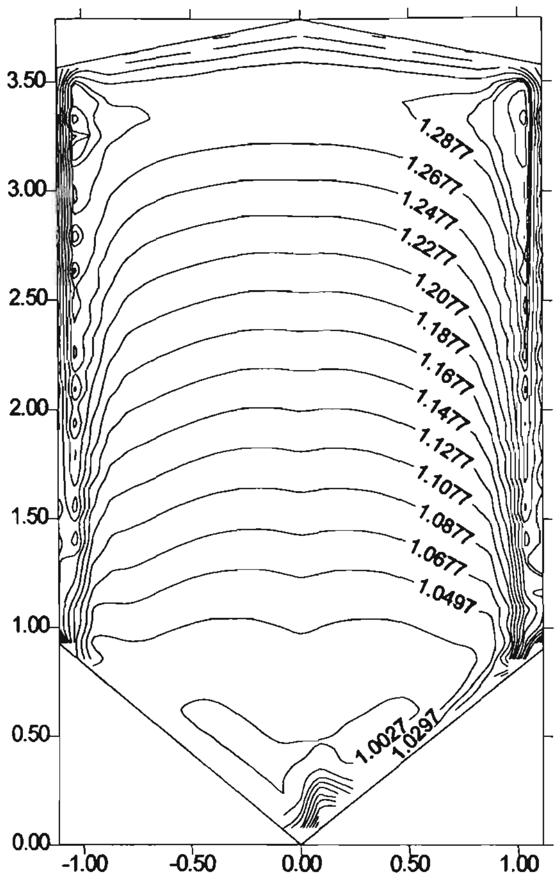


S. zeamais at day 40 (linear duct)

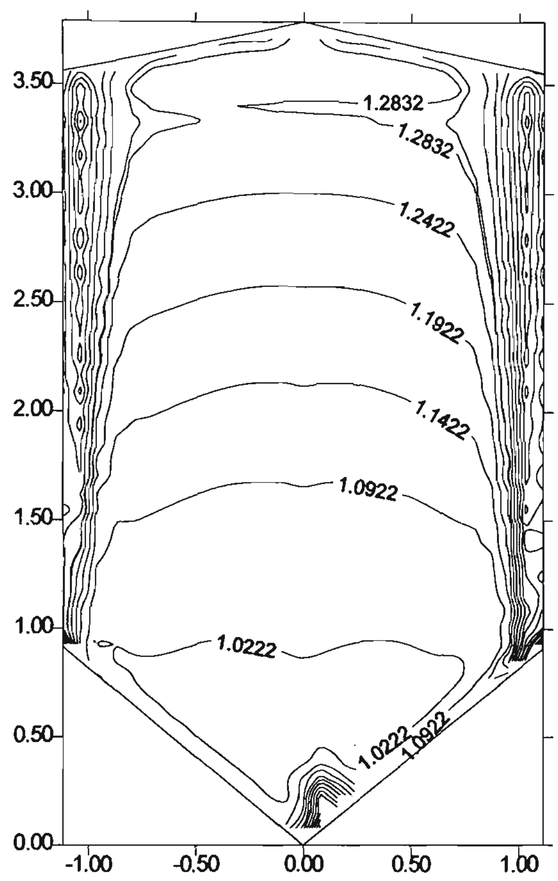


S. zeamais at day 60 (linear duct)

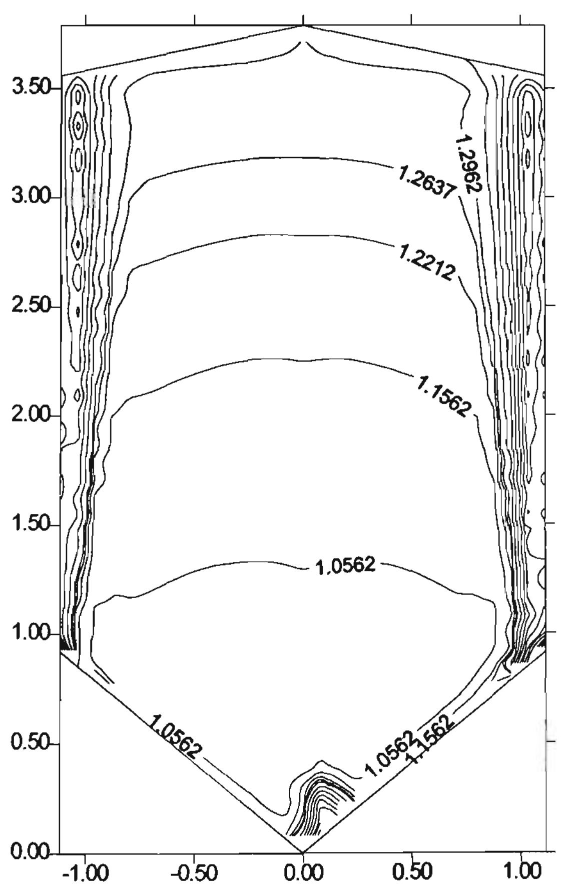
Figure 7.41 *S. zeamais* distributions in the silo fitted with a linear duct



S. zeamais at day 20 (annular duct)



S. zeamais at day 40 (annular duct)



S. zeamais at day 60 (annular duct)

Figure 7.42 *S. zeamais* distributions in the silo fitted with an annular duct

7.9 SEED VIABILITY DISTRIBUTIONS

The viability of seed grain is signified as the ability of the seed to develop into a young plant under favourable conditions in the absence of dormancy (*Brooker et al.*, 1992). The life of the viability of seed grain depends on temperature, moisture content and the length of time. Figures 7.46, 7.47, and 7.48 show, the variation of stored seed viability under three different storage conditions.

It is clearly to see, that after 60 days storage with an initial seed viability of initial 0.99, in the non-aerated silo, the centre area of seed viability is 0.95, which is 4.2% lower than initial condition. In the silo fitted with a linear aeration duct, the central zone of the seed viability is 0.97. In the silo fitted with an annular aeration duct, the centre region of seed viability is 0.98.

It is clear that aeration preserves seed viability of the rate of degradation the germ proteins and hydrolysis of the embryonic starch, and that the annular duct is slightly move effectively than the linear duct.

Figures 7.46, 7.46 and 7.47 reflect the seed viability responds to grain cooling. In the non-aerated silo as illustrated in figure 7.45, the viability of the grain is generally lower in the centre of the silo. This is because the grain remain warmest in this case.

Figure 7.46 clearly reflects the non-uniform passage of the cooling wave through the grain, and the viability of the seed is generally lower in the region furthest from the aeration duct,

The air flow distribution in the silo fitted with an annular duct is relatively uniform, and this is reflected in the seed viability patterns as shown in figure 7.47. Here we observe a large central region of the grain bulk that retains a large seed viability.

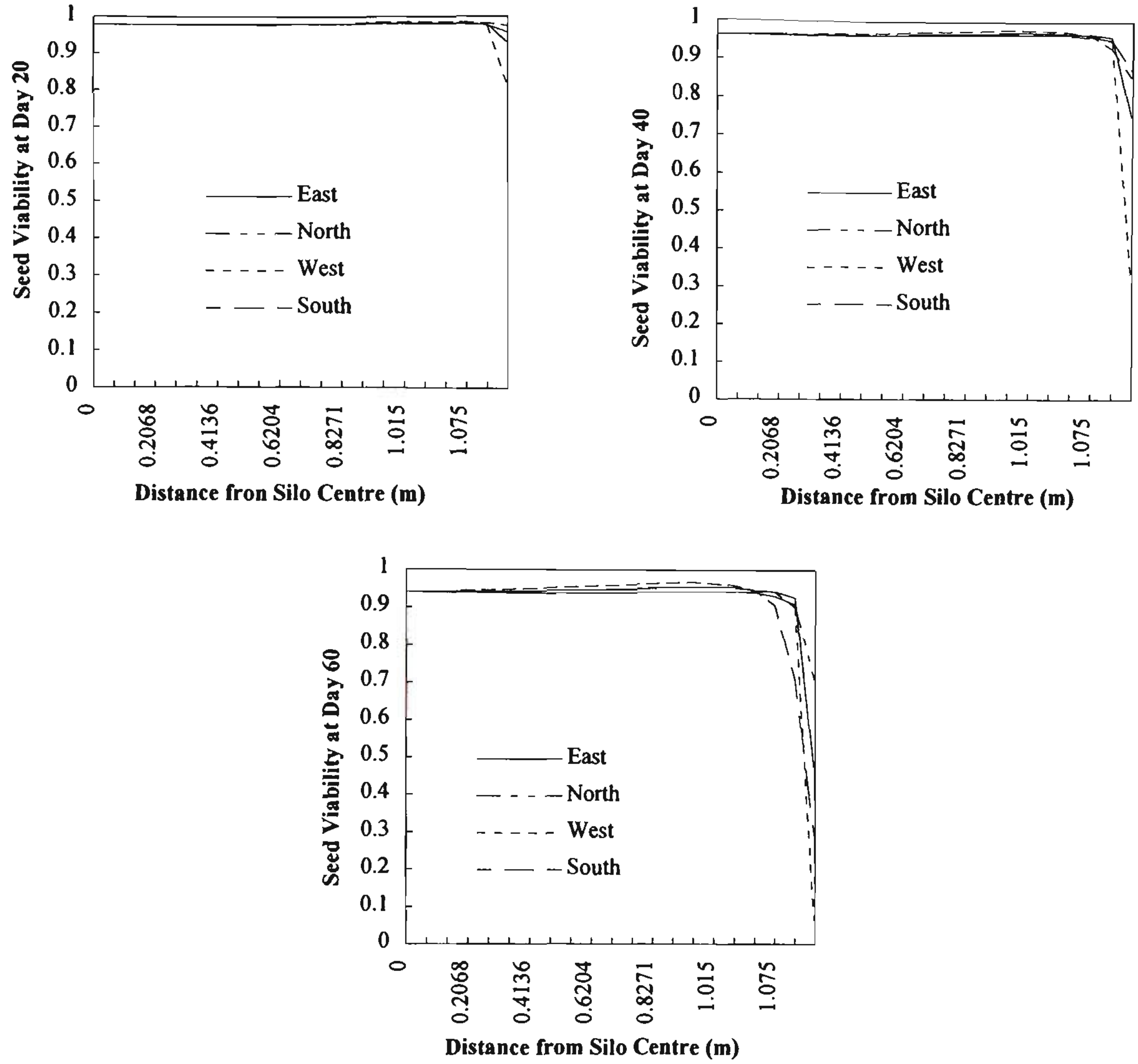
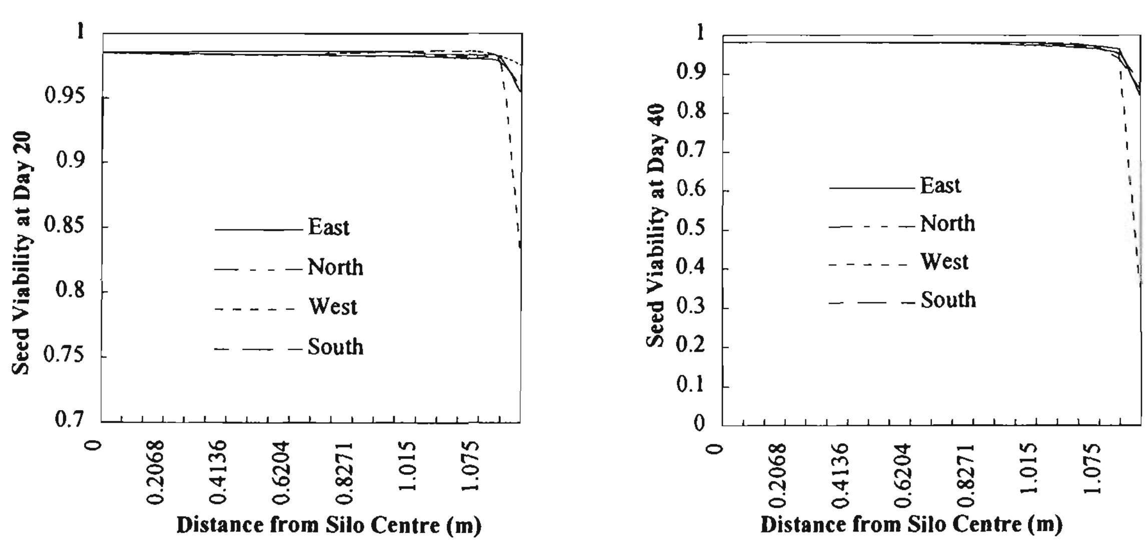


Figure 7.43 Radial profiles of the seed viability in the non-aerated case



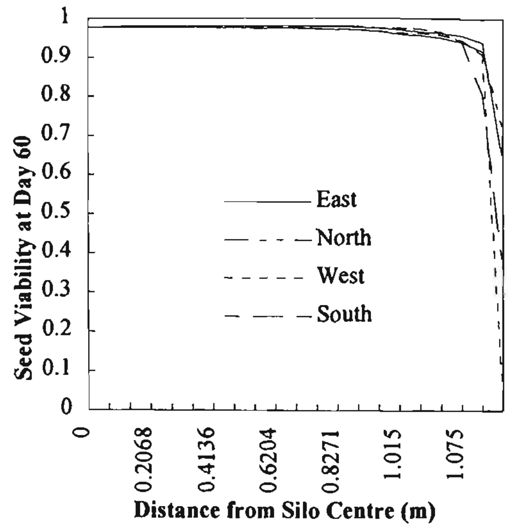


Figure 7.44 Radial profiles of the seed viability in the linear duct case

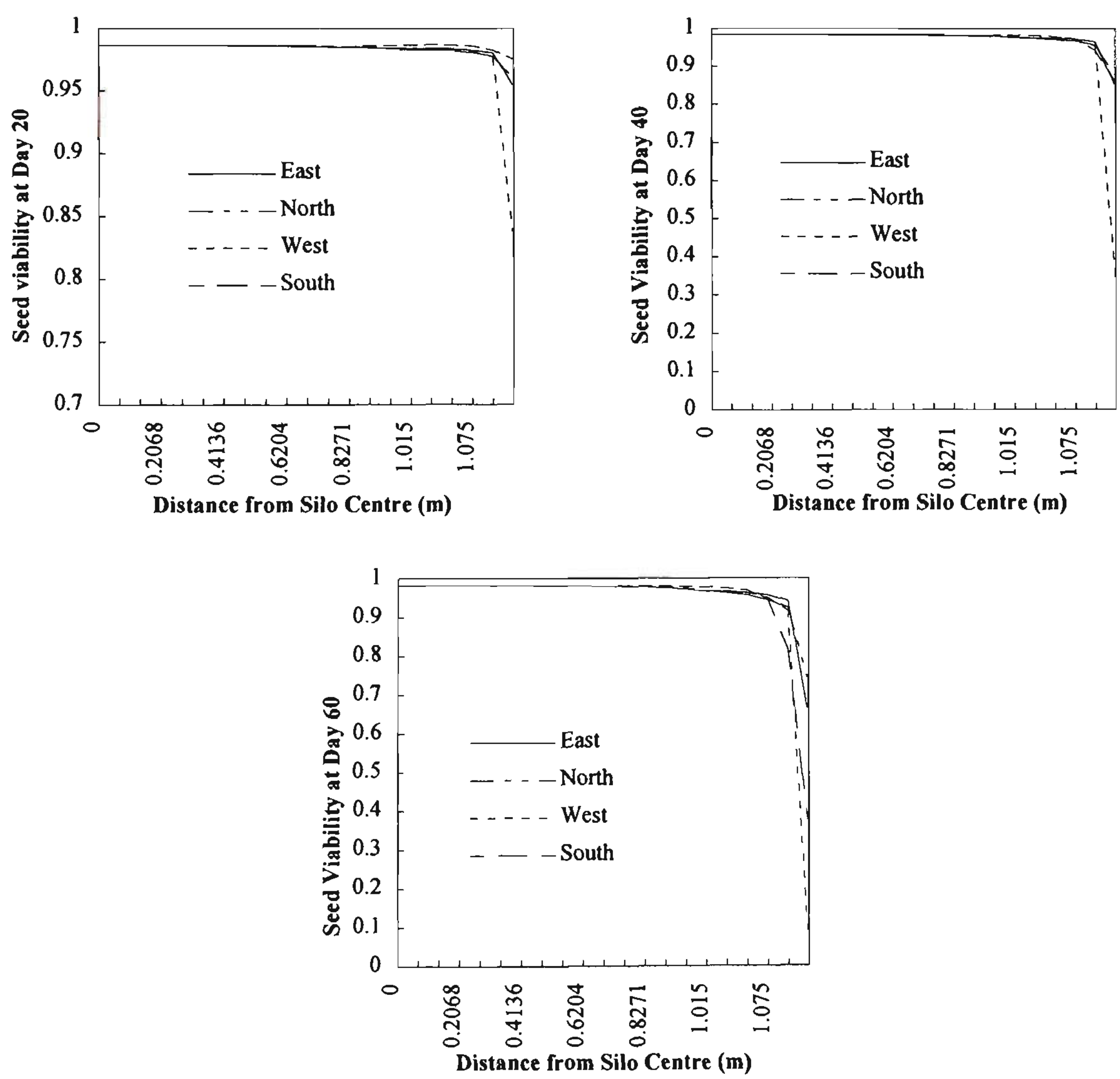
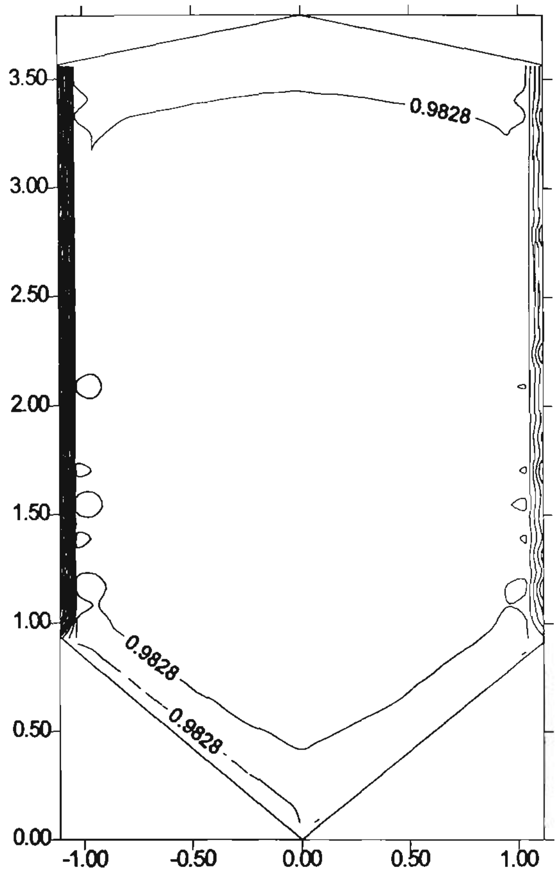
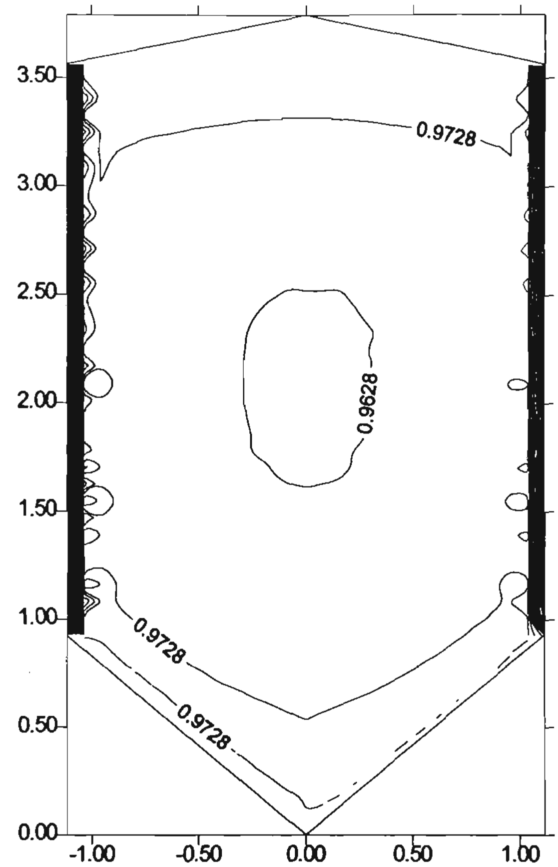


Figure 7.45 Radial profiles of the seed viability in the annular duct case

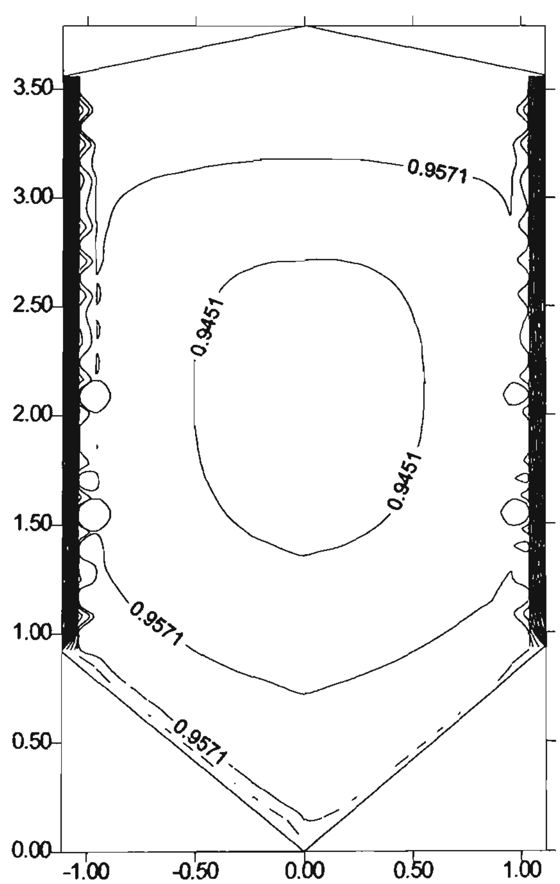
Figures 7.43, 7.44 and 7.45 show, the radial profiles of the seed viability along the north, south, east and west facing of the silo. Two observations have been made. Firstly, the seed viability in the central region of the grain bulk is higher in the aerated silos than in the non-aerated silo, and the silo fitted with the annular duct provide better result of seed preservation. Secondly, aeration appears to give litter protection to grain adjacent to those walls exposed to high temperatures. The stored seed near the west facing of the wall has a particularly low viability, because of the high temperature entering to which it has been exposed.



Seed viability at day 20 (non-aerated)

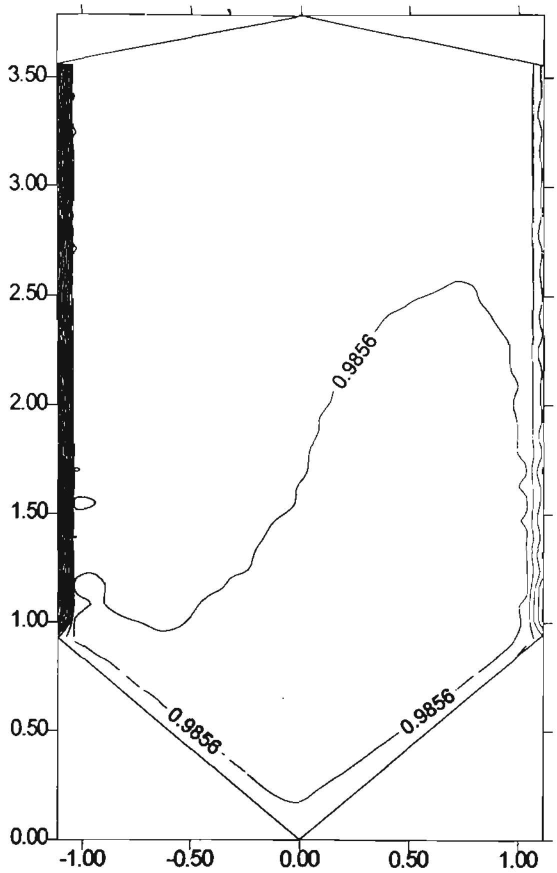


Seed viability at day 40 (non-aerated)

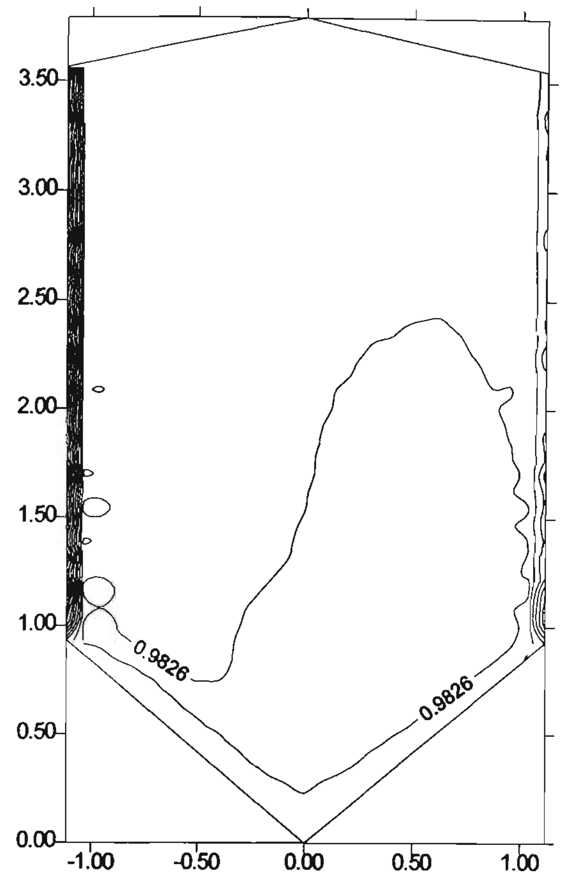


Seed viability at day 60 (non-aerated)

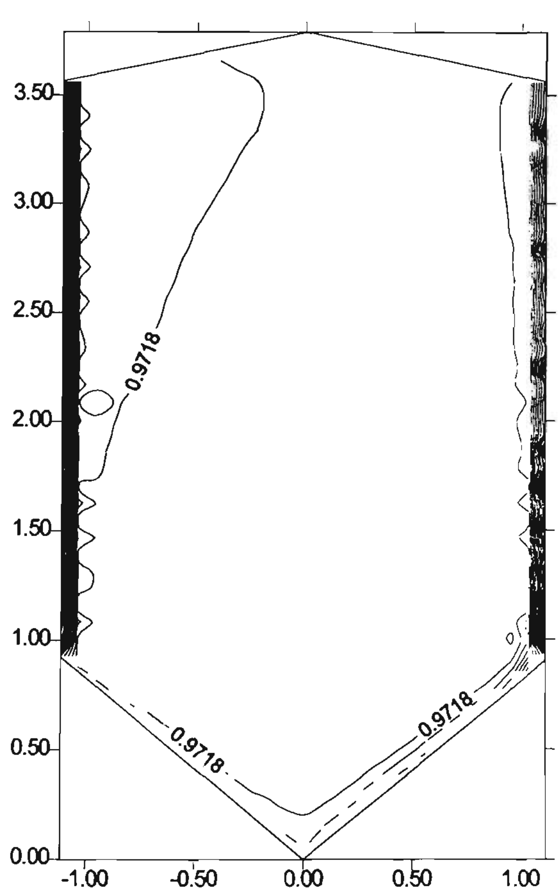
Figure 7.46 Seed viability distributions in the non-aerated silo



Seed viability at day 20 (linear duct)

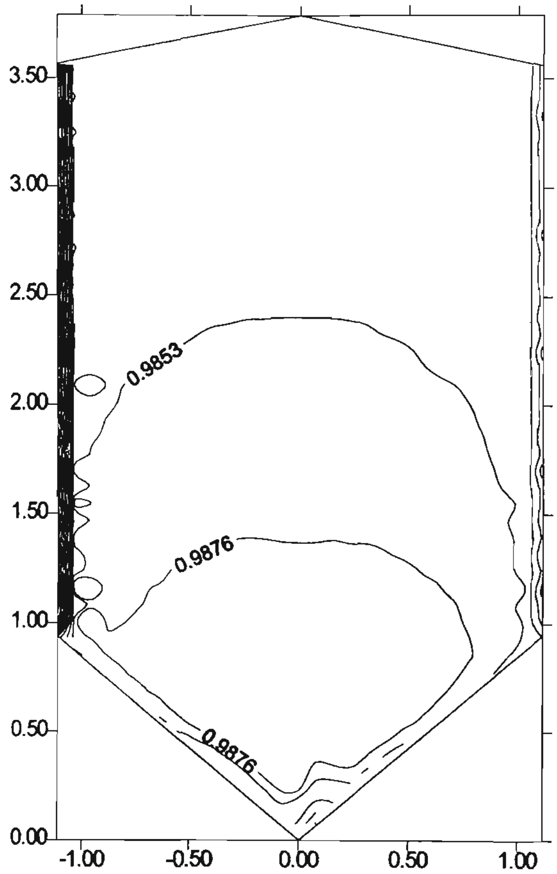


Seed viability at day 40 (linear duct)

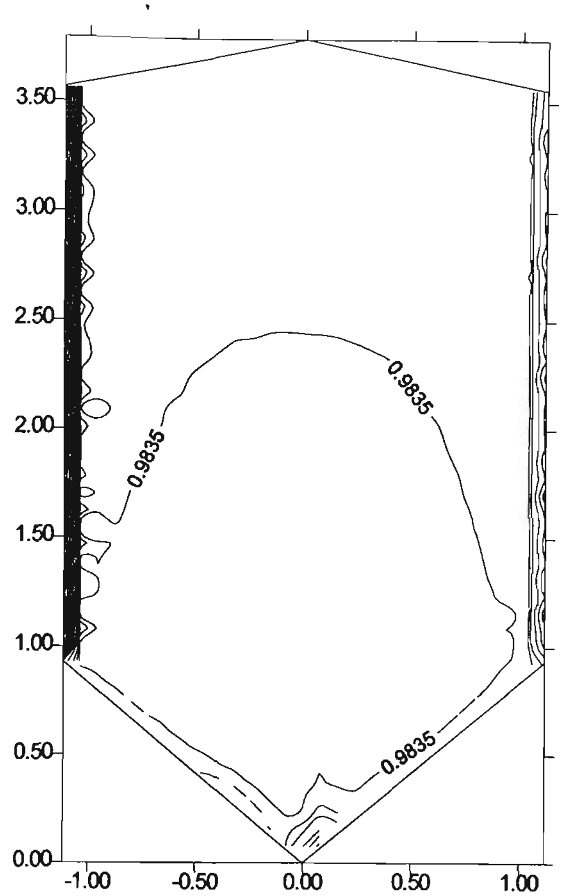


Seed viability at day 60 (linear duct)

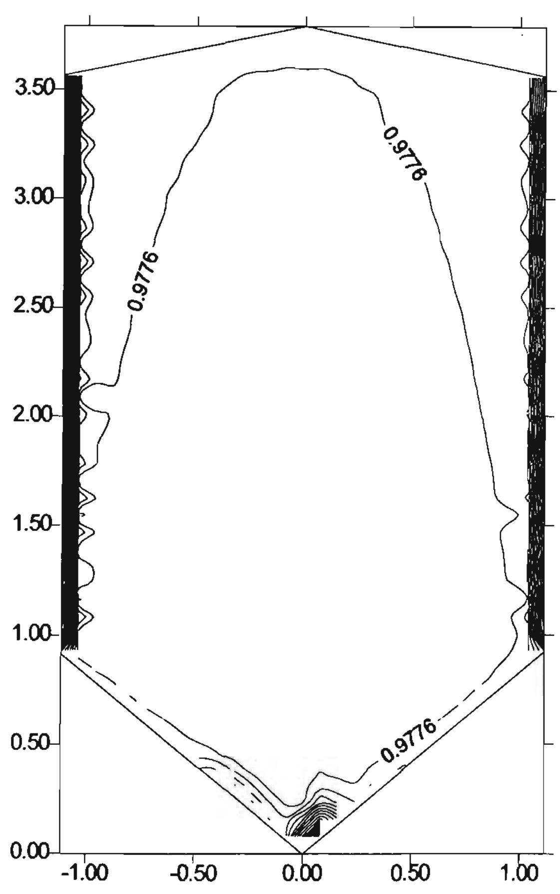
Figure 7.47 Seed viability distributions in the silo fitted with a linear duct



Seed viability at day 20 (annular duct)



Seed viability at day 40 (annular duct)



Seed viability at day 60 (annular duct)

Figure 7.48 Seed viability distributions in the silo fitted with an annular duct

7.10 DECAY OF CHEMICAL PESTICIDE DISTRIBUTIONS

Temperature and moisture content are the major factors in the degradation of contact pesticides applied to grain. Pesticides can be effectively against the insects for many month, and can be applied to grain held in poorly constructed store without specialised equipment for their application.

Figures 7.52, 7.53 and 7.54 illustrate the distribution of the pesticide concentration in the stored grains. It is clear that the pesticide decays rapidly in the non-aerated silo. Indeed, the concentration of pesticide in the central region has falls to about 69% of its initial value of the 60 days storage (as shown in figure 7.52), this is because that the warm air flow is effected on this central region of the silo.

The rate of pesticide decays in the silo fitted with a linear duct clearly reflects the non-uniform cooling, it is shown in figure 7.53, and the pesticide concentration in the region furthest from the aeration duct is about 77% of its initial condition.

By contrast, it can be seen in figure 7.54 that the pesticide concentration in the silo fitted with the annular duct is generally about 80% of its initial pesticide decay in the central area of the silo, and reflects the uniform cooling wave to the pesticide decays in the silo.

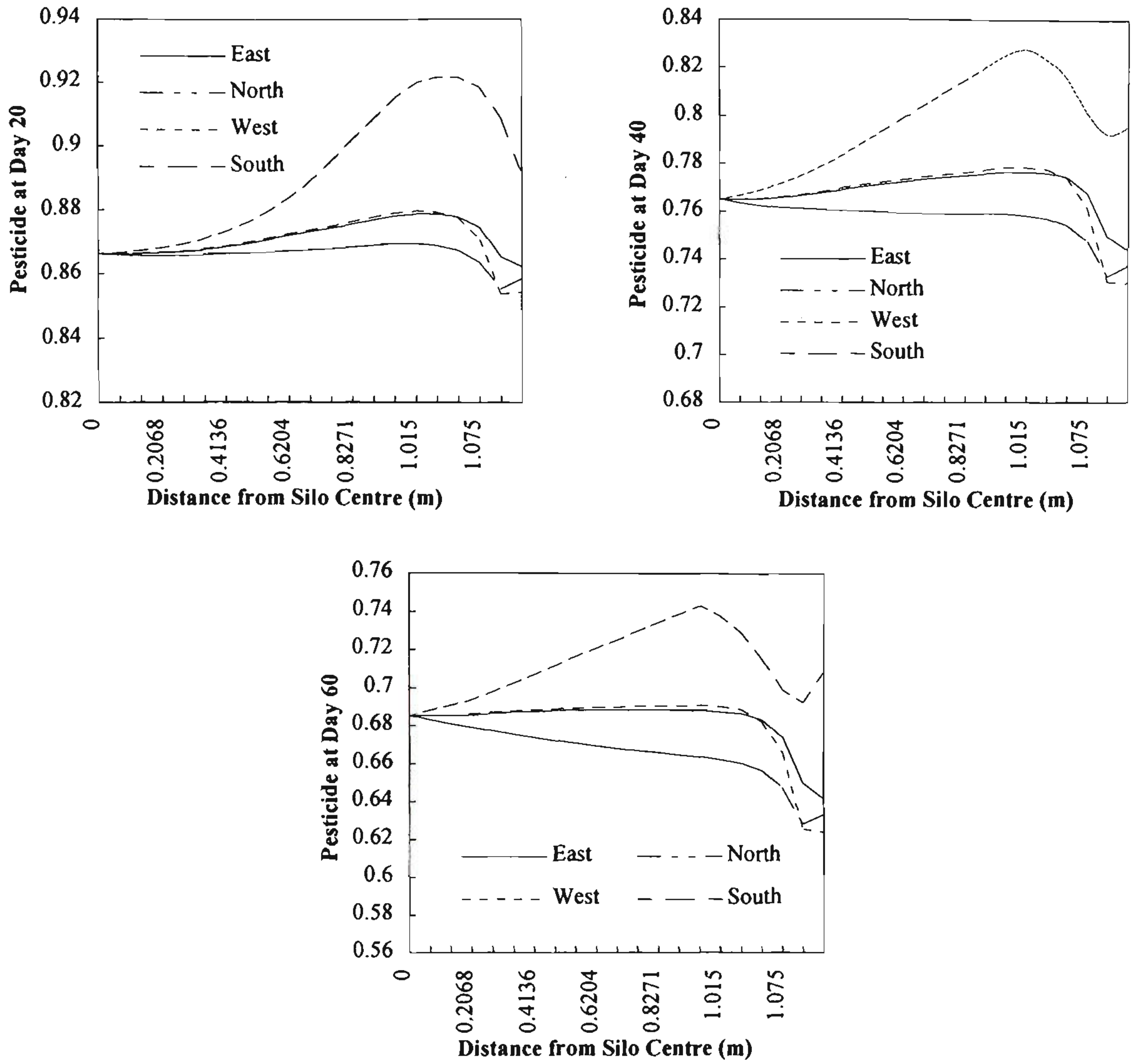
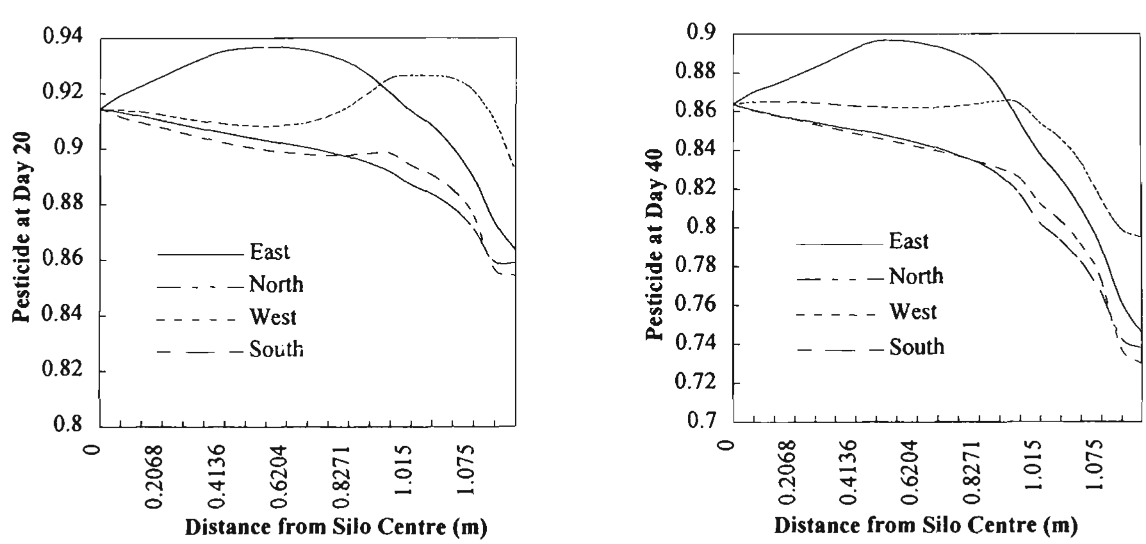


Figure 7.49 Radial profiles of the pesticide in the non-aerated case



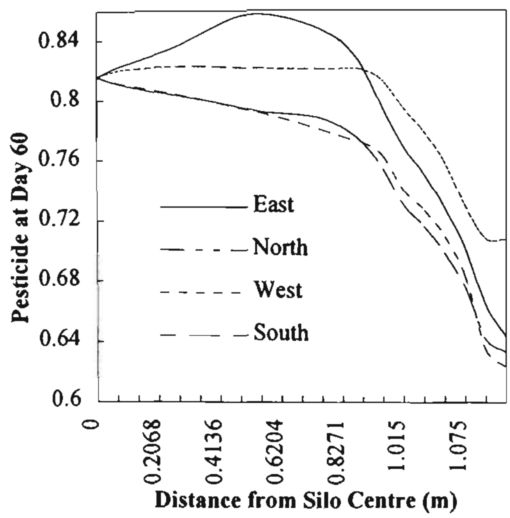


Figure 7.50 Radial profiles of the pesticide in the linear duct case

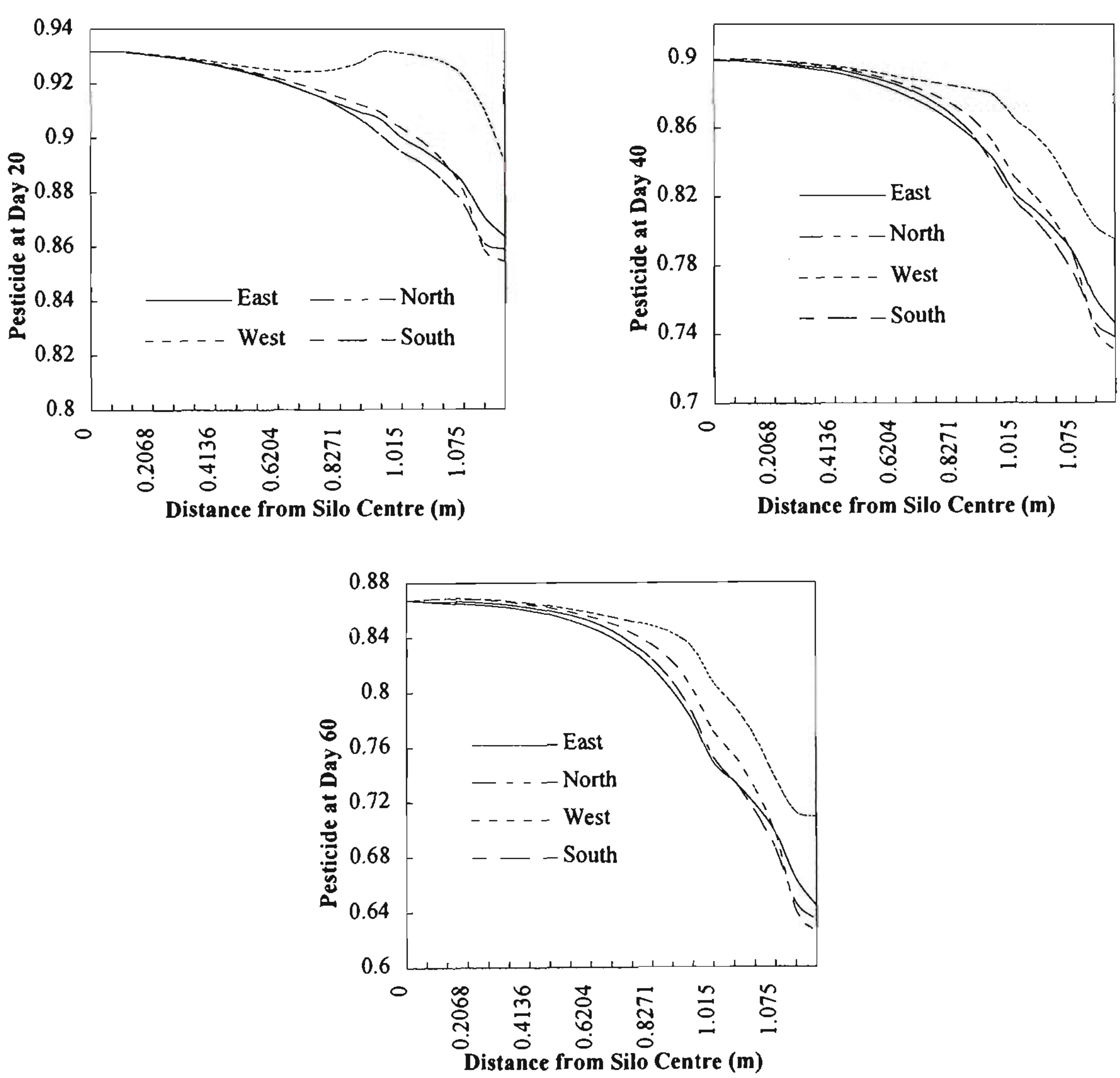
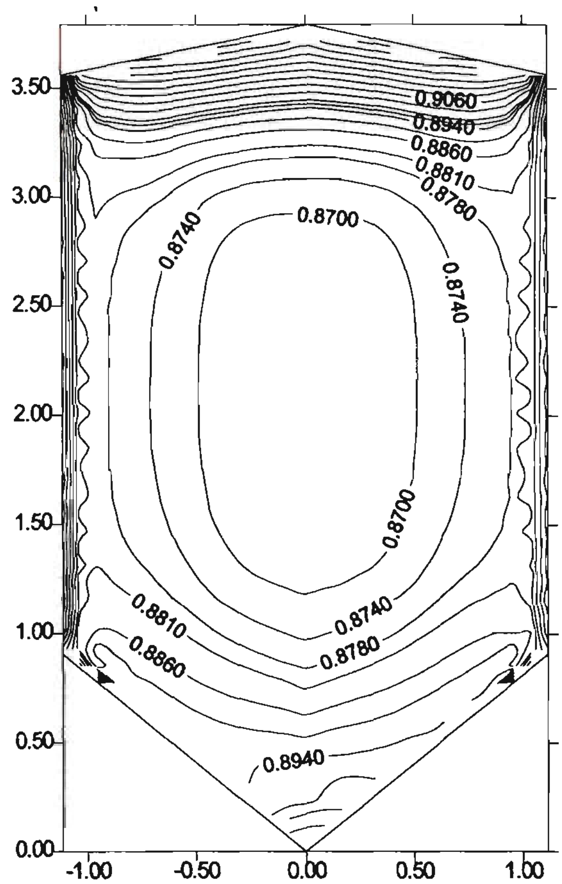


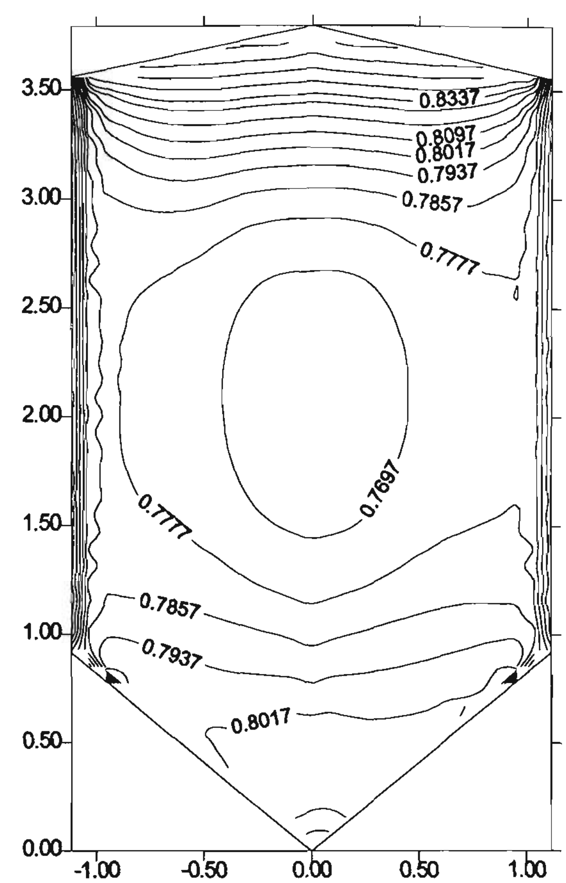
Figure 7.51 Radial profiles of the pesticide in the annular duct case

Figures 7.49, 7.50 and 7.51 show that the variation of the decay of chemical pesticide from the east, north, west and south facing portions of the wall. As would be expected, the pesticide concentration in the central region of the silo is higher in the aerated silos than in the non-aerated silo, and the silo fitted with an annular duct render the better results for protection the decay of pesticide than the silo fitted with a linear duct.

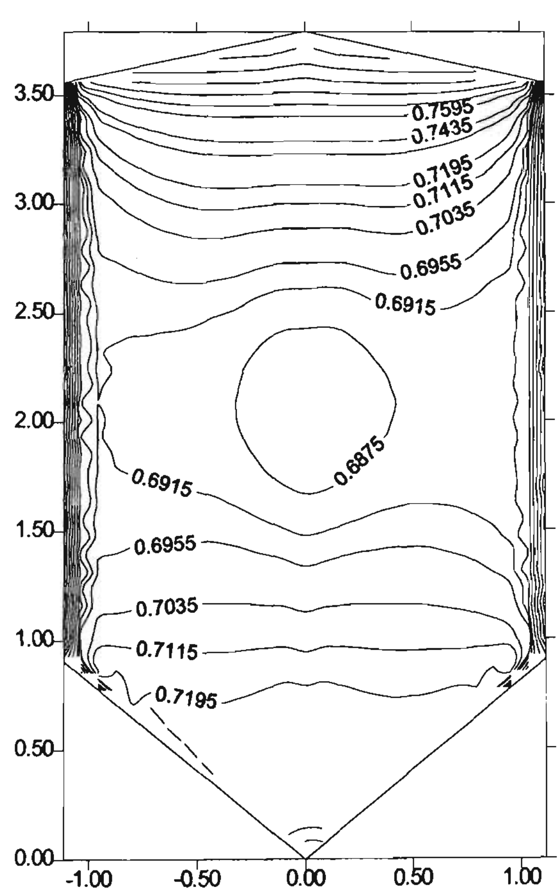
The rate of chemical pesticide is higher in the south side of the silo, this is because the sunlight is not strongly effected in this area, and the pesticide concentration is rapidly broken down at the east, west and north sides of the wall in the three cases by solar radiation heating during the day.



Pesticide at day 20 (non-aerated)

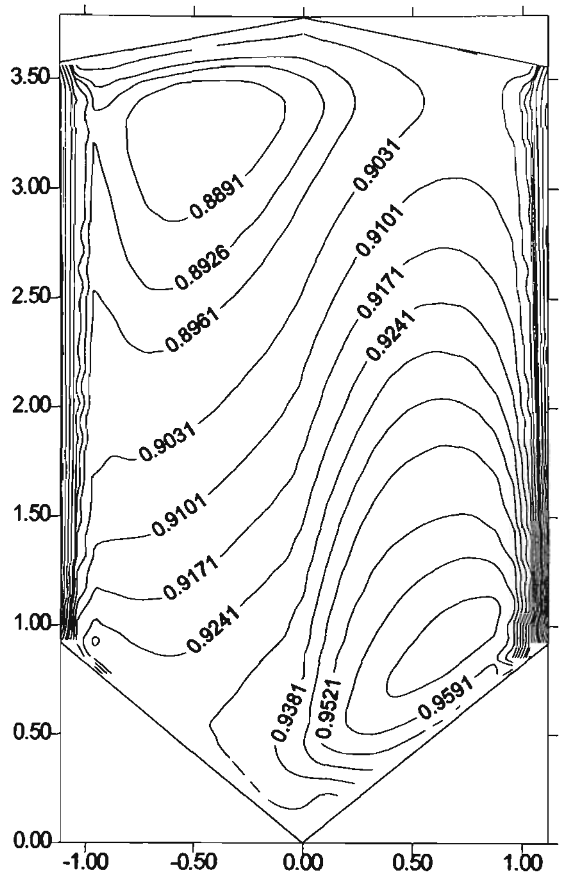


Pesticide at day 40 (non-aerated)

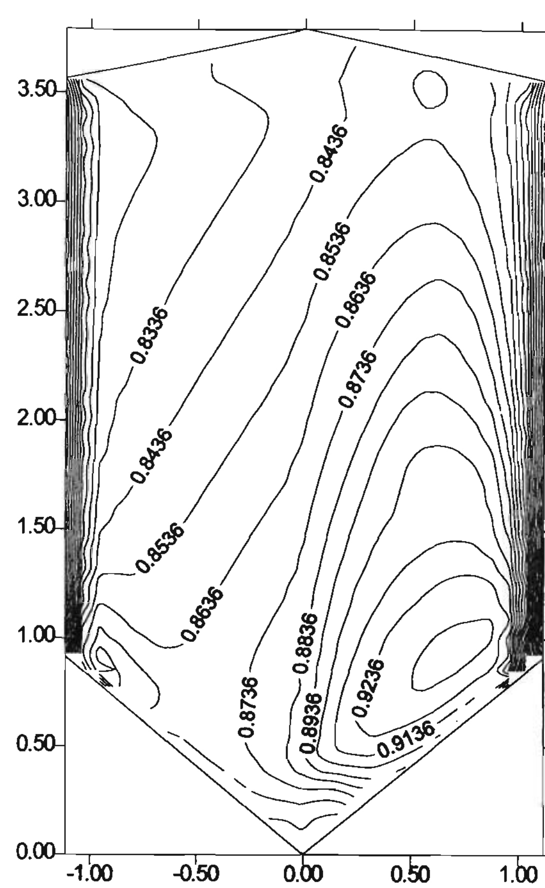


Pesticide at day 60 (non-aerated)

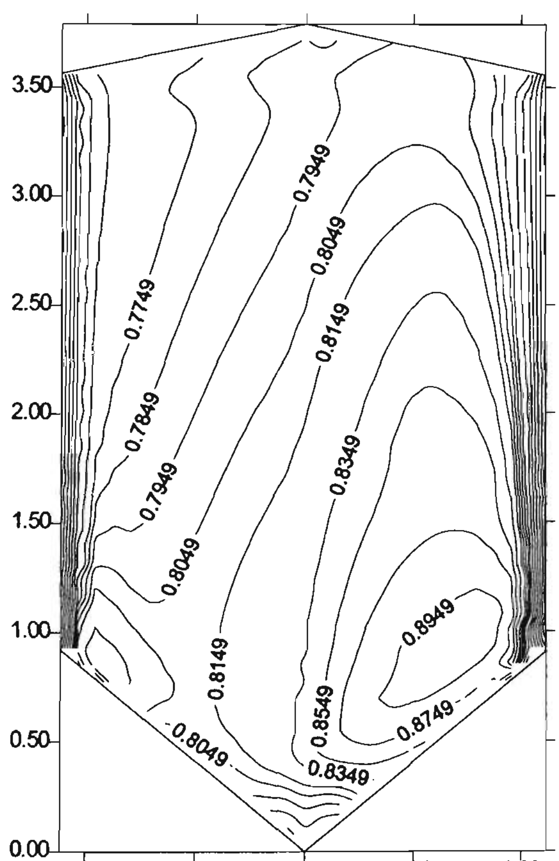
Figure 7.52 Pesticide distributions in the non-aerated silo



Pesticide at day 20 (linear duct)

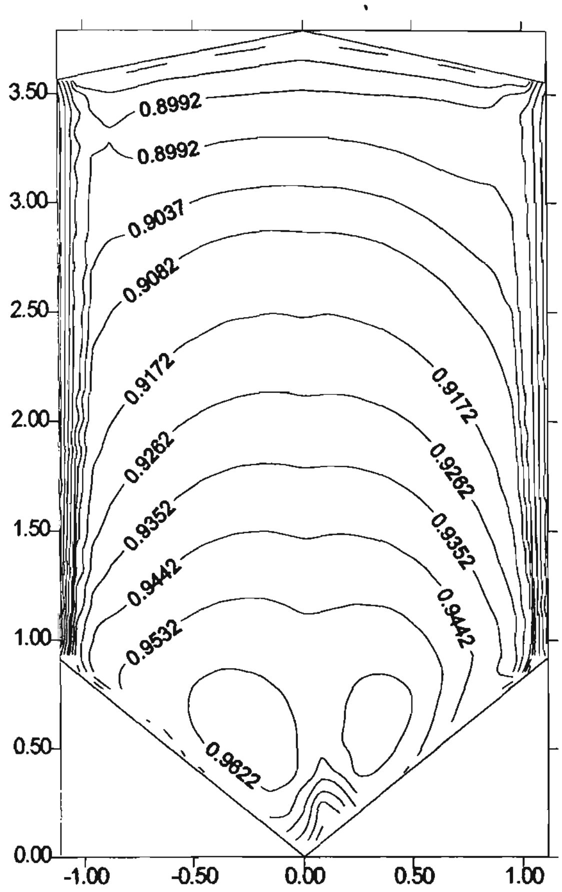


Pesticide at day 40 (linear duct)

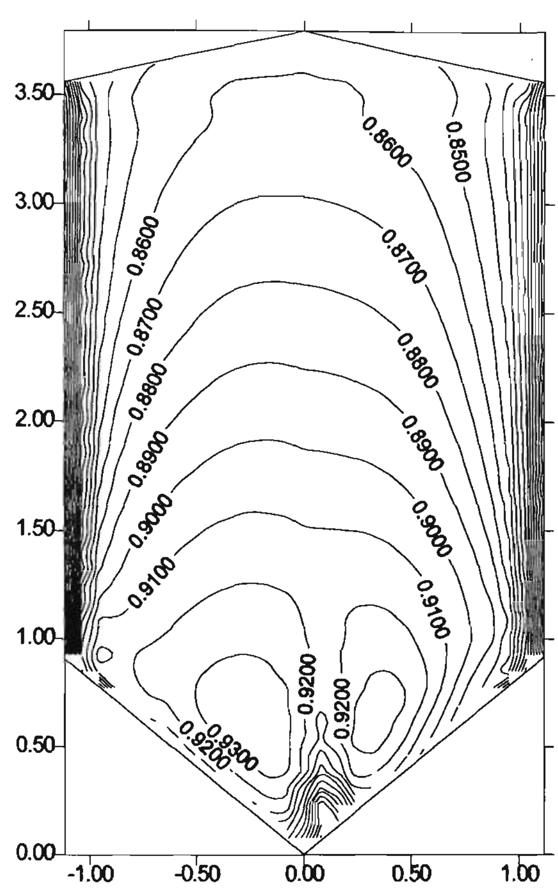


Pesticide at day 60 (linear duct)

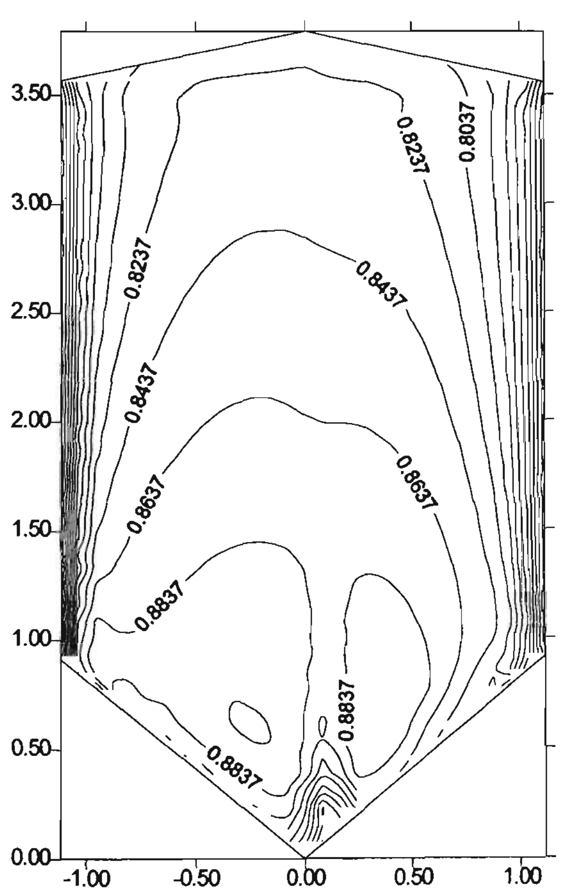
Figure 7.53 Pesticide distributions in the silo fitted with a linear duct



Pesticide at day 20 (annular duct)



Pesticide at day 40 (annular duct)



Pesticide at day 60 (annular duct)

Figure 7.54 Pesticide distributions in the silo fitted with an annular duct

7.11 OVERVIEW

After 9 groups of profiles discussed, the performances between non-aerated, linear aeration and annular aeration systems are evident:

- Annular aeration system can provide strong and uniform flow of air to occurring in the silo symmetrically, and gives best results for protection of the environment of stored grain.

- Linear aeration system may provide a non-uniform airflow distribution, and offers a specific causes of special cooling results.

- The grains in the non-aerated system can be quickly damaged by insect pests attacking, and loss in the quantity and quality.

- In addition, the performances of simulation results which are used program SILO3DFTN to predict the ecosystems in a non-aerated silo, the silos fitted with linear and annular aeration ducts are summarised in table 7.1.

Table 7.1 The performance of non-aeration and aeration silos

	Non-aerated	Linear duct aeration	Annular duct aeration
Mean dry matter loss	0.0039	0.0038	0.0031
Mean concentration of fenitrothion	0.7092	0.7885	0.8238
Mean viability	0.9453	0.9647	0.9686
Mean concentration of *R. dominica*	12.7431	3.7367	2.6713
Mean concentration of *S. oryzae 1*	128.5328	24.2884	18.1394
Mean concentration of *S. oryzae 2*	63.0503	13.1513	10.0286
Mean concentration of *S. zeamais*	2.4118	1.3611	1.3197

7.12 LIMITATIONS OF THIS WORK

In this work that the insects are immobile, whereas in reality they are likely to translocate to more favourable microclimates. However, the results for insect growth indicate the potential for the growth of insect populations. Furthermore, the effects of aeration on the movement of insects has also been ignored. Essentially, the reason for this is that the biological data are simply not available, and this is an area worthy of study by stored products entomologist. In this work we have used rate of insect population data presented by Desmarchelier (1988). In this work it is explicitly acknowledged that there are differences in the behaviour of different strains of the same species of insects. The data presented by Desmarchelier (1988) are for age stable populations, which are highly unlikely to occur during the initial stages of grain storage.

Fungal activity appears to depend on a large number of variables, many of which relate to the posthavest conditions of the grain. Any deterministic model of mould activity must therefore be treated with extreme caution. In this work we have relied on the expression for the rate of respiration of grains presented by Lacey *et al.* (1994). This appears to give a rate of respiration much higher than that predicted by the work of Thompson (1972). Once more there is clearly a need for a much more detailed and deeper understanding of biological phenomena.

The data presented on the rate of decay of chemical pesticides are relatively deterministic, and since we are dealing with contact pesticides that are not labile. However, in practice, pesticides are unlikely to be applied with a uniform concentration as implied by this work, but variations can be easily accommodated.

CHAPTER 8

CONCLUSIONS

This chapter presents conclusions stemming from the work of the previous chapters. Also some suggestions and some further work directions within the framework introduced in this thesis.

8.1 SUMMARY OF CONTRIBUTIONS

(1) A finite difference model for predicating the behaviour of the microclimate occurring within the conical bottomed circular silos was developed. The three-dimensional model was applied to estimate the actual variation of pressure, velocity, temperature, moisture content, absolute humidity, wet bulb temperature, dry matter loss, seed viability and pesticide decay in the grain storage.

(2) A mapping technique provides a convenient methodology, and has strong ability to analysis the classical mathematical model for obtaining the numerical solution of partial differential equations in the arbitrary geometries.

(3) Solar radiation is the major factor that influences the surface temperature on the silo wall. It can causes the temperature gradients raising to conduction, leading to

moisture transfer, and consequent grain damage. It is the important parameter for choice the colour of the silo, and the placement of the aeration duct.

(4) The research has shown that grain temperatures remain high prolonged period of time in the non-aerated silo. Aeration is successful in cooling the grain bulk, but the grain at the peripheries is little affected by aeration.

(5) Importantly, the results of the work demonstrated: The silo fitted with annular duct is more uniformly cooled that silo fitted with conventional linear duct. Grain in the annular aerated silo are stored in better condition with a higher viability and lower pesticide decay. Furthermore, insect population growth can also be better controlled in the silo fitted with an annular duct.

(6) The mathematical analysis has been encoded in FORTRAN, and the program listing forms a part of this thesis (see Appendix III).

8.2 FUTURE WORK

The future research should be carried out in the following areas:

(1) Research projects should validate the computer model by carrying out field experiment measurements, for investigating pressure, temperature and moisture content in a conical bottomed circular silo, with non-aerated, linear aeration and annular aeration systems. Moreover, the experiments should indicate the environmental variables which are relevant and pertain to practical situations.

(2) Ecological study of computerization model should be used to investigate the likely success of aeration system in a range of climates, specially the methods of augmenting aeration with ambient air by mechanical refrigeration and solar

cooling are also worthy. Besides, the study should include the effect of the size of the silo and the duct placement on the performance of aeration system .

(3) It is also important to study the interaction of the air in the headspace of a silo and the grain bulk.

REFERENCES

Abawi, G. Y. (1993) A Simulation Model of Wheat Harvesting and Drying in Northern Australia. Journal of Agricultural Engineering Research. Vol. 54, pp. 141-158.

Abbouda, S. K., Chung, D. S., Seib, P. A. and Song, A. (1992) Heat and Mass Transfer in Stored Milo. Part I. Heat Transfer Model. Transaction of American Society of Agricultural Engineers. Vol. 35:5, pp. 1569-1573.

Andales, S. C., Pettibone, C. A. and Davis, D. C. (1979) Two-dimensional cooling of bulk stored sugarbeets. ASAE Paper No. 79-4534. St. Joseph, Mich.: 49085.

Arinze, E. A., Sokhansanj, S., Schoenau, G. J. and Sumner, A. K. (1994) Control Strategies for Low Temperature In-bin of Barley for Feed and Malt. Journal of Agricultural Engineering Research. Vol. 58, pp. 73-88.

Barre, H. J. Baughman, G. R. and Hamdy, M. Y. (1971) Application of the Logarithmic Model to Cross- Flow Deep-Bed Grain Drying. Transaction of American Society of Agricultural Engineers. Vol. 14:6, pp. 1061-1064.

Bloome, P. D. and Shove, G. C. (1972) Simulation of Low Temperature Drying of Shelled Corn Leading to Optimization. Transaction of American Society of Agricultural Engineers. Vol. 15:2, pp. 310-316.

Bridges, T. C., Overhults, D. G., McNeill, S. G. and White, G. M. (1988) An Aeration Duct Design Model for Flat Grain Storage. Transaction of American Society of Agricultural Engineers. Vol. 31:4, pp. 1283-1288.

Brooker, D. B. (1961) Pressure Patterns in Grain - Drying Systems Established by Numerical Methods. Transactions of the American Society of Agricultural Engineers. Vol. 4:1, pp. 72-74, 77.

Brooker, D. B. (1969) Computing Air Pressure and Velocity Distribution When Air Flows Through a Porous Medium and Non-linear Velocity - Pressure Relationships Exist._Transactions of the American Society of Agricultural Engineers. Vol. 12:1, pp. 118-120.

Brooker, D. B., Bakker-Arkema, F. W. and Hall, C. W. (1992) Drying and Storage of Grains and Oilseeds. Van Nostrand Reinhold. New York.

Bushuk, W. and Rasper, V. F. (1994) Wheat: Production, Properties and Quality. Blackie Academic and Professional, An Imprint of Chapman & Hall. London.

Carnahan, B., Luther, H. A. and Wilkes, J. O. (1969) Applied Numerical Methods. John Wiley & Sons, Inc.

Casada, M. E. and Young, J. H. (1994) Model for Heat and Moisture Transfer in Arbitrarily Shaped Two-Dimensional Porous Media. Transaction of American Society of Agricultural Engineers. Vol. 37:6, pp. 1927-1938.

Chang, C. S., Converse, H. H. and Steele, J. L. (1994) Modelling of Moisture Content of Grain During Storage with Aeration. Transaction of American Society of Agricultural Engineers. Vol. 37:6, pp. 1891-1898.

Chapman, J. E., Morey, R. V., Cloud, H. A. and Nieber, J. L. (1989) Airflow Patterns in Flat Storage Aeration Systems. American Society of Agricultural Engineers. Vol. 32:4, pp. 1368-1376.

Chelkowski, J. (1991) Cereal Grain Mycotoxins, Fungi and Quality in Drying and Storage. Elsevier Science Publishers B. V. The Netherlands.

Chung, D. S. and Pfost, H. B. (1967a) Adsorption and Desorption of Water Vapour by Cereal Grains and Their Products. Part I. Heat and Free Energy Changes of Adsorption and Desorption. Transaction of American Society of Agricultural Engineers. Vol. 10, pp. 549-551, 555.

Chung, D. S. and Pfost, H. B. (1967c) Adsorption and Desorption of Water Vapour by Cereal Grains and Their Products. Part III. A Hypothesis for Explaining the Hysteresis Effect. Transaction of American Society of Agricultural Engineers. Vol. 10, pp. 556-557.

Collares-Pereira, M. and Rabl, A. (1979) The Average distribution of Solar Radiation - Correlations between Diffuse and Hemispherical and between Daily and Hourly Insolation Values. Solar Energy. Vol. 22, pp. 55.

Cooper, P. I. (1969) The Absorption of Solar Radiation in Solar Stills. Solar Energy. Vol. 12, pp. 333-346.

Crank, J. and Nicolson, P. (1947) A Practical Method for Numerical Evaluation of Solutions of Partial Differential Equations of the Heat Conduction Type. Process Cambridge Philosophical Society. Vol. 43, pp. 50-67.

Croft, D. R. and Lilley, D. G. (1977) Heat Transfer Calculations Using Finite Difference Equations. Applied Science Publishers Ltd. London.

Darcy, H. P. G. (1856) *Les Fontaines Publiques de la Ville de Dijon. Exposition et Application a Suiver et Des Formules a Employer Dans Les Questions de Distribution d' Eau.* Victor Dalamount.

Deshpande, S. D., Bal, S. and Ojha, T. P. (1993) Physical Properties of Soybean. Journal of Agricultural Engineering Research. Vol. 56, pp. 89-98.

Desmarchelier, J. M. and Bengston, M. (1979) Chemical Residues of Newer Grain Protectants. In: Australian Contributions to the Symposium on the Protection of Grain Against Insect Damage During Storage (Evans, D. E. Ed.), Moscow 1978. CSIRO Division of Entomology, Canberra, pp. 108-115.

Desmarchelier, J. M. (1988) The Relationship between Wet-bulb Temperature and the Intrinsic Rate of Increase of Eight Species of Stored-product Coleoptera. Journal of Stored Products Research. Vol. 24:3, pp 107-113.

Duffie, J. A. and Beckman, W. A. (1980) Solar Engineering in Thermal Processes. John Wiley and Sons, New York.

Dutta, S. K., Nema, V. K. and Bhardwaj, R. K. (1988) Physical Properties of Grain. Journal of Agricultural Engineering Research. Vol. 39, pp. 259-268.

Dutta, S. K., Nema, V. K. and Bhardwaj, R. K. (1988) Thermal Properties of Gram. Journal of Agricultural Engineering Research. Vol. 39, pp. 269-275.

Elder, W. B. (1978) The future of temperature controlled storages and heat treatment of grain. Australia Grain Institute Incorporated. "Grist", No. 19, pp. 8-10.

Ergun, S. (1952) Fluid Flow Through Packed Columns. Chemical Engineering Progress. Vol. 48:2, pp. 89-94.

Fohr, J. P. and Moussa, H. B. (1994) Heat and Mass Transfer in a Cylindrical Grain Silo Submitted to a Periodical Wall Heat Glux. International Journal of Heat and Mass Transfer. Vol. 37:12, pp. 1699-1712.

Fortes, M. and Okos, M. (1981) Heat and Mass Transfer in Hygroscopic Capillary Extruded Products. A. I. Chemical Engineering Journal. Vol. 27:2, pp. 255-262.

Foster, G. H. (1967) Moisture Changes During Aeration of Grain. Transaction of American Society of Agricultural Engineers. Vol. 10:3, pp. 344-347, 351.

Freer, M. W., Siebenmorgen, T. J., Couvillion, R. J. and Loewer, O. J. (1990) Modelling Temperature and Moisture Content Changes in Bunker-Stored Rice. Transaction of American Society of Agricultural Engineers. Vol. 33:1, pp. 211-220.

Gallaher, G. L. (1951) A Method of Determining the Latent Heat of Agricultural Crops. Agricultural Engineering. Vol. 32:1, pp. 34.

Goudie, J. M., Smith, E. A. and De Ville, A. (1995) Modelling the Velocity of Air Through Beds of Cereal Grains. IMA Journal of Mathematics Applied in Business & Industry. Vol. 5, pp. 325-335.

Godon, B. (1994) Bioconversion of Cereal Products. VCH Publishers, Inc. New York.

Henderson, S. M. (1952) A Basic Concept of Equilibrium Moisture. Agricultural Engineering. Vol. 33, pp. 29-31.

Henderson, J. M. and Henderson, S. M. (1968) A Computational Procedure for Deep-bed Drying Analysis. Journal of Agricultural Engineering Research. Vol. 13:2, pp. 87-95.

Holdsworth, S. D. (1983) The Preservation of Fruit and Vegetable Food Production. Macmillan Press. London.

Hukill, W. V. and Ives, N. C. (1955) Radial Air Flow Resistance of Grain. Agricultural Engineering. Vol. 36:5, pp. 332-335.

Hunter, A. J. and Taylor, P. A. (1980) Refrigerated aeration for the preservation of bulk grain. Journal of Stored Products Research. Vol. 16, pp. 123-131.

Hunter, A. J.(1983) Pressure Difference across an Aerated Seed Bulk for some Common Duct and Store Cross-sections. Journal of Agriculture Engineering Research. Vol. 28, pp. 437-450.

Hunter, A. J.(1986) Thermodynamic Criteria for Optimum Cooling Performance of Aeration Systems for Stored Seed. Journal of Agricultural Engineering Research. Vol. 33, pp. 83-99.

Hunter, A. J. (1987) An Isostere Equation for Some Common Seeds. Journal of Agriculture Engineering Research. Vol. 37, pp. 93-105.

Hunt, W. H. and Pixton, S. W. (1974) Moisture-Its Significance, Behaviour, and Measurement. In Storage of Cereal Grains and Their Products, education C. M. Christensen, No. 1-55. St. Paul, MN: American Associate Cereal Chemistry.

Ingram, G. W. (1979) Solution of Grain Cooling and Drying problems by the Method of Characteristics in Comparison with Finite Difference Solutions. Journal of Agricultural Engineering Research. Vol. 24, pp. 219-232.

Ismail, M. Z., Angus, D. E. and Thorpe, G. R. (1991) The Performance of A Solar-regenerated Open-cycle Desiccant Bed Grain Cooling System. Solar Energy. Vol. 46:2, pp. 63-70.

Jayas, D. S. and Muir, W. E. (1991) Airflow-Pressure Drop Data for Modelling Fluid Flow in Anisotropic Bulks. Transaction of American Society of Agricultural Engineers. Vol. 34:1, pp. 251-254.

Jayas, D. S., Alagusundaram, K. Shunmugam, G., Muir, W. E. and White, N. D. G. (1992) Simulation of Temperature in Stored Bulks of Wheat Using A Three-dimensional Finite Element Model. ASAE Paper No. 92-6527. St. Joseph, Mich.: ASAE.

Jindal, V. K. and Thompson, T. L. (1972). Air Pressure Patterns and Flow Paths in Two-Dimensional Triangular-Shaped Piles of Sorghum Using Forced Convection. Transaction of American Society of Agricultural Engineers. Vol. 15:4, pp. 737-744.

Kazarian, E. A. and Hall, C. W. (1965) Thermal Properties of Grain. Transaction of American Society of Agricultural Engineers. Vol. 8:1, pp. 33-48.

Kent, N. L. and Evers, A. D. (1994) The Technology of Cereals. Elsevier Science Ltd. Oxford.

Khankari, K. K., Morey, R. V. and Patankar, S. V. (1990) Moisture Diffusion in Stored Grain due to Temperature Gradients. ASAE Paper No. 90-6581. St. Joseph, Mich.: ASAE.

Khankari, K. K., Morey, R. V. and Patankar, S. V. (1993a) Application of A Numerical Model for Prediction of Moisture Migration in Stored Grain. ASAE Paper No. 93-6018. St. Joseph, Mich.: ASAE.

Khankari, K. K., Patankar, S. V. and Morey, R. V. (1993b) A Mathematical Model for Natural Convection Moisture Migration in Stored Grain. ASAE Paper No. 93-6017. St. Joseph, Mich.: ASAE.

Khankari, K. K., Morey, R. V. and Patankar, S. V. (1994) Mathematical Model for Moisture Diffusion in Stored Grain Due to Temperature Gradients. Transaction of American Society of Agricultural Engineers. Vol. 37:5, pp. 1591-1604.

Krishnamurthy, K. (1975) Post Harvest Losses in Food Grains. Bulletin of Grain Technology. Vol. 13:1, pp. 33-49.

Lacey, J., Hamer, A. and Magan, N. (1994) Respiration and Loss in Stored Wheat Grain Under Different Environmental Conditions. Proceedings of the 6th International Working Conference on Stored-product Protection. Vol. 2, pp. 1007-1013.

Lai, F. S. (1980) Three-Dimensional Flow of Air Through Nonuniform Grain Beds. Transaction of American Society of Agricultural Engineers. Vol. 23:3, pp. 729-734.

Liu, B. Y. H. and Jordan, R. C. (1960) The Long-term Average Performance of Flat Plate Solar Collectors. Solar Energy, Vol. 7:2, pp. 53-74.

Ma, C. C. Y. and Iqbal, M. (1984) Statistical Comparison of Solar Radiation Correlations. Monthly Average Global and Diffuse Radiation on Horizontal Surfaces. Solar Energy. Vol. 33, pp. 143-148.

Mallinson, G. and de Vahl Davis, G. (1973) The method of false transient for the solution of coupled elliptic equations. Journal of Computation Physics. Vol. 12, pp. 435-461.

Marchant, J. A. (1976b) The Prediction of Air Flows in Crop Drying Systems by the Finite Element Method. Journal of Agricultural Engineering Research. Vol. 21, pp. 417-429.

Mazza, G., Jayas, D. S. and White, N. D. G. (1990) Moisture Sorption Isotherms of Flax Seed. Transaction of American Society of Agricultural Engineering Research. Vol. 33:4, pp. 1313-1317.

Noyes, R. T., Clary, B. L., Cloud, H. A. and McKenzie, B. A. (1989) Aeration of Stored Wheat. In Wheat Pest Management, eds. G. Mckinney, K. Swisher and N. Beauprez, pp. 43-49. Cooperative Extension Service, Oklahoma State University, Stillwater.

Obaldo, L. G., Harner, J. P. and Converse, H. H. (1991) Prediction of Moisture Changes in Stored Corn. Transaction of American Society of Agricultural Engineers. Vol. 34:4, pp. 1850-1857.

Page, J. K. (1961) The Estimation of Monthly Mean Values of Short Wave Radiation on Vertical and Inclined Surfaces from Sunshine Records for Latitudes 40°N-40°S. Process UN Conference on New Sources of Energy. Paper No. S98, Vol. 4, pp. 378-390.

Parti, M. (1993) Selection of Mathematical Models for Drying Grain in Thin-Layers. Journal of Agricultural Engineering Research. Vol. 54, pp. 339-352.

Parry, J. L. (1985) Mathematical Modelling and Computer Simulation of Heat and Mass Transfer in Agricultural Grain Drying: A Review. Journal of Agricultural Engineering Research. Vol. 32:1, pp. 1-29.

Patankar, S. V. (1980) Numerical Heat Transfer and Fluid Flow. McGraw - Hill.

Peaceman, D. W. and Rachford, H. H. (1955) The Numerical Solution of Parabolic and Elliptic Differential Equations. Journal of Society Industry Applied Mathematics. Vol. 3, pp. 28-47.

Pierce, R. O. and Thompson, T. L. (1975) Airflow Patterns in Conical- Shaped Piles of Grain. Transaction of American Society of Agricultural Engineers. Vol. 18:5, pp. 946-949.

Pixton, S. W. and Griffiths, H. J. (1971) Diffusion of Moisture through Grain. Journal of Stored Products Research. Vol. 7, pp. 133-152.

Pixton, S. W. and Warburton, S. (1971) Moisture Content / Relative Humidity Equilibrium of Some Cereal Grains at Different Temperatures. Journal of Stored Products Research. Vol. 6, pp. 283-293.

Pomeranz, Y. (1987) Modern Cereal Science and Technology. VCH Publisher, Inc. New York.

Rietveld, M. R. (1978) A New Method of Estimating the Regression Coefficients in the Formula Relating Solar Radiation to Sunshine. Agricultural Meteorology. Vol. 19, pp. 243-252.

Roberts, E. H. (1960) The Viability of Cereal Seed in Relation to Temperature and Moisture. Annals of Botany, New Series. Vol. 24:93, pp. 12-31.

Rumsey, T. R. and Fortis, T. (1984) Improving the Air Flow Distribution in a Batch Walnut Dryer. American Society of Agricultural Engineers. Vol. 27:3, pp. 938.

Salunkhe, D. K., Chavan, J. K. and Kadam, S. S. (1985) Posthavest Biotechnology of Cereals. CRC Press, Inc. Boca Raton, Florida.

Samarskii, A. A. and Andreyev, V. B. (1963) On a High-Accuracy Difference Scheme for an Elliptic Equation with Several Space Variables. U. S. S. R. Journal of Computational Mathematics and Mathematical Physics. Vol. 3, pp. 1373-1382.

Segerlind, L. J. (1982) Solving the Nonlinear Air Flow Equation. ASAE Paper No. 82-3017, ASAE, St. Joseph, MI 49085.

Sharp, J. R. (1982) A review of low temperature drying simulation models. Journal of Agriculture Engineering Research. Vol. 27, pp. 169-190.

Shedd, C. K.(1953) Resistance of Grains and Seeds to Air Flow. Agricultural Engineering. Vol. 34:9, pp. 616-619.

Singh, A. K. and Thorpe, G. R. (1993a) A solution Procedure for Three-dimensional Free Convective Flows in Peaked Bulks of Grain. Journal of Stored Product Research. Vol. 28:3, pp. 221-235.

Singh, A. K. and Thorpe, G. R. (1993b) The Application of a Grid Generation Technique to the Numerical Modelling of Heat and Moisture Movement in Peaked Bulks of Grains. Journal of Food Process Engineering. Vol. 16:2, pp. 127-145.

Singh, A. K. Leonardi, E. and Thorpe, G. R. (1993) Solution Procedure for the Equations that Govern Three Dimensional Free Convection in Bulk Stored Grains. Transaction of American Society of Agricultural Engineers. Vol. 36:4, pp. 1159-1173.

Sinha, R. N. (1971) Fungus as Food for some Stored Product Insects. Journal of Economical Entomology. Vol. 64, pp. 3.

Sinha, R. N. and Mair, W. E. (1973) Grain Storage: Part of a System. The AVI Publishing Company, Inc. United States.

Smith, E. A., Jayas, D. S., Muir, W. E., Alagusundaram, K. and Kalbande, V. H. (1992) Simulation of Grain Drying in Bins with Partially Perforated Floors. American Society of Agricultural Engineers. Vol. 35:3, pp. 909-915.

Smith, E. A. (1982) 3-Dimensional Analysis of Air Velocity and Pressure in Beds of Grain and Hay. Journal of Agricultural Engineering Research. Vol. 27, pp. 101-117.

Smith, S. E. (1947) The sorption of water vapour by high polymer. Journal of American Chemical Society. Vol. 69, pp. 646-651.

Spencer, H. B. (1969) A Mathematical Simulation of Grain Drying. Journal of Agricultural Engineering Research. Vol. 14:3, pp. 226-235.

Stanek, V. and Szekely, J. (1972) The effect of nonuniform porosity in causing flow maldistributions in isothermal packed beds. Canadian Journal of Chemical Engineering. Vol. 50, pp. 9-14.

Stanek, V. and Szekely, J. (1973) Flow Maldistribution in Two Dimensional Packed Beds. Part V: The Behaviour of Nonisothermal Systems. Canadian Journal of Chemical Engineering. Vol: 51, pp. 22-30.

Stanek, V. and Szekely, J. (1974) Three-dimensional Flow of Fluid Through Nonuniform Packed Beds. American Institute Chemical Engineering Journal. Vol. 20:5, pp. 974-980.

Stewart, J. A. (1975) Moisture Migration During Storage of Preserved High Moisture Grains. Transaction of American Society of Agricultural Engineers. Vol. 18:2, pp. 387-393, 400.

Stroshine, R. L. and Yang, X. (1990) Effects of Hybrid and Grain Damage on Estimated Dry Matter Loss for High-Moisture Shelled Corn. Transaction of American Society of Agricultural Engineers. Vol. 33:4, pp. 1291-1298.

Sutherland, J. W., Banks, P. J. and Griffiths, H. J. (1971) Equilibrium Heat and Moisture Transfer in Air Flow through Grain. Journal of Agricultural Engineering Research. Vol. 16, pp. 368-386.

Sutherland, J. W., Banks, P. J. and Elder, W. B. (1983) Interation between Successive Temperature or Moisture Fronts during Aeration of Deep Grain Beds. Journal of Agricultural Engineering Research. Vol. 28, pp. 1-19.

Sun, D. W. and Woods, J. L. (1993) The Moisture Content / Relative Humidity Equilibrium Relationship of Wheat- A Review. Drying Technology. Vol. 11:7, pp. 1523-1551.

Sun, D. W. and Woods, J. L. (1994) Low Temperature Moisture Transfer Characteristics of Wheat in Thin Layers. Transaction of American Society of Agricultural Engineers. Vol. 37:6, pp. 1919-1926.

Swinbank, W. C. (1963) Long-wave Radiation from Clear Skies. Quarterly Journal of the Royal Meteorological Society. Vol. 89.

Tang, J. and Sokhansanj, S. (1993) Geometric Changes in Lentil Seeds Caused by Drying. Journal of Agricultural Engineering Research. Vol. 56, pp. 313-326.

Thomas, L. H. (1949) Elliptic Problems in Linear Difference Equations Over a Network. Watson Scientific Computing Laboratory, Columbia University, New York.

Thompson, T. L. Peart, R. M. and Foster, G. H. (1968) Mathematical Simulation of Corn Drying - A New Model. Transaction of American Society of Agricultural Engineers. Vol. 11:4, pp. 582-586.

Thompson, T. L. (1972) Temporary Storage of High- Moisture Shelled Corn Using Continuous Aeration. Transaction of American Society of Agricultural Engineers. Vol. 15:2, PP. 333-337.

Thorpe, G. R. and Hunter, A. J. (1977) Expressions for Pressure and Velocity Distributions in Non-uniformly Aerated Grain Bulks. Journal of agricultural Engineering Research. Vol. 22, pp. 27-35.

Thorpe, G. R. (1980) Moisture Diffusion through Bulk Grain. Journal of Stored Products Research. Vol. 17, pp. 39-42.

Thorpe, G. R. (1982) Moisture Diffusion through Bulk Grain Subjected to a Temperature Gradient. Journal of Stored Products Research. Vol. 18, pp. 9-12.

Thorpe, G. R. and Elder, W. B. (1982) Modelling the Effects of Aeration on the Persistence of Chemical Pesticides Applied to Stored Bulk Grain. Journal of Stored Products Research. Vol. 18, pp. 103-114.

Thorpe, G. R. (1986) Some Fundamental Principles and Benefits of Aeration of Stored Grain. Proceedings of an International Seminar on Preserving Grain Quality by Aeration and Instored Drying. Kuala Lumpur, Malaysia. October 1985.

Thorpe, G. R. (1989) Homegeneous Chemical Reaction in Porous Media - The Spatial Averaging Formalism. CHEME CA'89: Technology for Our Third Century: Gold Coast Queensland 23-25 August, pp. 866-873.

Thorpe, G. R., Stokes, A. N. and Wilson, S. G. (1990) The Integral Heat of Wetting of Food Grains. Journal of Agricultural Engineering Research. Vol. 46, pp. 71-76.

Thorpe, G. R., Ochoa, J. A. and Whitaker, S. (1991a) The Diffusion of Moisture in Food Grains. I The Development of a Mass Transfer Equation. Journal of Stored Products Research. Vol. 27, pp. 1-9.

Thorpe, G. R., Ochoa, J. A. and Whitaker, S. (1991b) The Diffusion of Moisture in Food Grains. II Estimation of the Effective Thermal Diffusivity. Journal of Stored Products Research. Vol. 27, pp. 11-30.

Thorpe, G. R. (1994) Modelling Heat and Mass Transfer Phenomena in Bulk Stored Grains. Proceedings of the 6th International Working Conference on Stored-product Protection. Canberra, Australia. 17th-23rd April.

Thorpe, G. R. (1995) Heat and Mass Transfer in Bulk Stored Grains. A Component of a Course Presented at the XVth Seminar on Chemical Engineering *Instituto Tecnologico de Celaya*, Mexico.

Versavel, P. A. and Muir, W. E. (1988) Moisture Absorption by Wheat Spikes. Transaction of American Society of Agricultural Engineering Research. Vol. 31:4, pp. 1255-1259.

Whitney, J. D. and Porterfield, J. G. (1968) Moisture Movement in a Porous, Hygroscopic Solid. Transaction of America Society of Agricultural Engineering Research. Vol. 11, pp. 716-719,723.

Yang, X. and Williams, D. L. (1990) Airflow Resistance of Grain Sorghum as Affected by Bulk Density. Transaction of American Society of Agricultural Engineers. Vol. 33:6, pp. 1966-1970.

APPENDIX I

THOMAS ALGORITHM

Thomas algorithm is a particularly efficient method for solving the system of tri-diagonal matrix, and it is relatively simple. The product procedure is expressed as following.

The discretized triangular factorisation of the linear system is taken the form as:

$$b_1\omega_2 + c_1\omega_3 = d_1 - a_1\omega_1 \quad \text{(I-1)}$$

$$a_2\omega_2 + b_2\omega_3 + c_2\omega_4 = d_2 \quad \text{(I-2)}$$

$$a_3\omega_3 + b_3\omega_4 + c_3\omega_5 = d_3 \quad \text{(I-3)}$$

......

$$a_{ii}\omega_{i-1} + b_{ii}\omega_i + c_{ii}\omega_{i+1} = d_{ii} \quad \text{(I-4)}$$

......

$$a_{n-3}\omega_{n-3} + b_{n-3}\omega_{n-2} + c_{n-3}\omega_{n-1} = d_{n-3} \quad \text{(I-5)}$$

$$a_{n-2}\omega_{n-2} + b_{n-2}\omega_{n-1} = d_{n-2} - c_{n-2}\omega_n \quad \text{(I-6)}$$

in which $ii = i-1$ $(ii = 1,...n;$ and $i = 1,...n-1)$

if $ii = 1,$ $d_1 = d_1 - a_1\omega_1$

if $ii = n-2$ $d_{n-2} = d_{n-2} - c_{n-2}\omega_n$

The matrix of coefficients is:

$$\begin{bmatrix} b_1 & c_1 & & & & & & & \\ & a_2 & b_2 & c_2 & & & & & \\ & & a_3 & b_3 & c_3 & & & & \\ & & & \cdots\cdots & & & & & \\ & & & a_{ii} & b_{ii} & c_{ii} & & & \\ & & & & & \cdots\cdots & & & \\ & & & & & a_{n-3} & b_{n-3} & c_{n-3} \\ & & & & & & a_{n-2} & b_{n-2} \end{bmatrix} \times \begin{bmatrix} \omega_2 \\ \omega_3 \\ \omega_4 \\ \cdots \\ \omega_i \\ \cdots \\ \omega_{n-2} \\ \omega_{n-1} \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \cdots \\ d_i \\ \cdots \\ d_{n-3} \\ d_{n-2} \end{bmatrix}$$

(I-7)

The basic idea for solving this matrix is to reduce one column at an equivalent triangular system by means of substitution and manipulation of the matrix coefficients into the form as below:

$$\begin{bmatrix} \beta_1 & c_1 & & & & & \\ & \beta_2 & c_2 & & & & \\ & & \beta_3 & c_3 & & & \\ & & & \cdots\cdots & & & \\ & & & & \beta_{ii} & c_{ii} & \\ & & & & & \cdots\cdots & \\ & & & & & \beta_{nn-1} & c_{nn-1} \\ & & & & & & \beta_{nn} \end{bmatrix} \times \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \cdots \\ X_{ii} \\ \cdots \\ X_{nn-1} \\ X_{nn} \end{bmatrix} = \begin{bmatrix} \delta_1 \\ \delta_2 \\ \delta_3 \\ \cdots \\ \delta_{ii} \\ \cdots \\ \delta_{nn-1} \\ \delta_{nn} \end{bmatrix}$$

(I-8)

Firstly, we define:

$$b_1 = \beta_1 \tag{I-9}$$

$$d_1 = \delta_1 \tag{I-10}$$

$$\omega_2 = X_1;\ \omega_3 = X_2;\ \ldots\ldots\omega_{n-1} = X_{nn} \tag{I-11}$$

So equation (I-1) and (I-2) can be written as:

$$\beta_1 X_1 + c_1 X_2 = \delta_1 \tag{I-12}$$

$$a_2 X_1 + b_2 X_2 + c_2 X_3 = d_2 \tag{I-13}$$

Multiplying equation (I-12) by $\frac{a_2}{\beta_1}$, we will get:

$$a_2 X_1 + \frac{a_2}{\beta_1} c_1 X_2 = \frac{a_2}{\beta_1} \delta_1 \tag{I-14}$$

Then subtracting equation (I-13) from equation (I-14) yields:

$$(b_2 - \frac{a_2}{\beta_1} c_1) X_2 + c_2 X_3 = d_2 - \frac{a_2}{\beta_1} \delta_1 \tag{I-15}$$

We define again:

$$\beta_2 = b_2 - \frac{a_2}{\beta_1} c_1 \tag{I-16}$$

$$\delta_2 = d_2 - \frac{a_2}{\beta_1} \delta_1 \tag{I-17}$$

Equation (I-15) becomes:

$$\beta_2 X_2 + c_2 X_3 = \delta_2 \tag{I-18}$$

However, the summarisation calculation of the coefficient matrix is subject to the following recursion relations:

$$\beta_{ii} = b_{ii} - \frac{c_{ii} a_{ii}}{\beta_{ii-1}} \tag{I-19}$$

$$\delta_{ii} = d_{ii} \frac{\delta_{ii} a_{ii}}{\beta_{ii-1}} \tag{I-20}$$

Continuing this procedure until an equation with only one unknown is obtained.

The solution of tri-diagonal system can be solved by using last equation to solve for X_{nn}, such as:

$$X_{nn} = \frac{\delta_{nn}}{\beta_{nn}} \tag{I-21}$$

Then substituting the result of X_{nn} into the second last equation and solving for X_{nn-1}, and so forth. The general form for doing this procedure is written as:

$$X_{ii-1} = \frac{\delta_{ii-1} - c_{ii-1} X_{ii}}{\beta_{ii-1}} \tag{I-22}$$

Remembering $\omega_{nn} = X_{nn-1}$. Update the value of X_{ii} to ω_i, then the tri-diagonal system is completely solved.

APPENDIX II

NEWTON-RAPHSON METHOD

Newton-Raphson method is applied to a non-linear equation as:

$$f(X) = 0 \quad \text{(II-1)}$$

The algebraic rearrangements of equation (II-1) take the form is:

$$X = g(x) \quad \text{(II-2)}$$

When equation (II-1) has an approximation to a root $X_n = \alpha$, the better approximation given by X_{n+1}, generally it is written as:

$$X_{n+1} = X_n - \frac{f(X_n)}{f'(X_n)} \quad \text{(II-3)}$$

Equation (II-3) is the Newton-Raphson scheme. This scheme can be explained as the form as following:

$$\textit{Nest Estimate} = \textit{Current Estimate} + \textit{Correction Term} \quad \text{(II-4)}$$

in which the *Correction Term* of $\frac{-f(X_n)}{f'(X_n)}$ must be small, when X_n is close to the root if convergence is to be achieved.

When provide $f'(X_n) \neq 0$, the Newton- Raphson method is at least a second order process. The iterative equation (II-3) can be repeatedly used to find improved approximations to the real root α.

APPENDIX III

A LISTING OF PROGRAM - SILO3DFTN

```
c#########################SILO3D##############################
c      Calculate pressure,velocity,temperature and moisture          #
c      content and humidity distributions in ventilated grain        #
c      storage                                                       #
c#############################################################
c
      Program silo3Dftn
      Include'dry.inc'
      Include'solar.inc'
c
      Open (unit=5,file='crop.in',status='unknown')
      Open (unit=6,file='crop.out',status='unknown')
      Open (unit=7,file='pv.out',status='unknown')
      Open (unit=8,file='pv.in',status='unknown')
      Open (unit=9,file='preplot.out',status='unknown')
      Open (unit=12,file='vciplot.out',status='unknown')
      Open (unit=13,file='verplot.out',status='unknown')
      Open (unit=15,file='surface.out',status='unknown')
      Open (unit=28,file='crop6.out',status='unknown')
      Open (unit=29,file='crop7.out',status='unknown')
      Open (unit=87,file='prewall.out',status='unknown')
      Open (unit=88,file='crop19.out',status='unknown')
      Open (unit=89,file='crop20.out',status='unknown')
      Open (unit=90,file='perr.out',status='unknown')
      Open (unit=91,file='pretime.out',status='unknown')
      Open (unit=92,file='vraplot.out',status='unknown')
      Open (unit=93,file='velwall.out',status='unknown')
      Open (unit=94,file='airflow.out',status='unknown')
      Open (unit=95,file='aerat.out',status='unknown')
c
       Call input
       Call display
       Call setcon
       Call props
       Call initial
       Call silosize
       Call surtemp
       Call ductbc(p,pduct)
c
c     calculate pressure difference
       if(pnotfld)then
          Call pressure
          Call pvout
        else
          Call pvin
          Call pressure
          Call pvout
        end if
c
          write(*,9550)
 9550     format(1x,'pressure filed determined')
c
        Call flowrate
c
c      Output pressure filed for plotting
         Call pvmod
         Call pressplot
         Call wallpress
c
c      Output velocity filed for plotting
         Call velraplot
         Call velverplot
         Call velcirplot
         Call wallvel
c
         Call velsave
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c      Begin heat and mass transfer calculations
          write(*,9600)
 9600     format(1x,'Begin heat and mass transfer calculations')
c
c      Convert moisture content to dry basis
       if(mcb)then
          winitd=winit/(1.0-winit)
          write(28,3100)
          write(28,3200)tinit
          write(28,3210)winit
          write(28,3220)winitd
          write(28,3230)qu
c
 3100     format(///1x,5x,'Mcbea cooler is invoked')
```

```
3200    format(1x,5x,'initial grain temperature',f7.1)
3210    format(1x,5x,'initial grain mc(wet bulb)',f7.4)
3220    format(1x,5x,'initial grain mc(dry bulb)',f7.4)
3230    format(1x,5x,'flow rate through cooler(m**3/s)',f7.4)
      else
c
        write(28,3240)
3240    format(1x,5x,'Mcbea cooler is not invoked')
      end if
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c     Calculate the amplitude of the diurnal temperature veriation
c     (amp) and the mean ambient temperature (tmean).
c
      amp=tmax(1)-tmin(1)
      tmean=0.5*(tmax(1)+tmin(1))
c
      aerationtime=0.0
      timer=0.0
      time=0.0
      hamb=humidity
      Call tprise
      dayhour=1.0
      kount=0
c
      tprint0=0.0
      tprint5=real(maxday)/12.0
      tprint10=real(maxday)/6.0
      tprint15=real(maxday)/4.0
      tprint20=real(maxday)/3.0
      tprint25=(real(maxday)*5.0)/12.0
      tprint30=real(maxday)/2.0
      tprint35=(real(maxday)*7.0)/12.0
      tprint40=(real(maxday)*2.0)/3.0
      tprint45=(real(maxday)*3.0)/4.0
      tprint50=(real(maxday)*5.0)/6.0
      tprint55=(real(maxday)*11.0)/12.0
      tprint60=real(maxday)
c
      kprint0=0
      kprint5=0
      kprint10=0
      kprint15=0
      kprint20=0
      kprint25=0
      kprint30=0
      kprint35=0
      kprint40=0
      kprint45=0
      kprint50=0
      kprint55=0
      kprint60=0
      kkp2=0
      kkp4=0
c
      do 50 kkk=1, ku1
       do 52 kk=1, ku2
c
          taerate1=aerate1*3600.0
          taerate2=aerate2*3600.0
c
          if((timer.le.taerate1).or.(timer.gt.taerate2))then
c
             write(*,9575)
9575         format(1x,'Aeration system off')
c
             noflow=1
             Call velconduct
             dtg=dtgconduct
          else
             write(*,,9595)
9595         format(1x,'Aeration system on')
c
             noflow=0
             Call velaerate
             dtg=dtgaerate
             aerationtime=aerationtime+dtg/3600.0
c
             write(95,5420)aerationtime
5420         format(1x,'Total aeration fan operation=',f14.8,
     &              'hours')
          end if
```

```
c
      ifreq=2.0*3600.0/dtg
      kount=kount+1
      kkount=kount/ifreq
      kkkount=kkount*ifreq
      time=time+dtg
      timer=timer+dtg
c
      if(timer.gt.86400.0)then
        timer=timer-86400.0
      end if
c
      timeh=time/3600.0
      timeday=time/(3600.0*24.0)
      tstop=3600.0*24.0*real(maxday)
c
      dayhour=dayhour+dtg/3600.0
c
      if(dayhour.ge.25.0)then
        dayhour=dayhour-24.0
      end if
c
      kh=dayhour
c
      tamb=tmean+0.5*amp*sin((timeh-9.0)*3.14159/12.0)
c
      if(mcb)then
        Call wetbulb(tamb,hamb,twamb)
        Call mcbeasr(twamb,tamb,twout)
        Call twout1(twout,tin,hin)
      else
        tin=tamb+trise
        hin=hamb
      end if
c
      if(smith)then
        Call wesmith(hin,tin,win)
      else
        Call we(tin,hin,win)
      end if
c
      Call tbc(tamb)
      if(noflow.eq.0)Call ductbc(t,tin)

c
      do k=1,nz
       do j=1,nphi
        do i=1,nrm1
          if(smith)then
            Call humsm(i,j,k)
          else
            Call hum(i,j,k)
          end if
        end do
       end do
      end do
c
      if(noflow.eq.0)Call ductbc(h,hin)
c
      do 54 k=2,nzm1
       do 56 j=1,nphi
        do 58 i=2,nrm1
c
          if(smith)then
            Call respl(i,j,k,time)
          else
            Call resp(i,j,k)
          end if
c
          dmloss(i,j,k)=dmloss(i,j,k)+ddmdt*dtg
c
c   Solve moisture conservation equation and thermal energy equation
c
          Call moistu(i,j,k)
          Call tempera(i,j,k)
c
58        continue
56       continue
54      continue
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c   Establish boundary conditions
```

```
          Call tbc(tamb)
          if(noflow.eq.0)Call ductbc(tnew,tin)
c
          Call humbc(hin)
          if(noflow.eq.0)Call ductbc(h,hin)
c
          Call mcbc(win)
c
c     Re-set value of t(i,j,k) and w(i,j,k) for start of next
c     integration through a time step of dtg
          do i=1, nr
            do j=1, nphi
              do k=1, nz
                  t(i,j,k)=tnew(i,j,k)
                  w(i,j,k)=wnew(i,j,k)
c
c     Calculate interstital wet-bulb temperatures
                  tdrybulb=t(i,j,k)
                  humintg=h(i,j,k)
                  Call wetbulb(tdrybulb,humintg,twbb)
c
                  twetbulb(i,j,k)=twbb
                  tdrybulb=t(i,j,k)
                  Call insect(twbb,tdrybulb,i,j,k)
                  if(kkkount.eq.kount)Call viable(i,j,k)
              end do
            end do
          end do
c
          Call pesti
          Call dmlossbc(time)
          Call dayplot(timeday)
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c
      if(timeday.ge.tprint20)then
        if(kkp2.eq.0)then
c
          Call dayqut(time,timeh,timeday,aerationtime,timer,
     &                dayhour,kount,tamb,hamb,twamb,twout,
     &                tin,hin)
c
            kkp2=1
          end if
c
        else
          continue
        end if
c
        if(timeday.ge.tprint40)then
          if(kkp4.eq.0)then
c
            Call dayhaf(time,timeh,timeday,aerationtime,timer,
     &                  dayhour,kount,tamb,hamb,twamb,twout,
     &                  tin,hin)
c
            kkp4=1
          end if
c
        else
          continue
        end if
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c
            if(time.ge.tstop)then
c
            Call fileopen
c
            write(28,3950)time,timeh,timeday
            write(28,3955)aerationtime
            write(28,3960)timer
            write(28,3965)dayhour
            write(28,3970)kount
c
            Call totals
            Call outputcooler(tamb,hamb,twamb,twout,tin,hin)
            Call outputhm(time,timeh)
c
            Call results
c
            go to 50
```

```
          else
            continue
          end if
c
 52    continue
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c
        write(28,3950)time,timeh,timeday
        write(28,3955)aerationtime
        write(28,3960)timer
        write(28,3965)dayhour
        write(28,3970)kount
c
 3950  format(1x,5x,'time(SECOND)',f12.4,2x,'timeh(HOURS)',f12.4,2x,
   &        'TIME(DAY)',f12.4)
 3955  format(1x,5x,'aerationtime',f10.3,'HOURS')
 3960  format(1x,5x,'timer',f12.4)
 3965  format(1x,5x,'dayhour',f8.4)
 3970  format(1x,5x,'kount',I5)
c
        Call fileopen
        Call totals
        Call outputcooler(tamb,hamb,twamb,twout,tin,hin)
        Call outputhm(time,timeh)
c
        Call results
c
 50   continue
c
      end
c
c######################################################################
c                          Read input file                             #
c######################################################################
c
      Subroutine input
      Include'dry.inc'
      Include'solar.inc'
c
        read(5,1000)
        read(5,1000)
        read(5,1000)
        read(5,1010)nr,nphi,nz
        read(5,1010)ku1,ku2,maxday
        read(5,1010)kp1,kp2
        read(5,1020)dtp,dtgaerate,dtgconduct
        read(5,1030)wR,wz1,specflo,res
        read(5,1040)gam1,gam2
        read(5,1050)tinit,winit
        read(5,1060)tmean,tamp,humidity
        read(5,1070)nphimindc,nphimaxdc,nrmindc,nrmaxdc
        read(5,1040)aerate1,aerate2
        read(5,1080)mcb
        read(5,1080)pnotfld
        read(5,1080)smith
c
 1000  format( )
 1010  format(45x,3I8)
 1020  format(45x,3f6.1)
 1030  format(45x,4f10.4)
 1040  format(45x,2f6.2)
 1050  format(45x,2f7.4)
 1060  format(45x,2f5.1,E7.1)
 1070  format(45x,4I6)
 1080  format(45x,L8)
c
        read(5,1000)
        read(5,1090)phi
        read(5,1100)(suns(k),k=1,6)
        read(5,1110)(nuda(k),k=1,6)
        read(5,1100)(tmin(k),k=1,6)
        read(5,1100)(tmax(k),k=1,6)
        read(5,1120)alpha,epsilon
        read(5,1000)
c
 1090  format(45x,f6.2)
 1100  format(45x,6f5.1)
 1110  format(45x,6I4)
 1120  format(45x,2f4.1)
c
      return
```

```
      end
c
c##################################################################
c                        Display input file                      #
c##################################################################
c
      Subroutine display
      Include'dry.inc'
      Include'solar.inc'
c
      write(28,*)nr,nphi,nz
      write(28,*)ku1,ku2,maxday
      write(28,*)kp1,kp2
      write(28,*)dtp,dtgaerate,dtgconduct
      write(28,*)wR,wz1,specflo,res
      write(28,*)gam1,gam2
      write(28,*)tinit,winit
      write(28,*)tmean,tamp,humidity
      write(28,*)nphimindc,nphimaxdc,nrmindc,nrmaxdc
      write(28,*)aerate1,aerate2
      write(28,*)mcb
      write(28,*)pnotfld
      write(28,*)smith
c
      write(28,*)phi
      write(28,*)(suns(k),k=1,6)
      write(28,*)(nuda(k),k=1,6)
      write(28,*)(tmin(k),k=1,6)
      write(28,*)(tmax(k),k=1,6)
      write(28,*)alpha,epsilon
c
      return
      end
c
c##################################################################
c                    Set up grain silo constants                  #
c##################################################################
c
      Subroutine setcon
      Include'dry.inc'
c
c     Dimension of bulk ( The radius of silo is wR; Thetall of silo's
c     wall is wz1; The total high of silo is wz; The corn of silo's
c     angels are gamma1 and gamma2 ).
      nrm1=nr-1
      nrm2=nr-2
      nzm1=nz-1
      nzm2=nz-2
      pi=3.14159
      wphi=2.0*pi
      gamma1=gam1*pi/180.0
      gamma2=gam2*pi/180.0
      wz=wz1+wR*(tan(gamma1)+tan(gamma2))
c
c     Spacing of nodes
      dphi=wphi/real(nphi)
c
c     Set uniform mesh for initial trials
      hr(nrm1)=0.02/wR
      hr(nrm2)=0.02/wR
      hr(nr-3)=0.02/wR
      hr(nr-4)=0.02/wR
      hr(nr-5)=0.02/wR
      hr(nr-6)=0.05/wR
      nrm7=nr-7
c
      do i=1, nrm7
         hr(i)=(1.0-0.15/wR)/(real(nrm7))
      end do
c
      do  k=1, nzm1
          hz(k)=1.0/real(nzm1)
      end do
c
c     Calculate the distance of radius r(i)
      do  i=1, nr
          if(i.eq.1)then
             r(1)=0.0
          else
             r(i)=r(i-1)+hr(i-1)*wR
          end if
      end do
```

```
c
c     Calculate the distance of vertical z(i,k)
      do  k=1, nz
        do  i=1, nr
           z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
        end do
      end do
c
c     Set mesh transformation constants
      do 7 k=1, nz
       do 7 i=1, nr
c
         urz=z(i,k)-r(i)*tan(gamma1)
         vrz=real(wz)-r(i)*(tan(gamma1)+tan(gamma2))
         durzdr=-tan(gamma1)
         dvrzdr=-(tan(gamma1)+tan(gamma2))
c
         alpha1(i,k)=1.0/(real(wR))
         alpha2(i,k)=(durzdr*vrz-urz*dvrzdr)/(vrz**2)
         alpha3(i,k)=0.0
         alpha4(i,k)=1.0/vrz
c
         beta1(i,k)=alpha1(i,k)**2+alpha3(i,k)**2
         beta2(i,k)=alpha2(i,k)**2+alpha4(i,k)**2
         beta3(i,k)=2.0*(alpha1(i,k)*alpha2(i,k)+alpha3(i,k)
     &              *alpha4(i,k))
         beta4(i,k)=0.0
         beta5(i,k)=(4.0*(-vrz*durzdr+urz*dvrzdr)*dvrzdr)/(vrz
     &              **3)
         if(i.eq.1)go to 7
         beta6(i,k)=beta4(i,k)+alpha1(i,k)/r(i)
         beta7(i,k)=beta5(i,k)+alpha2(i,k)/r(i)
         beta8(i)=1.0/(r(i)**2)
 7    continue
c
c     Set finite difference operators at the internodals
      do i=2, nrm1
         cr1(i)=-hr(i)/(hr(i-1)*(hr(i-1)+hr(i)))
         cr2(i)=(hr(i)-hr(i-1))/(hr(i-1)*hr(i))
         cr3(i)=hr(i-1)/(hr(i)*(hr(i-1)+hr(i)))
         cr4(i)=2.0/(hr(i-1)*(hr(i)+hr(i-1)))
         cr5(i)=-2.0/(hr(i-1)*hr(i))
         cr6(i)=2.0/(hr(i)*(hr(i)+hr(i-1)))
      end do
c
      do k=2, nzm1
         cz1(k)=-hz(k)/(hz(k-1)*(hz(k-1)+hz(k)))
         cz2(k)=(hz(k)-hz(k-1))/(hz(k-1)*hz(k))
         cz3(k)=hz(k-1)/(hz(k)*(hz(k-1)+hz(k)))
         cz4(k)=2.0/(hz(k-1)*(hz(k)+hz(k-1)))
         cz5(k)=-2.0/(hz(k-1)*hz(k))
         cz6(k)=2.0/(hz(k)*(hz(k)+hz(k-1)))
      end do
c
      do j=1, nphi
         cp1(j)=-1.0/(2.0*dphi)
         cp2(j)=0.0
         cp3(j)=1.0/(2.0*dphi)
         cp4(j)=1.0/(dphi**2)
         cp5(j)=-2.0/(dphi**2)
         cp6(j)=1.0/(dphi**2)
      end do
c
c     Set finite difference operators at the boundaries
      crb4=(2.0*hr(nrm1)+hr(nrm2))/(hr(nrm1)*(hr(nrm2)+hr(nrm1)))
      crb5=-(hr(nrm1)+hr(nrm2))/(hr(nrm1)*hr(nrm2))
      crb6=hr(nrm1)/(hr(nrm2)*(hr(nrm2)+hr(nrm1)))
c
      czb1=-(2.0*hz(1)+hz(2))/(hz(1)*(hz(2)+hz(1)))
      czb2=(hz(1)+hz(2))/(hz(1)*hz(2))
      czb3=-hz(1)/(hz(2)*(hz(2)+hz(1)))
      czb4=(2.0*hz(nzm1)+hz(nzm2))/(hz(nzm1)*(hz(nzm2)+hz(nzm1)))
      czb5=-(hz(nzm1)+hz(nzm2))/(hz(nzm1)*hz(nzm2))
      czb6=hz(nzm1)/(hz(nzm2)*(hz(nzm2)+hz(nzm1)))
c
      return
      end
c
c#################################################################
c                    Set up the properties of the system          - #
c#################################################################
```

```
c
      Subroutine props
      Include'dry.inc'
      Real keff
c
c      constants associated with sorption isotherm
       cw1=-0.5016001
       cw2=5.6323
       cw3=-0.09832
       cw4=3.104
       cw5=11.4758
       p0=115840.0
       w0=(cw2**cw1/cw4**cw3)**(1.0/(cw3-cw1))
       revm=3.788
       wsat=0.32235
       h0=cw1*cw3/(cw3-cw1)*log(cw2/cw4)
c
c      constants in integral heat of wetting equation
       hw1=0.0003476
       hw2=2.119
       hw3=-32.10
       hw4=303.24
       hw5=-1156.8
       hw6=0.01867
       hw7=0.1049
       hw8=-5.810
       hw9=14.975
       hw10=-14.868
c
c      Thermophysical and other properties of the grain/air/water system
       eps=0.4
       rhos=1300.0
       rhob=(1.0-eps)*rhos
       rhoa=1.2
       cg=1300.0
       cw=4186.0
       ca=1000.0
       keff=0.15
       deff=0.000005
       dhvdt=-2377.0
       hv=2502390.0
       gn=0.622
       patm=101325.0
c
c      Relative humidity of air leaving McBea cooler
       rhout=0.75
c
c      Properties of the materal
       detal=0.5
c
c      Constants in the Lacey, Hamer and Magan equation
       a1lhm=0.00034583*20.0/21.0
       a2lhm=0.0001252*20.0/(21.0*3600.0)
       a3lhm=0.1737
       a4lhm=20.33
       a5lhm=0.9143
       a6lhm=-0.001036/3600.0
       a7lhm=-0.013634
       a8lhm=24.38
c
c      Insect population growth parameters
       crdom=0.0435
       csoryz1=0.052
       csoryz2=0.048
       csz=0.018
       trdom=13.0
       tsoryz1=9.0
       tsoryz2=9.8
       tsz=14.0
c
c      Kinetic constants determine the rate of pesticide decay
       bfe=0.036
       thalfe=8.467E6
       pduct=1000.0
c
      return
      end
c
c######################################################################
c       Set up initial conditions of grain silo. Such as: pressure,   #
c       temperature, moisture content and dry matter loss, etc.       #
c######################################################################
```

```
c
    Subroutine initial
    Include'dry.inc'
c
      winitd=winit/(1.0-winit)
      do i=1,nr
       do j=1,nphi
        do k=1,nz
           p(i,j,k)=0.0
           t(i,j,k)=tinit
           w(i,j,k)=winitd
           dmloss(i,j,k)=0.0
c
           cfe(i,j,k)=1.0
           viability(i,j,k)=0.99
           savep(i,j,k)=0.0
c
           if(smith)then
             Call humsmith(i,j,k)
           else
             Call hum(i,j,k)
           end if
c
           rdomconc(i,j,k)=1.0
           soryz1c(i,j,k)=1.0
           soryz2c(i,j,k)=1.0
           szconc(i,j,k)=1.0
        end do
       end do
      end do
c
    return
    end
c
c##################################################################
c                     Calculate the volume of silo                #
c##################################################################
c
    Subroutine silosize
    Include'dry.inc'
c
      volume=3.14159*wR**2*(wz1+(wR*(tan(gamma1)+tan(gamma2))/3.0))
      gmass=volume*rhob/1000.0
      qu=specflo*gmass/1000.0
      heightc(1)=0.0
      heightr(1)=wR*tan(gamma1)
      slope(1)=tan(gamma1)
c
      do k=2, nz
        heightc(k)=heightc(k-1)+wz*hz(k-1)
        heightr(k)=heightr(k-1)+wz1*hz(k-1)
        slope(k)=(heightr(k)-heightc(k))/wR
      end do
c
      totvol=0.0
      do k=1, nzm1
       do j=1, nphi
        do i=1, nrm1
          vol(i,j,k)=2.0*3.14159*((slope(k+1)-slope(k))*
     &              (r(i+1)**3-r(i)**3)/3.0+(r(i+1)**2-r(i)**2)
     &              *(heightc(k+1)-heightc(k))/2.0)/nphi
          totvol=totvol+vol(i,j,k)
        end do
       end do
      end do
c
       write(28,3677)volume,totvol
3677   format(/1x,5x,'volume=',f10.4,'sum of volume elements=',f10.4)
c
    return
    end
c
c##################################################################
c        Calculate the temperature of surface subjected to        #
c        heating and cooling by radiation and convection          #
c##################################################################
c
    Subroutine surtemp
    Include'dry.inc'
    Include'solar.inc'
c
       write(15,1000)
```

```
      write(15,1010)phi
      write(15,1020)(suns(k),k=1,6)
      write(15,1030)(nuda(k),k=1,6)
      write(15,1040)(tmin(k),k=1,6)
      write(15,1050)(tmax(k),k=1,6)
      write(15,1060)alpha,epsilon
c
 1000 format(1x,'Calculate the temperature of inclined surface:')
 1010 format(/1x,'Latitude=',f6.2)
 1020 format(1x,'Sunshine hours per day=',6f5.1)
 1030 format(1x,'Number of day of year=',6I4)
 1040 format(1x,'Minimum daily temperature=',6f5.1)
 1050 format(1x,'Maximum daily temperature=',6f5.1)
 1060 format(1x,'Absorbtivity=',f4.1,1x,'Emissivity=',f4.1)
c
    do ll=1, 2
      chi=pi/2.0
      if(ll.eq.2)then
        chi=(180.0-gam1)*pi/180.0
      end if
c
      do mm=1, nphi
        gamma=-pi/2.0+2.0*pi*real(mm-1)/real(nphi)
        rltime=0.0
c
        do kh=1, 24
          rltime=(kh-0.5)*3600.0
          Call surface
        end do
      end do
    end do
c
    return
    end
c
c##################################################################
c                Calculate the solar radiation loads in grain silo      #
c##################################################################
c
    Subroutine surface
    Include'dry.inc'
    Include'solar.inc'
    Real nu
c
     month=1.0+rltime/(3600.0*24.0*30.0)
     khour=1.0+rltime/3600.0
     hc=7.0
     rho=0.2
     eta=5.67E-08
     arc=pi/180.0
c
c   Calculate sun's declination
     delt=23.45*sin(2.0*pi*(284.0+nuda(month))/365.0)
c
c   Convert angles to radians
     nu=delt*arc
     phi=phi*arc
     cbp1=0.5*(1.0+cos(chi))
     cbm1=0.5*(1.0-cos(chi))
     rsurf=0.0
     rb=0.0
     rd=0.0
     rt=0.0
c
c   Calculate the length of the day and sunset hour angle ws
     x=-tan(phi)*tan(nu)
     ws=acos(x)
c
c   Calculate ws in degree
     wsdeg=acos(x)/arc
     day=(2.0/15.0)*wsdeg
     nightday=int(day/2.0)
     light=12.0-nightday
     dark=12.0+nightday
c
     if((khour.lt.light).or.(khour.gt.dark))goto 650
        kday=abs(12-khour)
c
c   Calculate the total,ht,and diffuse,hdradiation on a horizontal
c   surface
     gsc=1353.0
     a1=(24.0*3600.0*gsc/pi)*(1.0+0.033*cos(2.0*pi*nuda(month))
```

```
     &      /365.0))
        a2=cos(phi)*cos(nu)*sin(ws)+ws*sin(phi)*sin(nu)
        ho=a1*a2
        ht=ho*(0.18+0.62*suns(month)/day)
        hd=ht*(1.0-1.13*ht/ho)
c
c       Cross-check with Iqbal's expression
        a3=ho*(0.163+0.478*suns(month)/day)
        a4=ho*(0.655*(suns(month)/day)**2)
        hiqbal=a3-a4
c
        wd=kday*15.0*arc
        a5=0.409+0.5016001*sin(ws-60.0*arc)
        a6=0.6609-0.4767*sin(ws-60.0*arc)
        rt=(pi/24.0)*(a5+a6*cos(wd))*(cos(wd)-cos(ws))/(sin(ws)-ws*
     &      cos(ws))
        rt=rt*ht/3600.0
        rd=(pi/24.0)*(cos(wd)-cos(ws))/(24.0*(sin(ws)-ws*cos(ws)))
        rd=rd*hd/3600.0
        rp=rt-rd
c
c      Calculate the beam radiation on a sloping surface
        ww=15.0*arc*(khour-12.0)
        a7=sin(nu)*sin(phi)*cos(chi)
        a8=-sin(nu)*cos(phi)*sin(chi)*cos(gamma)
        a9=cos(nu)*cos(phi)*cos(chi)*cos(ww)
        a10=cos(nu)*sin(phi)*sin(chi)*cos(gamma)*cos(ww)
        a11=cos(nu)*sin(chi)*sin(gamma)*sin(ww)
        cost=a7+a8+a9+a10+a11
c
        costz=cos(nu)*cos(phi)*cos(ww)+sin(nu)*sin(phi)
        rb=cost/costz
c
        if(rb.lt.0.0)then
          rb=0.0
        end if
c
          rsurf=rp*rb
650       tbar=0.5*(tmax(month)+tmin(month))
          amp=tmax(month)-tmin(month)
          ta=tbar+0.5*amp*sin(pi*(khour-9.0)/12.0)
          tsky=0.05522*(ta+273.15)**1.5
          ts1=ta-5.0
c
c      Perform energy balance on surface
        do 200 kk=1, 20
           an1=alpha*(rsurf+rd*cbp1+rt*rho*cbm1)
           an=an1-hc*(ts1-ta)-cbp1*eta*epsilon*((ts1+273.15)**4
     &        -tsky**4)-eta*cbm1*epsilon*((ts1+273.15)**4-
     &        (ta+273.15)**4)
c
           dan=-hc-4.0*cbp1*eta*epsilon*(ts1+273.15)**3-
     &        4.0*cbm1*eta*epsilon*(ts1+273.15)**3
           tsnew=ts1-an/dan
c
           if(abs(tsnew-ts1).lt.1.0E-4)goto 500
200           ts1=tsnew
c
500           ts1=tsnew
              ts(ll,mm,kh)=ts1
c
          write(15,5010)delt,wsdeg,day
          write(15,5020)ht,hd,hiqbal
          write(15,5030)rt,rd,rp
          write(15,5040)rsurf,rb
          write(15,5050)ta,tsky
          write(15,5060)khour
          write(15,5070)an1,an,dan
          write(15,5080)wd
          write(15,5090)ww
          write(15,5100)a7,a8,a9,a10,a11
          write(15,5110)ts(ll,mm,kh)
c
5010  format(/1x,'Declination=',f6.2,'/','sunset hour anglea=',f6.2,
     &        '/','Daylength=',f6.2)
5020  format(1x,'Horizontal surface-Total radiation=',E10.4,'/',
     &        'Diffuse radiation=',E10.4,'/','Iqbal=',E10.4)
5030  format(1x,'Instantaneous radiation-Total=',E12.4,'/',
     &        'Diffuse=',E12.4,'/','Beam=',E10.4)
5040  format(1x,'Beam radiation on sloping surface',E12.7,'/',
     &        'Ratio=',E12.7)
c
```

```
5050   format(1x,'Ambient temp=',f5.2,'/','Sky temp=',f6.2)
5060   format(1x,'Hour of day=',I3)
5070   format(1x,'An1=',E12.4,'/','An=',E10.4,'/','Dan=',E11.4)
5080   format(1x,'Wd=',f8.5)
5090   format(1x,'First w=',f8.5)
5100   format(1x,'A7,A8,A9,A10,A11',5E12.4)
5110   format(1x,'Surface temperature',f8.5)
c
      return
      end
c
c#################################################################
c                 Calculate the boundary conditions at the duct          #
c#################################################################
c
      Subroutine ductbc(g,ductin)
      Include'dry.inc'
      Dimension g(imax,jmax,kmax)
c
      do j=nphimindc,nphimaxdc
        do i=nrmindc,nrmaxdc
          g(i,j,1)=ductin
        end do
      end do
c
      return
      end
c
c#################################################################
c                 Calculate pressure distributions                       #
c#################################################################
c
      Subroutine pressure
      Include'dry.inc'
      Include'solar.inc'
c
      atd=detal*dtp
      time=0.0
c
      do 150 loop=1,kp1
       do 100 lop=1, kp2
c
c     Start of iterations implicit solution in r(xi) direction.
c     Explicit in phi and z(eta) direction.
      do  k=2,nzm1
        do  j=1, nphi
          do  i=2, nrm1
c
            if( j.eq.1 ) then
              pijm1k= p(i,nphi,k)
            else
              pijm1k= p(i,j-1,k)
            end if
c
            if( j.eq.nphi ) then
              pijp1k= p(i,1,k)
            else
              pijp1k= p(i,j+1,k)
            end if
c
            ii=i-1
            a(ii)=-atd*(beta6(i,k)*cr1(i)+beta1(i,k)*cr4(i))
c
            b(ii)=1.0-atd*(beta6(i,k)*cr2(i)+beta1(i,k)*cr5(i))
c
            c(ii)=-atd*(beta6(i,k)*cr3(i)+beta1(i,k)*cr6(i))
c
            d(ii)=beta1(i,k)*(cr4(i)*p(i-1,j,k)+cr5(i)*p(i,j,k)+
     &          cr6(i)*p(i+1,j,k))+beta2(i,k)*(cz4(k)*p(i,j,k-1)
     &          +cz5(k)*p(i,j,k)+cz6(k)*p(i,j,k+1))+beta3(i,k)*
     &          (cz1(k)*(cr1(i)*p(i-1,j,k-1)+cr2(i)*p(i,j,k-1)+
     &          cr3(i)*p(i+1,j,k-1))+cz2(k)*(cr1(i)*p(i-1,j,k)+
     &          cr2(i)*p(i,j,k)+cr3(i)*p(i+1,j,k))+cz3(k)*(cr1(
     &          i)*p(i-1,j,k+1)+cr2(i)*p(i,j,k+1)+cr3(i)*p(i+1,
     &          j,k+1)))+beta6(i,k)*(cr1(i)*p(i-1,j,k)+cr2(i)*p(
     &          i,j,k)+cr3(i)*p(i+1,j,k))+beta7(i,k)*(cz1(k)*p(
     &          i,j,k-1)+cz2(k)*p(i,j,k)+cz3(k)*p(i,j,k+1))+
     &          beta8(i)*(pijm1k-2.0*p(i,j,k)+pijp1k)/(dphi**2)
c
          end do
c
            call thomas (qq,nrm2)
```

```
          do  i=1, nrm2
               ii=i+1
              omega1(ii,j,k)=qq(i)
          end do
        end do
      end do
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c     Implicit solution in phi direction. Explicit in r(xi) and z
c     (eta) direction
c
      do  i=2, nrm1
        do  k=2, nzm1
          do  j=1, nphi
c
            a(j)=-atd*beta8(i)*cp4(j)
c
            b(j)=1.0-atd*beta8(i)*cp5(j)
c
            c(j)=-atd*beta8(i)*cp6(j)
c
            d(j)=omega1(i,j,k)

          end do
c
            call thomas (qq,nphi)
            do  j=1, nphi
                omega2(i,j,k)=qq(j)
            end do
        end do
      end do
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c     Implicit solution in z(eta) direction. Explicit in r(xi)
c     and phi direction
c
      do  i=2, nrm1
        do  j=1, nphi
          do  k=2,nzm1
c
             kk=k-1
c
              a(kk)=-atd*(beta2(i,k)*cz4(k)+beta7(i,k)*cz1(k))
c
              b(kk)=1.0-atd*(beta2(i,k)*cz5(k)+beta7(i,k)*cz2(k))
c
              c(kk)=-atd*(beta2(i,k)*cz6(k)+beta7(i,k)*cz3(k))
c
              d(kk)=omega2(i,j,k)
c
          end do
c
            call thomas (qq,nzm2)
            do  kl=1,  nzm2
                 k=kl+1
                omega3(i,j,k)=qq(kl)
                pnew(i,j,k)=p(i,j,k)+qq(kl)*atd
            end do
        end do
      end do
c
      do  i=2, nrm1
        do  j=1, nphi
          do  k=2, nzm1
              p(i,j,k)=pnew(i,j,k)
          end do
        end do
      end do
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c     Calculate pressure boundary conditions
c
       Call pbc
c
c     Calculate middle of centre conditions
       do  k=2,nzm1
         do  j=1,nphi
             p(1,j,k)= sigmaP(k)/real(nphi)
         end do
       end do
c
       time=time+dtp
c
```

```
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      Call ptime(time)
      Call iterp(psierr)
      psi(lop)=psierr
c
        write(90,5888)lop,psi(lop)
 5888   format(1x,I8,E16.7)
c
        if(psierr.lt.5E-15)then
          go to 780
        else
          continue
        end if
c
 100  continue
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
 780      write(6,2100)time
 2100     format(1x,//7x,'TIME STEP=',f8.1)
c
          write(6,2200)
          write(6,2300)(i,i=1,nr)
c
 2200     format(1x,75x,'THE PRESSURE DISTRIBUTION NODAL
   &              POINTS')
 2300     format(1x,'PHI\R',21(4x,I2,4x))
c
        do  k=nz,1,-1
            write(6,2400)k
 2400       format(1x,'k=',I2)
c
          do  j=nphi,1,-1
              write(6,2500)j,(p(i,j,k),i=1,nr)
 2500         format(1x,I2,21(f10.4))
          end do
        end do
 150  continue
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
        Call velocity
c
          write(6,2600)
 2600     format(/1x,35x,'THE RADIAL COMPONENT OF VELOCITY')
c
        do k=1,nz
          do j=1,nphi
             write(6,2620)k,j,(Vra(i,j,k),i=1,nr)
c
 2620        format(1x,'k=',I2,1x,'j=',I2,2x,21(E12.4,1x))
          end do
        end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
          write(6,2640)
 2640     format(/1x,35x,'THE ANGULAR COMPONENT OF VELOCITY')
c
        do i=1,nr
          do k=1,nz
             write(6,2660)i,k,(Vci(i,j,k),j=1,nphi)
c
 2660        format(1x,'i=',I2,1x,'k=',I2,2x,20(E12.4,1x))
          end do
        end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
          write(6,2680)
 2680     format(/1x,35x,'THE VERTICAL COMPONENT OF VELOCITY')
c
        do j=1,nphi
          do i=1,nr
             write(6,2700)j,i,(Ver(i,j,k),k=1,nz)
c
 2700        format(1x,'j=',I2,1x,'i=',I2,2x,21(E12.4,1x))
          end do
        end do
c
      return
      end
c
c################################################################
c                   Calculate the sum of pressure near          #
c                   the centre line of the silo                 #
c################################################################
c
```

```
      Function sigmaP (k)
      Include'dry.inc'
      Real sigmaP
c
      sigmaP=0.0
c
          do j=1,nphi
              sigmaP=sigmaP+p(2,j,k)
          end do
      return
      end
c
c#################################################################
c                    Solve the tridiagonal matrix                #
c#################################################################
c
      Subroutine thomas (xx,nn)
      Include'dry.inc'
      Dimension xx(ikmax)
c
c     Set the value of beta(1) and delta(1)
        beta(1)=b(1)
        delta(1)=d(1)
c
c     Calculate successive value of beta(ij)
c     and delta(ij)
        do  ij=2, nn
            eps=a(ij)/beta(ij-1)
            beta(ij)=b(ij)-eps*c(ij-1)
            delta(ij)=d(ij)-eps*delta(ij-1)
        end do
c
c     Solve for xx(nn)
        xx(nn)=delta(nn)/beta(nn)
c
c     Solve for xx(jk) by back to substitution
        do  jk=nn-1,1,-1
            xx(jk)=(delta(jk)-c(jk)*xx(jk+1))/beta(jk)
        end do
c
      return
      end
c
c#################################################################
c                Calculate the pressure boundary conditions      #
c#################################################################
c
      Subroutine pbc
      Include'dry.inc'
c
c     Calculate the pressure boundary conditions along the sloping
c     bottom
      max=nr+nz-1
      maxm1=max-1
      do 8 j=1,nphi
        do 9 i=2, maxm1
c
              il=i-1
              if(i.eq.nr) then
                go to 3
              end if
c
              if(i.gt.nr) then
                go to 4
              end if
c
              a(il)=-alpha1(i,1)*sin(gamma1)*cr1(i)
              b(il)=-alpha1(i,1)*sin(gamma1)*cr2(i)-alpha2(i,1)
     &             *sin(gamma1)*czb1+alpha4(i,1)*cos(gamma1)*czb1
c
              c(il)=-alpha1(i,1)*sin(gamma1)*cr3(i)
              d(il)=(alpha2(i,1)*sin(gamma1)-alpha4(i,1)*cos(gamma1))
     &             *(czb2*p(i,j,2)+czb3*p(i,j,3))
c
              if(i.eq.2) d(il)=d(il)-a(il)*p(1,1,1)
              go to 9
c
c     Calaulate the pressure boundary conditions at the intersection
c     of the floor and wall of the silo
3             a(il)=(hr(nr-1)/(hr(nr-1)+hz(1)))-1.0
              b(il)=1.0
              c(il)=-hr(nr-1)/(hr(nr-1)+hz(1))
```

```
          d(il)=0.0
          go to 9
c
c    Calculate the pressure boundary conditions at the vertical wall
 4        k=i-nr+1
          a(il)=alpha2(nr,k)*cz1(k)
          b(il)=alpha2(nr,k)*cz2(k)+alpha1(nr,k)*crb4
          c(il)=alpha2(nr,k)*cz3(k)
          d(il)=-alpha1(nr,k)*crb5*p(nrm1,j,k)-alpha1(nr,k)*crb6*
   &            p(nrm2,j,k)
c
          if(k.eq.nzm1) d(il)=d(il)-c(il)*p(nr,j,k+1)
 9     continue
c
      maxm2=max-2
      Call thomas (qq,maxm2)
        do ib=1, maxm2
          il=ib+1
c
          if(ib.le.nrm1) then
            p(il,j,1)=qq(ib)
          end if
c
          k=ib-nrm1+1
          if(ib.gt.nrm1) then
            p(nr,j,k)=qq(ib)
          end if
c
        end do
 8    continue
c
c    Calculate the pressure boundary conditions at the top of the
c    bottom
       do j=1, nphi
          p(1,j,1)=-(czb2*p(1,j,2)+czb3*p(1,j,3))/czb1
       end do
c
c    Set conditions at duct
       Call ductbc(p,pduct)
c
     return
      end
c
c#################################################################
c                       Calculate the three components        #
c                       of velocity distributions             #
c#################################################################
c
      Subroutine velocity
      Include'dry.inc'
c
      nphihalf=nphi/2
c
       do i=1,nr
         do j=1,nphi
           do k=1,nz
c
            if(i.eq.1)then
              if(j.lt.nphihalf)then
                dpdxi=(p(2,j,k)-p(2,j+nphihalf,k))/(2.0*hr(i))
              else
                dpdxi=(p(2,j,k)-p(2,j-nphihalf,k))/(2.0*hr(i))
              end if
            end if
c
            if(i.gt.1)then
              if(i.lt.nr)then
                dpdxi=cr1(i)*p(i-1,j,k)+cr2(i)*p(i,j,k)+
   &                   cr3(i)*p(i+1,j,k)
              end if
c
              if(i.eq.nr)then
                dpdxi=crb4*p(nr,j,k)+crb5*p(nrm1,j,k)+
   &                   crb6*p(nrm2,j,k)
              end if
            end if
c
            if(j.eq.1)then
              pijm1k=p(i,nphi,k)
            else
              pijm1k=p(i,j-1,k)
            end if
```

```
c
         if(j.eq.nphi)then
           pijp1k=p(i,1,k)
         else
           pijp1k=p(i,j+1,k)
         end if
c
         if(i.gt.1)then
           dpdphi=cp1(j)*pijm1k+cp2(j)*p(i,j,k)+cp3(j)*pijp1k
         end if
c
         if(k.gt.1.and.k.lt.nz)then
           dpdeta=cz1(k)*p(i,j,k-1)+cz2(k)*p(i,j,k)+cz3(k)*
   &              p(i,j,k+1)
         end if
c
         if(k.eq.1)then
           dpdeta=czb1*p(i,j,1)+czb2*p(i,j,2)+czb3*p(i,j,3)
         end if
c
         if(k.eq.nz)then
           dpdeta=czb4*p(i,j,nz)+czb5*p(i,j,nzm1)+
   &              czb6*p(i,j,nzm2)
         end if
c
c   Calculate the velocity in radial,circular and vertical direction
c
        Vra(i,j,k)=-1.0/res*(alpha1(i,k)*dpdxi+alpha2(i,k)*
   &              dpdeta)
c
        if(i.gt.1)then
          Vci(i,j,k)=-1.0/(res*r(i))*dpdphi
        end if
c
        Ver(i,j,k)=-1.0/res*(alpha4(i,k)*dpdeta)
c
       end do
      end do
     end do
c
    return
    end
c
c################################################################
c                   Plot pressure and velocity fields           #
c################################################################
c
    Subroutine pvout
    Include'dry.inc'
c
    do k=1, nz
     do j=1, nphi
      do i=1, nr
         write(7,1000)i,j,k,p(i,j,k),Vra(i,j,k),Vci(i,j,k),
   &          Ver(i,j,k)
1000     format(1x,3I3,f9.4,2x,3(E12.4))
      end do
     end do
    end do
c
    return
    end
c
c################################################################
c                   Input pressure and velocity fields          #
c################################################################
c
    Subroutine pvin
    Include'dry.inc'
c
    do k=1, nz
     do j=1, nphi
      do i=1, nr
         read(8,1000)i,j,k,p(i,j,k),Vra(i,j,k),Vci(i,j,k),
   &          Ver(i,j,k)
1000     format(1x,3I3,f9.4,2x,3(E12.4))
      end do
     end do
    end do
c
    return
    end
```

```
c
c####################################################################
c                    Iteration contral for convegenc of pressre        #
c####################################################################
c
      Subroutine iterp(psierr)
      Include'dry.inc'
      Include'solar.inc'
c
      sm=0.0
      ep=0.0
c
       do k=1, nz
         do j=1, nphi
           do i=1, nr
             sm=sm+p(i,j,k)
             ep=ep+abs(p(i,j,k)-savep(i,j,k))
           end do
         end do
       end do
c
c     Calculate psi-error of pressure
       psierr=ep/sm
c
c     Save old value of pressure distributions
       do k=1, nz
         do j=1, nphi
           do i=1, nr
             savep(i,j,k)=p(i,j,k)
           end do
         end do
       end do
c
      return
      end
c
c####################################################################
c                 Calculate the air flow rate through                  #
c                 the upper surface of the silo                        #
c####################################################################
c
      Subroutine flowrate
      Include'dry.inc'
c
      airflow=0.0
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
c
c     Airflow in outer of grain regins
      do j=1,nphi
        do i=2, nrm1
          if((i.gt.1).and.(i.lt.nr))then
            dpde(i,j)=czb4*p(i,j,nz)+czb5*p(i,j,nzm1)+czb6*
     &                p(i,j,nzm2)
        end if
c
c     Airflow in the central region
        if(i.eq.1)then
          dpde(1,j)=czb4*p(1,1,nz)+czb5*p(1,1,nzm1)+czb6*
     &              p(1,1,nzm2)
        end if
c
c     Airflow in region nearest the wall
        if(i.eq.nr)then
          dpde(nr,j)=czb4*p(nr,j,nz)+czb5*p(nr,j,nzm1)+czb6*
     &               p(nr,j,nzm2)
        end if
       end do
      end do
c
c     Mean normal pressure
      do j=1, nphi
        do i=2, nr
          dpdn(i,j)=0.5*(sin(gamma2)*(alpha2(i-1,nz)*dpde(i-1,j)+
     &          alpha2(i,nz)*dpde(i,j))+cos(gamma2)*(alpha4(i-1,nz)*
     &          dpde(i-1,nz)+alpha4(i,nz)*dpde(i,j)))
c
c     Calculate pressure gradients in vertical direction at the surface
c
          airflow=airflow+3.14159*dpdn(i,j)*(r(i)**2-r(i-1)**2)/
```

```
     &              (cos(gamma2)*real(nphi)*res)
c
            air(i,j)=airflow
          end do
        end do
c
        qup=-airflow
c
        do i=2, nr
            write(94,5777)dpdn(i,jphi),air(i,jphi),
     &                      dpdn(i,jphiby1),air(i,jphiby1),
     &                      dpdn(i,jphiby2),air(i,jphiby2),
     &                      dpdn(i,jphiby3),air(i,jphiby3)
c
5777      format(1x,'dpdn',f16.10,1x,'air',7(f16.10,1x))
        end do
c
        return
        end
c
c##################################################################
c                   Modify the pressure and velocity fields to        #
c                   accomodate the revised duct pressure              #
c##################################################################
c
        Subroutine pvmod
        Include'dry.inc'
c
        pduct=qu/qup*pduct
c
        do i=1, nr
          do j=1, nphi
            do k=1,nz
              p(i,j,k)=qu/qup*p(i,j,k)
              Vra(i,j,k)=qu/qup*Vra(i,j,k)
              Vci(i,j,k)=qu/qup*Vci(i,j,k)
              Ver(i,j,k)=qu/qup*Ver(i,j,k)
            end do
          end do
        end do
c
        return
        end
c
c##################################################################
c                   Calculate pressure drop across grain              #
c                   and temperature rise in the fan                   #
c##################################################################
c
        Subroutine tprise
        Include'dry.inc'
c
c       The efficiency of the fan is taken as being 60%.
        etafan=0.6
        trise=pduct*qu/(rhoa*ca*etafan)
c
        return
        end
c
c##################################################################
c                   Using initial wet bulb temperature to             #
c                   calculate the dry bulb temperature                #
c##################################################################
c
        Subroutine wetbulb(tamb,hamb,twb)
        Include'dry.inc'
c
        twb=tamb
        do 160 kwb=1, 20
              ps=6.0E25/((twb+273.0)**5)*exp(-6800.0/(twb+273.0))
              ws=ps*gn/(patm-ps)
              dwsdps=gn/(patm-ps)*(1.0+ps/(patm-ps))
              dpsdt=ps*(6800.0/(twb+273.0)-5.0)/(twb+273.0)
              dwsdt=dwsdps*dpsdt
c
              fh=(tamb-twb)+hamb*(2502.0+1.809*tamb-4.186*twb)-
     &           (2502.0-2.377*twb)*ws
              dfhdt=-1.0-4.186*hamb-2502.0*dwsdt+2.377*(twb*dwsdt+ws)
              twbn=twb-fh/dfhdt
c
              if(abs(twb-twbn).lt.1.0E-06) go to 162
                twb=twbn
```

```
 160  continue
c
 162        twb=twbn
c
      return
      end
c
c################################################################
c                    Calculate the thermal performance           #
c                    of the mcbea grain cooling unit             #
c################################################################
c
      Subroutine mcbeasr(twin,tamb,twout)
      Include'dry.inc'
c
      twout=167.0*qu**3-262.0*qu**2+149.0*qu-43+1.2*twin-
     &      0.045*(35.0-tamb)/qu
c
      if(twout.lt.5.0)then
        twout=5.0
      end if
c
      return
      end
c
c################################################################
c                 Using wet bulb temperature to calculate the    #
c                 dry bulb temperature and humidity of air       #
c################################################################
c
      Subroutine twout1(twb,tin,hin)
      Include'dry.inc'
c
      Call Psat(pswbt,twb)
      ws=gn*pswbt/(patm-pswbt)
      tout=twb
c
      do 164 jwb=1, 20
         Call Psat(pstout,tout)
         hout=rhout*pstout*gn/(patm-rhout*pstout)
         fh=(tout-twb)+hout*(2502.0+1.809*tout-4.186*twb)-
     &        (2502.0-2.377*twb)*ws
c
         dhdp=rhout*gn/(patm-rhout*pstout)*(1.0+rhout/(patm-rhout*
     &          pstout))
         dpdt=pstout/(tout+273.0)*(6800.0/(tout+273.0)-5.0)
         dhdt=dhdp*dpdt
         dfhdt=1.0+dhdt*(2502.0+1.809*tout-4.186*twb)+1.809*hout
         toutn=tout-fh/dfhdt
c
         if(abs(toutn-tout).le.1.0E-5)go to 166
           tout=toutn
 164  continue
c
 166     tout=toutn
c
         Call Psat(pstout,tout)
         hout=rhout*pstout*0.622/(patm-rhout*pstout)
         tin=tout
         hin=hout
c
        return
        end
c
c################################################################
c                    Calculate the saturation vapour pressure    #
c                    of water as a function of temperature       #
c################################################################
c
      Subroutine psat(ps,ts)
      Include'dry.inc'
c
      ps=6.0E25/((ts+273.0)**5)*exp(-6800.0/(ts+273.0))
c
      return
      end
c
c################################################################
c       Calculate the equilibrium grain moisture content from    #
c       the temperature and the humidity by means of Smith's     #
c       sorption isostere equation                               #
c################################################################
```

```
c
      Subroutine wesmith(hum,temp,weg)
      Include'dry.inc'
c
      pvap=patm*hum/(gn+hum)
      ps=6.0E25/((temp+273.0)**5)*exp(-6800.0/(temp+273.0))
      rh=pvap/ps
      aaa=9.304-0.052076*temp
      bbb=4.1483+0.0215*temp
      wew=aaa-bbb*log(1.0-rh)
      weg=wew/(100.0-wew)
c
      return
      end
c
c###############################################################
c        Calculate grain moisture content at the inlet of the      #
c        bed as a function of air temperature and humidty          #
c###############################################################
c
      Subroutine we(temp,hum,weg)
      Include'dry.inc'
c
      pvap=patm*hum/(gn+hum)
      ps=6.0E25/((temp+273.0)**5)*exp(-6800.0/(temp+273.0))
c
      rh=pvap/ps
      hrev=log(rh)/log(ps/p0)
      w1=(((exp(hrev/cw3))/cw4)-((hrev/h0)**revm)/cw2*exp(hrev/cw1))/
     &     (1.0-(hrev/h0)**revm)
c
      do jw=1, 10
         ratw=(w1/w0)**cw5
         den=1.0-ratw
c
         dhdw1=cw1/(w1*den)+(ratw*cw1*cw5*log(cw2*w1))/(w1*den**2)-
     &            ratw*cw3/(w1*den)-cw5*cw3*ratw*log(cw4*w1)/(w1*den)-
     &            ratw**2*cw3*cw5*log(cw4*w1)/(w1*den**2)
c
         hsdhvm1=(cw1*log(cw2*w1)-ratw*cw3*log(cw4*w1))/den
         dfdh1=(ps/p0)**hsdhvm1*log(ps/p0)
         dfhdw=dfdh1*dhdw1
c
         fhw=(ps/p0)**hsdhvm1-rh
         wm=w1-fhw/dfhdw
         if(abs(wm-w1).lt.1.0E-5)go to 168
           w1=wm
       end do
c
 168  weg=wm
      if(weg.gt.wsat)then
        weg=wsat
      end if
c
      return
      end
c
c###############################################################
c        Calculate the humidity of the interstitial air from       #
c        the temperature and moisture content of the grain         #
c###############################################################
c
      Subroutine humsm(i,j,k)
      Include'dry.inc'
c
c     Calculate saturation vapour pressure of water.
       ps=(6.0E25/((t(i,j,k)+273.0)**5))*exp(-6800.0/(t(i,j,k)+273.0))
       mw=100.0*w(i,j,k)/(1.0+w(i,j,k))
       aaa=9.104-0.052076*t(i,j,k)
       bbb=4.1483+0.0215*t(i,j,k)
       arg=(aaa-mw)/bbb
       rh=1.0-exp(arg)
c
       pvap=ps*rh
       h(i,j,k)=(pvap*gn)/(patm-pvap)
c
c     Calculate the heat of sorption hs
       hvap=hv+dhvdt*t(i,j,k)
       hs=hvap*(1.0+23.0*exp(-40.0*w(i,j,k)))
c
      return
      end
```

```
c
c###################################################################
c        Calculate the humidity of the intergranular air as a         #
c        function of temperature and grains moisture content          #
c###################################################################
c
      Subroutine hum(i,j,k)
      Include'dry.inc'
c
c     Calculate saturation vapour pressure of water.
      ps=(6.0E25/(t(i,j,k)+273.0)**5)*exp(-6800.0/(t(i,j,k)+273.0))
c
c     Calculate relative humidity of interstitial air .
      hshvm1=(cw1*log(cw2*w(i,j,k))-(w(i,j,k)/w0)**cw5*cw3*log(cw4*
     &              w(i,j,k)))/(1.0-(w(i,j,k)/w0)**cw5)
c
      rh=(ps/p0)**hshvm1
      hvap=hv+dhvdt*t(i,j,k)
      hs=hvap*(1.0+hshvm1)
      pvap=ps*rh
      h(i,j,k)=(gn*pvap)/(patm-pvap)
c
      return
      end
c
c###################################################################
c                  Calculate the rate of respiration of wheat           #
c###################################################################
c
      Subroutine respl(i,j,k,time)
      Include'dry.inc'
c
       dmw=100.0*w(i,j,k)/(1.0+w(i,j,k))
       if(dmw.lt.14.5)then
          ddmdt=0.0
       else
          arg=(a5lhm+a6lhm*time+a7lhm*t(i,j,k))*(a8lhm-dmw)
          z1=1.0+exp(arg)
          z2=a2lhm
          z3=a1lhm+a2lhm*time
          z4=a6lhm*(a8lhm-w(i,j,k))*exp(arg)
          y=1.0+exp(a3lhm*(a4lhm-t(i,j,k)))
          ddmdt=(z1*z2-z3*z4)/(z1**2*y)
       end if
c
      return
      end
c
c###################################################################
c        Calculate the rate of dry matter loss of grain and the       #
c        elapsed physiological time  as a function of moisture        #
c        content temperature and time increment                       #
c###################################################################
c
      Subroutine resp(i,j,k)
      Include'dry.inc'
      Real mcdb,mcwb
c
c     Using moisture contents on wet basis(mcwb) and dry basis(mcdb).
      mcwb=w(i,j,k)*100.0/(1.0+w(i,j,k))
      mcdb=w(i,j,k)*100.0
      ar=32.3*exp(-0.1044*t(i,j,k)-1.856)
c
      if(t(i,j,k).le.15.and.mcwb.le.19)then
         tmod=ar
      end if
c
      if(t(i,j,k).gt.15)then
        if(mcwb.gt.19.and.mcwb.le.28)then
          tmod=ar+((mcwb-19.0)/100.0)*exp(0.0183*t(i,j,k)-0.28437)
        end if
      end if
c
      if(t(i,j,k).gt.15.and.mcwb.gt.28)then
        tmod=ar+9.000001E-02*exp(0.0183*t(i,j,k)-0.2847)
      end if
c
      wmod=1.0
      if(mcwb.gt.13.and.mcwb.le.35)then
        wmod=0.103*(exp(455.0/mcdb**1.53)-8.449999E-03*mcdb+1.558)
      end if
c
```

```
c     Equivalent increment of time as a function of grain moisture
c     content and temperature.
      dteq=dtg/(tmod*wmod)
      if(mcwb.lt.14)then
         teq(i,j,k)=0.0
      end if
c
      teq(i,j,k)=teq(i,j,k)+dteq
      ddmteq=(1.472E-09*exp(1.667E-06*teq(i,j,k)-1.0)+2.833E-09)
      ddmdt=ddmteq/(tmod*wmod)
c
      if(mcwb.lt.14)then
         ddmdt=0.0
      end if
c
      return
      end
c
c##################################################################
c                   Solve moisture conservation equation          #
c##################################################################
c
      Subroutine moistu(i,j,k)
      Include'dry.inc'
c
c     Set matching conditions for the humidities(when j=1 and j=nphi)
            if( j.eq.1 ) then
               hijm1k= h(i,nphi,k)
            else
               hijm1k= h(i,j-1,k)
            end if
c
            if( j.eq.nphi ) then
               hijp1k= h(i,1,k)
            else
               hijp1k= h(i,j+1,k)
            end if
c
c     Calculate effective velocities for up-wind differencing of the
c     moisture conservation equation.
            xi=real(i-1)/real(nr-1)
            Vxi=alpha1(i,k)*Vra(i,j,k)+alpha3(i,k)*Ver(i,j,k)
     &            -deff*beta6(i,k)
            Veta=alpha2(i,k)*Vra(i,j,k)+alpha4(i,k)*Ver(i,j,k)
     &            -deff*beta7(i,k)
            Vphi=Vci(i,j,k)/(wR*xi)
c
c     Calculate up-wind difference operators for moisture balance
            if(Vxi.gt.0.0)then
               cxi1=-1.0/hr(i-1)
               cxi2=1.0/hr(i-1)
               cxi3=0.0
            else
               cxi1=0.0
               cxi2=-1.0/hr(i)
               cxi3=1.0/hr(i)
            end if
c
            if(Veta.gt.0.0)then
               ceta1=-1.0/hz(k-1)
               ceta2=1.0/hz(k-1)
               ceta3=0.0
            else
               ceta1=0.0
               ceta2=-1.0/hz(k)
               ceta3=1.0/hz(k)
            end if
c
            if(Vphi.gt.0.0)then
               cphi1=-1.0/nphi
               cphi2=1.0/nphi
               cphi3=0.0
            else
               cphi1=0.0
               cphi2=-1.0/nphi
               cphi3=1.0/nphi
            end if
c
c     Calculate first derivatives of humidities
            dhdxi=cxi1*h(i-1,j,k)+cxi2*h(i,j,k)+cxi3*
     &            h(i+1,j,k)
            dhdeta=ceta1*h(i,j,k-1)+ceta2*h(i,j,k)+ceta3*
```

```
     &                h(i,j,k+1)
           dhdphi=cphi1*hijm1k+cphi2*h(i,j,k)+cphi3*hijp1k
c
c    Dispersion terms
           gamaw=beta1(i,k)*(cr4(i)*h(i-1,j,k)+cr5(i)*
     &               h(i,j,k)+cr6(i)*h(i+1,j,k))+beta2(i,k)*
     &               (cz4(k)*h(i,j,k-1)+cz5(k)*h(i,j,k)+cz6(k)*
     &               h(i,j,k+1))+beta3(i,k)*(cz1(k)*(cr1(i)*
     &               h(i-1,j,k-1)+cr2(i)*h(i,j,k-1)+cr3(i)*
     &               h(i+1,j,k-1))+cz2(k)*(cr1(i)*h(i-1,j,k)+
     &               cr2(i)*h(i,j,k)+cr3(i)*h(i+1,j,k))+cz3(k)*
     &               (cr1(i)*h(i-1,j,k+1)+cr2(i)*h(i,j,k+1)+
     &               cr3(i)*h(i+1,j,k+1)))+beta8(i)*(cp4(j)*
     &               hijm1k+cp5(j)*h(i,j,k)+cp6(j)*hijp1k)
c
           dwdt=-rhoa/rhob*(Vxi*dhdxi+Veta*dhdeta+Vphi*
     &              dhdphi)+(rhoa/rhob)*deff*gamaw+ddmdt*(0.6+
     &              w(i,j,k))
           wnew(i,j,k)=w(i,j,k)+dwdt*dtg
c
      return
      end
c
c#################################################################
c                     Solve thermal energy equation              #
c#################################################################
c
      Subroutine tempera(i,j,k)
      Include'dry.inc'
      Real keff,iw
c
           premu=rhoa*(ca+h(i,j,k)*(cw+dhvdt))
           Vxit=premu*(alpha1(i,k)*Vra(i,j,k)+alpha3(i,k)*
     &             Ver(i,j,k))-keff*beta6(i,k)
           Vetat=premu*(alpha2(i,k)*Vra(i,j,k)+alpha4(i,k)*
     &              Ver(i,j,k))-keff*beta7(i,k)
           Vphit=premu*Vci(i,j,k)/(wR*xi)
c
c    Set matching conditions for temperature when j=1 and j=nphi
           if( j.eq.1 ) then
              tijm1k= t(i,nphi,k)
           else
              tijm1k= t(i,j-1,k)
           end if
c
           if( j.eq.nphi ) then
              tijp1k= t(i,1,k)
           else
              tijp1k= t(i,j+1,k)
           end if
c
c    Set up wind difference operators
           if(Vxit.gt.0.0)then
             cxi1=-1.0/hr(i-1)
             cxi2=1.0/hr(i-1)
             cxi3=0.0
           else
             cxi1=0.0
             cxi2=-1.0/hr(i)
             cxi3=1.0/hr(i)
           end if
c
           if(Vetat.gt.0.0)then
             ceta1=-1.0/hz(k-1)
             ceta2=1.0/hz(k-1)
             ceta3=0.0
           else
             ceta1=0.0
             ceta2=-1.0/hz(k)
             ceta3=1.0/hz(k)
           end if
c
           if(Vphit.gt.0.0)then
             cphi1=-1.0/nphi
             cphi2=1.0/nphi
             cphi3=0.0
           else
             cphi1=0.0
             cphi2=-1.0/nphi
             cphi3=1.0/nphi
           end if
c
```

```
c    Calculate first derivatives with respect to temperatures
          dTdxi=cxi1*t(i-1,j,k)+cxi2*t(i,j,k)+cxi3*
   &             t(i+1,j,k)
          dTdeta=ceta1*t(i,j,k-1)+ceta2*t(i,j,k)+ceta3*
   &             t(i,j,k+1)
          dTdphi=cphi1*tijm1k+cphi2*t(i,j,k)+cphi3*tijp1k
c
c    Dispersion terms
          gamat=beta1(i,k)*(cr4(i)*t(i-1,j,k)+cr5(i)*
   &             t(i,j,k)+cr6(i)*t(i+1,j,k))+beta2(i,k)*
   &             (cz4(k)*t(i,j,k-1)+cz5(k)*t(i,j,k)+cz6(k)*
   &             t(i,j,k+1))+beta3(i,k)*(cz1(k)*(cr1(i)*
   &             t(i-1,j,k-1)+cr2(i)*t(i,j,k-1)+cr3(i)*
   &             t(i+1,j,k-1))+cz2(k)*(cr1(i)*t(i-1,j,k)+
   &             cr2(i)*t(i,j,k)+cr3(i)*t(i+1,j,k))+cz3(k)*
   &             (cr1(i)*t(i-1,j,k+1)+cr2(i)*t(i,j,k+1)+
   &             cr3(i)*t(i+1,j,k+1)))+beta8(i)*(cp4(j)*
   &             tijm1k+cp5(j)*t(i,j,k)+cp6(j)*tijp1k)
c
          if(smith)then
            iw=0.575*(exp(-40.0*w(i,j,k))-1.0)
          else
            if(w(i,j,k).le.0.1)then
              iw=hw1+hw2*w(i,j,k)+hw3*w(i,j,k)**2+hw4*
   &             w(i,j,k)**3+hw5*w(i,j,k)**4
            end if
c
            if(w(i,j,k).gt.0.1)then
              iw=hw6+hw7*w(i,j,k)+hw8*w(i,j,k)**2+hw9*
   &             w(i,j,k)**3+hw10*w(i,j,k)**4
            end if
          end if
c
          dhwdt=dhvdt*iw
          denom=(cg+cw*w(i,j,k)+dhwdt)*rhob+eps*rhoa*(ca+
   &             h(i,j,k)*(cw+dhvdt))
c
          aa=rhob*hs*dwdt
          bb=-(Vxit*dTdxi+Vetat*dTdeta+Vphit*dTdphi)
          cc=keff*gamat
          dd=rhob*ddmdt*(1.5778E+07-0.6*hvap)
          ee=rhob*ddmdt*hvap*(iw-w(i,j,k))
          tnew(i,j,k)=t(i,j,k)+dtg*(aa+bb+cc+dd+ee)/denom
c
      return
      end
c
c##################################################################
c                  Calculate the temperature boundary conditions          #
c##################################################################
c
      Subroutine tbc(tamb)
      Include'dry.inc'
      Include'solar.inc'
c
c    Wall temperature
       do j=1, nphi
         do k=1, nz
           t(nr,j,k)=ts(1,j,kh)
           tnew(nr,j,k)=ts(1,j,kh)
         end do
       end do
c
c    Bottom of silo temperature
       do j=1, nphi
         do i=1, nr
           t(i,j,1)=ts(2,j,kh)
           tnew(i,j,1)=ts(2,j,kh)
         end do
       end do
c
c    Temperature at the upper surface
       do j=1, nphi
         do i=2, nrm1
           tnew(i,j,nz)=tnew(i,j,nzm1)
           if(noflow.eq.1)tnew(i,j,nz)=tamb
         end do
       end do
c
c    Temperature at centre of the bulk
       do j=1, nphi
c
```

```
c     At the upper surface
          t(1,j,nz)=-(czb5*t(1,j,nzm1)+czb6*t(1,j,nzm2))/czb4
          if(noflow.eq.1)t(1,j,nz)=tamb
          tnew(1,j,nz)=t(1,j,nz)
c
c     At bottom of the cone
          t(1,j,1)=ts(2,1,mm)
          tnew(1,j,1)=t(1,j,1)
       end do
c
c     At the centre
       do k=2, nzm1
         do j=1, nphi
           t(1,j,k)=sigmaT(k)/real(nphi)
           tnew(1,j,k)=t(1,j,k)
         end do
       end do
c
      return
      end
c
c####################################################################
c                Calculate the sum of temperature nera              #
c                the centre line of the silo                        #
c####################################################################
c
      Function sigmaT(k)
      Include'dry.inc'
      Real sigmaT
c
      sigmaT=0.0
c
          do j=1,nphi
             sigmaT=sigmaT+t(2,j,k)
          end do
      return
      end
c
c####################################################################
c                Calculate the humidity boundary conditions         #
c####################################################################
c
      Subroutine humbc(hin)
      Include'dry.inc'
c
c     Along the sloping bottom
      max=nr+nz-1
      maxm1=max-1
      do 8 j=1,nphi
        do 9 i=2, maxm1
c
            il=i-1
              if(i.eq.nr) then
                go to 3
              end if
c
              if(i.gt.nr) then
                go to 4
              end if
c
            a(il)=-alpha1(i,1)*sin(gamma1)*cr1(i)
            b(il)=-alpha1(i,1)*sin(gamma1)*cr2(i)-alpha2(i,1)
     &            *sin(gamma1)*czb1+alpha4(i,1)*cos(gamma1)*czb1
            c(il)=-alpha1(i,1)*sin(gamma1)*cr3(i)
            d(il)=(alpha2(i,1)*sin(gamma1)-alpha4(i,1)*cos(gamma1))
     &            *(czb2*h(i,j,2)+czb3*h(i,j,3))
c
            if(i.eq.2) d(il)=d(il)-a(il)*h(1,1,1)
            go to 9
c
c     Calaulate the humidity boundary condition at the intersection
c     of the floor and wall of the silo
3           a(il)=(hr(nr-1)/(hr(nr-1)+hz(1)))-1.0
            b(il)=1.0
            c(il)=-hr(nr-1)/(hr(nr-1)+hz(1))
            d(il)=0.0
            go to 9
c
c     Calculate the humidity boundary conditions at the vertical wall
4           k=i-nr+1
            a(il)=alpha2(nr,k)*cz1(k)
            b(il)=alpha2(nr,k)*cz2(k)+alpha1(nr,k)*crb4
```

```
          c(il)=alpha2(nr,k)*cz3(k)
         d(il)=-alpha1(nr,k)*crb5*h(nrm1,j,k)-alpha1(nr,k)*crb6*
   &             h(nrm2,j,k)
          if(k.eq.nzm1) d(il)=d(il)-c(il)*h(nr,j,k+1)
9     continue
c
      maxm2=21-2
      Call thomas (qq,maxm2)
        do ib=1, maxm2
           il=ib+1
c
           if(ib.le.nrm1) then
              h(il,j,1)=qq(ib)
           end if
c
           k=ib-nrm1+1
           if(ib.gt.nrm1) then
              h(nr,j,k)=qq(ib)
           end if
c
        end do
8     continue
c
c     Calculate middle of centre conditions
        do  k=2,nzm1
          do  j=1,nphi
              h(1,j,k)= sigmaH(k)/real(nphi)
          end do
        end do
c
c     Calculate humidity at the top of the grain
        do j=1, nphi
           h(nr,j,nz)=-(czb5*h(nr,j,nzm1)+czb6*h(nr,j,nzm2))/czb4
c
c     Calculate humidity at the peak of the grain
           h(1,j,nz)=-(czb5*h(1,j,nzm1)+czb6*h(1,j,nzm2))/czb4
c
c     Calculate humidity at the top of the bottom
           h(1,j,1)=-(czb2*h(1,j,2)+czb3*h(1,j,3))/czb1
c
        end do
c
c     Calculate humidity at duct
         if(noflow.eq.0)Call ductbc(h,hin)
c
      return
      end
c
c##################################################################
c                 Calculate the sum of humidity near                #
c                 the centre line of the silo                       #
c##################################################################
c
      Function sigmaH(k)
      Include'dry.inc'
      Real sigmaH
c
      sigmaH=0.0
c
          do j=1,nphi
             sigmaH=sigmaH+h(2,j,k)
          end do
      return
      end
c
c##################################################################
c               Calculate grain moisture content boundary conditions     #
c##################################################################
c
      Subroutine mcbc(win)
      Include'dry.inc'
c
c     Moisture contents in the centre of the silo
      do k=2, nzm1
        do j=1, nphi
           wnew(1,j,k)=sigmaW(k)/real(nphi)
        end do
      end do
c
c     Moisture content along the floor
      do j=1, nphi
        do i=1, nr
```

```
      wnew(i,j,1)=((hz(1)+hz(2))*wnew(i,j,2)-hz(1)*wnew(i,j,3))
     &            /hz(2)
c
c   Moisture content along the upper surface
      wnew(i,j,nz)=((hz(nzm1)+hz(nzm2))*wnew(i,j,nzm1)-hz(nzm1)
     &             *wnew(i,j,nzm2))/hz(nzm2)
c
     end do
   end do
c
c   Moisture content along the wall
   do j=1, nphi
     do k=1, nz
      wnew(nr,j,k)=((hr(nrm1)+hr(nrm2))*wnew(nrm1,j,k)-hr(nrm1)
     &             *wnew(nrm2,j,k))/hr(nrm2)
     end do
   end do
c
c   Moisture content at the duct
   if(noflow.eq.0)Call ductbc(wnew,win)
c
   return
   end
c
c###################################################################
c            Calculate the sum of moisture content                 #
c            near the centre of the silo                           #
c###################################################################
c
   Function sigmaW (k)
   Include'dry.inc'
   Real sigmaW
c
   sigmaW=0.0
      do j=1,nphi
        sigmaW=sigmaW+wnew(2,j,k)
      end do
c
   return
   end
c
c###################################################################
c            Calculate insect polpulation growth                   #
c###################################################################
c
   Subroutine insect(twbb,tdrybulb,i,j,k)
   Include'dry.inc'
c
    dtweek=dtg/(3600.0*24.0*7.0)
c
c   Increase rhyzopertha dominica
    rdom=crdom*(twbb-trdom)
    if(rdom.lt.0.0)rdom=0.0
    if(tdrybulb.ge.40.0)rdom=0.0
    rdomconc(i,j,k)=rdomconc(i,j,k)*exp(rdom*dtweek)
c
c   Increase sitophilus oryzael
    soryz1=csoryz1*(twbb-tsoryz1)
    if(soryz1.lt.0.0)soryz1=0.0
    if(tdrybulb.ge.35.0)soryz1=0.0
    soryz1c(i,j,k)=soryz1c(i,j,k)*exp(soryz1*dtweek)
c
c   Increase sitophilus oryzae2
    soryz2=csoryz2*(twbb-tsoryz2)
    if(soryz2.lt.0.0)soryz2=0.0
    if(tdrybulb.ge.35.0)soryz2=0.0
    soryz2c(i,j,k)=soryz2c(i,j,k)*exp(soryz2*dtweek)
c
c   Increase sitophilus zeamais
    sz=csz*(twbb-tsz)
    if(sz.lt.0.0)sz=0.0
    if(tdrybulb.ge.35.0)sz=0.0
    szconc(i,j,k)=szconc(i,j,k)*exp(sz*dtweek)
c
   return
   end
c
c###################################################################
c            Calculate dry matter loss along boundries             #
c###################################################################
c
   Subroutine dmlossbc(time)
```

```
      Include'dry.inc'
c
c     Along the wall
      do k=1, nz
        do j=1, nphi
          if(smith)then
            Call resplhm(nr,j,k,time,ddmdt)
          else
            Call resp(nr,j,k,ddmdt)
          end if
c
          dmloss(nr,j,k)=dmloss(nr,j,k)+ddmdt*dtg
        end do
      end do
c
c     Calculate upper surface of the grain
      do j=1, nphi
        do i=1, nr
          if(smith)then
            Call resplhm(i,j,nz,time,ddmdt)
          else
            Call resp(i,j,nz,ddmdt)
          end if
c
          dmloss(i,j,nz)=dmloss(i,j,nz)+ddmdt*dtg
        end do
      end do
c
c     Calculate along the floor
      do j=1, nphi
        do i=1, nr
          if(smith)then
            Call resplhm(i,j,1,time,ddmdt)
          else
            Call resp(i,j,1,ddmdt)
          end if
c
          dmloss(i,j,1)=dmloss(i,j,1)+ddmdt*dtg
        end do
      end do
c
      return
      end
c
c#################################################################
c                     Save the velocity fields during           #
c                     aeration fan is turned on                 #
c#################################################################
c
      Subroutine velsave
      Include'dry.inc'
c
      do i=1, nr
        do j=1, nphi
          do k=1, nz
              vrsave(i,j,k)=Vra(i,j,k)
              vpsave(i,j,k)=Vci(i,j,k)
              vzsave(i,j,k)=Ver(i,j,k)
          end do
        end do
      end do
c
      return
      end
c
c#################################################################
c                     Reinstate the velocity fields after       #
c                     aeration fan is turned off                #
c#################################################################
c
      Subroutine velaerate
      Include'dry.inc'
c
      do i=1, nr
        do j=1, nphi
          do k=1, nz
              Vra(i,j,k)=vrsave(i,j,k)
              Vci(i,j,k)=vpsave(i,j,k)
              Ver(i,j,k)=vzsave(i,j,k)
          end do
        end do
      end do
```

```
c
      return
      end
c
c#################################################################
c                Set the velocity fields to zero during              #
c                aeration fan is turned off                          #
c#################################################################
c
      Subroutine velconduct
      Include'dry.inc'
c
      do i=1, nr
        do j=1, nphi
          do k=1, nz
              Vra(i,j,k)=0.0
              Vci(i,j,k)=0.0
              Ver(i,j,k)=0.0
          end do
        end do
      end do
c
      return
      end
c
c#################################################################
c             Calculate the concentration of pesticide on the grains  #
c#################################################################
c
      Subroutine pesti
      Include'dry.inc'
c
      do i=1, nr
        do j=1, nphi
          do k=1, nz
              pvap=patm*h(i,j,k)/(0.622+h(i,j,k))
              ts=t(i,j,k)
              Call psat(ps,ts)
              rh=pvap/ps
              argfe=-(1.3863*rh*dtg*10.0**(bfe*(t(i,j,k)-30.0)))/
     &              thalfe
              cfe(i,j,k)=cfe(i,j,k)*exp(argfe)
          end do
        end do
      end do
c
      return
      end
c
c#################################################################
c       Using the input values of temperature, grain moisture        #
c       content and the initial germination viability of the         #
c       seeds to calculade the updated viability after a time        #
c       dtg*ifreq/(3600.0*24)                                        #
c#################################################################
c
      Subroutine viable(i,j,k)
      Include'dry.inc'
c
      heat=t(i,j,k)
      wc=100.0*w(i,j,k)/(1.0+w(i,j,k))
      dead=1.0-viability(i,j,k)
c
      Call lifedata(sigma,halif,heat,wc)
c
      timev=halif
      Call timod(timev,dead,halif,sigma)
      timev=timev+dtg*real(ifreq)/(3600.0*24.0)
c
      Call survivors(surv,dead,timev,halif,sigma)
      viability(i,j,k)=surv
c
      return
      end
c
c#################################################################
c       Calculates the half life of seeds and the standard           #
c       deviation of the mortality of the grains                     #
c#################################################################
c
      Subroutine lifedata(sigma,halif,heat,wc)
c
```

```
      halif=10.0**(5.067-0.108*wc-0.05*heat)
      sigma=0.35*halif
c
      return
      end
c
c##################################################################
c           Calculates the time and timev from                    #
c           when half the seeds are dead                          #
c##################################################################
c
      Subroutine timod(timev,dead,halif,sigma)
c
      arg=abs(timev-halif)/(sigma*sqrt(2.0))
       do j=1, 20
c
         Call erf(arg,erfx,derfx)
c
         if(dead.le.0.5)then
           func=dead-0.5*(1.0-erfx)
           x=arg-func/(0.5*derfx)
c
           if(x.lt.0.0)x=0.0
         else
c
           func=dead-0.5*(1.0+erfx)
           x=arg+func/(0.5*derfx)
c
           if(x.lt.0.0)x=0.0
         end if
c
         if(abs(arg-x).lt.1.0e-5) go to 15
           arg=x
       end do
c
 15   arg=x
c
      if(dead.ge.0.5)then
        timev=halif+sigma*sqrt(2.0)*arg
      else
        timev=halif-sigma*sqrt(2.0)*arg
      end if
c
      return
      end
c
c##################################################################
c         Calculates values of erf(arg)and derf(arg)               #
c         designated as erfx and derf respectively.                #
c         If arg is less than or equal tounity is                  #
c         evaluated from its series expansion which                #
c         converges rapidly. If arg is greater than                #
c         unity it is evaluated by integrating its                 #
c         integral representation using Simpson's rule,            #
c         with the lower bound on arg taken as unity.              #
c##################################################################
c
      Subroutine erf(arg,erfx,derfx)
c
      if(arg.le.1.0)then
        Call erf1(arg,erfx)
      else
c
c       In this integration of the error function by Simpson's rule.
c       we have:
c         n=the even number of integration intervals
c         h=lenght of the integration interval
c         x10=sum of the two extreme ordinates
c         x11=sum of the ordinates at even nodes
c         x12=sum of the ordinates at odd nodes
c
        n=int(arg)*20
        nm1=n-1
        hp=(arg-1.0)/real(n)
        x10=exp(-1.0)+exp(-(arg**2))
        x11=0.0
        x12=0.0
c
        do i=1, nm1
          x=1.0+real(i)*hp
          ii=0.5*i
          ii2=ii*2
```

```
c
          if(i.eq.ii2)then
            x12=x12+exp(-(x**2))
          else
            x11=x11+exp(-(x**2))
          end if
c
        end do
c
        erfx=0.8427008+2.0*hp*(x10+2.0*x12+4.0*x11)/
     &        (3.0*sqrt(3.14159))
c
      end if
c
      derfx=2.0/sqrt(3.14159)*exp(-arg**2)
c
      return
      end
c
c##################################################################
c        Calculates the error function as a direct integration      #
c        of a series expansion of the integrand                     #
c##################################################################
c
      Subroutine erf1(arg,erfx)
c
       nfact=1
       sum=arg
c
       do n=2, 100
          nfact=nfact*(n-1)
          add=(-1.0)**(n-1)*arg**(2*n-1)/((2*n-1)*nfact)
          sum=sum+add
          if(abs(add).lt.1.0e-6)go to 11
       end do
c
 11     erfx=2.0*sum/sqrt(3.14159)
c
      return
      end
c
c##################################################################
c                Calculate the number of dead and surviving insects  #
c##################################################################
c
      Subroutine survivors(surv,dead,timev,halif,sigma)
c
      arg=abs(timev-halif)/(sigma*sqrt(2.0))
      Call erf(arg,erfx,derfx)
c
      if(timev.le.halif)then
        dead=0.5*(1.0-erfx)
      else
        dead=0.5*(1.0+erfx)
      end if
c
      surv=1.0-dead
c
      return
      end
c
c##################################################################
c        Output the variation of grain silo conditions              #
c        by time increase                                           #
c##################################################################
c
      Subroutine dayplot(timeday)
      Include'dry.inc'
c
      nrby2=nr/2+1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      nzby1=nz/4+1
      nzby2=nz/2+1
      nzby3=(nz*3)/4+1
c
      if(timeday.ge.tprint0)then
        if(kprint0.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &                 w(1,1,nzby2), h(1,1,nzby2),
     &                 viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                 soryz2c(1,1,nzby2),szconc(1,1,nzby2),
```

```
     &              dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &              rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
      write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &              w(nrby2,jphiby1,nzby3),
     &              h(nrby2,jphiby1,nzby3),
     &              viability(nrby2,jphiby1,nzby3),
     &              soryz1c(nrby2,jphiby1,nzby3),
     &              soryz2c(nrby2,jphiby1,nzby3),
     &              szconc(nrby2,jphiby1,nzby3),
     &              dmloss(nrby2,jphiby1,nzby3),
     &              cfe(nrby2,jphiby1,nzby3),
     &              rdomconc(nrby2,jphiby1,nzby3),
     &              twetbulb(nrby2,jphiby1,nzby3)
c
      write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &              w(nrby2,jphiby2,nzby1),
     &              h(nrby2,jphiby2,nzby1),
     &              viability(nrby2,jphiby2,nzby1),
     &              soryz1c(nrby2,jphiby2,nzby1),
     &              soryz2c(nrby2,jphiby2,nzby1),
     &              szconc(nrby2,jphiby2,nzby1),
     &              dmloss(nrby2,jphiby2,nzby1),
     &              cfe(nrby2,jphiby2,nzby1),
     &              rdomconc(nrby2,jphiby2,nzby1),
     &              twetbulb(nrby2,jphiby2,nzby1)
c
      kprint0=1
    end if
  else
    continue
  end if
c
  if(timeday.ge.tprint5)then
    if(kprint5.eq.0)then
      write(29,7000)timeday,t(1,1,nzby2),
     &              w(1,1,nzby2), h(1,1,nzby2),
     &              viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &              soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &              dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &              rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
      write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &              w(nrby2,jphiby1,nzby3),
     &              h(nrby2,jphiby1,nzby3),
     &              viability(nrby2,jphiby1,nzby3),
     &              soryz1c(nrby2,jphiby1,nzby3),
     &              soryz2c(nrby2,jphiby1,nzby3),
     &              szconc(nrby2,jphiby1,nzby3),
     &              dmloss(nrby2,jphiby1,nzby3),
     &              cfe(nrby2,jphiby1,nzby3),
     &              rdomconc(nrby2,jphiby1,nzby3),
     &              twetbulb(nrby2,jphiby1,nzby3)
c
      write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &              w(nrby2,jphiby2,nzby1),
     &              h(nrby2,jphiby2,nzby1),
     &              viability(nrby2,jphiby2,nzby1),
     &              soryz1c(nrby2,jphiby2,nzby1),
     &              soryz2c(nrby2,jphiby2,nzby1),
     &              szconc(nrby2,jphiby2,nzby1),
     &              dmloss(nrby2,jphiby2,nzby1),
     &              cfe(nrby2,jphiby2,nzby1),
     &              rdomconc(nrby2,jphiby2,nzby1),
     &              twetbulb(nrby2,jphiby2,nzby1)
c
      kprint5=1
    end if
  else
    continue
  end if
c
  if(timeday.ge.tprint10)then
    if(kprint10.eq.0)then
      write(29,7000)timeday,t(1,1,nzby2),
     &              w(1,1,nzby2), h(1,1,nzby2),
     &              viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &              soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &              dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &              rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
      write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
```

```
     &                 w(nrby2,jphiby1,nzby3),
     &                 h(nrby2,jphiby1,nzby3),
     &                 viability(nrby2,jphiby1,nzby3),
     &                 soryz1c(nrby2,jphiby1,nzby3),
     &                 soryz2c(nrby2,jphiby1,nzby3),
     &                 szconc(nrby2,jphiby1,nzby3),
     &                 dmloss(nrby2,jphiby1,nzby3),
     &                 cfe(nrby2,jphiby1,nzby3),
     &                 rdomconc(nrby2,jphiby1,nzby3),
     &                 twetbulb(nrby2,jphiby1,nzby3)
c
        write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &                 w(nrby2,jphiby2,nzby1),
     &                 h(nrby2,jphiby2,nzby1),
     &                 viability(nrby2,jphiby2,nzby1),
     &                 soryz1c(nrby2,jphiby2,nzby1),
     &                 soryz2c(nrby2,jphiby2,nzby1),
     &                 szconc(nrby2,jphiby2,nzby1),
     &                 dmloss(nrby2,jphiby2,nzby1),
     &                 cfe(nrby2,jphiby2,nzby1),
     &                 rdomconc(nrby2,jphiby2,nzby1),
     &                 twetbulb(nrby2,jphiby2,nzby1)
c
        kprint10=1
       end if
      else
       continue
      end if
c
      if(timeday.ge.tprint15)then
       if(kprint15.eq.0)then
        write(29,7000)timeday,t(1,1,nzby2),
     &                 w(1,1,nzby2), h(1,1,nzby2),
     &                 viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                 soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &                 dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &                 rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
        write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &                 w(nrby2,jphiby1,nzby3),
     &                 h(nrby2,jphiby1,nzby3),
     &                 viability(nrby2,jphiby1,nzby3),
     &                 soryz1c(nrby2,jphiby1,nzby3),
     &                 soryz2c(nrby2,jphiby1,nzby3),
     &                 szconc(nrby2,jphiby1,nzby3),
     &                 dmloss(nrby2,jphiby1,nzby3),
     &                 cfe(nrby2,jphiby1,nzby3),
     &                 rdomconc(nrby2,jphiby1,nzby3),
     &                 twetbulb(nrby2,jphiby1,nzby3)
c
        write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &                 w(nrby2,jphiby2,nzby1),
     &                 h(nrby2,jphiby2,nzby1),
     &                 viability(nrby2,jphiby2,nzby1),
     &                 soryz1c(nrby2,jphiby2,nzby1),
     &                 soryz2c(nrby2,jphiby2,nzby1),
     &                 szconc(nrby2,jphiby2,nzby1),
     &                 dmloss(nrby2,jphiby2,nzby1),
     &                 cfe(nrby2,jphiby2,nzby1),
     &                 rdomconc(nrby2,jphiby2,nzby1),
     &                 twetbulb(nrby2,jphiby2,nzby1)
c
        kprint15=1
       end if
      else
       continue
      end if
c
      if(timeday.ge.tprint20)then
       if(kprint20.eq.0)then
        write(29,7000)timeday,t(1,1,nzby2),
     &                 w(1,1,nzby2), h(1,1,nzby2),
     &                 viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                 soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &                 dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &                 rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
        write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &                 w(nrby2,jphiby1,nzby3),
     &                 h(nrby2,jphiby1,nzby3),
     &                 viability(nrby2,jphiby1,nzby3),
     &                 soryz1c(nrby2,jphiby1,nzby3),
```

```
     &                  soryz2c(nrby2,jphiby1,nzby3),
     &                  szconc(nrby2,jphiby1,nzby3),
     &                  dmloss(nrby2,jphiby1,nzby3),
     &                  cfe(nrby2,jphiby1,nzby3),
     &                  rdomconc(nrby2,jphiby1,nzby3),
     &                  twetbulb(nrby2,jphiby1,nzby3)
c
          write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &                  w(nrby2,jphiby2,nzby1),
     &                  h(nrby2,jphiby2,nzby1),
     &                  viability(nrby2,jphiby2,nzby1),
     &                  soryz1c(nrby2,jphiby2,nzby1),
     &                  soryz2c(nrby2,jphiby2,nzby1),
     &                  szconc(nrby2,jphiby2,nzby1),
     &                  dmloss(nrby2,jphiby2,nzby1),
     &                  cfe(nrby2,jphiby2,nzby1),
     &                  rdomconc(nrby2,jphiby2,nzby1),
     &                  twetbulb(nrby2,jphiby2,nzby1)
c
          kprint20=1
        end if
      else
        continue
      end if
c
      if(timeday.ge.tprint25)then
        if(kprint25.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &                  w(1,1,nzby2), h(1,1,nzby2),
     &                  viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                  soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &                  dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &                  rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
          write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &                  w(nrby2,jphiby1,nzby3),
     &                  h(nrby2,jphiby1,nzby3),
     &                  viability(nrby2,jphiby1,nzby3),
     &                  soryz1c(nrby2,jphiby1,nzby3),
     &                  soryz2c(nrby2.jphiby1,nzby3),
     &                  szconc(nrby2,jphiby1,nzby3),
     &                  dmloss(nrby2,jphiby1,nzby3),
     &                  cfe(nrby2,jphiby1,nzby3),
     &                  rdomconc(nrby2,jphiby1,nzby3),
     &                  twetbulb(nrby2,jphiby1,nzby3)
c
          write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &                  w(nrby2,jphiby2,nzby1),
     &                  h(nrby2,jphiby2,nzby1),
     &                  viability(nrby2,jphiby2,nzby1),
     &                  soryz1c(nrby2,jphiby2,nzby1),
     &                  soryz2c(nrby2,jphiby2,nzby1),
     &                  szconc(nrby2,jphiby2,nzby1),
     &                  dmloss(nrby2,jphiby2,nzby1),
     &                  cfe(nrby2,jphiby2,nzby1),
     &                  rdomconc(nrby2,jphiby2,nzby1),
     &                  twetbulb(nrby2,jphiby2,nzby1)
c
          kprint25=1
        end if
      else
        continue
      end if
c
      if(timeday.ge.tprint30)then
        if(kprint30.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &                  w(1,1,nzby2), h(1,1,nzby2),
     &                  viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                  soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &                  dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &                  rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
          write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &                  w(nrby2,jphiby1,nzby3),
     &                  h(nrby2,jphiby1,nzby3),
     &                  viability(nrby2,jphiby1,nzby3),
     &                  soryz1c(nrby2,jphiby1,nzby3),
     &                  soryz2c(nrby2,jphiby1,nzby3),
     &                  szconc(nrby2,jphiby1,nzby3),
     &                  dmloss(nrby2,jphiby1,nzby3),
     &                  cfe(nrby2,jphiby1,nzby3),
```

```
     &             rdomconc(nrby2,jphiby1,nzby3),
     &             twetbulb(nrby2,jphiby1,nzby3)
c
        write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &             w(nrby2,jphiby2,nzby1),
     &             h(nrby2,jphiby2,nzby1),
     &             viability(nrby2,jphiby2,nzby1),
     &             soryz1c(nrby2,jphiby2,nzby1),
     &             soryz2c(nrby2,jphiby2,nzby1),
     &             szconc(nrby2,jphiby2,nzby1),
     &             dmloss(nrby2,jphiby2,nzby1),
     &             cfe(nrby2,jphiby2,nzby1),
     &             rdomconc(nrby2,jphiby2,nzby1),
     &             twetbulb(nrby2,jphiby2,nzby1)
c
        kprint30=1
      end if
    else
      continue
    end if
c
    if(timeday.ge.tprint35)then
      if(kprint35.eq.0)then
        write(29,7000)timeday,t(1,1,nzby2),
     &             w(1,1,nzby2), h(1,1,nzby2),
     &             viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &             soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &             dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &              rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
        write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &             w(nrby2,jphiby1,nzby3),
     &             h(nrby2,jphiby1,nzby3),
     &             viability(nrby2,jphiby1,nzby3),
     &             soryz1c(nrby2,jphiby1,nzby3),
     &             soryz2c(nrby2,jphiby1,nzby3),
     &             szconc(nrby2,jphiby1,nzby3),
     &             dmloss(nrby2,jphiby1,nzby3),
     &             cfe(nrby2,jphiby1,nzby3),
     &             rdomconc(nrby2,jphiby1,nzby3),
     &             twetbulb(nrby2,jphiby1,nzby3)
c
        write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &             w(nrby2,jphiby2,nzby1),
     &             h(nrby2,jphiby2,nzby1),
     &             viability(nrby2,jphiby2,nzby1),
     &             soryz1c(nrby2,jphiby2,nzby1),
     &             soryz2c(nrby2,jphiby2,nzby1),
     &             szconc(nrby2,jphiby2,nzby1),
     &             dmloss(nrby2,jphiby2,nzby1),
     &             cfe(nrby2,jphiby2,nzby1),
     &             rdomconc(nrby2,jphiby2,nzby1),
     &             twetbulb(nrby2,jphiby2,nzby1)
c
        kprint35=1
      end if
    else
      continue
    end if
c
    if(timeday.ge.tprint40)then
      if(kprint40.eq.0)then
        write(29,7000)timeday,t(1,1,nzby2),
     &             w(1,1,nzby2), h(1,1,nzby2),
     &             viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &             soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &             dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &              rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
        write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &             w(nrby2,jphiby1,nzby3),
     &             h(nrby2,jphiby1,nzby3),
     &             viability(nrby2,jphiby1,nzby3),
     &             soryz1c(nrby2,jphiby1,nzby3),
     &             soryz2c(nrby2,jphiby1,nzby3),
     &             szconc(nrby2,jphiby1,nzby3),
     &             dmloss(nrby2,jphiby1,nzby3),
     &             cfe(nrby2,jphiby1,nzby3),
     &             rdomconc(nrby2,jphiby1,nzby3),
     &             twetbulb(nrby2,jphiby1,nzby3)
c
        write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
```

```
     &                  w(nrby2,jphiby2,nzby1),
     &                  h(nrby2,jphiby2,nzby1),
     &                  viability(nrby2,jphiby2,nzby1),
     &                  soryz1c(nrby2,jphiby2,nzby1),
     &                  soryz2c(nrby2,jphiby2,nzby1),
     &                  szconc(nrby2,jphiby2,nzby1),
     &                  dmloss(nrby2,jphiby2,nzby1),
     &                  cfe(nrby2,jphiby2,nzby1),
     &                  rdomconc(nrby2,jphiby2,nzby1),
     &                  twetbulb(nrby2,jphiby2,nzby1)
c
          kprint40=1
        end if
      else
        continue
      end if
c
      if(timeday.ge.tprint45)then
        if(kprint45.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &                  w(1,1,nzby2), h(1,1,nzby2),
     &                  viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                  soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &                  dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &                  rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
          write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &                  w(nrby2,jphiby1,nzby3),
     &                  h(nrby2,jphiby1,nzby3),
     &                  viability(nrby2,jphiby1,nzby3),
     &                  soryz1c(nrby2,jphiby1,nzby3),
     &                  soryz2c(nrby2,jphiby1,nzby3),
     &                  szconc(nrby2,jphiby1,nzby3),
     &                  dmloss(nrby2,jphiby1,nzby3),
     &                  cfe(nrby2,jphiby1,nzby3),
     &                  rdomconc(nrby2,jphiby1,nzby3),
     &                  twetbulb(nrby2,jphiby1,nzby3)
c
          write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &                  w(nrby2,jphiby2,nzby1),
     &                  h(nrby2,jphiby2,nzby1),
     &                  viability(nrby2,jphiby2,nzby1),
     &                  soryz1c(nrby2,jphiby2,nzby1),
     &                  soryz2c(nrby2,jphiby2,nzby1),
     &                  szconc(nrby2,jphiby2,nzby1),
     &                  dmloss(nrby2,jphiby2,nzby1),
     &                  cfe(nrby2,jphiby2,nzby1),
     &                  rdomconc(nrby2,jphiby2,nzby1),
     &                  twetbulb(nrby2,jphiby2,nzby1)
c
          kprint45=1
        end if
      else
        continue
      end if
c
      if(timeday.ge.tprint50)then
        if(kprint50.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &                  w(1,1,nzby2), h(1,1,nzby2),
     &                  viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &                  soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &                  dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &                  rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
          write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &                  w(nrby2,jphiby1,nzby3),
     &                  h(nrby2,jphiby1,nzby3),
     &                  viability(nrby2,jphiby1,nzby3),
     &                  soryz1c(nrby2,jphiby1,nzby3),
     &                  soryz2c(nrby2,jphiby1,nzby3),
     &                  szconc(nrby2,jphiby1,nzby3),
     &                  dmloss(nrby2,jphiby1,nzby3),
     &                  cfe(nrby2,jphiby1,nzby3),
     &                  rdomconc(nrby2,jphiby1,nzby3),
     &                  twetbulb(nrby2,jphiby1,nzby3)
c
          write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &                  w(nrby2,jphiby2,nzby1),
     &                  h(nrby2,jphiby2,nzby1),
     &                  viability(nrby2,jphiby2,nzby1),
     &                  soryz1c(nrby2,jphiby2,nzby1),
```

```
     &               soryz2c(nrby2,jphiby2,nzby1),
     &               szconc(nrby2,jphiby2,nzby1),
     &               dmloss(nrby2,jphiby2,nzby1),
     &               cfe(nrby2,jphiby2,nzby1),
     &               rdomconc(nrby2,jphiby2,nzby1),
     &               twetbulb(nrby2,jphiby2,nzby1)
c
          kprint50=1
        end if
      else
        continue
      end if
c
      if(timeday.ge.tprint55)then
        if(kprint55.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &               w(1,1,nzby2), h(1,1,nzby2),
     &               viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &               soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &               dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &               rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
          write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &               w(nrby2,jphiby1,nzby3),
     &               h(nrby2,jphiby1,nzby3),
     &               viability(nrby2,jphiby1,nzby3),
     &               soryz1c(nrby2,jphiby1,nzby3),
     &               soryz2c(nrby2,jphiby1,nzby3),
     &               szconc(nrby2,jphiby1,nzby3),
     &               dmloss(nrby2,jphiby1,nzby3),
     &               cfe(nrby2,jphiby1,nzby3),
     &               rdomconc(nrby2,jphiby1,nzby3),
     &               twetbulb(nrby2,jphiby1,nzby3)
c
          write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &               w(nrby2,jphiby2,nzby1),
     &               h(nrby2,jphiby2,nzby1),
     &               viability(nrby2,jphiby2,nzby1),
     &               soryz1c(nrby2,jphiby2,nzby1),
     &               soryz2c(nrby2,jphiby2,nzby1),
     &               szconc(nrby2,jphiby2,nzby1),
     &               dmloss(nrby2,jphiby2,nzby1),
     &               cfe(nrby2,jphiby2,nzby1),
     &               rdomconc(nrby2,jphiby2,nzby1),
     &               twetbulb(nrby2,jphiby2,nzby1)
c
          kprint55=1
        end if
      else
        continue
      end if
c
      if(timeday.ge.tprint60)then
        if(kprint60.eq.0)then
          write(29,7000)timeday,t(1,1,nzby2),
     &               w(1,1,nzby2), h(1,1,nzby2),
     &               viability(1,1,nzby2),soryz1c(1,1,nzby2),
     &               soryz2c(1,1,nzby2),szconc(1,1,nzby2),
     &               dmloss(1,1,nzby2),cfe(1,1,nzby2),
     &               rdomconc(1,1,nzby2),twetbulb(1,1,nzby2)
c
          write(88,7000)timeday,t(nrby2,jphiby1,nzby3),
     &               w(nrby2,jphiby1,nzby3),
     &               h(nrby2,jphiby1,nzby3),
     &               viability(nrby2,jphiby1,nzby3),
     &               soryz1c(nrby2,jphiby1,nzby3),
     &               soryz2c(nrby2,jphiby1,nzby3),
     &               szconc(nrby2,jphiby1,nzby3),
     &               dmloss(nrby2,jphiby1,nzby3),
     &               cfe(nrby2,jphiby1,nzby3),
     &               rdomconc(nrby2,jphiby1,nzby3),
     &               twetbulb(nrby2,jphiby1,nzby3)
c
          write(89,7000)timeday,t(nrby2,jphiby2,nzby1),
     &               w(nrby2,jphiby2,nzby1),
     &               h(nrby2,jphiby2,nzby1),
     &               viability(nrby2,jphiby2,nzby1),
     &               soryz1c(nrby2,jphiby2,nzby1),
     &               soryz2c(nrby2,jphiby2,nzby1),
     &               szconc(nrby2,jphiby2,nzby1),
     &               dmloss(nrby2,jphiby2,nzby1),
     &               cfe(nrby2,jphiby2,nzby1),
```

```
     &                    rdomconc(nrby2,jphiby2,nzby1),
     &                    twetbulb(nrby2,jphiby2,nzby1)
c
          kprint60=1
        end if
      else
        continue
      end if
c
 7000 format(1x,'day',f12.4,'t',f12.4,'m',f12.4,'h',f12.4,
     &          'v',f12.4,'s1',f12.4,'s2',f12.4,'sz',f12.4,
     &          'd',f12.8,'cf',f12.4,'rd',f12.4,'wte',f12.4)
c
      return
      end
c
c###############################################################
c              Output the results in 1/3 period of running time          #
c###############################################################
c
      Subroutine dayqut(time,timeh,timeday,aerationtime,timer,
     &                  dayhour,kount,tamb,hamb,twamb,twout,tin,
     &                  hin)
      Include'dry.inc'
c
      Open (unit=41,file='qin.out',status='unknown')
      Open (unit=42,file='qte.out',status='unknown')
      Open (unit=43,file='qmo.out',status='unknown')
      Open (unit=44,file='qhu.out',status='unknown')
      Open (unit=45,file='qvi.out',status='unknown')
      Open (unit=46,file='qso1.out',status='unknown')
      Open (unit=47,file='qso2.out',status='unknown')
      Open (unit=48,file='qsz.out',status='unknown')
      Open (unit=49,file='qdm.out',status='unknown')
      Open (unit=50,file='qfe.out',status='unknown')
      Open (unit=51,file='qrd.out',status='unknown')
      Open (unit=52,file='qwte.out',status='unknown')
      Open (unit=53,file='qwallt.out',status='unknown')
      Open (unit=54,file='qwallm.out',status='unknown')
      Open (unit=55,file='qwallh.out',status='unknown')
      Open (unit=56,file='qwallvi.out',status='unknown')
      Open (unit=57,file='qwalls1.out',status='unknown')
      Open (unit=58,file='qwalls2.out',status='unknown')
      Open (unit=59,file='qwallsz.out',status='unknown')
      Open (unit=60,file='qwalldm.out',status='unknown')
      Open (unit=61,file='qwallfe.out',status='unknown')
      Open (unit=62,file='qwallrd.out',status='unknown')
      Open (unit=63,file='qwallwte.out',status='unknown')
c
      Call infqut(time,timeh,timeday,aerationtime,timer,
     &            dayhour,kount,tamb,hamb,twamb,twout,tin,
     &            hin)
c
      Call tmqutplot
      Call moqutplot
      Call huqutplot
      Call viqutplot
      Call so1qutplot
      Call so2qutplot
      Call szqutplot
      Call dmqutplot
      Call fequtplot
      Call rdqutplot
      Call wtequtplot
c
      Call walltqutplot
      Call wallmqutplot
      Call wallhqutplot
      Call wallviqutplot
      Call wallso1qutplot
      Call wallso2qutplot
      Call wallszqutplot
      Call walldmqutplot
      Call wallfequtplot
      Call wallrdqutplot
      Call wallwtequtplot
c
      close(41)
      close(42)
      close(43)
      close(44)
      close(45)
```

```
      close(46)
      close(47)
      close(48)
      close(49)
      close(50)
      close(51)
      close(52)
      close(53)
      close(54)
      close(55)
      close(56)
      close(57)
      close(58)
      close(59)
      close(60)
      close(61)
      close(62)
      close(63)
c
      return
      end
c
c#################################################################
c                 Output the results in 2/3 period of running time           #
c#################################################################
c
      Subroutine dayhaf(time,timeh,timeday,aerationtime,timer,
     &                  dayhour,kount,tamb,hamb,twamb,twout,tin,
     &                  hin)
      Include'dry.inc'
c
      Open (unit=64,file='hin.out',status='unknown')
      Open (unit=65,file='hte.out',status='unknown')
      Open (unit=66,file='hmo.out',status='unknown')
      Open (unit=67,file='hhu.out',status='unknown')
      Open (unit=68,file='hvi.out',status='unknown')
      Open (unit=69,file='hso1.out',status='unknown')
      Open (unit=70,file='hso2.out',status='unknown')
      Open (unit=71,file='hsz.out',status='unknown')
      Open (unit=72,file='hdm.out',status='unknown')
      Open (unit=73,file='hfe.out',status='unknown')
      Open (unit=74,file='hrd.out',status='unknown')
      Open (unit=75,file='hwte.out',status='unknown')
      Open (unit=76,file='hwallt.out',status='unknown')
      Open (unit=77,file='hwallm.out',status='unknown')
      Open (unit=78,file='hwallh.out',status='unknown')
      Open (unit=79,file='hwallvi.out',status='unknown')
      Open (unit=80,file='hwalls1.out',status='unknown')
      Open (unit=81,file='hwalls2.out',status='unknown')
      Open (unit=82,file='hwallsz.out',status='unknown')
      Open (unit=83,file='hwalldm.out',status='unknown')
      Open (unit=84,file='hwallfe.out',status='unknown')
      Open (unit=85,file='hwallrd.out',status='unknown')
      Open (unit=86,file='hwallwte.out',status='unknown')
c
      Call infhaf(time,timeh,timeday,aerationtime,timer,
     &            dayhour,kount,tamb,hamb,twamb,twout,tin,
     &            hin)
c
      Call tmhafplot
      Call mohafplot
      Call huhafplot
      Call vihafplot
      Call so1hafplot
      Call so2hafplot
      Call szhafplot
      Call dmhafplot
      Call fehafplot
      Call rdhafplot
      Call wtehafplot
c
      Call wallthafplot
      Call wallmhafplot
      Call wallhhafplot
      Call wallvihafplot
      Call wallso1hafplot
      Call wallso2hafplot
      Call wallszhafplot
      Call walldmhafplot
      Call wallfehafplot
      Call wallrdhafplot
      Call wallwtehafplot
```

```
c
      close(64)
      close(65)
      close(66)
      close(67)
      close(68)
      close(69)
      close(70)
      close(71)
      close(72)
      close(73)
      close(74)
      close(75)
      close(76)
      close(77)
      close(78)
      close(79)
      close(80)
      close(81)
      close(82)
      close(83)
      close(84)
      close(85)
      close(86)
c
      return
      end
c
c###############################################################
c                         Open files                           #
c###############################################################
c
      Subroutine fileopen
      Include'dry.inc'
c
      Open (unit=10,file='templot.out',status='unknown')
      Open (unit=11,file='misplot.out',status='unknown')
      Open (unit=14,file='humiplot.out',status='unknown')
      Open (unit=16,file='crop2.out',status='unknown')
      Open (unit=17,file='crop3.out',status='unknown')
      Open (unit=18,file='crop4.out',status='unknown')
      Open (unit=19,file='crop5.out',status='unknown')
      Open (unit=20,file='viaplot.out',status='unknown')
      Open (unit=21,file='so1plot.out',status='unknown')
      Open (unit=22,file='so2plot.out',status='unknown')
      Open (unit=23,file='szplot.out',status='unknown')
      Open (unit=24,file='dmplot.out',status='unknown')
      Open (unit=25,file='feplot.out',status='unknown')
      Open (unit=26,file='rdmplot.out',status='unknown')
      Open (unit=27,file='wteplot.out',status='unknown')
      Open (unit=30,file='crop8.out',status='unknown')
      Open (unit=31,file='crop9.out',status='unknown')
      Open (unit=32,file='crop10.out',status='unknown')
      Open (unit=33,file='crop11.out',status='unknown')
      Open (unit=34,file='crop12.out',status='unknown')
      Open (unit=35,file='crop13.out',status='unknown')
      Open (unit=36,file='crop14.out',status='unknown')
      Open (unit=37,file='crop15.out',status='unknown')
      Open (unit=38,file='crop16.out',status='unknown')
      Open (unit=39,file='crop17.out',status='unknown')
      Open (unit=40,file='crop18.out',status='unknown')
c
      return
      end
c
c###############################################################
c      Calculate the total amount of pesticide, the total      #
c      of dry matter loss, the total number of insects         #
c      and the total of seeds viability                        #
c###############################################################
c
      Subroutine totals
      Include'dry.inc'
c
      totalrd=0.0
      totalfe=0.0
      totaldml=0.0
      totalviab=0.0
      totalso1=0.0
      totalso2=0.0
      totalsz=0.0
c
```

```
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=j+1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totalrd=totalrd+(rdomconc(1,j,k)+rdomconc(1,j,k+1)+
     &                rdomconc(2,j,k)+rdomconc(2,j,k+1)+rdomconc(2,
     &                jp1,k)+rdomconc(2,jp1,k+1))/6.0*vol(i,j,k)
            else
              totalrd=totalrd+(rdomconc(i,j,k)+rdomconc(i+1,j,k)+
     &                rdomconc(i,j,k+1)+rdomconc(i+1,j,k+1)+
     &                rdomconc(i,jp1,k)+rdomconc(i,jp1,k+1)+
     &                rdomconc(i+1,jp1,k)+rdomconc(i+1,jp1,k+1))/
     &                8.0*vol(i,j,k)
            end if
          end do
        end do
      end do
c~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=j+1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totalfe=totalfe+(cfe(1,j,k)+cfe(1,j,k+1)+cfe(2,
     &                j,k)+cfe(2,j,k+1)+cfe(2,jp1,k)+cfe(2,jp1,
     &                k+1))/6.0*vol(i,j,k)
            else
              totalfe=totalfe+(cfe(i,j,k)+cfe(i+1,j,k)+cfe(i,j,
     &                k+1)+cfe(i+1,j,k+1)+cfe(i,jp1,k)+cfe(i,jp1,
     &                k+1)+cfe(i+1,jp1,k)+cfe(i+1,jp1,k+1))/8.0*
     &                vol(i,j,k)
            end if
          end do
        end do
      end do
c~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=j+1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totaldml=totaldml+(dmloss(1,j,k)+dmloss(1,j,k+1)+
     &                dmloss(2,j,k)+dmloss(2,j,k+1)+dmloss(2,
     &                jp1,k)+dmloss(2,jp1,k+1))/6.0*vol(i,j,k)
            else
              totaldml=totaldml+(dmloss(i,j,k)+dmloss(i+1,j,k)+
     &                dmloss(i,j,k+1)+dmloss(i+1,j,k+1)+
     &                dmloss(i,jp1,k)+dmloss(i,jp1,k+1)+
     &                dmloss(i+1,jp1,k)+dmloss(i+1,jp1,k+1))/
     &                8.0*vol(i,j,k)
            end if
          end do
        end do
      end do
c~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totalviab=totalviab+(viability(1,j,k)+viability(1,
     &                j,k+1)+viability(2,j,k)+viability(2,j,
     &                k+1)+viability(2,jp1,k)+viability(2,jp1,
     &                k+1))/6.0*vol(i,j,k)
            else
              totalviab=totalviab+(viability(i,j,k)+viability(i+1,
     &                j,k)+viability(i,j,k+1)+viability(i+1,j,
     &                k+1)+viability(i,jp1,k)+viability(i,jp1,
     &                k+1)+viability(i+1,jp1,k)+viability(i+1,
     &                jp1,k+1))/8.0*vol(i,j,k)
            end if
          end do
        end do
```

```
      end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totalso1=totalso1+(soryz1c(1,j,k)+soryz1c(1,j,k+1)+
     &                 soryz1c(2,j,k)+soryz1c(2,j,k+1)+soryz1c(2,
     &                 jp1,k)+soryz1c(2,jp1,k+1))/6.0*vol(i,j,k)
            else
              totalso1=totalso1+(soryz1c(i,j,k)+soryz1c(i+1,j,k)+
     &                 soryz1c(i,j,k+1)+soryz1c(i+1,j,k+1)+
     &                 soryz1c(i,jp1,k)+soryz1c(i,jp1,k+1)+
     &                 soryz1c(i+1,jp1,k)+soryz1c(i+1,jp1,k+1))/
     &                 8.0*vol(i,j,k)
            end if
          end do
        end do
      end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totalso2=totalso2+(soryz2c(1,j,k)+soryz2c(1,j,k+1)
     &                 +soryz2c(2,j,k)+soryz2c(2,j,k+1)+
     &                 soryz2c(2,jp1,k)+soryz2c(2,jp1,k+1))/
     &                 6.0*vol(i,j,k)
            else
              totalso2=totalso2+(soryz2c(i,j,k)+soryz2c(i+1,j,k)+
     &                 soryz2c(i,j,k+1)+soryz2c(i+1,j,k+1)+
     &                 soryz2c(i,jp1,k)+soryz2c(i,jp1,k+1)+
     &                 soryz2c(i+1,jp1,k)+soryz2c(i+1,jp1,k+1))/
     &                 8.0*vol(i,j,k)
            end if
          end do
        end do
      end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      do k=1, nzm1
        do j=1, nphi
          do i=1, nrm1
            jp1=1
            if(j.eq.nphi)jp1=1
c
            if(i.eq.1)then
              totalsz=totalsz+(szconc(1,j,k)+szconc(1,j,k+1)
     &                 +szconc(2,j,k)+szconc(2,j,k+1)+
     &                 szconc(2,jp1,k)+szconc(2,jp1,k+1))/
     &                 6.0*vol(i,j,k)
            else
              totalsz=totalsz+(szconc(i,j,k)+szconc(i+1,j,k)+
     &                 szconc(i,j,k+1)+szconc(i+1,j,k+1)+
     &                 szconc(i,jp1,k)+szconc(i,jp1,k+1)+
     &                 szconc(i+1,jp1,k)+szconc(i+1,jp1,k+1))/
     &                 8.0*vol(i,j,k)
            end if
          end do
        end do
      end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      totalrd=totalrd/volume
      totalfe=totalfe/volume
      totaldml=totaldml/volume
      totalviab=totalviab/volume
      totalso1=totalso1/volume
      totalso2=totalso2/volume
      totalsz=totalsz/volume
c
      return
      end
c
c###################################################################
c                  Output pressure fields for plotting             #
c###################################################################
c
```

```
      Subroutine pressplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(9,3950)i,jphi,k,r(i),z(i,k),p(i,jphi,k)
            write(9,3950)i,jphiby2,k,-r(i),z(i,k),p(i,jphiby2,k)
         end do
       end do
c
3950  format(1x,3I3,3f12.4)
c
      return
      end
c
c#################################################################
c                 Output pressure fields by time increas          #
c#################################################################
c
      Subroutine ptime(time)
      Include'dry.inc'
c
      nrby2=nr/2+1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      nzby1=nz/4+1
      nzby2=nz/2+1
      nzby3=(nz*3)/4+1
c
      if (time.eq.1.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.1000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.2000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.4000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.6000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.8000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.10000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.12000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.14000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.16000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
```

```
c
      if (time.eq.18000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.20000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.22000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.24000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.26000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.28000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.30000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.32000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.34000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.36000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.38000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.40000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.42000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.44000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.46000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.48000.0)then
         write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                  p(nrby2,jphiby2,nzby1)
      end if
c
      if (time.eq.50000.0)then
```

```
      write(91,5995)time,p(1,1,nzby2),p(nrby2,jphiby1,nzby3),
     &                p(nrby2,jphiby2,nzby1)
      end if
c
 5995 format(1x,f8.1,3f12.4)
c
      return
      end
c
c##################################################################
c        Output pressure fields along East, West,                 #
c        South and North sides of the silo wall                   #
c##################################################################
c
      Subroutine wallpress
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
c
      do k=1, nz
        do i=1,nr
          write(87,5550)i,k,z(i,k),p(nr,jphi,k),p(nr,jphiby1,k),
     &                  p(nr,jphiby2,k),p(nr,jphiby3,k)
        end do
      end do
c
 5550     format(1x,2I3,5f12.4)
c
      return
      end
c
c##################################################################
c                   Output the performance of grain cooler        #
c##################################################################
c
      Subroutine outputcooler(tamb,hamb,twamb,twout,tin,hin)
c
        write(28,3760)tamb,hamb,twamb
        write(28,3770)tin,hin
        write(28,3780)twout
c
 3760   format(1x,'ambient temperature:',f7.3,'ambient humidity:',
     &         f7.3,'ambient wet-bulb temperature:',f7.3)
 3770   format(1x,'temperature of air leaving cooler:',f7.3,
     &         'humidty of air entering grain:',f7.3)
 3780   format(1x,'wet-bulb temperature of the air leaving cooler:',
     &         f10.4)
c
      return
      end
c
c##################################################################
c        Output fields arising from heat and mass transfer        #
c        calculations,biological activity and decary of           #
c        pesticides                                               #
c##################################################################
c
      Subroutine outputhm(time,timeh)
      Include'dry.inc'
      Include'solar.inc'
c
        write(28,3400)time,timeh,kh
        write(28,3401)totalfe
        write(28,3402)totaldml
        write(28,3403)totalviab
        write(28,3404)totalrd
        write(28,3405)totalso1
        write(28,3406)totalso2
        write(28,3407)totalsz
c
 3400   format(1x,5x,'ELAPSED TIME=',f12.4,2x,'ELAPSED TIME(HOURS)=',
     &         f12.4,2x,'HOUR OF DAY=',I3)
 3401   format(1x,5x,'Mean concentration of fenitrothion=',f10.4)
 3402   format(1x,5x,'Mean dry matter loss=',f10.4)
 3403   format(1x,5x,'Mean viability=',f10.4)
 3404   format(1x,5x,'Mean concentration of R.dominica=',f10.4)
 3405   format(1x,5x,'Mean concentration of S.oryzael=',f10.4)
 3406   format(1x,5x,'Mean concentration of S.oryzae2=',f10.4)
 3407   format(1x,5x,'Mean concentration of S.zeamais=',f10.4)
```

```
c
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(16,3420)
3420  format(1x,75x,'TEMPERATURE FIELD')
c
      write(16,3440)(i,i=1,nr)
3440  format(1x,'PHI\R',21(4x,I2,4x))
c
    do k=nz,1,-1
       write(16,3460)k
3460   format(1x,'k=',I2)
c
     do j=nphi,1,-1
        write(16,3480)j,(t(i,j,k),i=1,nr)
3480    format(1x,I2,21(f9.4))
     end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(16,3500)
3500  format(1x,75x,'GRAIN MOISTURE CONTENT FIELD')
c
    write(16,3440)
    do k=nz,1,-1
       write(16,3460)k
       do j=nphi,1,-1
           write(16,3480)j,(w(i,j,k),i=1,nr)
       end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(17,3520)
3520  format(1x,75x,'HUMIDITY FIELD')
c
    write(17,3440)
    do k=nz,1,-1
       write(17,3460)k
       do j=nphi,1,-1
          write(17,3480)j,(h(i,j,k),i=1,nr)
       end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(17,3540)
3540  format(1x,75x,'DRY MATTER LOSS FIELD')
c
    write(17,3440)
    do k=nz,1,-1
       write(17,3460)k
       do j=nphi,1,-1
          write(17,3481)j,(dmloss(i,j,k),i=1,nr)
3481      format(1x,I2,21(f12.8))
       end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(17,3560)
3560  format(1x,75x,'INTERGRANULAR WET BULB TEMTERATURE')
c
    write(17,3440)
    do k=nz,1,-1
       write(17,3460)k
       do j=nphi,1,-1
          write(17,3480)j,(twetbulb(i,j,k),i=1,nr)
       end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(18,3580)
3580  format(1x,75x,'POPULATION INCREASE OF RHYOPERTHA
  &            DOMINICA')
c
    write(18,3440)
    do k=nz,1,-1
       write(18,3460)k
       do j=nphi,1,-1
          write(18,3480)j,(rdomconc(i,j,k),i=1,nr)
       end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
      write(18,3600)
3600  format(1x,75x,'POLPULATION INCREASE OF SITOPHILUS
  &            ORYZAE1')
c
    write(18,3440)
    do k=nz,1,-1
       write(18,3460)k
```

```
      do j=nphi,1,-1
         write(18,3480)j,(soryz1c(i,j,k),i=1,nr)
      end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
     write(19,3620)
3620 format(1x,75x,'POLPULATION INCREASE OF SITOPHILUS
   &          ORYZAE2')
c
    write(19,3440)
    do k=nz,1,-1
      write(19,3460)k
      do j=nphi,1,-1
         write(19,3480)j,(soryz2c(i,j,k),i=1,nr)
      end do
    end do
c^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
     write(19,3640)
3640 format(1x, 75x,'POLPULATION INCREASE OF SITOPHILUS
   &          ZEAMAIS')
c
    write(19,3440)
    do k=nz,1,-1
      write(19,3460)k
      do j=nphi,1,-1
         write(19,3480)j,(szconc(i,j,k),i=1,nr)
      end do
    end do
c
   return
   end
c
c#################################################################
c                 Output temperature fields for plotting          #
c#################################################################
c
   Subroutine templot
   Include'dry.inc'
c
    jphi=1
    jphiby2=nphi/2+1
    do k=1, nz
      do i=1, nr
         z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
   &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
         write(10,3900)i,jphi,k,r(i),z(i,k),t(i,jphi,k)
         write(10,3900)i,jphiby2,k,-r(i),z(i,k),t(i,jphiby2,k)
      end do
    end do
c
3900    format(1x,3I3,3f12.4)
c
   return
   end
c
c#################################################################
c              Output moisture content fields for plotting        #
c#################################################################
c
   Subroutine moistplot
   Include'dry.inc'
c
    jphi=1
    jphiby2=nphi/2+1
    do k=1, nz
      do i=1, nr
         z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
   &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
         write(11,3910)i,jphi,k,r(i),z(i,k),w(i,jphi,k)
         write(11,3910)i,jphiby2,k,-r(i),z(i,k),w(i,jphiby2,k)
      end do
    end do
c
3910    format(1x,3I3,3f12.4)
c
   return
   end
c
c#################################################################
c                Output radial velcity fields for plotting        #
c#################################################################
c
```

```
      Subroutine velraplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(92,3850)i,jphi,k,r(i),z(i,k),Vra(i,jphi,k)
            write(92,3850)i,jphiby2,k,-r(i),z(i,k),Vra(i,jphiby2,k)
         end do
       end do
c
 3850   format(1x,3I3,2f12.4,f12.6)
c
      return
      end
c
c###############################################################
c                 Output vertical velcity fields for plotting                  #
c###############################################################
c
      Subroutine velverplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(13,3800)i,jphi,k,r(i),z(i,k),Ver(i,jphi,k)
            write(13,3800)i,jphiby2,k,-r(i),z(i,k),Ver(i,jphiby2,k)
         end do
       end do
c
 3800   format(1x,3I3,2f12.4,f12.6)
c
      return
      end
c
c###############################################################
c                 Output circular velcity fields for plotting                  #
c###############################################################
c
      Subroutine velcirplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(12,5885)i,jphi,k,r(i),z(i,k),Vci(i,jphi,k)
            write(12,5885)i,jphiby2,k,-r(i),z(i,k),Vci(i,jphiby2,k)
         end do
       end do
c
 5885   format(1x,3I3,2f12.4,E16.4)
c
      return
      end
c
c###############################################################
c                        Output simulation results                             #
c###############################################################
c
      Subroutine results
      Include'dry.inc'
c
            Call templot
            Call moistplot
            Call humidiplot
            Call viablplot
            Call rdomplot
            Call so1plot
            Call so2plot
            Call szplot
            Call feplot
            Call dmplot
```

```
            Call wteplot
c
            Call walltplot
            Call wallmplot
            Call wallhplot
            Call wallviaplot
            Call wallso1plot
            Call wallso2plot
            Call wallszplot
            Call walldmplot
            Call wallfeplot
            Call wallrdmplot
            Call wallwteplot
c
      return
      end
c
c###############################################################
c                 Output velcity fields along certre line of silo              #
c###############################################################
c
      Subroutine wallvel
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(93,5267)r(i),Vra(i,jphi,nzby2),Vra(i,jphiby1,nzby2),
     &                Vra(i,jphiby2,nzby2),Vra(i,jphiby3,nzby2),
     &                Vci(i,jphi,nzby2),Vci(i,jphiby1,nzby2),
     &                Vci(i,jphiby2,nzby2),Vci(i,jphiby3,nzby2),
     &                Ver(i,jphi,nzby2),Ver(i,jphiby1,nzby2),
     &                Ver(i,jphiby2,nzby2),Ver(i,jphiby3,nzby2)
c
      end do
c
5267  format(1x,f10.6,12(E16.4))
c
        close(6)
        close(7)
        close(9)
        close(12)
        close(13)
        close(15)
        close(87)
        close(90)
        close(91)
        close(92)
        close(93)
        close(94)
c
      return
      end
c
c###############################################################
c                 Output humidity fields for plotting                          #
c###############################################################
c
      Subroutine humidiplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &           (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
            write(14,3750)i,jphi,k,r(i),z(i,k),h(i,jphi,k)
            write(14,3750)i,jphiby2,k,-r(i),z(i,k),h(i,jphiby2,k)
         end do
       end do
c
3750    format(1x,3I3,3f12.4)
c
      return
      end
c
```

```
c################################################################
c                  Output seeds viability fields for plotting              #
c################################################################
c
    Subroutine viablplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
         z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
  &             (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
         write(20,3700)i,jphi,k,r(i),z(i,k),viability(i,jphi,k)
         write(20,3700)i,jphiby2,k,-r(i),z(i,k),
  &              viability(i,jphiby2,k)
       end do
     end do
c
3700    format(1x,3I3,2f12.4,f12.6)
c
    return
    end
c
c################################################################
c                  Output S. oryzae1 fields plotting                 #
c################################################################
c
    Subroutine so1plot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
         z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
  &             (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
         write(21,3650)i,jphi,k,r(i),z(i,k),soryz1c(i,jphi,k)
         write(21,3650)i,jphiby2,k,-r(i),z(i,k),
  &              soryz1c(i,jphiby2,k)
       end do
     end do
c
3650    format(1x,3I3,3f12.4)
c
    return
    end
c
c################################################################
c                  Output S. oryzae2 fields plotting                 #
c################################################################
c
    Subroutine so2plot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
         z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
  &             (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
         write(22,3600)i,jphi,k,r(i),z(i,k),soryz2c(i,jphi,k)
         write(22,3600)i,jphiby2,k,-r(i),z(i,k),
  &              soryz2c(i,jphiby2,k)
       end do
     end do
c
3600    format(1x,3I3,3f12.4)
c
    return
    end
c
c################################################################
c                  Output S. zeamais fields for plotting              #
c################################################################
c
    Subroutine szplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
```

```
        do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &         (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(23,3550)i,jphi,k,r(i),z(i,k),szconc(i,jphi,k)
          write(23,3550)i,jphiby2,k,-r(i),z(i,k),szconc(i,jphiby2,k)
        end do
      end do
c
 3550 format(1x,3I3,3f12.4)
c
      return
      end
c
c###################################################################
c                 Output dry matter loss fields for plotting        #
c###################################################################
c
      Subroutine dmplot
      Include'dry.inc'
c
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &         (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(24,3500)i,jphi,k,r(i),z(i,k),dmloss(i,jphi,k)
          write(24,3500)i,jphiby2,k,-r(i),z(i,k),
     &          dmloss(i,jphiby2,k)
        end do
      end do
 3500 format(1x,3I3,2f12.4,f12.8)
c
      return
      end
c
c###################################################################
c          Output fenitrothion concentration fields for plotting    #
c###################################################################
c
      Subroutine feplot
      Include'dry.inc'
c
      jphi=1
      jphiby2=nphi/2+1
c
      do k=1, nz
        do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &         (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
          write(25,3450)i,jphi,k,r(i),z(i,k),cfe(i,jphi,k)
          write(25,3450)i,jphiby2,k,-r(i),z(i,k),
     &          cfe(i,jphiby2,k)
        end do
      end do
c
 3450 format(1x,3I3,3f12.4)
c
      return
      end
c
c###################################################################
c               Output R. dominica fields for plotting              #
c###################################################################
c
      Subroutine rdomplot
      Include'dry.inc'
c
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &         (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
          write(26,3400)i,jphi,k,r(i),z(i,k),rdomconc(i,jphi,k)
          write(26,3400)i,jphiby2,k,-r(i),z(i,k),rdomconc(i,jphiby2,k)
        end do
      end do
c
 3400 format(1x,3I3,3f12.4)
```

```
c
      return
      end
c
c##############################################################
c          Outputs the intergranular wet bulb                 #
c          temperature field for plotting                     #
c##############################################################
c
      Subroutine wteplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(27,3350)i,jphi,k,r(i),z(i,k),twetbulb(i,jphi,k)
            write(27,3350)i,jphiby2,k,-r(i),z(i,k),twetbulb(i,jphiby2,k)
         end do
       end do
c
3350      format(1x,3I3,3f12.4)
c
      return
      end
c
c##############################################################
c          Output the temperature fields along East, West,    #
c          South and North sides of the silo wall             #
c##############################################################
c
      Subroutine walltplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(30,8000)r(i),t(i,jphi,nzby2),t(i,jphiby1,nzby2),
     &                  t(i,jphiby2,nzby2),t(i,jphiby3,nzby2)
       end do
c
8000   format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c          Output the moisture content fields along East,     #
c          West, South and North sides of the silo wall       #
c##############################################################
c
      Subroutine wallmplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(31,8000)r(i),w(i,jphi,nzby2),w(i,jphiby1,nzby2),
     &                  w(i,jphiby2,nzby2),w(i,jphiby3,nzby2)
       end do
c
8000   format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c          Output the humidity fields along East, West,       #
c          South and North sides of the silo wall             #
c##############################################################
c
      Subroutine wallhplot
```

```
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
           write(32,8000)r(i),h(i,jphi,nzby2),h(i,jphiby1,nzby2),
     &                   h(i,jphiby2,nzby2),h(i,jphiby3,nzby2)
       end do
c
 8000    format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c         Output the seeds viability fields along East,       #
c         West, South and North sides of the silo wall        #
c##############################################################
c
      Subroutine wallviaplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
           write(33,8000)r(i),viability(i,jphi,nzby2),
     &                   viability(i,jphiby1,nzby2),
     &                   viability(i,jphiby2,nzby2),
     &                   viability(i,jphiby3,nzby2)
       end do
c
 8000    format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c         Output the S. oryzae1 fields along East, West,      #
c         South and North sides of the silo wall              #
c##############################################################
c
      Subroutine wallso1plot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
           write(34,8000)r(i),soryz1c(i,jphi,nzby2),
     &                   soryz1c(i,jphiby1,nzby2),
     &                   soryz1c(i,jphiby2,nzby2),
     &                   soryz1c(i,jphiby3,nzby2)
       end do
c
 8000    format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c         Output the S. oryzae2 fields along East, West,      #
c         South and North sides of the silo wall              #
c##############################################################
c
      Subroutine wallso2plot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
```

```
      nzby2=nz/2+1
c
       do i=1,nr
          write(35,8000)r(i),soryz2c(i,jphi,nzby2),
     &                 soryz2c(i,jphiby1,nzby2),
     &                 soryz2c(i,jphiby2,nzby2),
     &                 soryz2c(i,jphiby3,nzby2)
c
       end do
c
 8000   format(1x,5f12.4)
c
      return
      end
c
c##################################################################
c        Output the S. zeamais fields along East, West,           #
c        South and North sides of the silo wall                   #
c##################################################################
c
      Subroutine wallszplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(36,8000)r(i),szconc(i,jphi,nzby2),
     &                 szconc(i,jphiby1,nzby2),
     &                 szconc(i,jphiby2,nzby2),
     &                 szconc(i,jphiby3,nzby2)
       end do
c
 8000   format(1x,5f12.4)
c
      return
      end
c
c##################################################################
c        Output the dry matter loss fields along East,            #
c        West, South and North sides of the silo wall             #
c##################################################################
c
      Subroutine walldmplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(37,8000)r(i),dmloss(i,jphi,nzby2),
     &                 dmloss(i,jphiby1,nzby2),
     &                 dmloss(i,jphiby2,nzby2),
     &                 dmloss(i,jphiby3,nzby2)
       end do
c
 8000   format(1x,f12.4,4f12.8)
c
      return
      end
c
c##################################################################
c        Output the fenitrothion concentration fields along       #
c        East, West, South and North sides of the silo wall       #
c##################################################################
c
      Subroutine wallfeplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
```

```
      write(38,8000)r(i),cfe(i,jphi,nzby2),
   &                    cfe(i,jphiby1,nzby2),
   &                    cfe(i,jphiby2,nzby2),
   &                    cfe(i,jphiby3,nzby2)
    end do
 8000  format(1x,5f12.4)
c
     return
     end
c
c##################################################################
c        Output the R. dominica fields along East, West,          #
c        South and North sides of the silo wall                   #
c##################################################################
c
     Subroutine wallrdmplot
     Include'dry.inc'
c
     jphi=1
     jphiby1=nphi/4+1
     jphiby2=nphi/2+1
     jphiby3=(nphi*3)/4+1
     nzby2=nz/2+1
c
      do i=1,nr
        write(39,8000)r(i),rdomconc(i,jphi,nzby2),
   &                    rdomconc(i,jphiby1,nzby2),
   &                    rdomconc(i,jphiby2,nzby2),
   &                    rdomconc(i,jphiby3,nzby2)
      end do
c
 8000  format(1x,5f12.4)
c
     return
     end
c
c##################################################################
c        Output the intergranular wet bulb temperature            #
c        fields along East, West, South and North                 #
c        sides of the silo wall                                   #
c##################################################################
c
     Subroutine wallwteplot
     Include'dry.inc'
c
     jphi=1
     jphiby1=nphi/4+1
     jphiby2=nphi/2+1
     jphiby3=(nphi*3)/4+1
     nzby2=nz/2+1
c
      do i=1,nr
        write(40,8000)r(i),twetbulb(i,jphi,nzby2),
   &                    twetbulb(i,jphiby1,nzby2),
   &                    twetbulb(i,jphiby2,nzby2),
   &                    twetbulb(i,jphiby3,nzby2)
      end do
c
 8000  format(1x,5f12.4)
c
     return
     end
c
c##################################################################
c        Output the performance of grain silo at 1/3              #
c        period of running time                                   #
c##################################################################
c
     Subroutine infqut(time,timeh,timeday,aerationtime,timer,
   &           dayhour,kount,tamb,hamb,twamb,twout,tin,
   &           hin)
     Include'dry.inc'
c
       write(41,3900)
       write(41,3950)time,timeh,timeday
       write(41,3955)aerationtime
       write(41,3960)timer
       write(41,3965)dayhour
       write(41,3970)kount
c
 3900  format(1x,'In 20 days')
 3950  format(1x,5x,'time(SECOND)',f12.4,2x,'timeh(HOURS)',f12.4,2x,
```

```
  &         'TIME(DAY)',f12.4)
 3955  format(1x,5x,'aerationtime',f10.3,'HOURS')
 3960  format(1x,5x,'timer',f12.4)
 3965  format(1x,5x,'dayhour',f8.4)
 3970  format(1x,5x,'kount',I5)
c
       write(41,3760)tamb,hamb,twamb
       write(41,3770)tin,hin
       write(41,3780)twout
c
 3760  format(1x,'ambient temperature:',f7.3,'ambient humidity:',
  &         f7.3,'ambient wet-bulb temperature:',f7.3)
 3770  format(1x,'temperature of air leaving cooler:',f7.3,
  &         'humidty of air entering grain:',f7.3)
 3780  format(1x,'wet-bulb temperature of the air leaving cooler:',
  &         f10.4)
c
    return
    end
c
c################################################################
c          Output temperature fields at 1/3 period of running time       #
c################################################################
c
    Subroutine tmqutplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
  &              (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(42,3900)i,jphi,k,r(i),z(i,k),t(i,jphi,k)
          write(42,3900)i,jphiby2,k,-r(i),z(i,k),t(i,jphiby2,k)
       end do
     end do
c
 3900     format(1x,3I3,3f12.4)
c
    return
    end
c
c################################################################
c          Output moisture content fields at 1/3                         #
c          period of running time                                        #
c################################################################
c
    Subroutine moqutplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
  &              (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(43,3910)i,jphi,k,r(i),z(i,k),w(i,jphi,k)
          write(43,3910)i,jphiby2,k,-r(i),z(i,k),w(i,jphiby2,k)
       end do
     end do
c
 3910     format(1x,3I3,3f12.4)
c
    return
    end
c
c################################################################
c          Output humidity fields at 1/3                                 #
c          period of running time                                        #
c################################################################
c
    Subroutine huqutplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
  &              (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(44,3750)i,jphi,k,r(i),z(i,k),h(i,jphi,k)
```

```
            write(44,3750)i,jphiby2,k,-r(i),z(i,k),h(i,jphiby2,k)
          end do
        end do
c
 3750      format(1x,3I3,3f12.4)
c
      return
      end
c
c##################################################################
c            Output seeds viability fields at                     #
c            1/3 period of running time                           #
c##################################################################
c
      Subroutine viqutplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
           z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &           (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
           write(45,3700)i,jphi,k,r(i),z(i,k),viability(i,jphi,k)
           write(45,3700)i,jphiby2,k,-r(i),z(i,k),
     &            viability(i,jphiby2,k)
         end do
       end do
c
 3700      format(1x,3I3,2f12.4,f12.6)
c
      return
      end
c
c##################################################################
c            Output S. oryzae1 fields at 1/3 period of running time   #
c##################################################################
c
      Subroutine so1qutplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
           z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &           (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
           write(46,3650)i,jphi,k,r(i),z(i,k),soryz1c(i,jphi,k)
           write(46,3650)i,jphiby2,k,-r(i),z(i,k),
     &            soryz1c(i,jphiby2,k)
         end do
       end do
c
 3650      format(1x,3I3,3f12.4)
c
      return
      end
c
c##################################################################
c            Output S. oryzae2 fields at 1/3 period of running time   #
c##################################################################
c
      Subroutine so2qutplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
           z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &           (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
           write(47,3600)i,jphi,k,r(i),z(i,k),soryz2c(i,jphi,k)
           write(47,3600)i,jphiby2,k,-r(i),z(i,k),
     &            soryz2c(i,jphiby2,k)
         end do
       end do
c
 3600      format(1x,3I3,3f12.4)
c
      return
      end
c
```

```
c##################################################################
c           Output S. zeamais fields at 1/3 period of running time        #
c##################################################################
c
    Subroutine szqutplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
   &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(48,3550)i,jphi,k,r(i),z(i,k),szconc(i,jphi,k)
          write(48,3550)i,jphiby2,k,-r(i),z(i,k),szconc(i,jphiby2,k)
       end do
     end do
c
 3550      format(1x,3I3,3f12.4)
c
    return
    end
c
c##################################################################
c           Output dry matter loss fields at                              #
c           1/3 period of running time                                    #
c##################################################################
c
    Subroutine dmqutplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
   &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
          write(49,3500)i,jphi,k,r(i),z(i,k),dmloss(i,jphi,k)
          write(49,3500)i,jphiby2,k,-r(i),z(i,k),
   &                    dmloss(i,jphiby2,k)
       end do
     end do
c
 3500      format(1x,3I3,2f12.4,f12.8)
c
    return
    end
c
c##################################################################
c           Output fenitrothion concentration fields                      #
c           at 1/3 period of running time                                 #
c##################################################################
c
    Subroutine fequtplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
   &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
          write(50,3450)i,jphi,k,r(i),z(i,k),cfe(i,jphi,k)
          write(50,3450)i,jphiby2,k,-r(i),z(i,k),
   &                cfe(i,jphiby2,k)
       end do
     end do
c
 3450      format(1x,3I3,3f12.4)
c
    return
    end
c
c##################################################################
c           Output R. dominica fields at 1/3 period of running time       #
c##################################################################
c
    Subroutine rdqutplot
    Include'dry.inc'
c
```

```
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
          write(51,3400)i,jphi,k,r(i),z(i,k),rdomconc(i,jphi,k)
          write(51,3400)i,jphiby2,k,-r(i),z(i,k),rdomconc(i,jphiby2,k)
        end do
      end do
c
 3400     format(1x,3I3,3f12.4)
c
      return
      end
c
c#################################################################
c       Output the intergranular wet bulb temperature              #
c       fields at 1/3 period of running time                       #
c#################################################################
c
      Subroutine wtequtplot
      Include'dry.inc'
c
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(52,3350)i,jphi,k,r(i),z(i,k),twetbulb(i,jphi,k)
          write(52,3350)i,jphiby2,k,-r(i),z(i,k),twetbulb(i,jphiby2,k)
        end do
      end do
c
 3350     format(1x,3I3,3f12.4)
c
      return
      end
c
c#################################################################
c       Output the temperature fields at 1/3 period                #
c       of running time in East, West, South and                   #
c       North sides of the silo wall                               #
c#################################################################
c
      Subroutine walltqutplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(53,8000)r(i),t(i,jphi,nzby2),t(i,jphiby1,nzby2),
     &                 t(i,jphiby2,nzby2),t(i,jphiby3,nzby2)
      end do
c
 8000   format(1x,5f12.4)
c
      return
      end
c
c#################################################################
c       Output the moisture content fields at 1/3 period           #
c       of running time in East, West, South and                   #
c       North sides of the silo wall                               #
c#################################################################
c
      Subroutine wallmqutplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
```

```
      write(54,8000)r(i),w(i,jphi,nzby2),w(i,jphiby1,nzby2),
   &                  w(i,jphiby2,nzby2),w(i,jphiby3,nzby2)
     end do
c
8000  format(1x,5f12.4)
c
     return
     end
c
c#################################################################
c       Output the humidity fields at 1/3 period of             #
c       running time in East, West, South and                   #
c       North sides of the silo wall                            #
c#################################################################
c
     Subroutine wallhqutplot
     Include'dry.inc'
c
     jphi=1
     jphiby1=nphi/4+1
     jphiby2=nphi/2+1
     jphiby3=(nphi*3)/4+1
     nzby2=nz/2+1
c
      do i=1,nr
         write(55,8000)r(i),h(i,jphi,nzby2),h(i,jphiby1,nzby2),
   &                  h(i,jphiby2,nzby2),h(i,jphiby3,nzby2)
      end do
c
8000  format(1x,5f12.4)
c
     return
     end
c
c#################################################################
c       Output the seeds viability fields at 1/3 period         #
c       of running time in East, West, South and                #
c       North sides of the silo wall                            #
c#################################################################
c
     Subroutine wallviqutplot
     Include'dry.inc'
c
     jphi=1
     jphiby1=nphi/4+1
     jphiby2=nphi/2+1
     jphiby3=(nphi*3)/4+1
     nzby2=nz/2+1
c
      do i=1,nr
         write(56,8000)r(i),viability(i,jphi,nzby2),
   &                  viability(i,jphiby1,nzby2),
   &                  viability(i,jphiby2,nzby2),
   &                  viability(i,jphiby3,nzby2)
c
      end do
c
8000  format(1x,5f12.4)
c
     return
     end
c
c#################################################################
c       Output the S. oryzae1 fields at 1/3 period              #
c       of running time in East, West, South and                #
c       North sides of the silo wall                            #
c#################################################################
c
     Subroutine wallso1qutplot
     Include'dry.inc'
c
     jphi=1
     jphiby1=nphi/4+1
     jphiby2=nphi/2+1
     jphiby3=(nphi*3)/4+1
     nzby2=nz/2+1
c
      do i=1,nr
         write(57,8000)r(i),soryz1c(i,jphi,nzby2),
   &                  soryz1c(i,jphiby1,nzby2),
   &                  soryz1c(i,jphiby2,nzby2),
   &                  soryz1c(i,jphiby3,nzby2)
```

```
      end do
c
 8000    format(1x,5f12.4)
c
      return
      end
c
c##################################################################
c         Output the S. oryzae2 fields at 1/3 period              #
c         of running time in East, West, South and                #
c         North sides of the silo wall                            #
c##################################################################
c
      Subroutine wallso2qutplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(58,8000)r(i),soryz2c(i,jphi,nzby2),
     &                  soryz2c(i,jphiby1,nzby2),
     &                  soryz2c(i,jphiby2,nzby2),
     &                  soryz2c(i,jphiby3,nzby2)
       end do
c
 8000    format(1x,5f12.4)
c
      return
      end
c
c##################################################################
c         Output the S. zeamais fields at 1/3 period              #
c         of running time in East, West, South and                #
c         North sides of the silo wall                            #
c##################################################################
c
      Subroutine wallszqutplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(59,8000)r(i),szconc(i,jphi,nzby2),
     &                  szconc(i,jphiby1,nzby2),
     &                  szconc(i,jphiby2,nzby2),
     &                  szconc(i,jphiby3,nzby2)
       end do
c
 8000    format(1x,5f12.4)
c
      return
      end
c
c##################################################################
c         Output the dry matter loss fields at 1/3 period         #
c         of running time in East, West, South and                #
c         North sides of the silo wall                            #
c##################################################################
c
      Subroutine walldmqutplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
          write(60,8000)r(i),dmloss(i,jphi,nzby2),
     &                  dmloss(i,jphiby1,nzby2),
     &                  dmloss(i,jphiby2,nzby2),
     &                  dmloss(i,jphiby3,nzby2)
c
```

```
      end do
c
8000  format(1x,f12.4,4f12.8)
c
      return
      end
c
c#################################################################
c        Output the fenitrithion concentration fields at         #
c        1/3 period of running time in East, West, South         #
c        and North sides of the silo wall                        #
c#################################################################
c
      Subroutine wallfequtplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(61,8000)r(i),cfe(i,jphi,nzby2),
     &                     cfe(i,jphiby1,nzby2),
     &                     cfe(i,jphiby2,nzby2),
     &                     cfe(i,jphiby3,nzby2)
      end do
c
8000  format(1x,5f12.4)
c
      return
      end
c
c#################################################################
c        Output the R. dominica fields at 1/3 period             #
c        of running time in East, West, South and                #
c        North sides of the silo wall                            #
c#################################################################
c
      Subroutine wallrdqutplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(62,8000)r(i),rdomconc(i,jphi,nzby2),
     &                     rdomconc(i,jphiby1,nzby2),
     &                     rdomconc(i,jphiby2,nzby2),
     &                     rdomconc(i,jphiby3,nzby2)
      end do
c
8000  format(1x,5f12.4)
c
      return
      end
c
c#################################################################
c        Output the intergranular wet bulb temperature           #
c        fields at 1/3 period of running time in East,           #
c        West, South and North sides of the silo wall            #
c#################################################################
c
      Subroutine wallwtequtplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(63,8000)r(i),twetbulb(i,jphi,nzby2),
     &                     twetbulb(i,jphiby1,nzby2),
     &                     twetbulb(i,jphiby2,nzby2),
     &                     twetbulb(i,jphiby3,nzby2)
c
```

```
      end do
c
 8000     format(1x,5f12.4)
c
      return
      end
c
c#################################################################
c          Output the performance of grain silo at 2/3                 #
c          period of running time                                      #
c#################################################################
c
      Subroutine infhaf(time,timeh,timeday,aerationtime,timer,
     &            dayhour,kount,tamb,hamb,twamb,twout,tin,
     &            hin)
      Include'dry.inc'
c
        write(64,3900)
        write(64,3950)time,timeh,timeday
        write(64,3955)aerationtime
        write(64,3960)timer
        write(64,3965)dayhour
        write(64,3970)kount
c
 3900   format(1x,'In 40 days')
 3950   format(1x,5x,'time(SECOND)',f12.4,2x,'timeh(HOURS)',f12.4,2x,
     &          'TIME(DAY)',f12.4)
 3955   format(1x,5x,'aerationtime',f10.3,'HOURS')
 3960   format(1x,5x,'timer',f12.4)
 3965   format(1x,5x,'dayhour',f8.4)
 3970   format(1x,5x,'kount',I5)
c
        write(64,3760)tamb,hamb,twamb
        write(64,3770)tin,hin
        write(64,3780)twout
c
 3760   format(1x,'ambient temperature:',f7.3,'ambient humidity:',
     &          f7.3,'ambient wet-bulb temperature:',f7.3)
 3770   format(1x,'temperature of air leaving cooler:',f7.3,
     &          'humidty of air entering grain:',f7.3)
 3780   format(1x,'wet-bulb temperature of the air leaving cooler:',
     &             f10.4)
c
      return
      end
c
c#################################################################
c          Output temperature fields at 2/3 period of running time     #
c#################################################################
c
      Subroutine tmhafplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
           z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
           write(65,3900)i,jphi,k,r(i),z(i,k),t(i,jphi,k)
           write(65,3900)i,jphiby2,k,-r(i),z(i,k),t(i,jphiby2,k)
         end do
       end do
c
 3900     format(1x,3I3,3f12.4)
c
      return
      end
c
c#################################################################
c          Output moisture content fields at 2/3                       #
c          period of running time                                      #
c#################################################################
c
      Subroutine mohafplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
           z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
```

```
     &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(66,3910)i,jphi,k,r(i),z(i,k),w(i,jphi,k)
          write(66,3910)i,jphiby2,k,-r(i),z(i,k),w(i,jphiby2,k)
        end do
      end do
c
 3910     format(1x,3I3,3f12.4)
c
    return
    end
c
c##################################################################
c           Output humidity fields at 2/3                         #
c           period of running time                                #
c##################################################################
c
    Subroutine huhafplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(67,3750)i,jphi,k,r(i),z(i,k),h(i,jphi,k)
          write(67,3750)i,jphiby2,k,-r(i),z(i,k),h(i,jphiby2,k)
       end do
     end do
c
 3750     format(1x,3I3,3f12.4)
c
    return
    end
c
c##################################################################
c           Output seeds viability fields at                      #
c           2/3 period of running time                            #
c##################################################################
c
    Subroutine vihafplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(68,3700)i,jphi,k,r(i),z(i,k),viability(i,jphi,k)
          write(68,3700)i,jphiby2,k,-r(i),z(i,k),
     &                    viability(i,jphiby2,k)
       end do
     end do
c
 3700     format(1x,3I3,2f12.4,f12.6)
c
    return
    end
c
c##################################################################
c           Output S. oryzae1 fields at 2/3 period of running time      #
c##################################################################
c
    Subroutine so1hafplot
    Include'dry.inc'
c
     jphi=1
     jphiby2=nphi/2+1
     do k=1, nz
       do i=1, nr
          z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &          (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
          write(69,3650)i,jphi,k,r(i),z(i,k),soryz1c(i,jphi,k)
          write(69,3650)i,jphiby2,k,-r(i),z(i,k),
     &                    soryz1c(i,jphiby2,k)
       end do
     end do
c
 3650     format(1x,3I3,3f12.4)
c
    return
```

```
      end
c
c#################################################################
c          Output S. oryzae2 fields at 2/3 period of running time          #
c#################################################################
c
      Subroutine so2hafplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
             z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &             (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
             write(70,3600)i,jphi,k,r(i),z(i,k),soryz2c(i,jphi,k)
             write(70,3600)i,jphiby2,k,-r(i),z(i,k),
     &                     soryz2c(i,jphiby2,k)
         end do
       end do
c
 3600   format(1x,3I3,3f12.4)
c
      return
      end
c
c#################################################################
c             Output S. zeamais fields at 2/3 period of running time        #
c#################################################################
c
      Subroutine szhafplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
             z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &             (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
             write(71,3550)i,jphi,k,r(i),z(i,k),szconc(i,jphi,k)
             write(71,3550)i,jphiby2,k,-r(i),z(i,k),szconc(i,jphiby2,k)
         end do
       end do
c
 3550   format(1x,3I3,3f12.4)
c
      return
      end
c
c#################################################################
c          Output dry matter loss fields at                                  #
c          2/3 period of running time                                        #
c#################################################################
c
      Subroutine dmhafplot
      Include'dry.inc'
c
       jphi=1
       jphiby2=nphi/2+1
       do k=1, nz
         do i=1, nr
             z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &             (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
             write(72,3500)i,jphi,k,r(i),z(i,k),dmloss(i,jphi,k)
             write(72,3500)i,jphiby2,k,-r(i),z(i,k),
     &                     dmloss(i,jphiby2,k)
         end do
       end do
c
 3500   format(1x,3I3,2f12.4,f12.8)
c
      return
      end
c
c#################################################################
c          Output fenitrothion concentration fields                          #
c          at 2/3 period of running time                                     #
c#################################################################
c
      Subroutine fehafplot
      Include'dry.inc'
c
```

```
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
c
            write(73,3450)i,jphi,k,r(i),z(i,k),cfe(i,jphi,k)
            write(73,3450)i,jphiby2,k,-r(i),z(i,k),
     &                    cfe(i,jphiby2,k)
        end do
      end do
c
 3450 format(1x,3I3,3f12.4)
c
      return
      end
c
c##################################################################
c         Output R. dominica fields at 2/3 period of running time  #
c##################################################################
c
      Subroutine rdhafplot
      Include'dry.inc'
c
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(74,3400)i,jphi,k,r(i),z(i,k),rdomconc(i,jphi,k)
            write(74,3400)i,jphiby2,k,-r(i),z(i,k),rdomconc(i,jphiby2,k)
        end do
      end do
c
 3400 format(1x,3I3,3f12.4)
c
      return
      end
c
c##################################################################
c         Output the intergranular wet bulb temperature            #
c         fields at 2/3 period of running time                     #
c##################################################################
c
      Subroutine wtehafplot
      Include'dry.inc'
c
      jphi=1
      jphiby2=nphi/2+1
      do k=1, nz
        do i=1, nr
            z(i,k)=r(i)*tan(gamma1)+real(k-1)/real(nz-1)*
     &            (real(wz)-r(i)*(tan(gamma1)+tan(gamma2)))
            write(75,3350)i,jphi,k,r(i),z(i,k),twetbulb(i,jphi,k)
            write(75,3350)i,jphiby2,k,-r(i),z(i,k),twetbulb(i,jphiby2,k)
        end do
      end do
c
 3350 format(1x,3I3,3f12.4)
c
      return
      end
c
c##################################################################
c         Output the temperature fields at 2/3 period              #
c         of running time in East, West, South and                 #
c         North sides of the silo wall                             #
c##################################################################
c
      Subroutine wallthafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
        write(76,8000)r(i),t(i,jphi,nzby2),t(i,jphiby1,nzby2),
```

```
&                   t(i,jphiby2,nzby2),t(i,jphiby3,nzby2)
   end do
c
8000    format(1x,5f12.4)
c
      return
      end
c
c###################################################################
c        Output the moisture content fields at 2/3 period           #
c        of running time in East, West, South and                   #
c        North sides of the silo wall                               #
c###################################################################
c
      Subroutine wallmhafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(77,8000)r(i),w(i,jphi,nzby2),w(i,jphiby1,nzby2),
     &                  w(i,jphiby2,nzby2),w(i,jphiby3,nzby2)
      end do
c
8000    format(1x,5f12.4)
c
      return
      end
c
c###################################################################
c        Output the humidity fields at 2/3 period of                #
c        running time in East, West, South and                      #
c        North sides of the silo wall                               #
c###################################################################
c
      Subroutine wallhhafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(78,8000)r(i),h(i,jphi,nzby2),h(i,jphiby1,nzby2),
     &                  h(i,jphiby2,nzby2),h(i,jphiby3,nzby2)
      end do
c
8000    format(1x,5f12.4)
c
      return
      end
c
c###################################################################
c        Output the seeds viability fields at 2/3 period            #
c        of running time in East, West, South and                   #
c        North sides of the silo wall                               #
c###################################################################
c
      Subroutine wallvihafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(79,8000)r(i),viability(i,jphi,nzby2),
     &                  viability(i,jphiby1,nzby2),
     &                  viability(i,jphiby2,nzby2),
     &                  viability(i,jphiby3,nzby2)
      end do
c
8000    format(1x,5f12.4)
c
```

```
      return
      end
c
c##############################################################
c       Output the S. oryzae1 fields at 2/3 period            #
c       of running time in East, West, South and              #
c       North sides of the silo wall                          #
c##############################################################
c
      Subroutine wallso1hafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(80,8000)r(i),soryz1c(i,jphi,nzby2),
     &                 soryz1c(i,jphiby1,nzby2),
     &                 soryz1c(i,jphiby2,nzby2),
     &                 soryz1c(i,jphiby3,nzby2)
      end do
c
 8000 format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c       Output the S. oryzae2 fields at 2/3 period            #
c       of running time in East, West, South and              #
c       North sides of the silo wall                          #
c##############################################################
c
      Subroutine wallso2hafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(81,8000)r(i),soryz2c(i,jphi,nzby2),
     &                 soryz2c(i,jphiby1,nzby2),
     &                 soryz2c(i,jphiby2,nzby2),
     &                 soryz2c(i,jphiby3,nzby2)
      end do
 8000 format(1x,5f12.4)
c
      return
      end
c
c##############################################################
c       Output the S. zeamais fields at 2/3 period            #
c       of running time in East, West, South and              #
c       North sides of the silo wall                          #
c##############################################################
c
      Subroutine wallszhafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
      do i=1,nr
         write(82,8000)r(i),szconc(i,jphi,nzby2),
     &                 szconc(i,jphiby1,nzby2),
     &                 szconc(i,jphiby2,nzby2),
     &                 szconc(i,jphiby3,nzby2)
      end do
c
 8000 format(1x,5f12.4)
c
      return
      end
```

```
c
c################################################################
c        Output the dry matter loss fields at 2/3 period        #
c        of running time in East, West, South and               #
c        North sides of the silo wall                           #
c################################################################
c
      Subroutine walldmhafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
         write(83,8000)r(i),dmloss(i,jphi,nzby2),
     &                  dmloss(i,jphiby1,nzby2),
     &                  dmloss(i,jphiby2,nzby2),
     &                  dmloss(i,jphiby3,nzby2)
       end do
c
 8000  format(1x,f12.4,4f12.8)
c
      return
      end
c
c################################################################
c        Output the fenitrithion concentration fields at        #
c        2/3 period of running time in East, West, South        #
c        and North sides of the silo wall                       #
c################################################################
c
      Subroutine wallfehafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
         write(84,8000)r(i),cfe(i,jphi,nzby2),
     &                  cfe(i,jphiby1,nzby2),
     &                  cfe(i,jphiby2,nzby2),
     &                  cfe(i,jphiby3,nzby2)
       end do
c
 8000  format(1x,5f12.4)
c
      return
      end
c
c################################################################
c        Output the R. dominica fields at 2/3 period            #
c        of running time in East, West, South and               #
c        North sides of the silo wall                           #
c################################################################
c
      Subroutine wallrdhafplot
      Include'dry.inc'
c
      jphi=1
      jphiby1=nphi/4+1
      jphiby2=nphi/2+1
      jphiby3=(nphi*3)/4+1
      nzby2=nz/2+1
c
       do i=1,nr
         write(85,8000)r(i),rdomconc(i,jphi,nzby2),
     &                  rdomconc(i,jphiby1,nzby2),
     &                  rdomconc(i,jphiby2,nzby2),
     &                  rdomconc(i,jphiby3,nzby2)
       end do
c
 8000  format(1x,5f12.4)
c
      return
      end
c
```

```
c##################################################################
c        Output the intergranular wet bulb temperature           #
c        fields at 2/3 period of running time in East,           #
c        West, South and North sides of the silo wall            #
c##################################################################
c
     Subroutine wallwtehafplot
     Include'dry.inc'
c
     jphi=1
     jphiby1=nphi/4+1
     jphiby2=nphi/2+1
     jphiby3=(nphi*3)/4+1
     nzby2=nz/2+1
c
     do i=1,nr
        write(86,8000)r(i),twetbulb(i,jphi,nzby2),
    &                   twetbulb(i,jphiby1,nzby2),
    &                   twetbulb(i,jphiby2,nzby2),
    &                   twetbulb(i,jphiby3,nzby2)
     end do
c
8000 format(1x,5f12.4)
c
     return
     end
c
c##################END OF PROGRAM##############################
```

```
c############################DRY.INC##############################
c
    parameter (imax=31, jmax=31, kmax=41, ikmax=imax+kmax)
c
    common/heat/ nr,nphi,nz,wR,wz,wz1,nrm1,nrm2,nzm1,nzm2,dphi,pi,
   &            gamma1,gamma2,crb4,crb5,crb6,czb1,czb2,czb3,czb4,
   &            czb5,czb6,gam1,gam2,nphimindc,nphimaxdc,nrmindc,
   &            nrmaxdc,kprint5,kprint10,kprint15,kprint20,
   &            kprint25,kprint30,kprint35,kprint40,kprint45,
   &            kprint50,kprint55,kprint60,kprint0
c
    common/prop/ cw1,cw2,cw3,cw4,cw5,hw1,hw2,hw3,hw4,hw5,hw6,hw7,
   &            hw8,hw9,hw10,p0,w0,h0,revm,wsat,eps,rhoa,rhos,rhob,
   &            cg,ca,cw,keff,deff,dhvdt,hv,gn,patm,rhout,hs,hvap,
   &            a1lhm,a2lhm,a3lhm,a4lhm,a5lhm,a6lhm,a7lhm,a8lhm,
   &            crdom,csoryz1,csoryz2,csz,trdom,tsoryz1,tsoryz2,
   &            tsz,xi,dwdt,tprint5,tprint10,tprint15,tprint20,
   &            tprint25,tprint30,tprint35,tprint40,tprint45,
   &            tprint50,tprint55,tprint60,tprint0
c
    common/soil/ dtg,dtp,pduct,trise,qu,res,tinit,winit,ku1,ku2,
   &            kp1,kp2,tmean,tamp,humidity,detal,qup,specflo,bfe,
   &            thalfe,maxday,aerate1,aerate2,dtgaerate,ifreq,
   &            dtgconduct,noflow,volume,totalrd,totalfe,totaldml,
   &            totalviab,totalso1,totalso2,totalsz,ddmdt
c
    common/room/ hr(imax),hz(kmax),r(imax),z(imax,kmax),cr1(imax),
   &            cr2(imax),cr3(imax),cr4(imax),cr5(imax),cr6(imax),
   &            cz1(kmax),cz2(kmax),cz3(kmax),cz4(kmax),cz5(kmax),
   &            cz6(kmax),cp1(jmax),cp2(jmax),cp3(jmax),cp4(jmax),
   &            cp5(jmax),cp6(jmax),heightc(kmax),heightr(kmax),
   &            slope(kmax)
c
    common/seed/ alpha1(imax,kmax),alpha2(imax,kmax),dpdn(imax,jmax),
   &            alpha3(imax,kmax),alpha4(imax,kmax),air(imax,jmax),
   &            beta1(imax,kmax),beta2(imax,kmax),beta3(imax,kmax),
   &            beta4(imax,kmax),beta5(imax,kmax),beta6(imax,kmax),
   &            beta7(imax,kmax),beta8(imax),dpde(imax,jmax)
c
    common/sign/ rdomconc(imax,jmax,kmax),soryz1c(imax,jmax,kmax),
   &            soryz2c(imax,jmax,kmax),szconc(imax,jmax,kmax),
   &            cfe(imax,jmax,kmax),cpes(imax,jmax,kmax)
c
    common/desk/ p(imax,jmax,kmax),pnew(imax,jmax,kmax),
   &            Vra(imax,jmax,kmax),Vci(imax,jmax,kmax),
   &            Ver(imax,jmax,kmax),t(imax,jmax,kmax),
   &            dmloss(imax,jmax,kmax),h(imax,jmax,kmax),
   &            tnew(imax,jmax,kmax),teq(imax,jmax,kmax)
c
    common/tree/ wnew(imax,jmax,kmax),w(imax,jmax,kmax),
   &            twetbulb(imax,jmax,kmax),omega1(imax,jmax,kmax),
   &            omega2(imax,jmax,kmax),omega3(imax,jmax,kmax)
c
    common/duct/ vol(imax,jmax,kmax),vrsave(imax,jmax,kmax),
   &            vpsave(imax,jmax,kmax),vzsave(imax,jmax,kmax),
   &            viability(imax,jmax,kmax),savep(imax,jmax,kmax)
c
    common/corn/ qq(ikmax),a(ikmax),b(ikmax),c(ikmax),d(ikmax),
   &            delta(ikmax),beta(ikmax)
c
    common/wood/ pnotfld,mcb,smith
c
    logical  pnotfld,mcb,smith
c
c###################END OF INCLUDE FILE########################

c#########################SOLAR.INC##############################
c
    common/tree/ chi,gamma,rltime,phi,alpha,epsilon
c
    common/wood/ nuda(6),suns(6),tmin(6),tmax(6)
c
    common/desk/ ts(2,30,24),ll,mm,kh

    common/iter/ psi(80000)
c
c###################END OF INCLUDE FILE##########################
```

```
c########################CROP.IN################################
c    STATEMENTS                              INPUT DATA
c###############################################################
c
 nr,nphi,nz                                 21,20,21
 ku1,ku2,maxday                             1,200000,60
 kp1,kp2                                    1,2
 dtp,dtgaerate,dtgconduct                   0.5,90.0,900.0
 wR,wz1,specflo,res                         1.115,2.7,1.5708,2000.0
 gam1,gam2                                  38.67,10.17
 tinit,winit                                30.0,0.12
 tmeam,tamp,humidity                        20.0,5.0,0.008
 nphimindc,nphimaxdc,nrmindc,nrmaxdc        1,1,2,15
 aerate1,aerate2                            0.0,6.0
 mcb                                        .true.
 pnotfld                                    .false.
 smith                                      .true.
c~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 phi                                        -37.67
 daily sunshine hours                       8.1,7.5,6.2,4.9,3.8,3.1
 day of the year                            17,47,75,105,135,162
 daily minimum temperature                  14.2,15.1,12.1,10.7,7.6,5.1
 daily maximum temperature                  26.3,26.4,23.2,20.7,16.5,13.4
 alpha,epsilon                              0.8,0.9
c
c###################END OF INPUT FILE###########################
```