EVALUATION OF GAS SPRING HYSTERESIS LOSSES IN STIRLING CRYOCOOLERS

by

ALBERT CHIN-MIN WANG

B.S.M.E., Massachusetts Institute of Technology
(1988)

Submitted to the Department of Mechanical
Engineering in partial fulfillment of the
Requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 1989

© Massachusetts Institute of Technology 1989
All rights reserved

Signature of Author_____
Department of Mechanical Engineering
September 14, 1989

Certified by_____
Joseph L. Smith, Jr.
Thesis Supervisor

Accepted by_____
Ain A. Sonin
Chairman, Mechanical Engineering Department Committee

ARCHIVES
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
AUG 13 1990
LIBRARIES

EVALUATION OF GAS SPRING HYSTERESIS LOSSES IN STIRLING CRYOCOOLERS

by

Albert Chin-min Wang

Submitted to the Department of Mechanical Engineering
on September 14, 1989 in partial fulfillment of the
requirements for the Degree of Master of Science
in Mechanical Engineering

Abstract

This thesis presents a numerical analysis of gas spring hysteresis losses in Stirling cryocoolers for a wide range of operating conditions which are characterized by an oscillating flow Peclet number. Originally from a model that assumes adiabatic working spaces and decouples heat exchange component irreversibilities from a basic cycle, the governing equations are extended to include gas-to-wall heat transfer in the piston cylinders. By basing the heat transfer upon a complex Nusselt number, the new model is able to predict the phase shift between heat transfer and gas-to-wall temperature difference. For low values of Peclet, a theoretical expression is used to determine the complex Nusselt number; for high Peclet, an empirical correlation is used. The greatest losses can occur for intermediate values of the Peclet number. The attribution of losses to several causes is explored by calculating entropy generation for selected control volumes. Between extreme situations, performance may vary by as much as a factor of two. Therefore, when designing a Stirling cryocooler, one must pay attention to the conditions under which heat transfer occurs. Merely increasing heat transfer rates will not necessarily improve the performance of Stirling cryocooler. The cylinder wall must be anchored to the temperature of the adjacent heat exchanger in order for additional heat transfer from the gas to have any positive effect.

Thesis Supervisor: Dr. Joseph L. Smith
Title: Professor of Mechanical Engineering

Acknowledgements

I would like to thank my advisor Joseph Smith for his stimulating ideas. I appreciate the company and advice of my fellow students at the Cryogenics Lab. I am indebted to Alan for starting me off on this project. I am especially grateful for Ho-Myung's insight of physical phenomena. Finally, I wish to acknowledge International Business Machines for support of this project.

Table of Contents

Abstract 2
Acknowlegdements 3
List of Figures 6
List of Common Symbols 7

1. BACKGROUND 9
   1.1 Stirling Cycle Description 9
   1.2 Previous Analytical Studies 9
   1.3 Gas Spring Hysteresis Losses 11
   1.4 Objective 12

2. GOVERNING EQUATIONS 13
   2.1 Introduction 13
   2.2 Basic Model with Ideal Components 13
      2.2.1 Assumptions 13
      2.2.2 Mass and Pressure Equations 13
   2.3 Heat Transfer Model 15
      2.3.1 Complex Nusselt Number 15
      2.3.2 Heat Transfer Equation 15
      2.3.3 Theoretical and Experimental Correlation 16

3. SOLUTION METHOD 19
   3.1 Introduction 19
   3.2 Perturbed 19
   3.3 Full 20
   3.4 Solution Algorithm 21

4. INCORPORATING SOLUTION WITH DECOUPLED LOSSES . . . 22
   4.1 Introduction 22
   4.2 Design Approach 22
   4.3 Pressure Drop 23
   4.4 Imperfect Heat Exchange 23
   4.5 Axial Conduction 24

5. TRENDS AND INTERPRETATION 25
5.1 Introduction 25
5.2 Inputs 25
5.3 Coefficient of Performance 25
5.4 Entropy Generation 28
5.5 Temperature Variation 30
5.6 Watts Power per Watt Refrigeration 31

6. CONCLUSION 32

Figures 34

Appendix A: STIRLING MODELS 50
A.1 Introduction 50
A.2 Dimensionless Values 50
A.3 Assumptions 52
A.4 Model with Heat Transfer 52

Appendix B: HEAT TRANSFER MODEL 57
B.1 Introduction 57
B.2 Parameter Definitions 57
B.3 Equation Development 58

Appendix C: COMPUTER CODE 61
D.1 Introduction 61
D.2 Header file 61
D.3 Driver code 63
D.4 Cycling routine 70
D.5 Function library 72
D.6 Integrator 76

References 79

List of Figures

1. Components of a Stirling Refrigerator 34
2. Complex Nusselt versus Peclet 35
3. Coefficient of Performance versus Peclet 36
4. Adiabatic Wall Temperature 37
5. Comparison of Pressure Ratios 38
6. Entropy Generation for Cold End with Isothermal Wall 39
7. Entropy Generation for Warm End with Isothermal Wall 40
8. Entropy Generation for Cold End with Adiabatic Wall 41
9. Entropy Generation for Warm End with Adiabatic Wall 42
10. Temperature Variation for Cold Cylinder with Isothermal Wall 43
11. Temperature Variation for Warm Cylinder with Isothermal Wall 44
12. Temperature Variation for Cold Cylinder with Adiabatic Wall 45
13. Temperature Variation for Warm Cylinder with Adiabatic Wall 46
14. Relative Power Loss with Isothermal Wall 47
15. Relative Power Loss with Adiabatic Wall 48
16. Trends in Relative Power Loss 49

List of Common Symbols

A	Heat transfer area
c_p	Specific Heat at Constant Pressure
c_v	Specific Heat at Constant Volume
COP	Coefficient of Performance
COP_{Car}	Coefficient of Performance for a Carnot refrigerator
D	Diameter
D_h	Hydraulic diameter at piston mid-stroke $= 4V/A$
D/L_s	Bore-to-stroke ratio
E	Energy
h	Specific enthalpy; convective heat transfer coefficient
$\mathcal{H}$	Dimensionless expression involving volume and Peclet
i	Imaginary unit, square root of negative one
k	Thermal conductivity
L	Length
L_s	Stroke length
m	Mass
$\mathcal{M}$	Dimensionless mass $= mRT^*/p_{max}V_a$
Nu	Nusselt number $= h\,D_h/k$
p	Pressure
$\mathcal{P}$	Dimensionless pressure $= p/p_{max}$
Pe	Peclet number $= \bar{v}\,D_h/\alpha$
Pe_ω	Oscillating flow Peclet number $= \frac{\pi}{4} Pe_h \frac{D_h}{L_s}$
Q	Heat transfer
$\mathcal{Q}$	Dimensionless heat transfer $= Q/p_{max}V_a$
R	Specific gas constant
$\mathcal{R}_{vt}$	Displaced mass ratio $= \frac{V_{ac}}{V_{aw}} \frac{T_w^*}{T_c^*}$
s	Entropy
$\mathcal{S}$	Dimensionless entropy $= s \cdot T^*/p_{max}\,V_a$
S_{gen}	Dimensionless entropy generated per cycle
t	Time
T	Temperature
T^*	Temperature of adjacent heat exchanger
$\mathcal{T}$	Dimensionless temperature $= T/T^*$
$\mathcal{T}^*$	Dimensionless temperature of adjacent heat exchanger $= 1.0$
V	Volume
$\bar{v}$	Mean velocity
V_a	Cylinder volume amplitude $= (V_{max} - V_{min})/2$
$\mathcal{V}$	Dimensionless volume $= V/V_a$
$\mathcal{V}_d$	Reduced dead volume $= m_d \frac{R\,T_w^*}{p\,V_{aw}}$
W	Work
$\mathcal{W}$	Dimensionless work per cycle $= \oint W/p_{max}V_a$

Greek Symbols

α	Thermal diffusivity $= k/\rho\, c_p$
γ	Ratio of specific heats $= c_p/c_v$
θ	Crank angle $= \omega\, t$
ρ	Density
ω	Angular frequency

Subscripts

c	Conditions in cold end
C	Complex variable
d	Conditions in lumped dead volume
gen	Generation as in entropy
in	Flow into a control volume
I	Imaginary part of complex variable
max	Maximum value of a property during a cycle
min	Minimum value of a property during a cycle
out	Flow out of a control volume
R	Real part of complex variable
w	Conditions in warm end
$wall$	Cylinder wall
o	Reference value

Chapter 1
BACKGROUND

1.1 Stirling Cycle Description

The ideal Stirling cycle is a completely reversible cycle that has two isothermal heat transfer processes involving compression and expansion, and two regenerative isochoric processes. The working fluid flows between two variable volumes via a regenerator. For a refrigerator, the volume used for expansion work is maintained at a low temperature, whereas the compression volume is at a high temperature. The temperature in each variable volume is kept constant by unrestricted exchange with a heat reservoir. The regenerator acts as a thermal capacitance which removes heat energy from the working fluid during one part of the cycle and transfers the energy back during another part. An ideal regenerator has no volume and introduces no pressure drop. Heat transfer occurs across an infinitesimal temperature difference. Because the processes are thermodynamically reversible, the performance of an ideal Stirling cycle is the same as that of a Carnot cycle. In practice, a cooler and a heater are added to improve the actual heat exchange.

Among the many configurations used for a Stirling refrigerator, the two cylinder type shown in Figure 1 represents the most general case, for the piston displacements of this type may be altered to simulate the volume variation of the other configurations. The setup, commonly referred to as the alpha configuration, has five axially aligned components. In between the two opposed cylinder working spaces are the heat exchange components. The regenerator, located at the center, is flanked on either side by a heat exchanger which is often the compact tubular type. The heat exchanger next to the warm compression space cools the working fluid by dumping heat energy to the environment. The heat exchanger bordering the cold expansion space extracts energy from the refrigeration load.

1.2 Previous Analytical Studies

The simplest analysis available is the application of the Second Law of Thermodynamics to the ideal cycle. This gives Carnot coefficient of performance. Analyses of greater complexity are required to determine operating conditions. Schmidt (1871)

devised a method that accounted for the physical dimensions of a machine. His model included the volume of the heat exchange components. Because the temperature of each component is assigned, the expansion and compression processes are isothermal. Closed form solutions are obtained when the volume variations in these spaces are taken to be sinusoidal. The dead volume of the heat exchange components affects the pressure ratio and power requirements. However, since the heat exchange components are still assumed to be ideal, the coefficient of performance is for a reversible cycle and is a function of temperature.

Finkelstein (1960) went a step further by considering working spaces that were adiabatic. While the Schmidt model relies on the ideal gas relation to determine pressure and mass, the Finkelstein adiabatic model requires conservation of energy. The coefficient of performance resulting from the adiabatic model is lower than that for a reversible cycle since mixing losses are included. Mixing loss occurs when gas from one component enters an adjacent component that has gas at a different temperature. A temperature discontinuity exists between a working space and its adjacent heat exchanger, because each is considered a separate control volume. The discrepancy entails that the temperature of the mass flux be dependent on flow direction. The fluid assumes the temperature of the component from which it is flowing out.

Although mixing losses are significant, other losses should be considered. In the Finkelstein analysis, the heat exchange components are assumed to be ideal. Nonideal behavior of the heat exchange components can create substantial losses. In actual heat exchange components, fluid friction is one major source of irreversibilities. Another cause is the heat transfer across finite temperature differences. An analysis that calculates these irreversibilities is needed.

The problem of modelling real Stirling cycles is quite complex. To accurately model conditions, the machine may be divided into many nodes to which conservation of mass, energy and momentum are applied. The general problem involves a large number of simultaneous differential and algebraic equations that are mostly nonlinear. Solution requires numerical methods. The problem may be simplified by selecting pertinent control volumes and independent variables. Further reduction may be achieved by assuming steady state conditions. Typically the equations are initialized and integrated repeatedly until convergence is attained. Solving these simultaneous equations is time

consuming and expensive. Often simplifying assumptions are made to obtain results. The accuracy of high order analyses is often compromised by such assumptions.

To analyze the losses effectively without extensive computational time, Qvale (1967) decoupled irreversibilities due to pressure drop and ineffective heat exchange from a basic model that had ideal heat heat exchange components. The cylinders are assumed to be adiabatic. First, the basic performance with only adiabatic compression and expansion losses is calculated. Then the irreversibilities due to nonideal components are determined and used to modify the basic performance. Pressure and mass variations were assumed to be sinusoids leaving volume to be determined.

Rios (1969) eliminated the sinusoidal restraint of this scheme by using volume as the independent variable. All the heat exchange components are lumped into a single dead volume. Variables for temperature and dead volume mass are eliminated by nondimensionalization. This produced a basic performance calculation that quickly converged within several cycles. In the basic cycle, the only irreversibility is due to adiabatic compression and expansion. The theory was supported by experimental data.

Nonideal behavior of the heat exchange components is another significant source of losses. Companion papers have been submitted to evaluate these decoupled losses. Qvale and Smith (1969) evaluated losses due to imperfect heat transfer in the regenerator. Rios and Smith (1969) presents an approximate method to calculate pressure drop. Regenerator design was addressed in Harris, Rios and Smith (1971). Regenerator geometry was varied to minimize the sum of losses.

1.3 Gas Spring Hysteresis Losses

When a gas spring is first compressed and then allowed to expand back to the initial rest state, a certain amount of work is lost. The irreversibility is due to gas-to-wall heat transfer which is inherent in machinery with reciprocating pistons. Chafe (1988) has shown that under certain operating conditions, gas spring hysteresis losses can seriously affect the performance of cryogenic machinery. Machine speed had a primary influence on the losses. The phase shift between the heat transfer and the gas-to-wall temperature difference is not adequately explained by conventional convective relations. Lee (1983) theoretically developed a complex heat transfer model that predicted losses reasonably well. Kornhauser (1989) proposed a complex Nusselt number that is a strong function

of an oscillating flow Peclet number. A semi-empirical correlation was obtained for this dependence. The time parameter required when computing heat transfer is supplied by the Peclet number. Fast cycling times, characterized by high Peclet numbers, have near adiabatic conditions because there is no time for heat transfer to occur. Low Peclet numbers indicate slow cycling times with near isothermal conditions.

1.4 Objective

It has been experimentally shown that gas-to-wall heat transfer in the reciprocating machinery can be a principal source of losses. The purpose of this thesis is to develop a model that includes this kind of heat transfer. The number of parameters required to describe heat transfer in the cylinders has been reduced to minimize solution complexity. With this improved model, one might gain a qualitative understanding of how design parameters affect performance. Contrary to the assumptions of the aforementioned models, actual conditions in cryogenic machinery are neither completely adiabatic nor isothermal. The decoupling of irreversibilities in heat exchange components can still be employed. While not as rigorous other analyses, the model provides fast yet illuminating results.

The formulation of the system equations is presented in Chapter 2. Detailed derivations are given in Appendixes A and B. In Chapter 3, the method of solving the differential equations is described. Chapter 4 summarizes how the decoupled losses are calculated and how they modify the model performance. Chapter 5 provides an interpretation of the numerical results. The results account for losses caused by heat transfer from cylinder wall to working gas as well as adiabatic compression and expansion. Under certain conditions, gas spring hysteresis loss can be quite significant.

Chapter 2
GOVERNING EQUATIONS

2.1 Introduction

In this chapter, the governing equations for the Stirling system are summarized. Pressure, mass, and heat transfer are the independent variables. Temperature is eliminated by nondimensionaliziation. Conservation of energy and mass are used to obtain the differential equations for pressure and mass. The form of each equation is dependent on the direction of mass flow. The heat transfer equation is based upon a complex Nusselt number that is correlated as a function of an oscillating flow Peclet number.

2.2 Basic Model with Ideal Components

2.2.1 Assumptions

The model for actual operating conditions is a modification of Rios's adiabatic model which is based on the Stirling alpha configuration as shown in Figure 1. The components of the Stirling refrigerator are divided into three primary control volumes. The warm and cold volumes at either end are each considered to be active volumes in which piston work is performed. The gas in each active volume is assumed to be perfectly mixed. Losses due to shuttle heat leak and axial conduction are neglected. The third volume, a lumped dead volume, consists of the dead volume in both heat exchangers and in the regenerator plus any dead volume presiding in the working volumes. The heat exchangers and the regenerator are idealized by assuming that neither heat exchange temperature differences nor pressure drops exist. The working fluid is assumed to be an ideal gas.

2.2.2 Pressure and Mass Equations

By including heat transfer in the energy equation and introducing ideal gas relation, differential equations for the mass in a working cylinder may by derived.

$$\begin{aligned} dM &= P\,d\mathcal{V} + \frac{1}{\gamma}\,\mathcal{V}\,dP - \frac{\gamma-1}{\gamma}\,d\mathcal{Q} && \text{for } dM > 0, \\ dM &= \frac{M}{P\,\mathcal{V}}\left(P\,d\mathcal{V} + \frac{1}{\gamma}\,\mathcal{V}\,dP - \frac{\gamma-1}{\gamma}\,d\mathcal{Q}\right) && \text{for } dM < 0. \end{aligned} \tag{2.1}$$

where γ is the ratio of specific heats and $\mathcal{P}$, $\mathcal{M}$, $\mathcal{V}$ and $\mathcal{Q}$ represent the respective dimensionless pressure, mass, volume and heat transfer. The variable $\mathcal{Q}$ represents gas-to-wall heat transfer only in the cylinder, not in the heat exchanger. Other forms of heat transfer are not considered in this idealized model. To account for adiabatic compression and expansion losses, two separate equations that depend on direction of mass flow are needed. When mass is entering a cylinder, its temperature is determined by the nearby heat exchanger. When mass is leaving the cylinder, gas temperature depends on the pressure, volume and mass in the cylinder. The mixing of gases at different temperatures leads to losses. The above equations apply to both warm and cold cylinders.

The total gas mass of the system is assumed to be constant. The differential form of the mass conservation equation may be written in dimensionless terms:

$$\mathcal{R}_{vt}\, d\mathcal{M}_c + \mathcal{V}_d\, d\mathcal{P} + d\mathcal{M}_w = 0. \tag{2.2}$$

The subscripts w and c respectively denote the warm and cold ends. Since pressure drops are assumed to be nonexistent, pressure $\mathcal{P}$ is the same for both cylinders. The geometry-related parameters are the reduced dead volume $\mathcal{V}_d$ and the displaced mass ratio $\mathcal{R}_{vt}$. The reduced dead volume is the ratio of the mass contained in the dead space to the mass contained in one half the warm cylinder volume at the same standard pressure and temperature. The parameter is related to the amount of gas that enters the dead volume from one cylinder but does not travel all the way to the other cylinder. No refrigeration is realized from the work expended to move this gas. The displaced mass ratio represents the ratio of the cold mass to the warm mass contained in their respective half volumes at their respective heat exchanger temperatures.

Because there are two working volumes, four possible mass flow combinations exist. The pressure derivative for each set of mass flows is obtained by substitution of the appropriate mass derivatives into Equation (2.2), followed by algebraic rearrangement: Pressure is the same for both warm and cold ends since pressure drops have been assumed to be nonexistent in the model. The resulting pressure equations are

$$
\begin{aligned}
dP &= -\gamma \frac{\mathcal{R}_{vt}\, P\, d\mathcal{V}_c + P\, d\mathcal{V}_w}{\mathcal{R}_{vt}\, \mathcal{V}_c + \mathcal{V}_w + \gamma\, \mathcal{V}_d} + \quad (\gamma - 1) \frac{\mathcal{R}_{vt}\, d\mathcal{Q}_c + d\mathcal{Q}_w}{\mathcal{R}_{vt}\, \mathcal{V}_c + \mathcal{V}_w + \gamma\, \mathcal{V}_d} \qquad \text{for } \begin{matrix} dM_c > 0 \\ dM_w > 0 \end{matrix}, \\
dP &= -\gamma \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{V}_c}{\mathcal{V}_c} + M_w \dfrac{d\mathcal{V}_w}{\mathcal{V}_w}}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} + (\gamma - 1) \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{Q}_c}{P\mathcal{V}_c} + M_w \dfrac{d\mathcal{Q}_w}{P\mathcal{V}_w}}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} \text{ for } \begin{matrix} dM_c < 0 \\ dM_w < 0 \end{matrix}, \\
dP &= -\gamma \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{V}_c}{\mathcal{V}_c} + P\, d\mathcal{V}_w}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \mathcal{V}_w + \gamma\, \mathcal{V}_d} + \quad (\gamma - 1) \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{Q}_c}{P\mathcal{V}_c} + d\mathcal{Q}_w}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \mathcal{V}_w + \gamma\, \mathcal{V}_d} \qquad \text{for } \begin{matrix} dM_c < 0 \\ dM_w > 0 \end{matrix}, \\
dP &= -\gamma \frac{\mathcal{R}_{vt}\, P\, d\mathcal{V}_c + M_w \dfrac{d\mathcal{V}_w}{\mathcal{V}_w}}{\mathcal{R}_{vt}\, \mathcal{V}_c + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} + \quad (\gamma - 1) \frac{\mathcal{R}_{vt}\, d\mathcal{Q}_c + M_w \dfrac{d\mathcal{Q}_w}{P\mathcal{V}_w}}{\mathcal{R}_{vt}\, \mathcal{V}_c + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} \qquad \text{for } \begin{matrix} dM_c > 0 \\ dM_w < 0 \end{matrix}.
\end{aligned}
\tag{2.3}
$$

2.3 Heat Transfer Model

2.3.1 Complex Nusselt Number

The heat transfer phenomenon in a gas spring is predicted reasonably well by a complex heat transfer model. Such a model is used to estimate the heat transfer that occurs in the working cylinders of Stirling refrigerators. Near the isothermal limit, heat transfer losses are low if the gas-to-wall temperature difference is small. Near the adiabatic end of the spectrum, the rate of heat transfer is small. Previous Stirling cycle analyses have not adequately modeled this gas-to-wall heat transfer in the working spaces. The primary reason is that an ordinary convective heat exchange model, based on a heat transfer coefficient and a gas-to-wall temperature difference, is incapable of predicting the heat transfer phase shift.

2.3.2 Heat Transfer Equation

The heat transfer model employed here uses a complex Nusselt number with an imaginary term involving the temperature derivative. The dimensionless heat transfer relation and appropriate dimensionless parameters are derived in Appendix B for simple piston-cylinder compressors and expanders:

$$
\frac{d\,\mathcal{Q}}{d\theta} = -\frac{\gamma}{\gamma - 1} \frac{\mathcal{V} + D/L_s}{1 + D/L_s} \frac{1}{Pe_\omega} \left[Nu_R\, (\mathcal{T} - \mathcal{T}_{wall}) d\theta + Nu_I \frac{d\mathcal{T}}{d\theta} \right] \tag{2.4}
$$

where Pe_ω is an oscillating flow Peclet number, Nu_R and Nu_I are the real and imaginary parts of the Nusselt number, $\mathcal{T}$ is dimensionless temperature, and subscript *wall* refers to the cylinder wall. The heat transfer area is described in terms of the volume $\mathcal{V}$ within the working space and the bore-to-stroke ratio D/L_s. In the heat transfer equation, we make the simplifying assumption that the cylinder wall temperature remains constant throughout each cycle. As discussed in Appendix B, Pe_ω is assumed constant over a cycle and is evaluated at peak pressure and exchanger temperature.

The oscillating flow Peclet number Pe_ω and the bore-to-stroke ratio D/L_s are two independent dimensionless parameters necessary to describe the heat transfer in the working volumes, since the two Nusselt numbers are functions of Pe_ω. The Peclet number reflects the rate of heat transfer in relation to the operating speed of the cooler. The bore-to-stroke ratio is associated only with the geometry of the working volume. The left-hand term within the square brackets is the real part of the complex model and is associated with conventional heat transfer which is proportional to gas-to-wall temperature difference. The right-hand or imaginary term relates heat transfer to the rate of change of temperature. By including the imaginary term, the phase shift that exists between heat transfer rate and gas-to-wall temperature difference may be predicted.

Temperature is not a primary variable and may be eliminated by substitution:

$$d\mathcal{Q} = -\mathcal{H}\left[Nu_R\left(\frac{\mathcal{P}\mathcal{V}}{\mathcal{M}} - \mathcal{T}_{wall}\right)d\theta + Nu_I\left(\frac{d\mathcal{P}}{\mathcal{P}} + \frac{d\mathcal{V}}{\mathcal{V}} - \frac{d\mathcal{M}}{\mathcal{M}}\right)\left(\frac{\mathcal{P}\mathcal{V}}{\mathcal{M}}\right)\right]. \qquad (2.5)$$

where

$$\mathcal{H} = \frac{\gamma}{\gamma - 1}\,\frac{\mathcal{V} + D/L_s}{1 + D/L_s}\,\frac{1}{Pe_\omega}. \qquad (2.6)$$

2.3.3 Theoretical and Experimental Correlation

The Peclet number reflects the rate of heat transfer in relation to the operating speed of the cooler. The complex Nusselt number has been related to the Peclet number by both theory and experiment. The complex Nusselt number may obtained from either the analytical results of Lee (1983) or the empirical results of Kornhauser (1989) depending on an oscillating flow Peclet number Pe_ω, defined by

$$Pe_\omega = \frac{\omega D_h^2}{4\alpha}, \qquad (2.7)$$

where ω is the angular speed of the piston motion, α is the thermal diffusivity of gas, and D_h is the mean hydraulic diameter of the cylinder. The oscillating flow Peclet number for adiabatic conditions is infinity. The corresponding value for isothermal conditions is zero. These values can be physically explained by considering a cylinder with a reciprocating piston. As the piston is speeded up, there is little time for heat transfer to occur and conditions become adiabatic. When the piston is slowed down, there is more time for heat transfer and isothermal conditions are approached. For the remaining chapters, the name Peclet will refer to Pe_ω, for convenience.

Lee (1983) derived a theoretical expression for the complex Nusselt number. Lee's model is based on a one dimensional energy equation for an ideal gas without convection in the boundary layer. Heat transfer is related to the difference between a constant wall temperature and a mixed mean bulk temperature. The amplitude of pressure and temperature fluctuations is assumed to be small so that variations in gas density may be neglected.

$$Nu_C = \sqrt{2\,Pe_\omega}\,\frac{(1+i)\,\tanh z}{1-(\tanh z\,/\,z)}, \tag{2.8}$$

where

$$z = (1+i)\sqrt{Pe_\omega/8}.$$

Kornhauser (1989) compared predictions by Lee's expressions and found good correlation for intermediate values of Pe_ω and large deviations for high and low ranges. Kornhauser obtained a semi-empirical power law expression for data points with $Pe_\omega >$ 100 by assuming that the real and imaginary parts of the complex Nusselt number were equal:

$$Nu_R = Nu_I = 0.56\,Pe_\omega^{0.69}. \tag{2.9}$$

The fit is for experimental cases with a volume ratio of two. For a small Pe_ω, the imaginary part of Nu_C in Equation (2.8) is much smaller than the real part, which means that in Equation (2.4) the heat transfer is proportional to the temperature difference and no phase shift occurs. For a large Pe_ω, the real and imaginary parts are equal according to Equation (2.9) and the phase lead of the heat transfer is 45° from the gas-to-wall temperature difference. In Figure 2, the Nusselt numbers from both correlations are plotted against Peclet in the appropriate ranges from 10 to 1000.

The complex theoretical expression of Equation (2.8) may be expressed in terms of a real and an imaginary part:

$$Nu_C = \frac{4\,u\,\sqrt{2\,Pe_\omega}\,\left(u\,b(\tanh\,u - a) + i[u\,b(\tanh\,u + a) - \tanh^2 u - a^2]\right)}{(2\,u\,b - \tanh\,u - a)^2 + (\tanh\,u - a)^2}, \qquad (2.10)$$

where

$$u = \sqrt{Pe_\omega/8},$$
$$a = \sin\,u\,\cos\,u\,\mathrm{sech}^2\,u,$$
$$b = \cos^2\,u + \sin^2\,u\,\tanh^2\,u.$$

This expression is used in plotting Figure 2.

The Nusselt-Peclet correlations presented above are for a closed gas spring system in which the amount of mass is fixed and the volume variation is small compared to the average volume. Unlike closed gas springs, the working volume of a Stirling machine has continuously varying mass and large volume ratios. Consequently when mass and volume are near zero, large rates of temperature change arise and cause numerical problems during simulation. The temperature derivative term in Equation (2.4) is physically related to the pressure fluctuation. This pressure fluctuation causes a near adiabatic temperature swing in the turbulent gas core in the cylinder. The sudden change in temperature is not due to the pressure fluctuation, but rather, due to the reversal in flow direction between the working volume and the adjacent heat exchanger. Therefore, evaluating the heat transfer rate based upon the temperature derivative is not suitable when the volume is near a minimum. The derivative of pressure may be a more befitting term.

Chapter 3
SOLUTION METHOD

3.1 Introduction

To facilitate further discussion, the differential equations are described as functions of the dependent variables below. Function f_q is obtained from Equation (2.5). Function f_p is directly taken from Equations (2.3). Function f_m is from Equations (2.1).

$$\begin{aligned} d\mathcal{Q} &= f_q(\mathcal{P}, d\mathcal{P}, \mathcal{M}, d\mathcal{M}), \\ d\mathcal{P} &= f_p(\mathcal{P}, \mathcal{M}, d\mathcal{Q}), \\ d\mathcal{M} &= f_m(\mathcal{P}, d\mathcal{P}, \mathcal{M}, d\mathcal{Q}). \end{aligned} \tag{3.1}$$

Because volume and its derivative are given as prescribed inputs, they are considered not as variables, but rather as parameters much like $\mathcal{R}_{vt}$, $\mathcal{V}_d$, Pe_ω and D/L_s. The equations as they stand cannot be solved by numerical integration because the functions depend on derivative terms shown on the right hand side.

Since the ideal adiabatic case has no heat transfer, all the derivatives may be eliminated by substitution:

$$\begin{aligned} d\mathcal{Q} &= 0, \\ d\mathcal{P} &= f_{p_{ideal}}(\mathcal{P}, \mathcal{M}), \\ d\mathcal{M} &= f_{m_{ideal}}(\mathcal{P}, f_{p_{ideal}}, \mathcal{M}) = f_{m_{ideal}}(\mathcal{P}, \mathcal{M}). \end{aligned} \tag{3.2}$$

When heat transfer is included, a nondeterminate loop results. The pressure derivative becomes a function of the heat transfer derivative, and visa versa. The heat transfer derivative must either be approximated or eliminated.

3.2 Perturbation Method

Since the heat transfer correlations (2.8) and (2.9) are approximate for averaged conditions, it is appropriate to use a perturbation method to obtain numerical results. The perturbation methods consists of estimating the heat transfer derivative using the heat transfer relation (2.5) with pressure and mass waves from previously calculated

points. For example, the dP and M may be calculated using the ideal adiabatic equations (3.2). Alternatively, dQ may be extrapolated from previously calculated points. Because dQ becomes an input, function f_q is eliminated from the integration scheme.

Solutions using the perturbation scheme exhibited fluctuations when the gas mass in a cylinder approached a minimum. This instability indicates that the heat transfer model used is inadequate for large compression ratios. Lee (1983) assumed that density changes were negligible in deriving his heat transfer relation. Kornhauser's (1989) correlation was fitted for cases that had a volume ratio of 2.0. The present heat transfer model clearly needs to be improved in order to apply to systems with large volume ratios.

3.3 Full Method

To achieve smooth solutions with the present model, heat transfer is eliminated algebraically from the set of equations. After the heat transfer relation (2.5) is substituted into mass equations (2.1), the mass derivative becomes:

$$dM = F\left(P\,d\mathcal{V} + \frac{1}{\gamma}\mathcal{V}\,dP + \frac{\gamma-1}{\gamma}\mathcal{H}\left[Nu_R\left(\frac{P\,\mathcal{V}}{M} - \mathcal{T}_{wall}\right)d\theta + \frac{Nu_I}{M}\left(P\,d\mathcal{V} + \mathcal{V}\,dP\right)\right]\right). \tag{3.3}$$

The factor F depends on mass flow:

$$F = \frac{M^2}{\beta + \frac{\gamma-1}{\gamma}\,Nu_I\,\mathcal{H}\,P\,\mathcal{V}}, \tag{3.4}$$

where

$$\beta = \begin{cases} M^2, & \text{for } dM > 0; \\ M\,P\,\mathcal{V}, & \text{for } dM < 0. \end{cases} \tag{3.5}$$

To solve for pressure, the mass derivative for each cylinder is substituted into the continuity equation (2.2). The general form for the pressure differential equation is:

$$dP = -\frac{\mathcal{R}_{vt}\,F_c\,N_c + F_w\,N_w}{\mathcal{R}_{vt}\,F_c\,D_c + F_w\,D_w}. \tag{3.6}$$

where

$$N = P\,d\mathcal{V} + \frac{\gamma-1}{\gamma}\,\mathcal{H}\left[Nu_R\left(\frac{P\,\mathcal{V}}{M} - \mathcal{T}_{wall}\right)d\theta + Nu_I\,\frac{P}{M}\,d\mathcal{V}\right], \tag{3.7}$$

$$D = \frac{1}{\gamma}\,\mathcal{V} + \frac{\gamma-1}{\gamma}\,\mathcal{H}\,Nu_I\frac{\mathcal{V}}{M}. \tag{3.8}$$

The quantities N and D apply to either cylinder. Once the pressure derivative is known, it is back substituted into Equation (3.3) to solve for the mass derivative.

3.4 Solution Algorithm

Since the governing equations are for a cyclic system, they may be solved as a steady state boundary value problem. A typical method for solving boundary value problems involves setting the derivatives as finite differences and then solving the resulting set of nonlinear equations. Instead of using finite differences, the problem has been transformed into an initial value problem to be solved using one point iteration. After the state variables are given initial values, the set of first order differential equations are numerically integrated over one cycle. The end values (at 360°) of pressure and mass are compared with the initial values (at 0°). If the values do not match, the state variables are reinitialized with the end values, and the integration is performed again. The procedure is repeated until convergence is attained.

To obtain the pressure and mass variations of the basic cycle, several inputs are required. The properties of the working gas must be known. The piston displacements and speed need to be specified. The relative proportions of the cylinders are taken as inputs. The operating temperatures of the heat exchangers are required. The size of the total dead volume must also be selected.

In Rios's analysis, the dead volume of both cylinders is added to the reduced dead volume $\mathcal{V}_d$. Thus, the minimum volume of each cylinder is zero containing no residual gas. This step accelerates convergence of the solution because as the last bit of the mass leaves the cylinder, no memory of temperature is retained. When there is no mass, there is physically no temperature. The returning mass has a temperature of unity, that of the warm heat exchanger. In the present analysis, a small dead volume is kept in the cylinders to prevent instabilities. To account for heat transfer, temperature is used indirectly in the governing equations causing some terms to have mass as a denominator. If the volume in a cylinder decreased to zero, no mass would remain and singularities would result.

Chapter 4
INCORPORATING SOLUTION WITH DECOUPLED LOSSES

4.1 Introduction

The model presented in previous chapters already includes irreversibilities caused by gas-to-wall heat transfer in the cylinders and by adiabatic compression and expansion. These two irreversibilities have a major effect on overall performance. Other major losses in Stirling-type refrigerators occur in the heat exchange components. This chapter will outline how these other losses are calculated and used to modify the performance of the cycle with perfect components. Previous analyses permit decoupling of heat exchange component irreversibilities from the basic performance of the adiabatic model. Addition of gas-to-wall heat transfer in the cylinder will not alter the method by which the decoupled losses are determined.

4.2 Design Approach

The basic performance is found by assuming perfect heat exchange components that have no frictional losses, no gas-to-wall temperature differences, nor axial conduction, but have dead volume. The components include the regenerator and the two heat exchangers. Imperfect components affect the basic performance in two ways. Three major losses are associated with the heat exchange components: (1) the loss due to imperfect heat exchange, (2) that due to pressure drop, and (3) the loss due to axial conduction. Pressure drop in all the components and imperfect heat transfer in the heat exchangers influence the pressure-volume relationship. Other losses affect the heat loads to the heat exchangers. Axial conduction, imperfect heat exchange in the regenerator, and the effect of piston motion belong to this second category. Once these losses are calculated, they may be added directly to the heat exchanger loads.

In the design of refrigerators, the conditions at the warm end compressor are often selected first. Various heat exchanger designs are then evaluated in conjunction with the chosen compressor. The calculation of component losses may be carried out in the same fashion. The design approach would be to use the same warm end geometry for both basic and real cycle calculations. By retaining warm end conditions, one would alter the cold end design to satisfy mass conservation and pressure drop considerations.

Once the basic calculations are done, the merits of different kinds of heat exchange components may be compared. The geometry of these components may be optimized within the constraints imposed by the dead volume input. Harris (1970) developed a regenerator optimization algorithm which was verified experimentally.

4.2 Pressure Drop

Pressure drop losses are treated by Rios and Smith (1970). The refrigeration load is equal to the work produced in the cold expander. Consider a control volume containing the cold cylinder and the adjacent heat exchanger. For cyclic steady conditions with perfect components, the net flow through the regenerator interface is zero because temperature is specified to be constant. The heat flow into the control volume through the walls of the heat exchanger and the cylinder must equal the work output. The refrigeration load is, therefore, directly affected by pressure drop in imperfect components. A pressure correction term and a volume correction term are used to correct the work output. When evaluating performance, the volume correction term may be neglected if pressure drops are small compared to the cyclic pressure variation.

The pressure drop is based upon local friction factors. The friction factors may be determined from the steady state correlations of Kays and London (1984). The local velocity needed for calculations is obtained from the mass flux. Gas flow is assumed to be one dimensional. The geometry and temperature profile of the heat exchange components are necessary to completely describe the pressure drop.

4.4 Imperfect Heat Exchange

The loss due to imperfect heat exchange has been treated by Rios, Qvale and Smith (1969). Imperfect heat transfer occurs across finite temperature differences between bulk gas and wall. Because the temperature difference is small compared to the temperature spanned by the regenerator, the impact on regenerator performance is small. Because the gross heat load of the regenerator is larger than the load of either heat exchanger, the regenerator accounts for a major fraction of dead volume losses. Correction involves determining an effective average temperature for the gas that flows out of the heat exchanger into the cylinder. The average temperature is based on the heat load of and the number of transfer units of each heat exchanger. The ratio of new

cold to new warm temperature is a factor used to correct the cold work.

Loss due to imperfect heat exchange may be treated as an enthalpy flow. The net enthalpy flow is necessary to heat the gas as it flows from cold end to warm end of the regenerator. Qvale and Smith (1969) examined the behavior of regenerators subjected to sinusoidal pressure and mass variations. They obtained an approximate closed form solution for the enthalpy flux. Rios (1969) extended the solution to arbitrary volume variations. Basic performance is modified by subtracting the enthalpy flux from the refrigeration load per cycle and adding the same amount of the cooling requirement of the warm heat exchanger.

4.5 Axial Conduction

Axial conduction is estimated by using a bulk thermal conductivity for the regenerator matrix and for the regenerator shell, and by finding the temperature profile of the regenerator. The temperature gradient may be obtained from regenerator theory. Alternatively, a linear temperature gradient provides fair results. Axial conduction may also be viewed as an enthalpy flux which decreases the refrigeration load while increasing the cooling requirements.

Chapter 5
TRENDS AND INTERPRETATION

5.1 Introduction

The purpose of introducing heat transfer in the cylinder walls is to better estimate overall efficiencies. Experimental results by Chafe (1988) have shown that gas spring hysteresis losses are greatest for intermediate operating speeds. In this chapter, the complex heat transfer model is used to confirm this trend. Two operating conditions are examined. One is with an effective isothermal wall; the other is with an effective adiabatic wall. The distribution of losses among heat transfer and mixing irreversibilities is determined by calculating entropy generation. Finally, entropy generation is related to coefficient of performance by computing the power input required for a given unit of refrigeration.

5.2 Inputs

The curves presented in this chapter evaluate performance for a wide range of Peclet values. In computing a given data point, the value of Pe_ω is set to be the same for both cylinders. For $Pe_\omega < 100$, Lee's theoretical correlation (2.8) is used to determine the complex Nusselt number. For $Pe_\omega \geq 100$, Kornhauser's experimental relation (2.9) is used. The fixed parametric inputs are $\gamma = 1.665$, $\mathcal{R}_{vt} = 1.0$, $\mathcal{V}_d = 1.0$, and $D/L_s = 1.0$ for both cylinders. For a displaced mass ratio of unity, pressurization of the lumped dead space of the heat exchange components will be approximately equal from both warm and cold ends. The volume variation is sinusoidal. At a crank angle of 0°, the cold cylinder volume is at a minimum. The phase lead of the cold cylinder volume variation over the warm end is 90°. Stirling machines operate near their maximum potential for this phase difference. To prevent instability problems a dead volume is introduced so that a finite amount of mass is present in each cylinder at top dead center. The amount of dead volume residing in each cylinder is five percent of the total cylinder volume.

5.3 Coefficient of Performance

The coefficient of performance (COP) is a yardstick for comparing all refrigerators. It is defined as the ratio of the refrigeration heat load to the required power input. The

energy extracted as the refrigeration load is equivalent to the work done by the gas at the cold end. Consider the cold heat exchanger and the cold cylinder as one control volume. Because the heat exchangers and the regenerator are ideal, there is no net enthalpy flux into the control volume under periodic steady state conditions. The gas passing through the interface between the heat exchanger and the regenerator is always at the same temperature, regardless of flow direction. By energy conservation, the work expended per cycle in the cold cylinder must equal the net heat transfer per cycle which is the heat load to the specified control volume. The heat load is transferred primarily through the cold heat exchanger and partly through the walls of the cold cylinder. Thus, the coefficient of performance for a nonideal cycle may be expressed in terms of work:

$$COP = \frac{refrigeration\ load}{net\ power\ input} = -\frac{\mathcal{R}_{vt}\frac{T_c^*}{T_w^*}\mathcal{W}_c}{\mathcal{W}_w - \mathcal{R}_{vt}\frac{T_c^*}{T_w^*}\mathcal{W}_c}. \tag{5.1}$$

Dimensionless work is converted to real units when multiplied by the maximum pressure p_{max} and the corresponding half swept volume V_a. Since COP is itself a ratio, the term p_{max} is canceled out and a volume ratio remains. In calculating the COP, the environment temperature is assumed to be $300K$; the refrigeration load temperature is taken to be $80K$. For an ideal Stirling cycle operating between two temperature limits, T_c^* and T_w^*, the coefficient of performance is the same as that of a Carnot refrigerator and is obtained by use of the second law of thermodynamics:

$$COP_{Car} = \frac{1}{\frac{T_w^*}{T_c^*} - 1}. \tag{5.2}$$

In Figure 3 the operating limits of the Stirling model performance are drawn for the gamut of Peclet conditions. The upper curve represents the case when the cylinder wall heat transfer contributes toward the refrigeration load. The lower curve represents the case when the net cyclic heat transfer through the cylinder wall is zero. The sudden change in slope at $Pe_\omega = 100$ arises from the use of two heat transfer correlations. As seen in Figure 2, the transition from one correlation to another is discontinuous.

The curve denoted as *isothermal* represents the condition when the cylinder wall temperature is set to be equal to the temperature of the bordering heat exchanger. The dimensionless wall temperature becomes unity. The wall temperature is assumed to be

constant throughout the entire cycle. The heat energy transferred from cylinder wall to working fluid contributes to the heat load. As Pe_ω is decreased towards the lower limit, the refrigeration load shifts from the cold heat exchanger to the cold cylinder wall. In addition, the increase in gas-to-wall heat transfer dampens the amplitude of temperature swing experienced by the cylinder gas. The mixing losses due to compression and expansion, which are maximum for high Pe_ω, gradually diminish to zero at the isothermal end.

The other limiting value for the cylinder wall temperature is the adiabatic wall temperature which is the temperature the wall attains when there is no net heat transfer per cycle. The curve representing this condition is denoted by *adiabatic*. Any net heat transfer experienced should occur through the heat exchanger, not the cylinder. The adiabatic wall temperature is determined by iteratively adjusting the wall temperatures of both cylinders until each cylinder has zero net heat transfer. In reality, the wall temperature would not be constant over an entire cycle; however, the variation is quite small. For the warm cylinder, the adiabatic wall temperature must be increased above unity because the warm gas temperature over most of the cycle is above one. At the cold end, the cylinder wall temperature is below unity. The adiabatic wall temperatures for the warm and cold cylinder walls are shown in Figure 4 as a function of Peclet.

The pressure ratios for the two limiting cases are compared in Figure 5. As Peclet decreases, one sees a general decrease in pressure ratio. The pressure ratios for the adiabatic wall cases are somewhat higher since damping due to heat transfer is lower.

The curves denoted by *isothermal* and *adiabatic* represent the limiting conditions in real operations. Figure 3 shows that the coefficient of performance for a Stirling refrigerator can be decreased by a factor of two if it is not designed properly. To obtain peak performance, all heat transfer across cylinder walls should contribute to the heat load. This measure is especially important for small refrigerators that have low range Peclet numbers. Because it is proportional to the square of the hydraulic diameter, the Peclet number is strongly influenced by size. Most Stirling refrigerators, however, tend to operate with $Pe_\omega > 100$. The difference in performance between the *isothermal* and the *adiabatic* curves is still significant in this range.

As Peclet decreases below 100, predictions by the two heat transfer relations begin

to diverge for the adiabatic wall cases. By assuming that the real and imaginary parts of the Nusselt number are equal, Kornhauser's correlation has a fixed 45° phase shift between heat transfer and gas-to-wall temperature difference. The resulting predictions show gradual drop in performance for $Pe_\omega < 100$. Lee's correlation, on the other hand, has a changing phase shift and predicts a substantial drop in performance. Instead of decreasing mixing losses as in the isothermal wall temperature case, the phase shift in heat transfer acts to augment the total losses. Losses grow as heat transfer increases with decreasing Peclet. At very low Peclet, the phase shift is small and the losses diminish.

5.3 Entropy Generation

The origin of losses may be better understood by examining entropy generation. Entropy is generated in the basic model by the mixing gases of different temperatures and by cylinder wall heat transfer. The losses due to mixing may be separated into an inflow part and an outflow part. When mass is flowing into the cylinder, consider a control volume of infinitesimal width that is placed just inside the cylinder head boundary. Pressure is equal on both sides of the control volume and does not contribute to a net entropy change. Temperature, however, does. Gas enters the control volume at the heat exchanger temperature, but exits at the bulk mean gas temperature in the cylinder. An entropy change is associated with the temperature difference. Energy conservation dictates that heat transfer to the control volume must accompany the change in temperature. The heat transfer occurs at the bulk mean gas temperature. The entropy generated per cycle in the control volume may be expressed as follows:

$$\begin{aligned} S_{gen,in} &= \int_{in} \frac{\gamma}{\gamma-1}(\ln \mathcal{T} - \ln \mathcal{T}^*)d\mathcal{M} - \int_{in} \frac{d\mathcal{Q}}{\mathcal{T}}, \\ &= \int_{in} \frac{\gamma}{\gamma-1}(\ln \mathcal{T} - \ln \mathcal{T}^* - \frac{\mathcal{T}-\mathcal{T}^*}{\mathcal{T}})d\mathcal{M}, \\ &= \int_{in} \frac{\gamma}{\gamma-1}(\ln \mathcal{T} - \frac{\mathcal{T}-1}{\mathcal{T}})d\mathcal{M}, \end{aligned} \tag{5.3}$$

where $\mathcal{T}$ is the temperature of the gas in the cylinder and $\mathcal{T}^*$ is the temperature of the gas in the heat exchanger. On the last step, the dimensionless temperature of the heat exchanger is replaced by the value one. Note that the integral is evaluated only when mass is flowing into the cylinder, that is when $d\mathcal{M} > 0$. A similar control volume may

be placed on the heat exchanger side of the interface when considering irreversibilities due to flow out of the cylinder. Gas enters the control volume at the cylinder mean temperature, but exits at the heat exchanger temperature. Heat energy is transferred across the control volume surfaces at the heat exchanger temperature. The expression for entropy generation due to outflow is

$$\begin{aligned} S_{gen,out} &= \int_{out} \frac{\gamma}{\gamma-1}(\ln \mathcal{T} - \ln \mathcal{T}^*)d\mathcal{M} - \int_{out} \frac{d\mathcal{Q}}{\mathcal{T}^*}, \\ &= \int_{out} \frac{\gamma}{\gamma-1}(\ln \mathcal{T} - \ln \mathcal{T}^* - \frac{\mathcal{T}-\mathcal{T}^*}{\mathcal{T}^*})d\mathcal{M}, \\ &= \int_{out} \frac{\gamma}{\gamma-1}(\ln \mathcal{T} - (\mathcal{T}-1))d\mathcal{M}, \end{aligned} \tag{5.4}$$

where $d\mathcal{M}$, which is the change in cylinder mass, is negative during outflow. The integral is evaluated only when $d\mathcal{M} < 0$.

The losses due to heat transfer to and from the cylinder wall may be determined by examining a control volume of infinitesimal thickness along the cylinder walls. Since all gas flows are one dimensional and parallel to the cylinder wall, no mass flows through the control volume. Therefore, entropy change is only associated with heat transfer at different temperatures. Heat transfer with the cylinder wall occurs at the wall temperature; heat transfer with the gas occurs at the gas temperature. The third part of entropy generation is then

$$S_{gen,wall} = \oint_{wall} \left(\frac{1}{\mathcal{T}} - \frac{1}{\mathcal{T}^*}\right) d\mathcal{Q}. \tag{5.5}$$

In Figures 6 and 7, the three individual losses and their total sum are calculated for the isothermal wall case at the cold and warm ends. Figures 8 and 9 have the same curves for the adiabatic wall case. In all instances, the entropy generation due to cylinder wall heat transfer is maximum for an intermediate value of Peclet. The magnitude of generation is slightly less for the adiabatic wall case because the gross heat transfer rate is smaller. At the high Peclet limit, gas-to-wall heat transfer is practically nil and leads to no irreversibilities. As Peclet is lowered, the increase in heat transfer has a dominant effect on gas-to-wall losses. The effect reaches a maximum in the intermediate Peclet range. For the cases examined here, the peak is at about $Pe_\omega = 20$. When Peclet is lowered further, the lack of temperature variation becomes

the predominant effect. Because gas temperature remains constant throughout the cycle, the cyclic integral of Equation (5.5) is zero.

5.4 Temperature Variation

The inclusion of heat transfer dampens the magnitude of the temperature swings. Whenever gas temperatures rise above that of the wall, heat transfer to the wall reduces the amount of rise. Whenever, temperatures go below that of the wall, heat transfer from the wall lessens the extent of drop.

Unlike the gas-to-wall heat transfer losses, mixing losses behave differently for the two cases. The isothermal wall case will be discussed first. At the lower Peclet limit, both inflow and outflow mixing losses diminish to zero because temperature does not fluctuate. The temperature remains constant at a value equal to the heat exchanger temperature. Because gas on either side of the cylinder head boundary is always at the same temperature, no entropy generation due to mixing occurs. At the high Peclet limit, inflow losses are much smaller than outflow losses. This difference in magnitude may be better understood by inspecting Figures 10 and 11, which show temperature histories at several values of Pe_ω for the respective cold and warm ends. The mass history for $Pe_\omega = 10$ is also included. The mass curve does not change much for different values of Peclet. When mass is accumulating in a cylinder, the temperature difference across the cylinder head interface is small. For the cold end the portion of the cycle is approximately between 180° and 360°. For the warm, the period is approximately from 90° to 270°.

The case with an adiabatic wall exhibits different behavior in the low to intermediate Peclet range. The temperature histories for this case are shown in Figures 12 and 13. The major difference is that mixing losses do not diminish to zero at the low Peclet limit. Although gas temperature in the cylinder remains constant throughout the cycle, its value differs from the temperature of the adjacent heat exchanger. Outflow and inflow mixing losses are about the same magnitude. The peaking of gas-to-wall heat transfer losses for intermediate Peclet becomes apparent since mixing losses no longer decrease to zero for low Peclet.

5.5 Watts Power per Watt Refrigeration

Measuring a system's power requirement for a given unit of refrigeration will offer an alternative perspective of the system's performance. The quantity under investigation is the inverse of the coefficient of performance and has units of watts per watt. For a reversible cycle, the quantity is simple the inverse of COP_{Car}. However, when entropy is generated, additional power input is required to produce the same amount of refrigeration.

To derive a relation that links entropy generation to the inverse of COP, the first and second laws of thermodynamics are applied the entire system. The net work input is balanced by the net heat transfer into the cold end and out of the warm end. The summation of entropy transfers is balanced by the total entropy generated. The two laws of thermodynamics may be combined with the definitions of Equations (5.1) and (5.2) to produce

$$\frac{1}{COP} = \frac{1}{COP_{Car}} + \left(\frac{T_w^*}{T_c^*}\right)\left(\frac{s_{gen}}{s_{in}}\right)_{total}. \tag{5.6}$$

where s_{in} is the entropy transferred intc the cold end. Dividing through by COP_{Car}^{-1} and converting to dimensionless terms gives

$$\begin{aligned}
\frac{COP^{-1}}{COP_{Car}^{-1}} &= 1 + COP_{Car}^{-1}\left(\frac{T_w^*}{T_c^*}\right)\left(\frac{s_{gen}}{s_{in}}\right)_{total}, \\
&= 1 + \frac{T_w^*}{T_w^* - T_c^*}\frac{\mathcal{R}_{vt}\, S_{gen,c} + S_{gen,w}}{\mathcal{R}_{vt}\, \mathcal{W}_c}, \\
\frac{Power}{Carnot\ power} &= 1 + \frac{Power\ loss}{Carnot\ power}.
\end{aligned} \tag{5.7}$$

The quantity one on the right-hand side represents the ideal reversible portion. Irreversibilities, represented by the right most term, increase the amount of work required. The components of the above relation are shown in Figures 14 and 15 for the respective isothermal wall and adiabatic wall cases. The value one represents the power necessary for an ideal cycle. The power compensations for both cold end and warm end irreversibilities are added one at a time to show their relative magnitudes. Because $\mathcal{R}_{vt} = 1$, the relative magnitudes of the dimensionless cold and the dimensionless warm entropy generations are the same in real dimensions.

Chapter 6
CONCLUSION

A Stirling cycle model that is capable of predicting gas spring hysteresis losses has been developed. The gas-to-wall heat transfer in the working cylinders is based on a complex Nusselt number which has been previously correlated as a function of an oscillating flow Peclet number. This Peclet number is related to the operating speed of a system. For a system with the same geometric dimensions and working fluid, a high value of Peclet indicates a high cycling rate, while a low value corresponds to a low cycling rate. The complex Nusselt number accounts for the phase shift that exists between heat transfer rate and gas-to-wall temperature difference. The regenerator and heat exchangers are idealized so that no frictional and heat exchange losses occur within these components.

The origin of losses is investigated using entropy generation analysis. Specifically, three causes are examined: (1) mixing during inflow to the cylinder; (2) mixing during outflow from the cylinder; and (3) gas-to-wall heat transfer in the cylinder. The degree of irreversibility depends upon the conditions under which gas-to-wall heat transfer takes place. Two hypothetical extreme conditions are considered. For one extreme, referred to as the isothermal wall case, the cylinder wall acts as an extension of the adjacent heat exchanger by assuming the heat exchanger temperature. For the other extreme, the cylinder wall acts adiabatically by attaining the temperature for which the *net* heat transfer per cycle in the cylinder is zero, even though there is heat transfer through out the cycle.

The additional power requirements (loss) due to the three sources of irreversibility are calculated for each extreme along the gamut of Peclet values. The major trends in the warm end are highlighted in the bar graph of Figure 16. The losses for various cases are shown along the same axis so that their relative magnitudes may be compared. Altogether, eighteen cases are shown. Three Peclet values are presented: a low value of 0.2, and intermediate value of 20, and a high value of 1,000,000. The power loss for inflow, outflow and wall flux are given for both isothermal and adiabatic conditions at each Peclet value. For both isothermal and adiabatic wall conditions, power losses due to cylinder wall heat flux are highest for the intermediate Peclet range, but are near

zero at the high and low ends. The inflow and outflow mixing losses act differently for the two wall conditions, notably in the low and intermediate Peclet ranges. As the lower limit of Peclet is approached, mixing losses diminish to zero for the isothermal wall condition, but either increase or level off for the adiabatic condition.

The combination of peak wall flux losses for an intermediate Peclet value and high mixing losses in the low to intermediate range produces an adiabatic condition that compares unfavorably to the isothermal condition. Computational results show that performance between the two conditions may differ by as much as a factor of two. Therefore, the designer should beware that enhancing heat transfer will not necessarily improve the performance of Stirling cryocooler. The cylinder wall must be anchored to the temperature of the adjacent heat exchanger in order for additional heat transfer from the gas to be of any use.

The incorporation of a complex heat transfer model does not complicate existing methods for decoupling losses due to nonideal behavior of the regenerator and heat exchangers. The combination of the present model with the decoupling methods should provide an efficient tool suitable for analyzing the principal irreversibilities. The current equations, however, are not adequate for making accurate predictions. Based upon low volume ratios, the heat transfer correlations restrict the breadth of the model's applications. Further work to revise the heat transfer model is recommended.

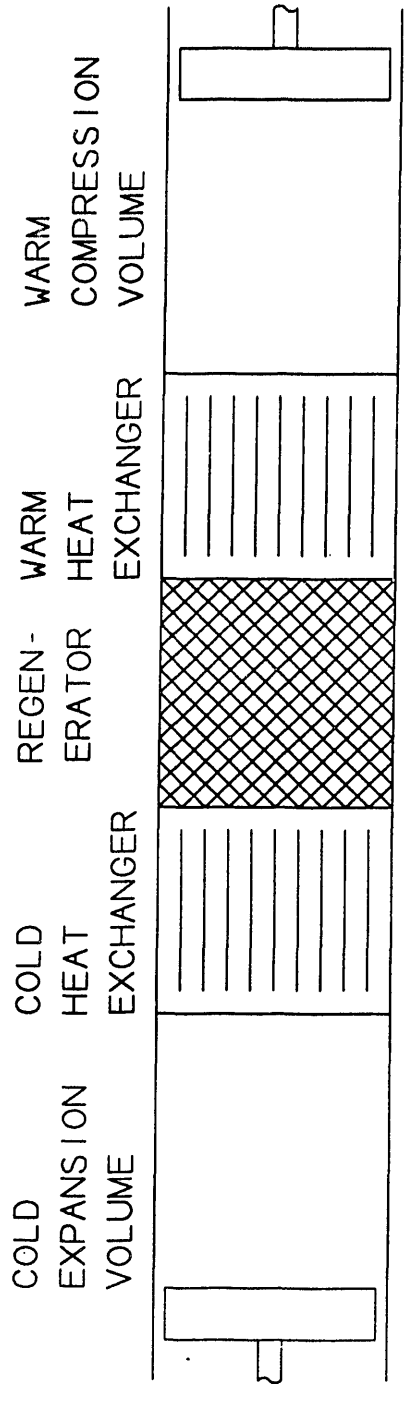

Figure 1. Components of a Stirling Refrigerator

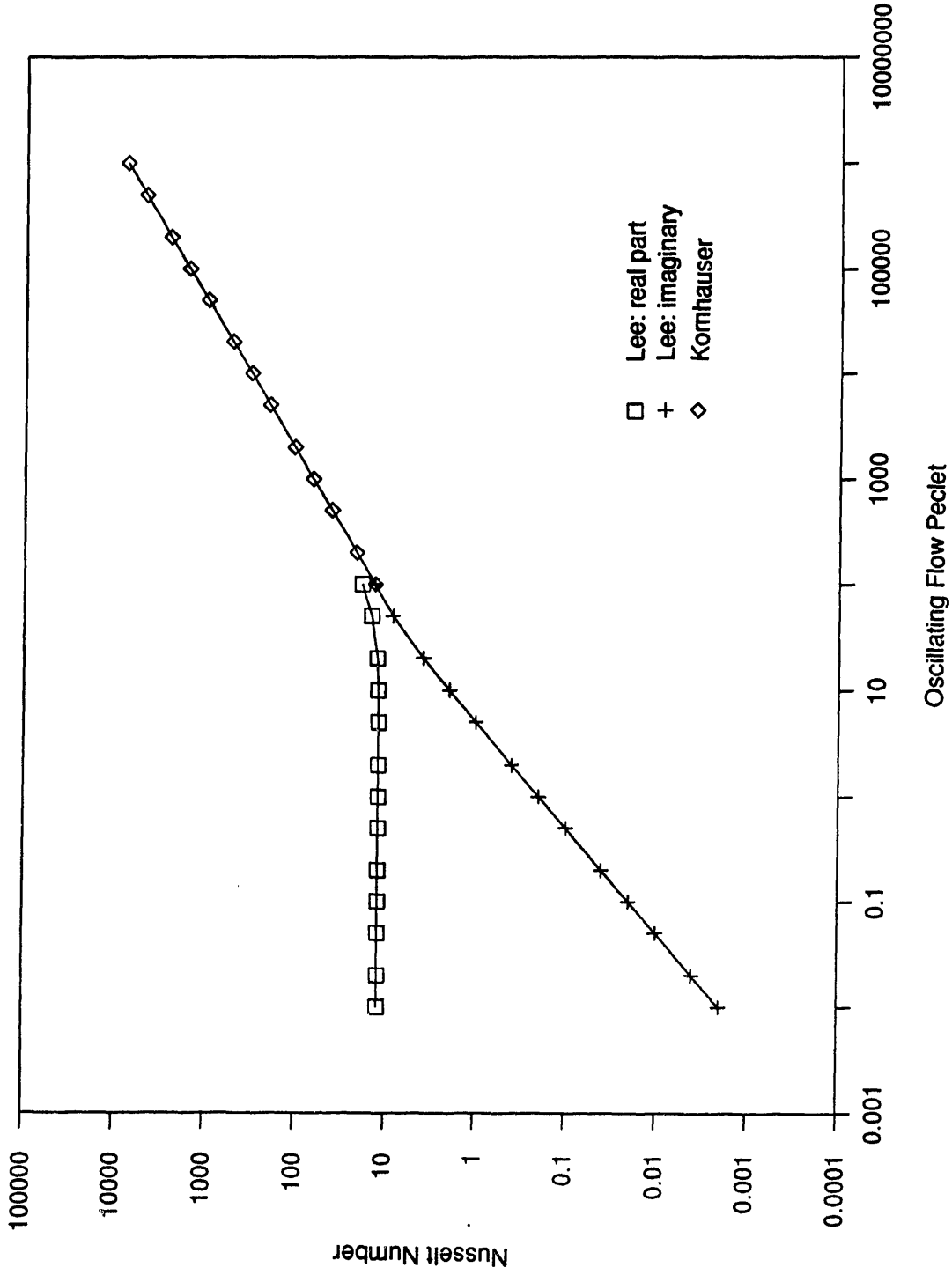

Figure 2. Complex Nusselt versus Peclet

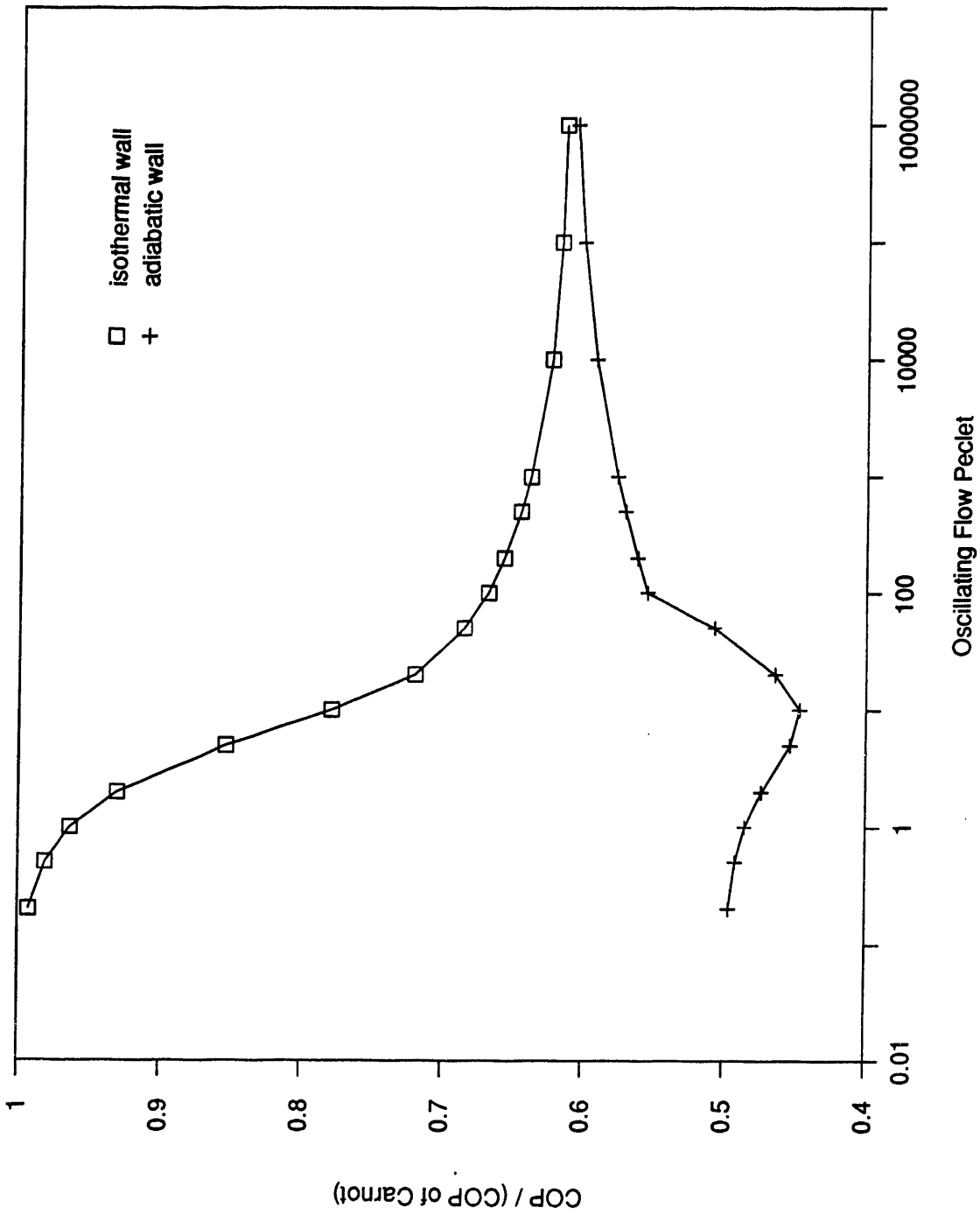

Figure 3. Coefficient of Performance versus Peclet

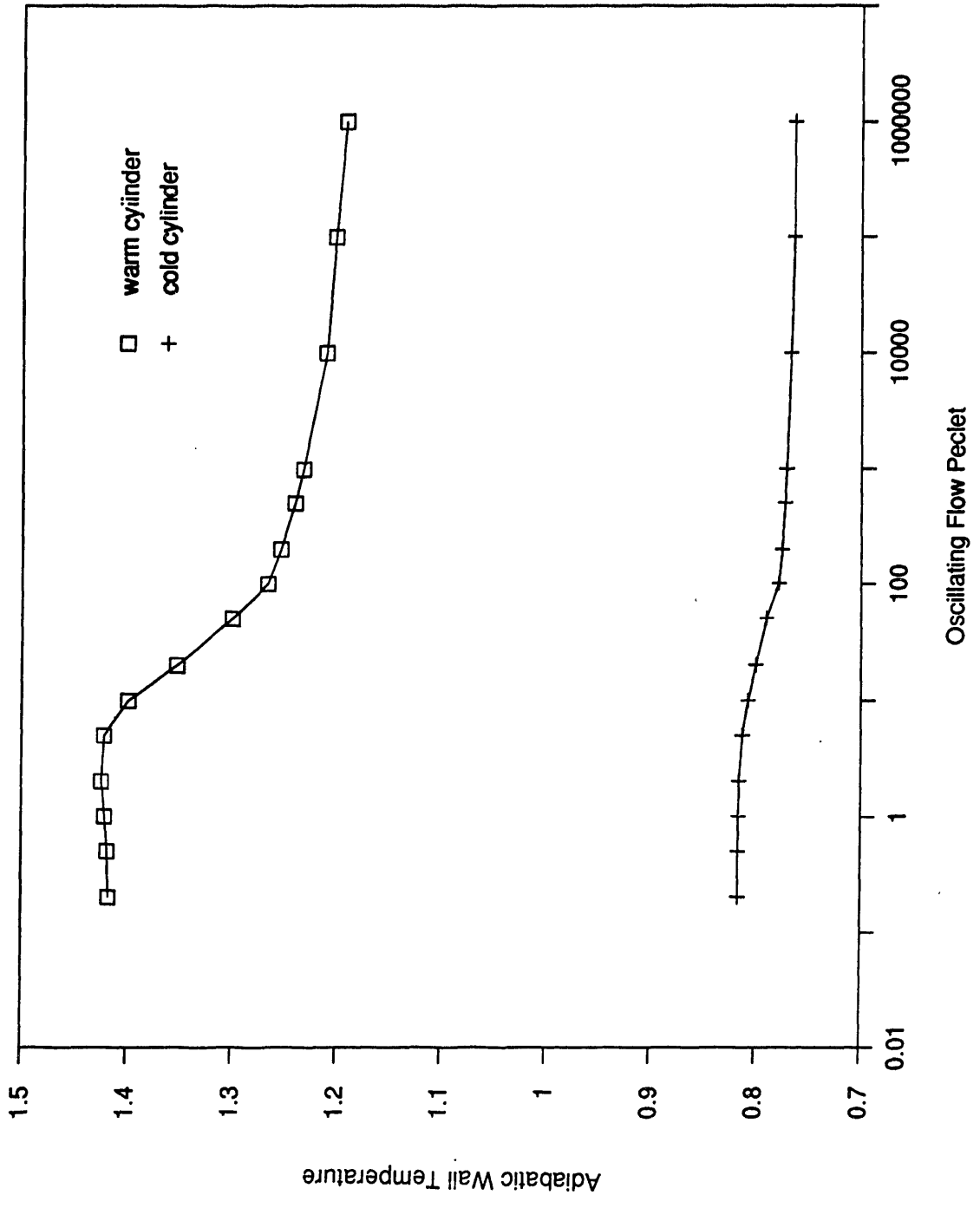

Figure 4. Adiabatic Wall Temperature

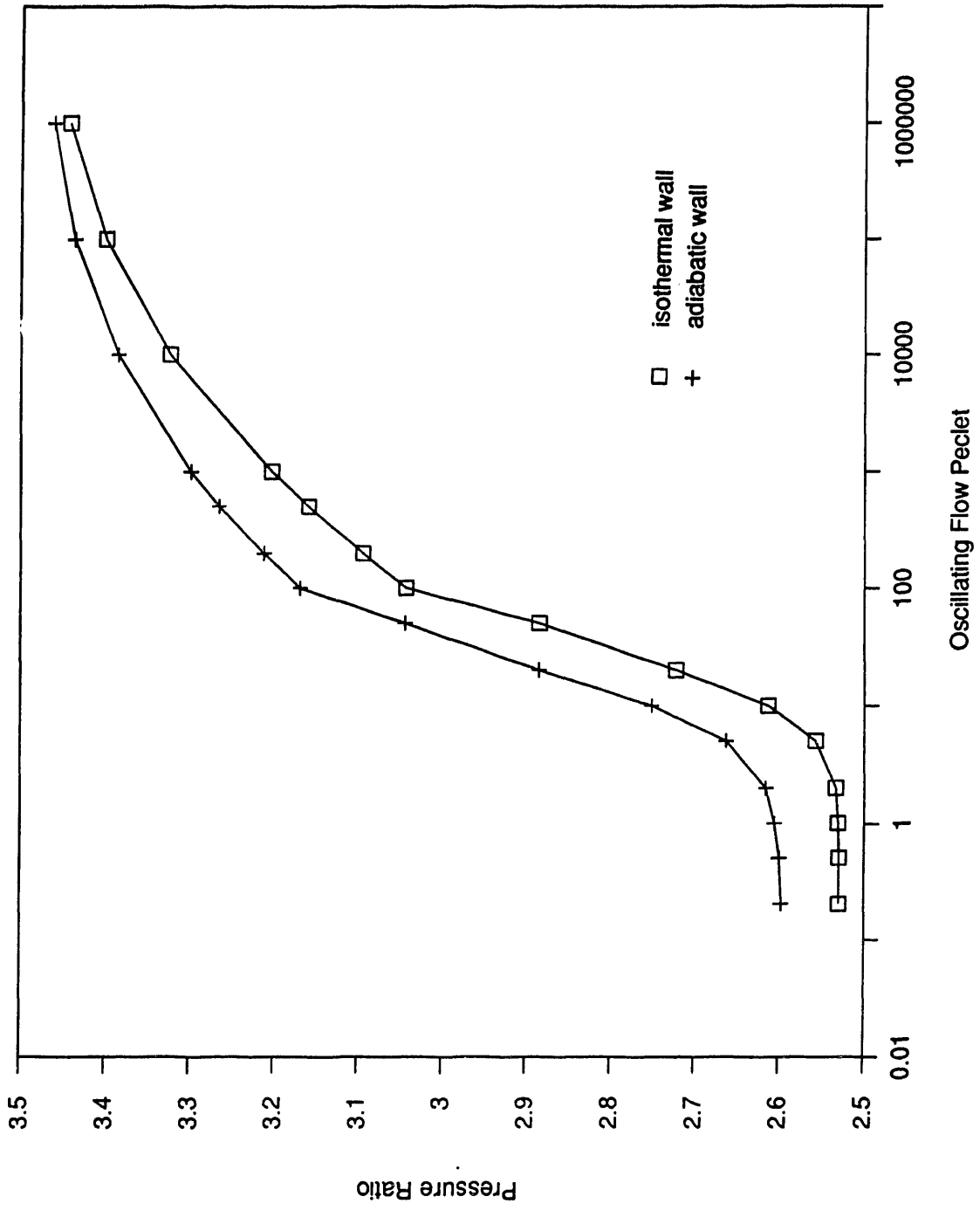

Figure 5. Comparison of Pressure Ratios

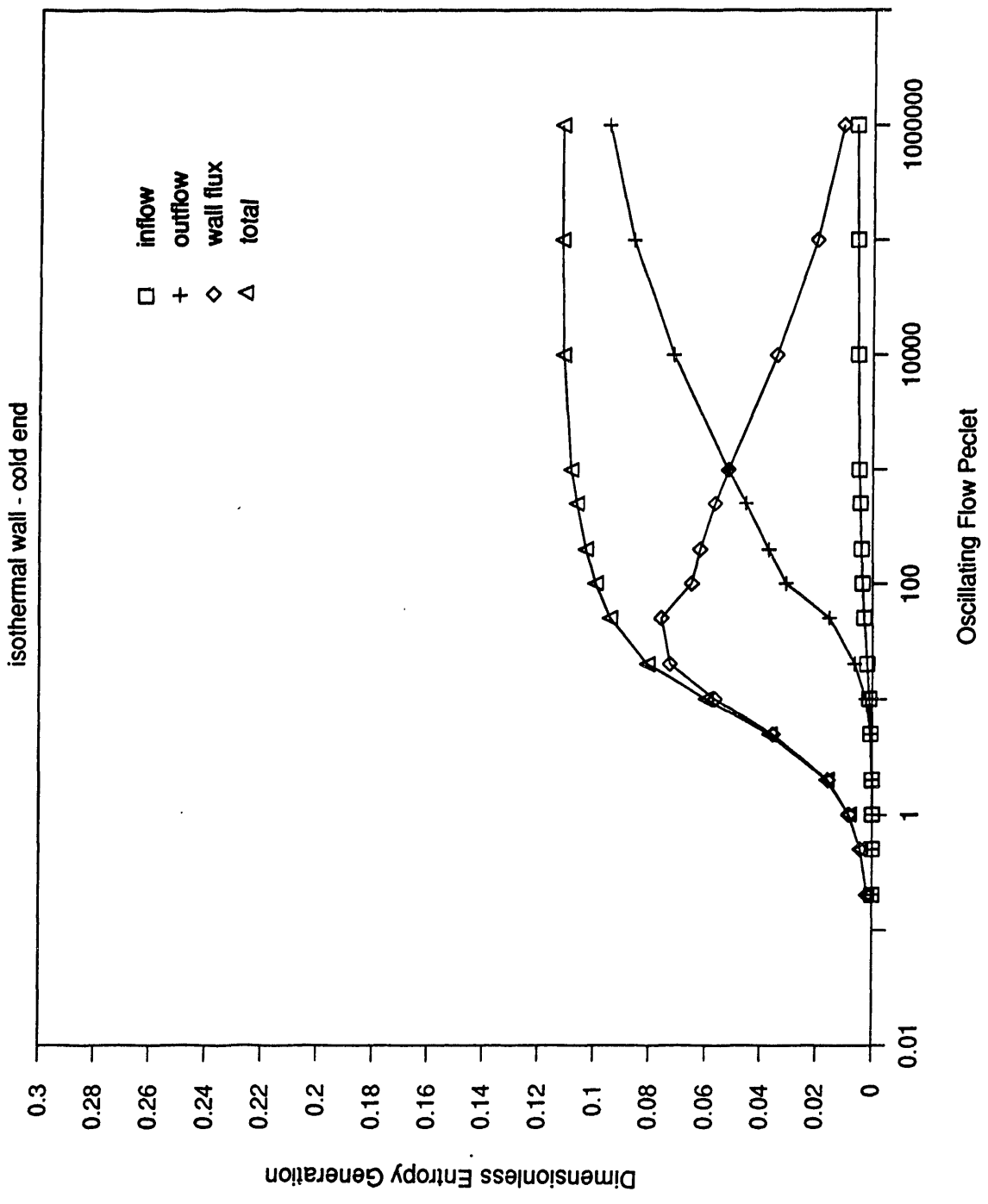

Figure 6. Entropy Generation for Cold End with Isothermal Wall

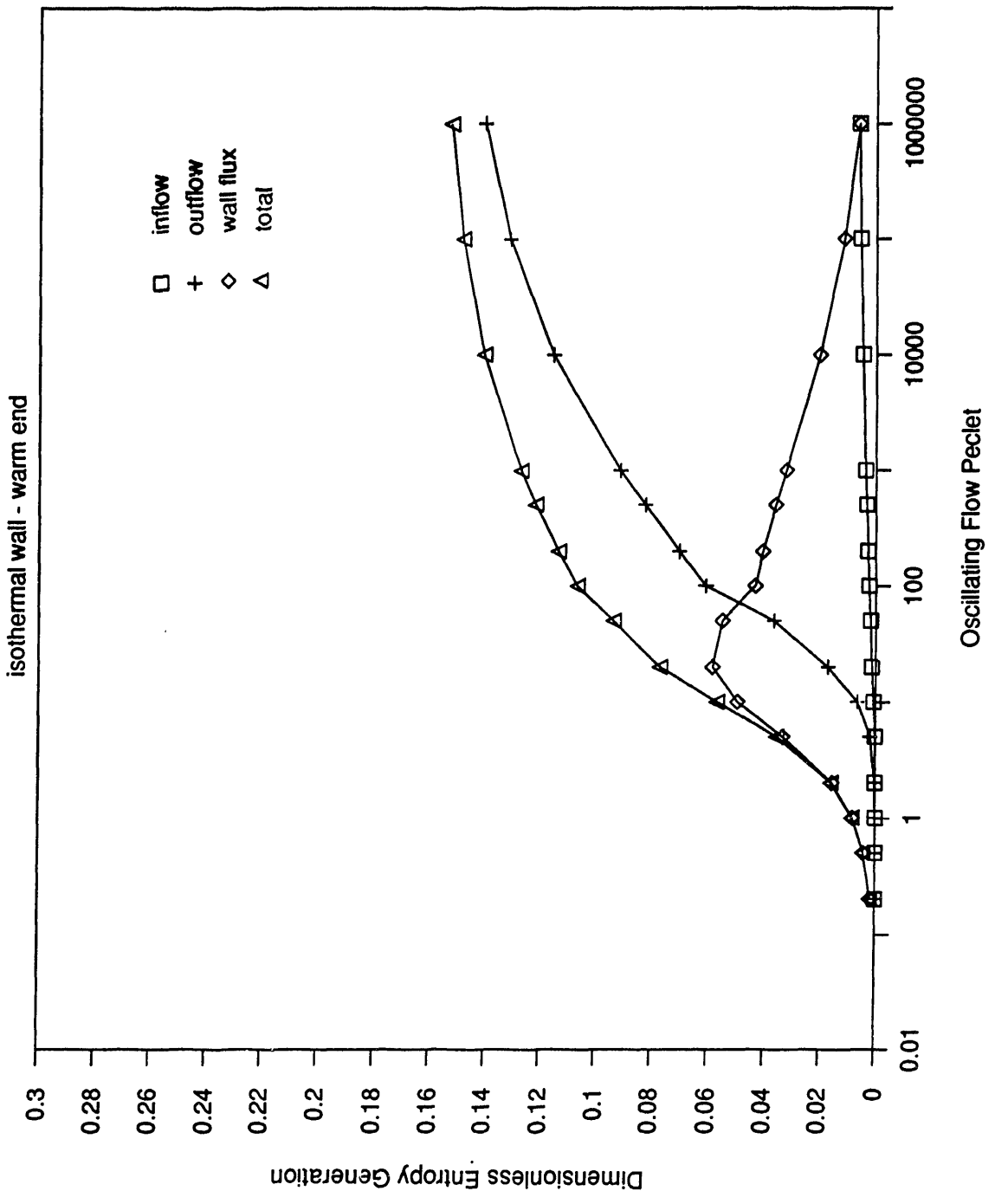

Figure 7. Entropy Generation for Warm End with Isothermal Wall

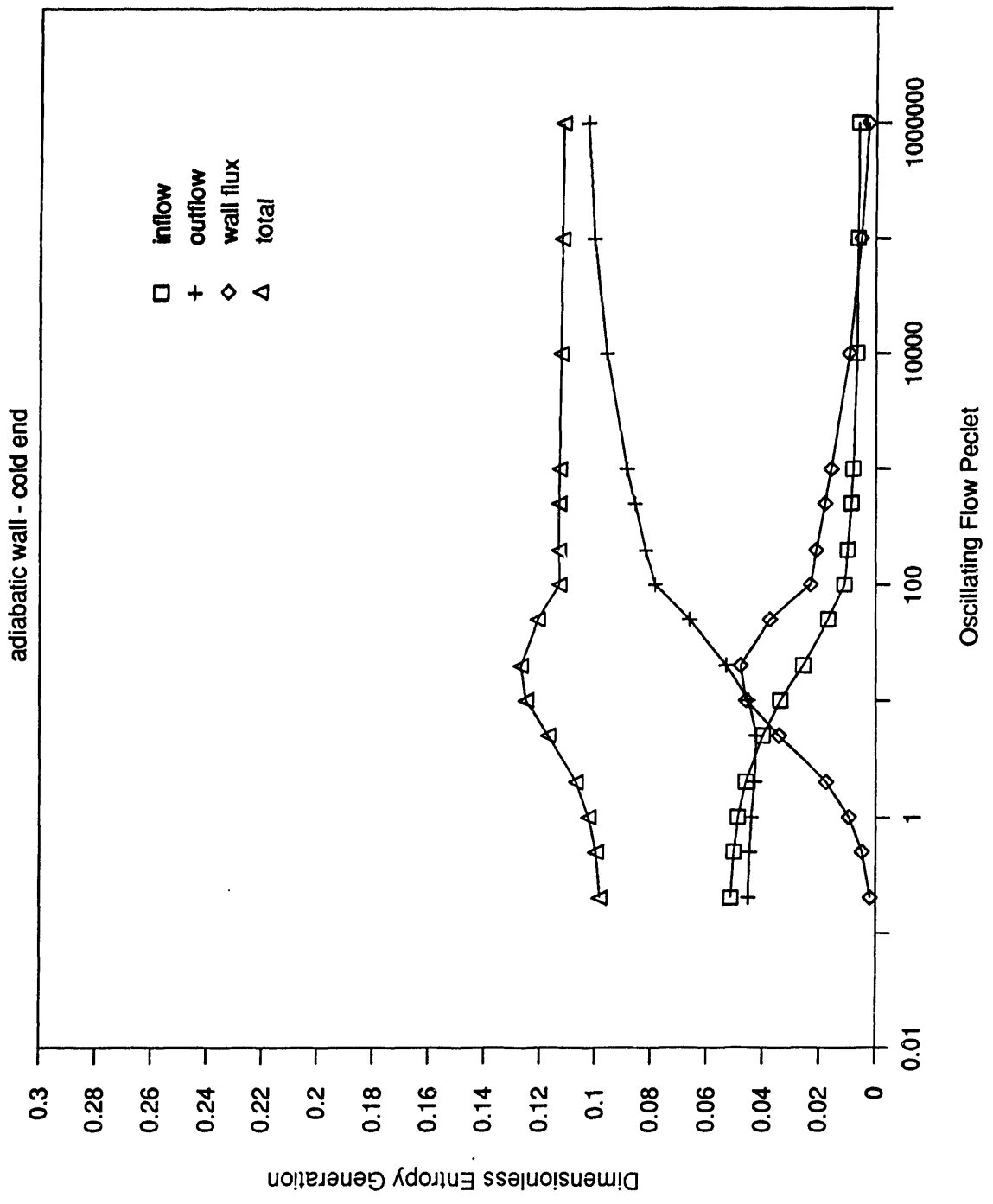

Figure 8. Entropy Generation for Cold End with Adiabatic Wall

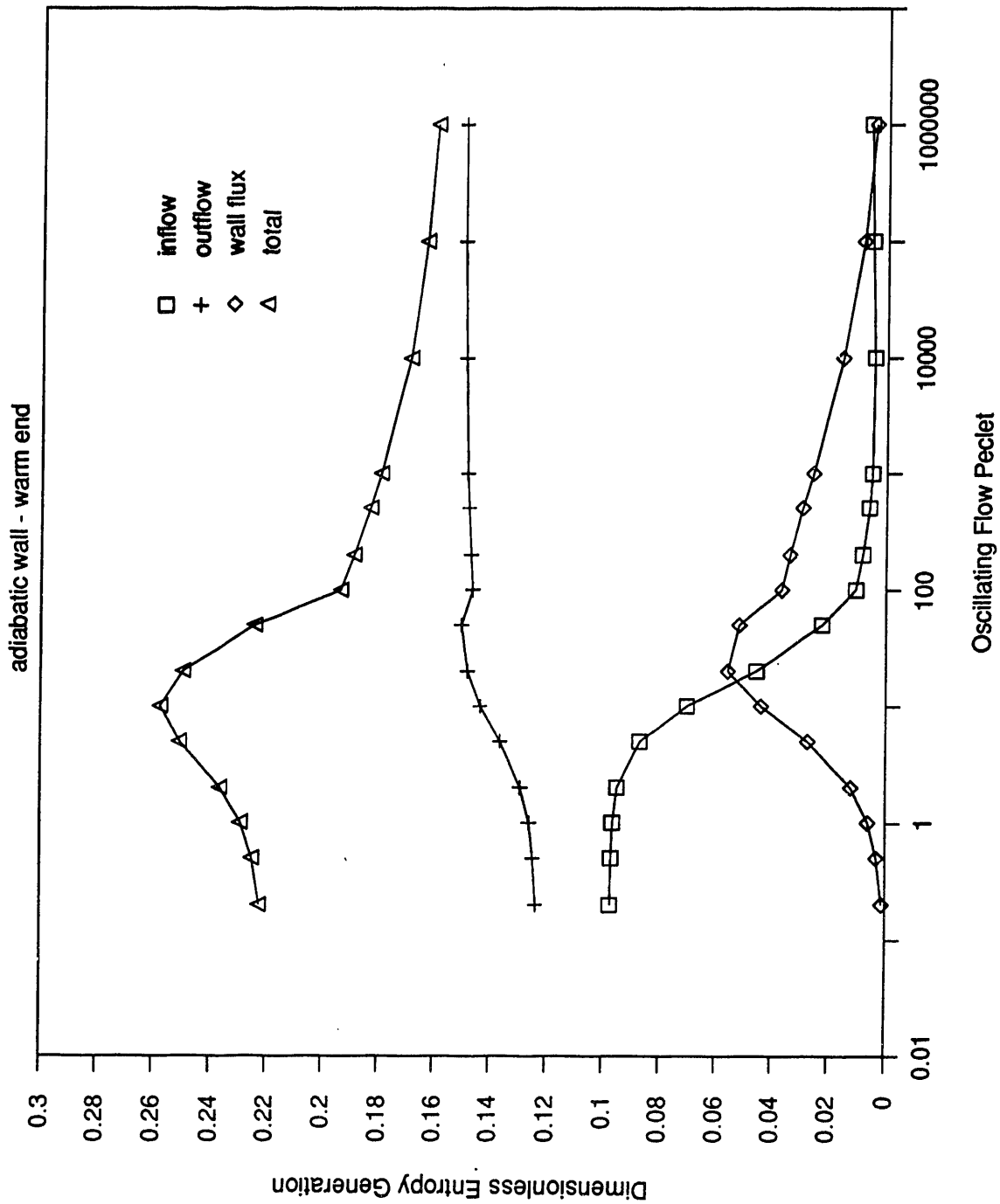

Figure 9. Entropy Generation for Warm End with Adiabatic Wall

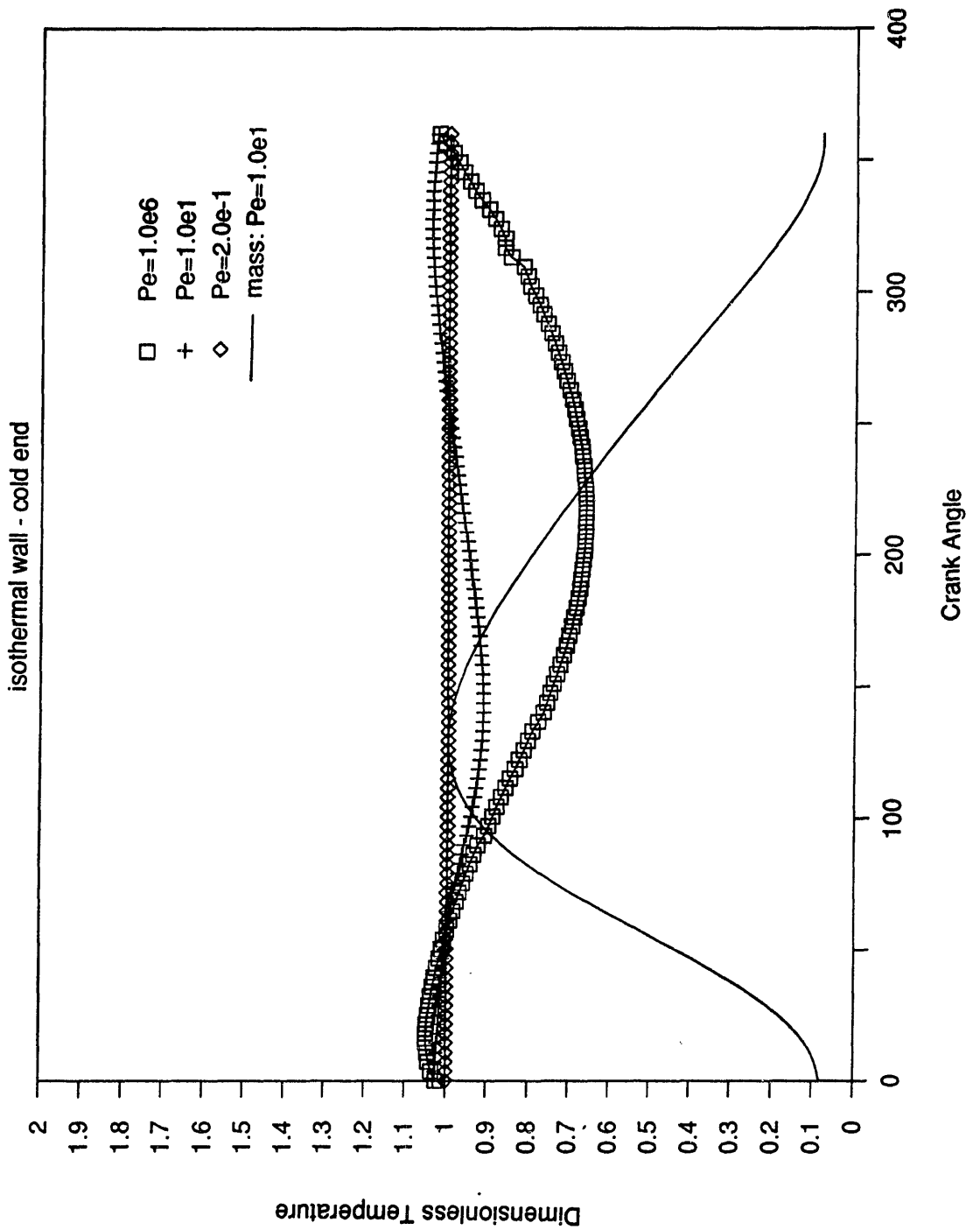

Figure 10. Temperature Variation for Cold Cylinder with Isothermal Wall

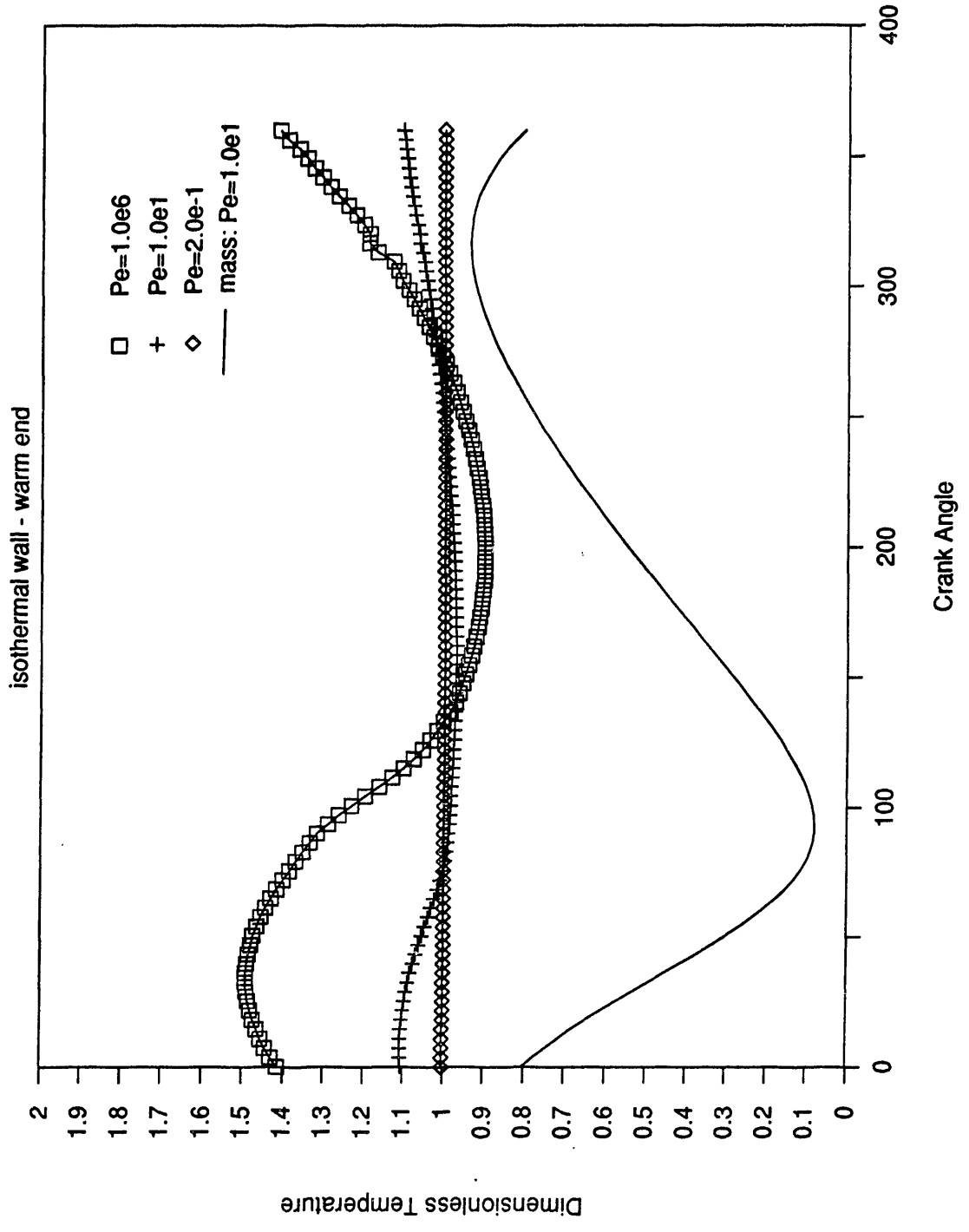

Figure 11. Temperature Variation for Warm Cylinder with Isothermal Wall

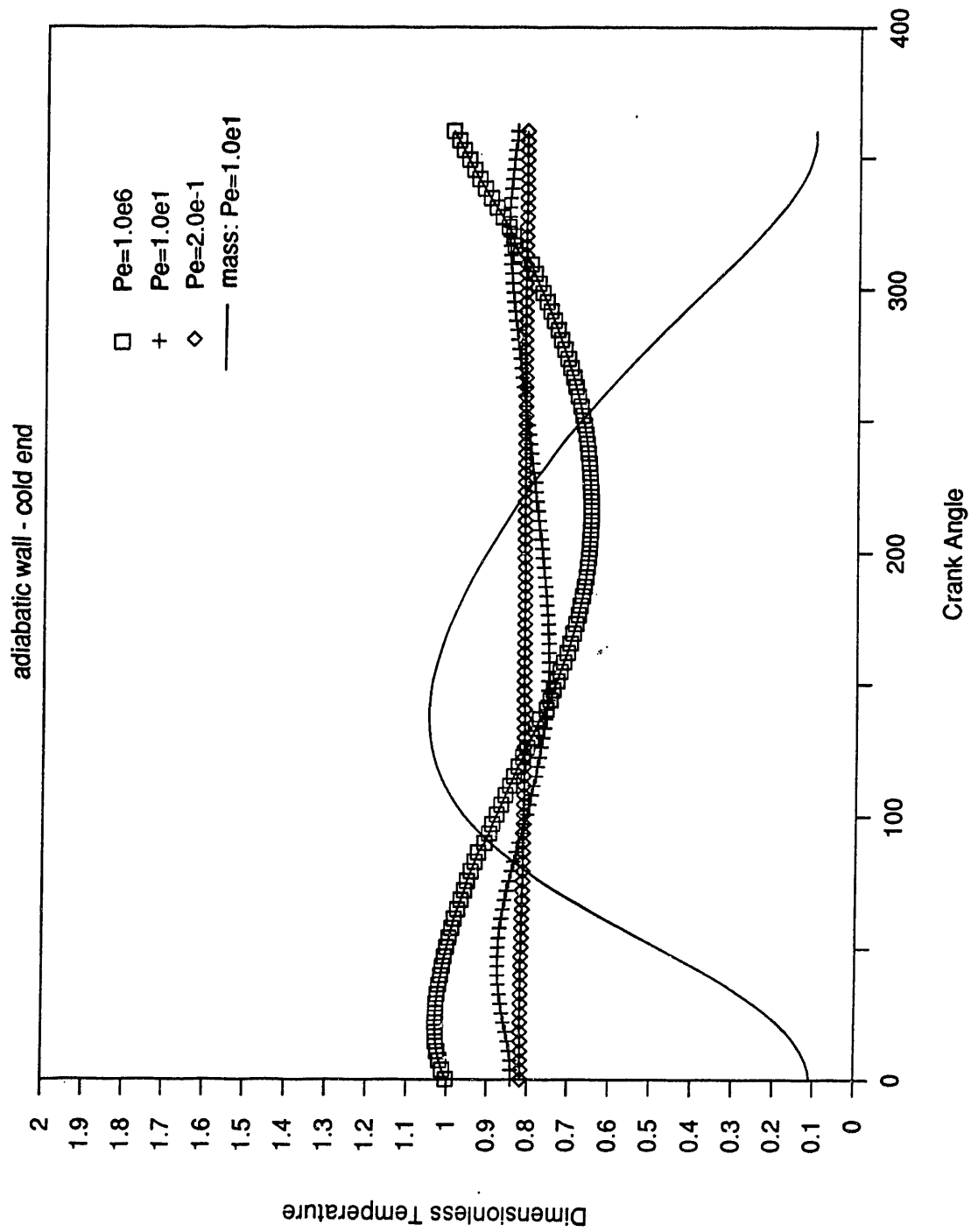

Figure 12. Temperature Variation for Cold Cylinder with Adiabatic Wall

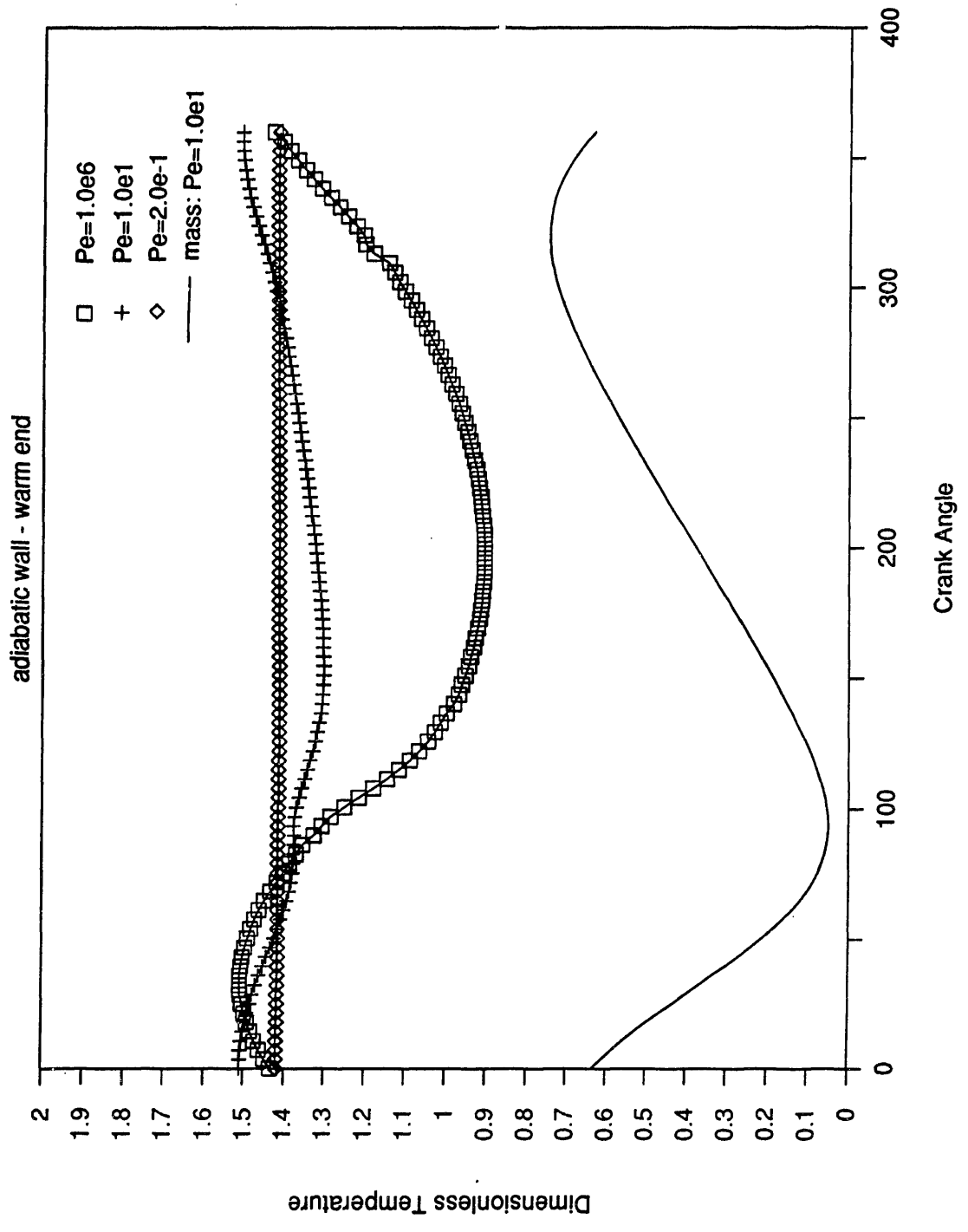

Figure 13. Temperature Variation for Warm Cylinder with Adiabatic Wall

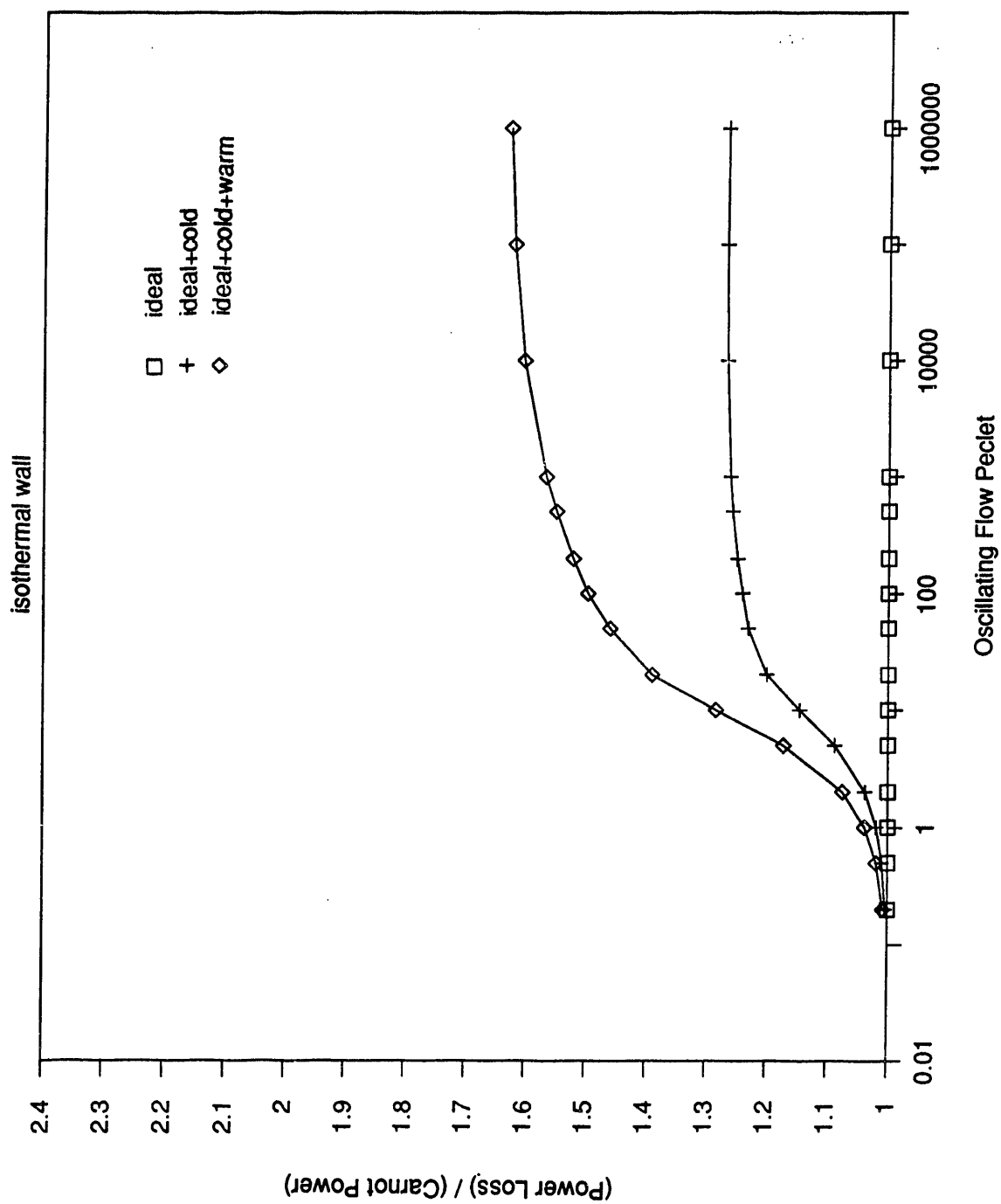

Figure 14. Relative Power Loss with Isothermal Wall

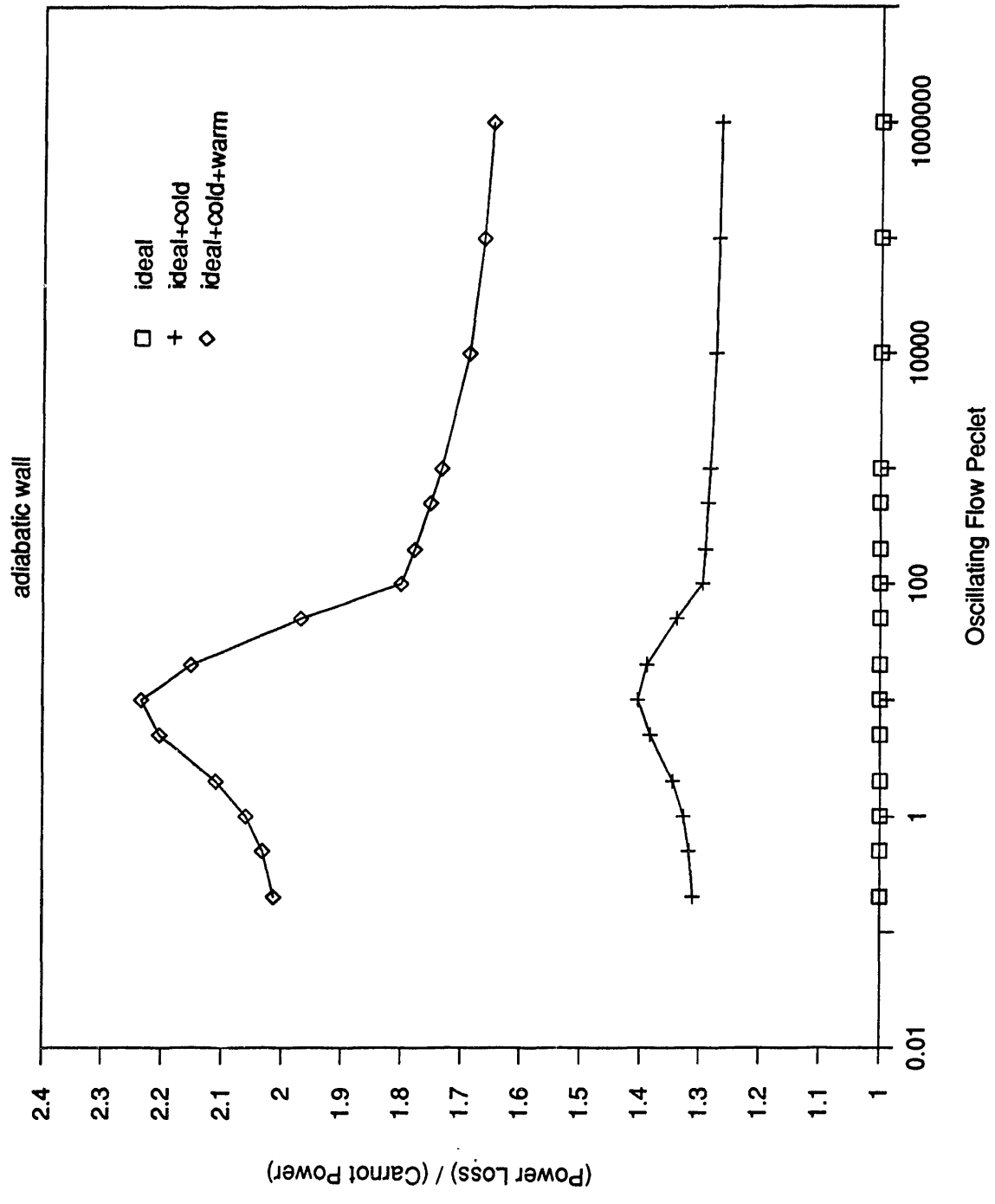

Figure 15. Relative Power Loss with Adiabatic Wall

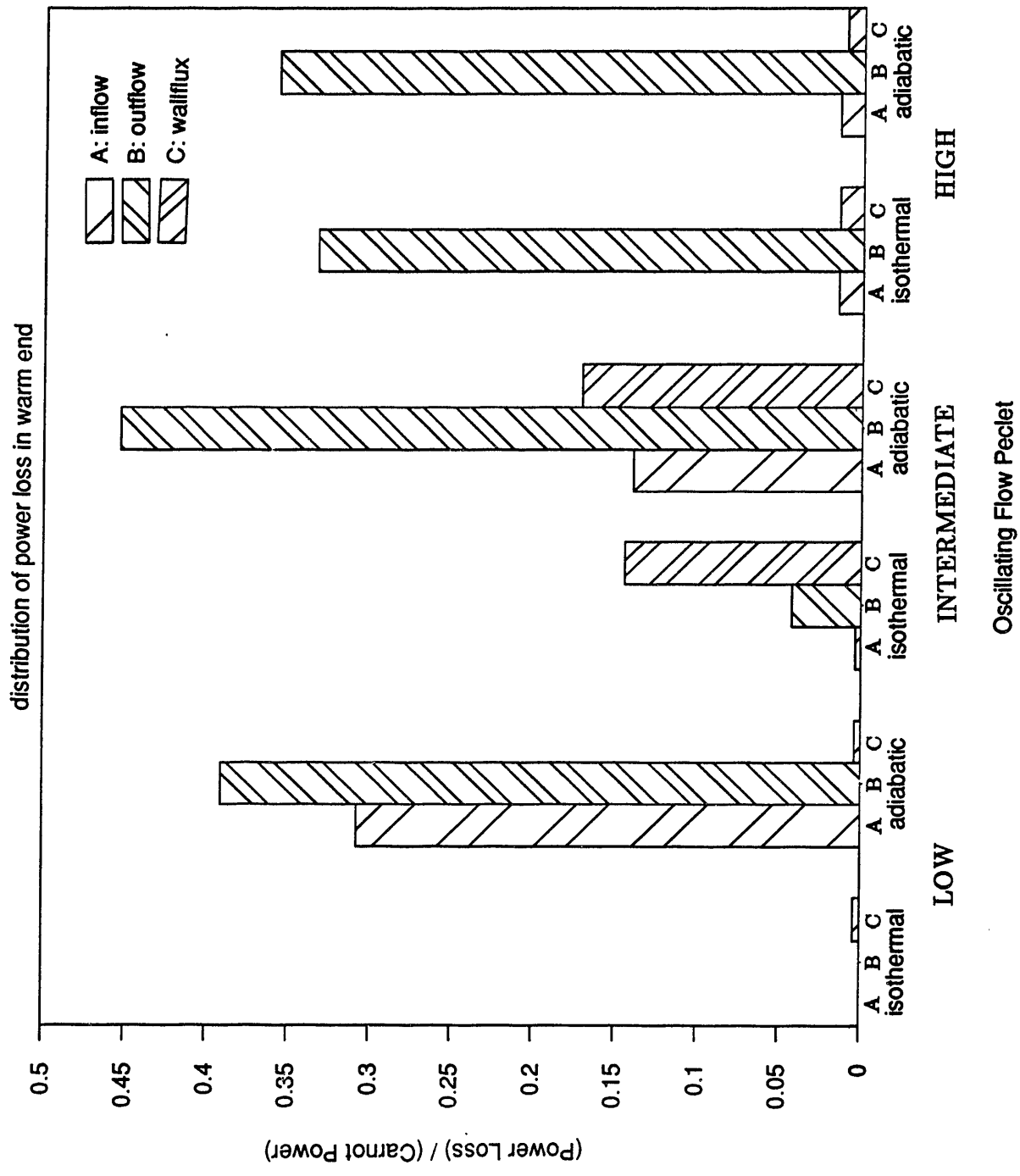

Figure 16. Trends in Relative Power Loss

Appendix A
STIRLING MODELS

A.1 Introduction

In this appendix the basic equations for the Stirling analysis will be derived. To reduce the number of dependent variables, the equations are converted to dimensionless terms. Energy and mass conservation plus the ideal gas relation are applied to control volumes resulting in differential equations for pressure and mass. These equations are an extension of the adiabatic model developed by Rios (1969).

A.2 Dimensionless Values

The model equations are written in dimensionless form to reduce the number of parameters and dependent variables. Properties are combined with the ideal gas constant and reference values of pressure, mass and volume to produce a pure number. For each work space, variables are nondimensionalized with respect to the volume amplitude V_a of the piston displacement. The volume amplitude of a cylinder space is defined as one half the difference between the maximum and minimum volumes. The corresponding value for temperature is the temperature of the adjacent heat exchanger denoted by T^*. The temperatures of the heater and cooler in this model taken to be fixed. The maximum value of pressure p_{max} over a cycle is used as a reference. Because the model assumes no pressure drop (explained below) in the heat transfer components, the maximum pressure is the same for both compression and expansion spaces.

The primary variables describing the work spaces are pressure, mass, volume and heat transfer. Pressure p and volume V are simply normalized respectively with p_{max} and V_a. The ideal gas law is used to normalize mass m. The product $p_{max} V_a$ has units of energy and is combined with the heat transfer variable Q. The primary dimensionless variables, denoted by calligraphic lettering, are defined as follows:

$$\mathcal{P} \equiv \frac{p}{p_{max}}, \qquad \mathcal{V} \equiv \frac{V}{V_a}, \qquad \mathcal{M} \equiv \frac{m R T^*}{p_{max} V_a}, \qquad \mathcal{Q} \equiv \frac{Q}{p_{max} V_a}. \tag{A.1}$$

Other properties may be obtained from the above variables. Gas temperature T is normalized with respect to heat exchanger temperature T^* and is equivalent to the

product of pressure and volume divided by mass:

$$\mathcal{T} \equiv \frac{T}{T^*} = \frac{\mathcal{P}\,\mathcal{V}}{\mathcal{M}}. \tag{A.2}$$

The temperature derivative $d\mathcal{T}$ may be determined by differentiating the natural logarithm of the right-hand side in Equation (A.2):

$$\begin{aligned} d\left(\ln\frac{\mathcal{P}\,\mathcal{V}}{\mathcal{M}}\right) &= d\left(\ln\mathcal{P} + \ln\mathcal{V} - \ln\mathcal{M}\right), \\ d\left(\frac{\mathcal{P}\,\mathcal{V}}{\mathcal{M}}\right) &= \left(\frac{d\mathcal{P}}{\mathcal{P}} + \frac{d\mathcal{V}}{\mathcal{V}} - \frac{d\mathcal{M}}{\mathcal{M}}\right)\left(\frac{\mathcal{P}\,\mathcal{V}}{\mathcal{M}}\right). \end{aligned} \tag{A.3}$$

Work W is normalized to the product $p_{max}\,V_a$. The dimensionless work work per cycle $\mathcal{W}$ is found by the cyclic integration of dimensionless pressure with respect to dimensionless volume:

$$\mathcal{W} \equiv \frac{\oint W}{p_{max}\,V_a} = \oint \mathcal{P}\,d\mathcal{V}. \tag{A.4}$$

Entropy s, which has units of energy over time, may be normalized as follows:

$$\mathcal{S} \equiv s \cdot \frac{T^*}{p_{max}\,V_a}. \tag{A.5}$$

Using a bulk flow model, the change in entropy of the gas within the control volume is dependent on mass flow, heat transfer and entropy generation:

$$ds = \sum(s\,dm)_{in} - \sum(s\,dm)_{out} + \sum\frac{dQ}{T} + ds_{gen}, \tag{A.6}$$

where T is the temperature at which each heat transfer occurs. Under steady state conditions, the cyclic integral of the change in entropy is zero. For a control volume with one inflow port, one outflow port and one heat reservoir, the entropy generated per cycle is

$$\oint ds_{gen} = \oint(s\,dm)_{out} - \oint(s\,dm)_{in} - \oint\frac{dQ}{T}. \tag{A.7}$$

The entropy of an ideal gas with constant specific heats may be expressed to a reference value s_o:

$$s - s_o = -R\,\ln\frac{p}{p_o} + c_p\ln\frac{T}{T_o}. \tag{A.8}$$

With Equations (A.7) and (A.8), the dimensionless entropy per cycle may be derived:

$$S_{gen} = \oint_{out} (-\ln P + \frac{\gamma}{\gamma - 1} \ln T) dM - \oint_{in} (-\ln P + \frac{\gamma}{\gamma - 1} \ln T) dM - \oint \frac{d\mathcal{Q}}{T}. \quad (A.9)$$

A.3 Assumptions

The assumptions used in this analysis are the same as those in Rios's with the exception that the cylinders are no longer adiabatic:

1. The gas in each cylinder is instantaneously and perfectly mixed.
2. Losses due to shuttle heat leak and axial conduction in the cylinders are neglected.
3. All the interactions in the heat exchange components are reversible having
   a. no pressure drop,
   b. no gas to wall temperature difference,
   c. no axial conduction.
4. The temperature profile of the heat transfer components is constant with time.
5. The system is one dimensional with temperature uniform along any plane perpendicular to the axis.
6. The working fluid is an ideal gas.

A.4 Model with Heat Transfer

The components of the Stirling refrigerator are shown in Figure 1. They are divided into three primary control volumes. The compression and expansion spaces at either end are each considered to be active volumes in which piston work is performed. The third volume, a lumped dead volume, consists of the dead volume in both heat exchangers and in the regenerator plus any dead volume presiding in the working volumes. In the derivation below, energy conservation is applied to the cylinders to obtain differential equations for mass. With assumption (4) above, the mass of the lumped dead volume may be expressed as a function of pressure without applying conservation of energy.

The ideal gas law is

$$pV = mRT, \tag{A.10}$$

where p is pressure, V is volume, m is mass, and T is temperature. The ideal gas constant R may be expressed as the difference between the specific heat at constant pressure c_p and the specific heat at constant volume c_v:

$$R = c_p - c_v. \tag{A.11}$$

The ratio of specific heats γ is defined as

$$\gamma = c_p/c_v. \tag{A.12}$$

The first law of thermodynamics is applied to the two adiabatic volumes:

$$dE = dQ - dW + h\,dm. \tag{A.13}$$

Note that by assigning a constant temperature gradient to the regenerator and heat exchangers, applying the first law to the dead volume is avoided. Energy E, work W and enthalpy h are simplified in the energy equation to produce

$$c_v\,d(mT) = dQ - p\,dV + c_p\,T\,dm. \tag{A.14}$$

Substituting Equation (A.10) into the left side of Equation (A.13) gives

$$\begin{aligned} c_v\,d\left(\frac{pV}{R}\right) &= dQ - p\,dV + c_p\,T\,dm, \\ \frac{c_v}{R}\,(p\,dV + V\,dp) &= dQ - p\,dV + c_p\,T\,dm, \\ \left(\frac{c_v}{R} + 1\right)\,p\,dV + \frac{c_v}{R}\,V\,dp - dQ &= c_p\,T\,dm. \end{aligned} \tag{A.15}$$

Substitution with Equations (A.11) and (A.12) and solving for the change in mass yields

$$dm = \frac{1}{RT}\left(p\,dV + \frac{1}{\gamma}\,V\,dp - \frac{\gamma - 1}{\gamma}\,dQ\right). \tag{A.16}$$

When gas is moving into the active volumes, it is at the same temperature as the adjacent heat exchanger wall. Denoting the heat exchanger wall temperature by T^*, Equation (A.16) becomes

$$dm = \frac{1}{RT^*}\left(p\,dV + \frac{1}{\gamma}V\,dp\right) \qquad \text{for } dm > 0. \tag{A.17}$$

When gas is flowing out of the control volume, the ideal gas law (A.10) may be used to further reduce Equation (A.16) by eliminating temperature:

$$dm = \frac{m}{pV}\left(p\,dV + \frac{1}{\gamma}V\,dp - \frac{\gamma-1}{\gamma}dQ\right) \qquad \text{for } dm < 0. \tag{A.18}$$

The heat exchanger temperature T^* may be eliminated by nondimensionalization. Rewriting the mass derivative equations (A.17) and (A.18) in terms of the definitions in Equation (A.1):

$$\begin{aligned} d\mathcal{M} &= \mathcal{P}\,d\mathcal{V} + \frac{1}{\gamma}\mathcal{V}\,d\mathcal{P} - \frac{\gamma-1}{\gamma}d\mathcal{Q} && \text{for } d\mathcal{M} > 0, \\ d\mathcal{M} &= \frac{\mathcal{M}}{\mathcal{P}\mathcal{V}}\left(\mathcal{P}\,d\mathcal{V} + \frac{1}{\gamma}\mathcal{V}\,d\mathcal{P} - \frac{\gamma-1}{\gamma}d\mathcal{Q}\right) && \text{for } d\mathcal{M} < 0. \end{aligned} \tag{A.19}$$

To obtain a differential equation for pressure, mass continuity is used. Since the entire system is closed, total mass is conserved:

$$m_{total} = m_c + m_d + m_w = constant. \tag{A.20}$$

Upon differentiation,

$$dm_c + dm_d + dm_w = 0. \tag{A.21}$$

where subscript c denotes the cold end, d the dead volume, and w the warm end. Introduction of another dimensionless term may be used to eliminate the dead volume mass m_d:

$$m_d \equiv \mathcal{V}_d \frac{p V_{aw}}{R T_w^*}, \tag{A.22}$$

where V_{aw} is the warm volume amplitude. The reduced dead volume $\mathcal{V}_d$ may considered as the ratio of the volume due to m_d under warm end conditions (at pressure p and

temperature T_w^*) to the average warm volume V_{aw}. The parameter is related to the amount of gas that enters the dead volume from one cylinder but does not travel all the way to the other cylinder. Differentiating Equation (A.22) gives the change in the dead volume mass:

$$dm_d = \mathcal{V}_d \frac{V_{aw}}{R\,T_w^*}\,dp. \tag{A.23}$$

Mass continuity may now be converted into dimensionless form:

$$\mathcal{R}_{vt}\,d\mathcal{M}_c + \mathcal{V}_d\,d\mathcal{P} + d\mathcal{M}_w = 0, \tag{A.24}$$

where the displaced mass ratio is defined as:

$$\mathcal{R}_{vt} \equiv \frac{V_{ac}}{V_{aw}}\,\frac{T_w^*}{T_c^*}. \tag{A.25}$$

The displaced mass ratio represents the ratio of the cold mass to the warm mass contained in their respective half volumes at their respective heat exchanger temperatures.

Since there are two active volumes (a cold cylinder and a warm cylinder) that can have mass flowing either in or out, there are four possible combinations of mass flow. By introducing Equation (A.19) for the case when $d\mathcal{M}_c > 0$ and $d\mathcal{M}_w > 0$, a differential equation for pressure may be obtained .

$$\begin{aligned}
\mathcal{R}_{vt}\left(\mathcal{P}\,d\mathcal{V}_c + \frac{1}{\gamma}\,\mathcal{V}_c\,d\mathcal{P} - \frac{\gamma-1}{\gamma}\,d\mathcal{Q}_c\right) + \mathcal{V}_d d\mathcal{P} + \left(\mathcal{P}\,d\mathcal{V}_w + \frac{1}{\gamma}\,\mathcal{V}_w\,d\mathcal{P} - \frac{\gamma-1}{\gamma}\,d\mathcal{Q}_w\right) &= 0,\\
\mathcal{P}\left(\mathcal{R}_{vt}\,d\mathcal{V}_c + d\mathcal{V}_w\right) + \frac{1}{\gamma}\left(\mathcal{R}_{vt}\,\mathcal{V}_c + \mathcal{V}_w + \gamma\,\mathcal{V}_d\right)d\mathcal{P} - \frac{\gamma-1}{\gamma}\left(\mathcal{R}_{vt}\,d\mathcal{Q}_c + d\mathcal{Q}_w\right) &= 0,\\
-\gamma\,\mathcal{P}\,\frac{\mathcal{R}_{vt}\,d\mathcal{V}_c + d\mathcal{V}_w}{\mathcal{R}_{vt}\,\mathcal{V}_c + \mathcal{V}_w + \gamma\,\mathcal{V}_d} + (\gamma-1)\,\frac{\mathcal{R}_{vt}\,d\mathcal{Q}_c + d\mathcal{Q}_w}{\mathcal{R}_{vt}\,\mathcal{V}_c + \mathcal{V}_w + \gamma\,\mathcal{V}_d} &= d\mathcal{P}.
\end{aligned} \tag{A.26}$$

Pressure equations for the three other mass flow combinations may be derived in the same fashion. The result for each of the four mass flow combinations are summarized below:

$$
\begin{aligned}
dP &= -\gamma \frac{\mathcal{R}_{vt}\, P\, d\mathcal{V}_c + P\, d\mathcal{V}_w}{\mathcal{R}_{vt}\, \mathcal{V}_c + \mathcal{V}_w + \gamma\, \mathcal{V}_d} + \quad (\gamma - 1) \frac{\mathcal{R}_{vt}\, d\mathcal{Q}_c + d\mathcal{Q}_w}{\mathcal{R}_{vt}\, \mathcal{V}_c + \mathcal{V}_w + \gamma\, \mathcal{V}_d} \qquad \text{for } \begin{matrix} dM_c > 0 \\ dM_w > 0 \end{matrix}, \\
dP &= -\gamma \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{V}_c}{\mathcal{V}_c} + M_w \dfrac{d\mathcal{V}_w}{\mathcal{V}_w}}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} + (\gamma - 1) \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{Q}_c}{P\mathcal{V}_c} + M_w \dfrac{d\mathcal{Q}_w}{P\mathcal{V}_w}}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} \text{ for } \begin{matrix} dM_c < 0 \\ dM_w < 0 \end{matrix}, \\
dP &= -\gamma \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{V}_c}{\mathcal{V}_c} + P\, d\mathcal{V}_w}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \mathcal{V}_w + \gamma\, \mathcal{V}_d} + \quad (\gamma - 1) \frac{\mathcal{R}_{vt}\, M_c \dfrac{d\mathcal{Q}_c}{P\mathcal{V}_c} + d\mathcal{Q}_w}{\mathcal{R}_{vt} \dfrac{M_c}{P} + \mathcal{V}_w + \gamma\, \mathcal{V}_d} \qquad \text{for } \begin{matrix} dM_c < 0 \\ dM_w > 0 \end{matrix}, \\
dP &= -\gamma \frac{\mathcal{R}_{vt}\, P\, d\mathcal{V}_c + M_w \dfrac{d\mathcal{V}_w}{\mathcal{V}_w}}{\mathcal{R}_{vt}\, \mathcal{V}_c + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} + \quad (\gamma - 1) \frac{\mathcal{R}_{vt}\, d\mathcal{Q}_c + M_w \dfrac{d\mathcal{Q}_w}{P\mathcal{V}_w}}{\mathcal{R}_{vt}\, \mathcal{V}_c + \dfrac{M_w}{P} + \gamma\, \mathcal{V}_d} \qquad \text{for } \begin{matrix} dM_c > 0 \\ dM_w < 0 \end{matrix}.
\end{aligned}
\tag{A.27}
$$

For the adiabatic case, the heat transfer term $d\mathcal{Q}$ is zero. This simplified version is the one that Rios developed.

Appendix B
HEAT TRANSFER MODEL

B.1 Introduction

In this appendix, the relation describing heat transfer from wall to gas is derived. In order to maintain simplicity, the heat transfer is described in terms of pure numbers related to geometry, speed and gas properties. The heat transfer relation, when combined with the pressure and mass equations, fully describe the work spaces of the Stirling system.

B.2 Parameter Definitions

Because the Nusselt-Peclet correlations (2.8) and (2.9) are based entire cycles and not local conditions, Peclet should be found in terms of average conditions. To obtain a better correlation, an oscillating flow Peclet number is used in place of the conventional Peclet number. The condition assumed to represent the approximate average will be when the piston is located at mid-stroke, the pressure is at the cycle maximum, and the temperature equals that of the heat exchanger. The conventional Peclet number is a pure number that represents the ratio of heat capacity to axial heat conduction:

$$Pe = \frac{velocity \times length}{thermal\ diffusivity}. \tag{B.1}$$

For reciprocating machinery the characteristic length is the hydraulic diameter. The hydraulic is defined as four times fluid volume divided by heat transfer area. The piston crown, the cylinder head, and the side wall of the cylinder comprise the heat transfer area. When the piston is halfway through its stroke, the hydraulic diameter becomes

$$\begin{aligned} D_h &= \frac{4V}{A}, \\ &= \frac{4\left(\pi D^2/4\right)\left(L_s/2\right)}{2\left(\pi D^2/4\right) + \left(\pi D\right)\left(L_s/2\right)}, \\ &= \frac{D}{1 + D/L_s}. \end{aligned} \tag{B.2}$$

where A is area, L_s is the stroke length of the piston and D is the cylinder diameter. The characteristic mean velocity $\bar{v}$ is an angle averaged velocity:

$$\bar{v} = \frac{1}{2\pi}\frac{2L_s}{1/\omega},$$

$$\bar{v} = \frac{\omega L_s}{\pi}, \tag{B.3}$$

where ω is the angular frequency of compression and expansion. The conventional Peclet number then becomes

$$Pe = \frac{\omega L_s D_h}{\pi \alpha}, \tag{B.4}$$

where α is the thermal diffusivity of the gas and is equal to $k/\rho c_p$. The oscillating flow Peclet number Pe_ω results when Pe is multiplied by D_h/L_s. A factor of $\pi/4$ is added so that the expression is consistent with established conventions:

$$\begin{aligned} Pe_\omega &= \frac{\pi}{4} Pe \frac{D_h}{L_s}, \\ &= \frac{\omega D_h^2}{4\alpha}, \\ &= \frac{\omega D_h^2}{4} \frac{\rho c_p}{k}. \end{aligned} \tag{B.5}$$

When expanded, Pe_ω becomes a function of hydraulic diameter, frequency, and gas properties. Even though gas density will change with pressure and temperature, density may be assumed to be constant for the purposes of observing trends affected by gas-to-wall heat transfer. Average gas density ρ is approximately that at maximum pressure and at the temperature of the adjacent heat exchanger

$$\rho = \frac{p_{max}}{R T^*}. \tag{B.6}$$

This approximation is consistent with the approximations used by Kornhauser in developing his expressions for Nu_C as a function of Pe_ω. By inserting Equation (B.6) into (B.5), oscillating flow Peclet then becomes

$$Pe_\omega = \frac{\omega D_h^2 p_{max}}{4 k T^*} \frac{\gamma}{\gamma - 1}. \tag{B.7}$$

B.3 Equation Development

The complex heat transfer model in this analysis is based upon a complex Nusselt number Nu_C which allows for prediction of heat transfer phase shift. Kornhauser (1989) writes Newton's law of convection in terms of complex heat transfer,

temperature and Nusselt number:

$$\frac{dQ_C}{dt} = -h_C\, A\,(T_C - T_{wall}),$$
$$\frac{dQ_C}{dt} = -A\,\frac{k}{D_h}\, Nu_C\,(T_C - T_{wall}). \qquad (B.8)$$

where h_C is a complex convective heat transfer coefficient. The negative sign indicates that heat transfer from the wall to the gas is positive. The wall temperature T_{wall} is assumed to be constant and, therefore has no complex part. By expressing complex temperature as a sinusoidal variation, the real part of heat transfer may be simplified to

$$\frac{dQ}{dt} = -A\frac{k}{D_h}\left(Nu_R\,(T - T_{wall}) + \frac{Nu_I}{\omega}\,\frac{dT}{dt}\right), \qquad (B.9)$$

where Nu_R and Nu_I are the real and imaginary parts of Nu_C, T and T_{wall} are respectively the mean bulk gas temperature and wall temperature, ω is the angular frequency of the cycle, k is the thermal conductivity of gas, and D_h is the mean hydraulic diameter of the space. The Nusselt number is strongly dependent on Peclet number. Equation (B.9) includes the phase shift that exists between heat transfer and temperature difference. The heat transfer area includes the piston crown and the cylinder head as well as the side wall of the cylinder. The side wall area is the product of instantaneous length and cylinder perimeter. Dividing the instantaneous volume by the cross sectional area will give the instantaneous length. The total area at any point in time is

$$A = \left(\frac{V}{\pi D^2/4}\right)(\pi D) + 2\left(\pi D^2/4\right),$$
$$= \frac{4V}{D} + \frac{\pi D^2}{2}. \qquad (B.10)$$

After substituting for hydraulic diameter and area, the heat transfer equation (B.9) becomes

$$\frac{dQ}{dt} = -\left(\frac{4V}{D} + \frac{\pi D^2}{2}\right)\frac{1 + D/L_s}{D}\,k\left[Nu_R\,(T - T_{wall}) + \frac{Nu_I}{\omega}\,\frac{dT}{dt}\right]. \qquad (B.11)$$

Heat transfer may be nondimensionalized by the term $p_{max}\,V_a$ which has units of energy. Since volume will be an input that is a function of the crank angle θ, it is appropriate to take the heat transfer derivative with respect to θ:

$$\frac{dQ}{d\theta} = \frac{dQ}{dt}\,\frac{dt}{d\theta} = \frac{dQ/dt}{p_{max}\,V_a}\,\frac{1}{\omega}. \qquad (B.12)$$

A dimensionless equation results when temperature is normalized to the temperature of the adjacent heat exchanger temperature. Algebraic manipulation will transform the complex heat transfer relation into

$$\frac{d\mathcal{Q}}{d\theta} = -\frac{\gamma}{\gamma-1}\frac{\mathcal{V}+D/L_s}{1+D/L_s}\frac{1}{Pe_\omega}\left[Nu_R\left(\mathcal{T}-\mathcal{T}_{wall}\right)+Nu_I\frac{d\mathcal{T}}{d\theta}\right]. \qquad (B.13)$$

Equation (B.13) can also be written in terms of dimensionless mass, pressure and volume using the definitions in (A.2) and (A.3):

$$d\mathcal{Q} = -\mathcal{H}\left[Nu_R\left(\frac{\mathcal{P}\mathcal{V}}{\mathcal{M}}-\mathcal{T}_{wall}\right)d\theta + Nu_I\left(\frac{d\mathcal{P}}{\mathcal{P}}+\frac{d\mathcal{V}}{\mathcal{V}}-\frac{d\mathcal{M}}{\mathcal{M}}\right)\left(\frac{\mathcal{P}\mathcal{V}}{\mathcal{M}}\right)\right]. \qquad (B.14)$$

where $\mathcal{H}$ is an abbreviation for

$$\mathcal{H} = \frac{\gamma}{\gamma-1}\frac{\mathcal{V}+D/L_s}{1+D/L_s}\frac{1}{Pe_\omega}. \qquad (B.15)$$

Equation (B.14) is the desired nondimensional expression for heat transfer in the working volumes. We will observe that two independent dimensionless parameters, Pe_ω and D/L_s, are necessary to describe the heat transfer in the working volumes, since the two Nusselt numbers are functions of Peclet. The Peclet number reflects the rate of heat transfer in relation to the operating speed of the cooler. The bore-to-stroke ratio is associated only with the geometry of the working volume.

Appendix C
COMPUTER CODE

D.1 Introduction

This appendix lists the computer routines used for running the cycle simulations. The code was written to be compiled with the Microsoft C 5.0 compiler.

D.2 Header file

The header file contains all the global definitions, constants and external variables. It is included in all the other modules.

```
#include <math.h>
#include <stdio.h>
#include <string.h>

#define PI              3.14159265358979
#define SIZE            200                 /* size of data array */
#define NODES           2                   /* number of cylinders */
#define VARS            9                   /* number of variables to be calculated for each node */
#define sqr(x)  ((x) * (x))                 /* macro for square function */

enum node       {H,L};
enum var        {V,P,M,T,Q,W,SI,SO,SW};

extern void     error       (char name[],char message[]);
extern void     correct     (int iter,double eps,double hmin,int maxsize);
extern void     odeint      (int i[],int j[],int nvar,double eps,double hmin,int maxsize);

/* derivative functions */
extern double entpy_infl (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double entpy_outf (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double entpy_wall (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double heat_deriv (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double mass_deriv (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double pres_deriv (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double temp_deriv (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double volm_deriv (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double work_deriv (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);

/* value functions */
extern double pres_value (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double temp_value (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
extern double volm_value (int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);
```

```
extern int    kount;                    /* counter for data array size */
extern double xp[SIZE];                 /* data array for storing crank angle */
extern double yp[NODES][VARS][SIZE];    /* data array for storing values */
extern double dyp[NODES][VARS][SIZE];   /* data array for storing derivatives */

/* array of pointers to value functions */
extern double (* f[NODES][VARS])(int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);

/* array of pointers to derivative functions */
extern double (*df[NODES][VARS])(int i,int j,double x,double y[NODES][VARS],double dy[NODES][VARS]);

extern double G;                        /* specific heat ratio */
extern double RVT;                      /* displaced mass ratio */
extern double VDTOT;                    /* reduced dead volume */
extern double phaselag;                 /* phaselag of warm piston to cold piston */
extern double PE[NODES];                /* Peclet number */
extern double DL[NODES];                /* bore-to-stroke ratio */
extern double NUR[NODES];               /* real part of complex Nusselt number */
extern double NUI[NODES];               /* imaginary part of complex Nusselt number */
extern double DEADVOLM[NODES];          /* cylinder dead volume */
extern double WALLTEMP[NODES];          /* cylinder wall temperature */
```

D.3 Driver code

The driver is used for inputting and outputting data files, assigning function pointers, initializing variables, and setting parameters. The subroutine input reads in the input parameters and data points from a previous simulation. All function assignments and any desired parameter changes are made in the assign subroutine. After initialization is completed, the correct subroutine is called to run the cyclic simulations. In adiabtemp, the cylinder wall temperature is adjusted iteratively until there is no net heat transfer to and from the cylinder wall. The output subroutine prints out the input parameters, the converged data points and the calculated performance to specified files.

```
#include "defs.h"

void   assign     (double peclet, double eps, double hmin, int maxsize);
void   adiabtemp  (FILE *ptr, int iter);
void   error      (char name[], char message[]);
void   fenter     (FILE *ptr, char name[], double *variable);
void   ienter     (FILE *ptr, char name[], int *variable);
void   input      (FILE *ptr1, FILE *ptr2);
void   output     (FILE *ptr1, FILE *ptr2, FILE *ptr3);
int    maxindex   (int i, int j);
int    minindex   (int i, int j);

int    iter;                    /* maximum number of cycle iterations */
int    maxsize;                 /* maximum number of data points */
double eps;                     /* error criterion for integration */
double hmin;                    /* minimum stepsize for integration */
double G, RVT, VDTOT, phaselag;
double PE[NODES], DL[NODES], NUR[NODES], NUI[NODES], DEADVOLM[NODES], WALLTEMP[NODES];

main(void)                              /* sample run */
{
   int i;
   FILE *ptr1, *ptr2, *ptr3, *ptr4;

   ptr1 = fopen("a1p6.dat", "r");
   ptr2 = fopen("a1p5.ful", "w");
   ptr3 = fopen("t1p5.ful", "w");
   ptr4 = fopen("cop.ful", "w");

   input(ptr1, ptr1);                   /* read parameters and data points from a1p6.dat */
   assign(1.0e5, eps, hmin, maxsize);   /* assign new Peclet number */
   correct(iter, eps, hmin, maxsize);   /* run cycle simulation with new inputs */
   output(ptr2, ptr2, ptr4);            /* output data to a1p5.ful */
   adiabtemp(ptr4, 10)                  /* calculate cycle with adiabatic temperature */
   output(ptr3, ptr3, ptr4);            /* output data to t1p5.ful */

   fclose(ptr1);   fclose(ptr2);    fclose(ptr3);   fclose(ptr4);
}
```

```
void assign(double pe, double ee, double hh, int mm)
{
   int i, j, k;

   PE[H] = pe;                                  /* assign new values to parameters */
   PE[L] = pe;
   eps = ee;
   hmin = hh;
   maxsize = mm;
   for (i=0; i<NODES; i++){                     /* assign function pointers */
      f[i][V]  = volm_value;
      df[i][V] = volm_deriv;
      f[i][P]  = NULL;
      df[i][P] = pres_deriv;
      f[i][M]  = NULL;
      df[i][M] = mass_deriv;
      f[i][T]  = temp_value;
      df[i][T] = temp_deriv;
      f[i][Q]  = NULL;
      df[i][Q] = heat_deriv;
      f[i][W]  = NULL;
      df[i][W] = work_deriv;
      f[i][SI]  = NULL;
      df[i][SI] = entpy_infl;
      f[i][SO]  = NULL;
      df[i][SO] = entpy_outf;
      f[i][SW]  = NULL;
      df[i][SW] = entpy_wall;
   }
   f[L][P]  = pres_value;

   for (i=0; i<NODES; i++){                     /* calculate complex Nusselt number */
      if (PE[i] >= 100){                        /* from Kornhauser's correlation */
         NUR[i] = 0.56 * pow(PE[i], 0.69);
         NUI[i] = NUR[i];
      }
      else{                                     /* from Lee's correlation */
         double u, a, b;
         double tanhu, coef;

         u = sqrt(PE[i] / 8.0);
         tanhu = tanh(u);
         a = sin(u) * cos(u) / sqr(cosh(u));
         b = sqr(cos(u)) + sqr(sin(u)) * sqr(tanhu);
         coef = 4.0*u*sqrt(2.0 * PE[i]);
         coef /= sqr(2*u*b-tanhu-a) + sqr(tanhu - a);
         NUR[i] = coef * (u*b*(tanhu - a));
         NUI[i] = coef * (u*b*(tanhu + a) - sqr(tanhu) - sqr(a));
      }
   }
}
```

```
/* enter variables of type double */
void fenter(FILE *ptr, char name[], double *variable)
{
   char s[12];

   fscanf(ptr, "%s", s);
   if (strcmp(name, s) == 0)
      fscanf(ptr, "%*[ =\t]%lf%*[^\n]\n", variable);
   else
      error("fenter", strcat(name, " not read"));
}

/* enter variables of type int */
void ienter(FILE *ptr, char name[], int *variable)
{
   char s[12];

   fscanf(ptr, "%s", s);
   if (strcmp(name, s) == 0)
      fscanf(ptr, "%*[ =\t]%i%*[^\n]\n", variable);
   else
      error("ienter", strcat(name, " not read"));
}

/* print error message and function in which the error occured */
void error(char name[], char message[])
{
   fprintf(stderr, "\n%s:  %s\n", name, message);
   exit(1);
}

/* return index of data point with maximum value during a cycle */
int maxindex(int i, int j)
{
   int k, kmax;

   kmax = 1;
   for (k=1; k<=kount; k++)
      if (yp[i][j][k] > yp[i][j][kmax])
          kmax = k;
   return(kmax);
}

/* return index of data point with minimum value during a cycle */
int minindex(int i, int j)
{
   int k, kmin;

   kmin = 1;
   for (k=1; k<=kount; k++)
      if (yp[i][j][k] < yp[i][j][kmin])
          kmin = k;
   return(kmin);
}
```

```
/* determine adiabatic wall temperature of cold and warm cylinders by iteration */
void adiabtemp(FILE *ptr, int adiabiter)
{
   double q1[NODES], q2[NODES], qlow[NODES], qhigh[NODES];
   double tlow[NODES], thigh[NODES], increment=0.15;
   int    i, n;

   n = 0;
   for (i=0; i<NODES; i++){            /* initialize guesses */
      q1[i] = yp[i][Q][kount];
      thigh[i] = 0.0;
      tlow[i] = 0.0;
   }
   for (;;){
      for (i=0; i<NODES; i++)
         q2[i] = yp[i][Q][kount];
      fprintf(ptr, "%6.4e %6.4e ", PE[H], PE[L]);
      fprintf(ptr, "%6.4f %6.4f ", WALLTEMP[H], WALLTEMP[L]);
      fprintf(ptr, "%6.4f %6.4f\n", q2[H], q2[L]);
      printf("%6.4f %6.4f ", WALLTEMP[H], WALLTEMP[L]);
      printf("%6.4f %6.4f\n", q2[H], q2[L]);
      if ((fabs(q2[H])<0.001 && fabs(q2[L])<0.001) || n >= adiabiter)
         break;
      for (i=0; i<NODES; i++){         /* reassign upper limit if net heat transfer postive */
         if (q2[i] > 0){
            thigh[i] = WALLTEMP[i];
            qhigh[i] = q2[i];
         }
         if (q2[i] < 0){               /* reassign lower limit if net heat transfer negative */
            tlow[i] = WALLTEMP[i];
            qlow[i] = q2[i];
         }
         if (q2[i]/q1[i] < 0){         /* determine new wall temperature by interpolation
                                          if sign of heat transfer changes */
            WALLTEMP[i]  = - qlow[i]/(qhigh[i]-qlow[i])*(thigh[i]-tlow[i]);
            WALLTEMP[i] += tlow[i];
         }
         if (q2[i]/q1[i] > 0)          /* if sign of heat transfer does not change */
            if (thigh[i] != 0.0  && tlow[i] != 0.0)
               WALLTEMP[i] = (thigh[i] + tlow[i]) / 2.0;
            else
               WALLTEMP[i] -= q2[i]/fabs(q2[i]) * increment;
            q1[i] = q2[i];
      }
      correct(iter, eps, hmin, maxsize);/* simulate cycle with new wall temperatures */
      n += 1;
   }
}
```

```
void input(FILE *ptr1, FILE *ptr2)
{
   int i, j, k;
   double x, y, dy, dummy;

   if (ptr1 != NULL){                                  /* scan parameters */
      fenter(ptr1, "PE[L]",         &PE[L]);
      fenter(ptr1, "PE[H]",         &PE[H]);
      fenter(ptr1, "DEADVOLM[L]",   &DEADVOLM[L]);
      fenter(ptr1, "DEADVOLM[H]",   &DEADVOLM[H]);
      fenter(ptr1, "WALLTEMP[L]",   &WALLTEMP[L]);
      fenter(ptr1, "WALLTEMP[H]",   &WALLTEMP[H]);
      fenter(ptr1, "DL[L]",         &DL[L]);
      fenter(ptr1, "DL[H]",         &DL[H]);
      fenter(ptr1, "G",             &G);
      fenter(ptr1, "RVT",           &RVT);
      fenter(ptr1, "VDTOT",         &VDTOT);
      fenter(ptr1, "phaselag",      &phaselag);
      fenter(ptr1, "eps",           &eps);
      fenter(ptr1, "hmin",          &hmin);
      ienter(ptr1, "iter",          &iter);
      ienter(ptr1, "maxsize",       &maxsize);

      phaselag *= PI / 180.0;                          /* convert degrees to rad */
   }
   if (ptr2 != NULL){
      fscanf(ptr2, "[^\n]\n");
      for (k=1; ; k++){                                /* scan variable values */
         fscanf(ptr2, "%lf  ", &x);
         xp[k] = x / 180.0 * PI;
         for (j=0; j<=W; j++)
            for (i=0; i<NODES; i++){
               fscanf(ptr2, "%lf  ", &y);
               yp[i][j][k] = y;
            }
         fscanf(ptr2, "\n");
         if (x >= 360.0) break;                        /* stop if end of cycle is reached */
      }
      kount = k;
      fscanf(ptr2, "\n");
      for (k=1; k<=kount; k++){                        /* scan variable derivatives */
         fscanf(ptr2, "%*f  ");
         for (j=0; j<=W; j++)
            for (i=0; i<NODES; i++){
               fscanf(ptr2, "%lf  ", &dy);
               dyp[i][j][k] = dy;
            }
         fscanf(ptr2, "\n");
      }
   }
}
```

```
void output(FILE *ptr1, FILE *ptr2, FILE *ptr3)
{
   int i, j, k;
   double pmaximum, rp, cop, tc=80.0, tw=300.0;
   double y[NODES][VARS], dy[NODES][VARS], sgen[NODES];

   /* normalize dimensionless variables with respect to maximum pressure
      except for volume and temperature */
   pmaximum = yp[H][P][maxindex(H, P)];
   for (i=0; i<NODES; i++)
      for (j=0; j<VARS; j++)
         for (k=1; k<=kount; k++)
            if (((enum var) j != T) && ((enum var) j != V))
               yp[i][j][k]  /= pmaximum;

   /* output parameters used for latest cycle simulation */
   if (ptr1 != NULL){
      fprintf(ptr1, "PE[L]        = %4.2e\n",        PE[L]);
      fprintf(ptr1, "PE[H]        = %4.2e\n",        PE[H]);
      fprintf(ptr1, "DEADVOLM[L]  = %4.2e\n",  DEADVOLM[L]);
      fprintf(ptr1, "DEADVOLM[H]  = %4.2e\n",  DEADVOLM[H]);
      fprintf(ptr1, "WALLTEMP[L]  = %9.5f\n",  WALLTEMP[L]);
      fprintf(ptr1, "WALLTEMP[H]  = %9.5f\n",  WALLTEMP[H]);
      fprintf(ptr1, "DL[L]        = %9.5f\n",        DL[L]);
      fprintf(ptr1, "DL[H]        = %9.5f\n",        DL[H]);
      fprintf(ptr1, "G            = %9.5f\n",           G);
      fprintf(ptr1, "RVT          = %9.5f\n",         RVT);
      fprintf(ptr1, "VDTOT        = %9.5f\n",       VDTOT);
      fprintf(ptr1, "phaselag     = %9.5f\n", phaselag * 180.0/PI);
      fprintf(ptr1, "eps          = %4.2e\n",         eps);
      fprintf(ptr1, "hmin         = %4.2e\n",        hmin);
      fprintf(ptr1, "iter         = %9i\n",          iter);
      fprintf(ptr1, "maxsize      = %9i\n",       maxsize);
      fprintf(ptr1, "\n");
   }

   if (ptr2 != NULL){
      for (k=1; k<=kount; k++){                        /* output variable values */
         fprintf(ptr2, "%12.8f  ", xp[k] * 180.0 / PI);
         for (j=0; j<=W; j++)
            for (i=0; i<NODES; i++)
               fprintf(ptr2, "%12.8f  ", yp[i][j][k]);
         fprintf(ptr2, "\n");
      }
      fprintf(ptr2, "\n");
      for (k=1; k<=kount; k++){                        /* output variable derivatives */
         fprintf(ptr2, "%12.8f  ", xp[k] * 180.0 / PI);
         for (j=0; j<=W; j++)
            for (i=0; i<NODES; i++)
               fprintf(ptr2, "%12.8f  ", dyp[i][j][k]);
         fprintf(ptr2, "\n");
      }
```

```
      fprintf(ptr2, "\n");
      for (k=1; k<=kount; k++){                          /* output entropy values */
         fprintf(ptr2, "%12.8f  ", xp[k] * 180.0 / PI);
         for (j=W+1; j<VARS; j++)
            for (i=0; i<NODES; i++)
               fprintf(ptr2, "%12.8f  ", yp[i][j][k]);
         fprintf(ptr2, "\n");
      }
      fprintf(ptr2, "\n");
      for (k=1; k<=kount; k++){                          /* output entropy derivatives */
         fprintf(ptr2, "%12.8f  ", xp[k] * 180.0 / PI);
         for (j=W+1; j<VARS; j++)
            for (i=0; i<NODES; i++)
               fprintf(ptr2, "%12.8f  ", dyp[i][j][k]);
         fprintf(ptr2, "\n");
      }
      fprintf(ptr2, "\n");

      /* calculate pressure ratio, coefficient of performance, and total
         entropy generation */
      rp = yp[H][P][maxindex(H,P)] / yp[H][P][minindex(H,P)];
      cop = (tc/tw) * RVT * yp[L][W][kount];
      cop = - cop / (yp[H][W][kount] + cop);
      cop *= (tw/tc) - 1.0;
      for (i=0; i<NODES; i++)
         sgen[i] = yp[i][SI][kount] + yp[i][SO][kount] + yp[i][SW][kount];
      fprintf(ptr2, "%6.4f  %6.4f  ", sgen[H], sgen[L]);
      fprintf(ptr2, "%6.4f  %6.4f\n", rp, cop);
      fprintf(ptr2, "\n");
      printf("%6.4f  %6.4f\n", rp, cop);
   }

   if (ptr3 != NULL && ptr1 != NULL){
      fprintf(ptr3, "%4.2f %4.2f %4.2f %4.2e ", RVT, VDTOT, DL[H], PE[H]);
      fprintf(ptr3, "%4.2f ", log10(DEADVOLM[H]));
      fprintf(ptr3, "%4.2f %4.2f ", log10(hmin), log10(eps));
      fprintf(ptr3, "%6.4f  %6.4f  ", WALLTEMP[H], WALLTEMP[L]);
      fprintf(ptr3, "%6.4f  %6.4f  ", yp[H][Q][kount], yp[L][Q][kount]);
      fprintf(ptr3, "%6.4f  %6.4f  ", yp[H][W][kount], yp[L][W][kount]);
      fprintf(ptr3, "%6.4f  %6.4f  ", yp[H][SI][kount], yp[L][SI][kount]);
      fprintf(ptr3, "%6.4f  %6.4f  ", yp[H][SO][kount], yp[L][SO][kount]);
      fprintf(ptr3, "%6.4f  %6.4f  ", yp[H][SW][kount], yp[L][SW][kount]);
      fprintf(ptr3, "%6.4f  %6.4f  ", sgen[H], sgen[L]);
      fprintf(ptr3, "%6.4f %6.4f\n", rp, cop);
   }
}
```

D.4 Cycling routine

In the subroutine `cycle`, cyclic integration is repeated until the property values at the end of the integration equal those at the beginning. Before the integrator `odeint` is called, variable that require integration are identified. Work, heat and entropy are initialized to zero. Unless convergence is reached, the cycle is repeated starting with improved starting values. The subroutine `correct` repeats `cycle` to double check convergence.

```
#include "defs.h"

void   cycle      (int iter, double eps, double hmin, int maxsize);
void   correct    (int iter, double eps, double hmin, int maxsize);

int    kount;
double xp[SIZE], yp[NODES][VARS][SIZE], dyp[NODES][VARS][SIZE];
double (*  f[NODES][VARS]) (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double (* df[NODES][VARS]) (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);

/* run cycle simulation a couple times to double checke convergence */
void correct(int iter, double eps, double hmin, int maxsize)
{
   int i, j, k, n;
   double mass, pres, temp, work;
   double mdiff, pdiff, tdiff, wdiff;

   printf("correct: PE=%f PE=%f", PE[H], PE[L]);
   n = 0;
   do {
      mass = yp[H][M][kount];
      pres = yp[H][P][kount];
      temp = yp[H][T][kount];
      work = yp[H][W][kount];
      n += 1;
      printf("\n%i-", n);
      cycle(iter, eps, hmin, maxsize);
      mdiff = fabs((yp[H][M][kount] - mass) / yp[H][M][kount]);
      pdiff = fabs((yp[H][P][kount] - pres) / yp[H][P][kount]);
      tdiff = fabs((yp[H][T][kount] - temp) / yp[H][T][kount]);
      wdiff = fabs((yp[H][W][kount] - work) / yp[H][W][kount]);
   } while ((mdiff>0.01 || pdiff>0.01 || tdiff>0.01 || wdiff>0.01) && n<iter);
   if (n==iter){
      printf("pdiff=%6.4f mdiff=%6.4f ", pdiff, mdiff);
      printf("tdiff=%6.4f wdiff=%6.4f ", tdiff, wdiff);
   }
   printf("\n");
}
```

```
/* repeat cycle simulation until end values equal initial values */
void cycle(int iter, double eps, double hmin, int maxsize)
{
   int i, j, k, n, nvar;
   int g[VARS*NODES], h[VARS*NODES];
   double pdiff, mdiff, tdiff, wdiff, work;

   /* search for variables requiring numerical integration and note
      their array indexes */
   nvar = 0;
   for (j=0; j<VARS; j++)
      for (i=0; i<NODES; i++)
         if ((f[i][j] == NULL) && (df[i][j] != NULL)) {
            nvar += 1;
            g[nvar] = i;
            h[nvar] = j;
         }

   /* repeat cycle until convergence */
   printf("cycle:");
   n = 0;
   do {
      work = yp[H][W][kount];
      /* reset work, heat, and entropy integrals */
      for (i=0; i<NODES; i++){
         yp[i][Q][kount] = 0.0;
         yp[i][W][kount] = 0.0;
         yp[i][SI][kount] = 0.0;
         yp[i][SO][kount] = 0.0;
         yp[i][SW][kount] = 0.0;
      }
      /* call cycle integrator */
      odeint(g,h,nvar,eps,hmin,maxsize);
      n += 1;
      printf("%i,", n);
      mdiff = fabs((yp[H][M][kount] - yp[H][M][1]) / yp[H][M][kount]);
      pdiff = fabs((yp[H][P][kount] - yp[H][P][1]) / yp[H][P][kount]);
      tdiff = fabs((yp[H][T][kount] - yp[H][T][1]) / yp[H][T][kount]);
      wdiff = fabs((yp[H][W][kount] - work) / yp[H][W][kount]);
   } while ((mdiff>0.01 || pdiff>0.01 || tdiff>0.01 || wdiff>0.01) && n<iter);
   if (n==iter){
      printf("pdiff=%6.4f mdiff=%6.4f ", pdiff, mdiff);
      printf("tdiff=%6.4f wdiff=%6.4f ", tdiff, wdiff);
   }
   printf("                ");
}
```

D.5 Property Functions

The functions used to calculate properties of the working gas are listed in this section. The functions describing the volume variation are also given.

```
#include "defs.h"

double entpy_infl (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double entpy_outf (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double entpy_wall (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double heat_deriv (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double mass_deriv (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double pres_deriv (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double pres_value (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double temp_deriv (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double temp_value (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double volm_deriv (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double volm_value (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);
double work_deriv (int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS]);

double heat_deriv(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double heatcoef, dheat;

   switch ((enum node) i){
   case H :
   case L : dy[i][T] = (*df[i][T])(i,T,x,y,dy);
            heatcoef = G/(G-1) / PE[i];
            heatcoef *= (y[i][V] + DL[i]) / (1.0 + DL[i]);
            dheat  = NUR[i] * (y[i][T] - WALLTEMP[i]) + NUI[i] * dy[i][T];
            dheat *= -heatcoef;
            return(dheat);
   default : error("heat_deriv", "no case matches i");
   }
}

double mass_deriv(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double coef, factor, dmass;

   coef  = G / (G-1) / PE[i];
   coef *= (y[i][V] + DL[i]) / (1.0 + DL[i]);
   if (dy[i][M] >= 0.0)
      factor = sqr(y[i][M]);
   else
      factor = y[H][P] * y[i][V] * y[i][M];
   factor += (G-1) / G * coef * NUI[i] * y[H][P] * y[i][V];
   factor = sqr(y[i][M]) / factor;
   dmass  = NUR[i] * (y[i][T] - WALLTEMP[i]);
   dmass += NUI[i] / y[i][M] * (y[H][P] * dy[i][V] + y[i][V] * dy[H][P]);
   dmass *= coef * (G-1) / G;
   dmass += y[H][P] * dy[i][V] + y[i][V] * dy[H][P] / G;
   dmass *= factor;
   return(dmass);
}
```

```
/* entropy generation due to mixing when gas flows into a cylinder */
double entpy_infl(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double dentp;

   switch ((enum node) i){
   case H :
   case L : if (dy[i][M] > 0.0){
               dentp  = log(y[i][T]) - (y[i][T] - 1.0) / y[i][T];
               dentp *= G / (G-1) * dy[i][M];
            }
            else
               dentp = 0.0;
            return(dentp);
   default : error("entpy_infl", "no case matches i");
   }
}

/* entropy generation due to mixing when gas flows out of a cylinder */
double entpy_outf(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double dentp;

   switch ((enum node) i){
   case H :
   case L : if (dy[i][M] < 0.0){
               dentp  = log(y[i][T]) - (y[i][T] - 1.0);
               dentp *= G / (G-1) * dy[i][M];
            }
            else
               dentp = 0.0;
            return(dentp);
   default : error("entpy_outf", "no case matches i");
   }
}

/* entropy generation due to gas-to-wall heat transfer in a cylinder */
double entpy_wall(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double dentp;

   switch ((enum node) i){
   case H :
   case L : dentp = (1.0 / y[i][T] - 1.0) * dy[i][Q];
            return(dentp);
   default : error("entpy_wall", "no case matches i");
   }
}
```

```
double pres_value(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   switch ((enum node) i){
   case L : return(y[H][P]);
   default : error("pres_value", "no case matches i");
   }
}

double pres_deriv(int ii, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double coef[NODES], factor[NODES], denom[NODES], numer[NODES], dpres;
   int i;

   for (i=0; i<NODES; i++){
      y[i][V]  = (*f[i][V])(i,V,x,y,dy);
      dy[i][V] = (*df[i][V])(i,V,x,y,dy);
      y[i][T]  = (*f[i][T])(i,T,x,y,dy);
      coef[i]  = G / (G-1) / PE[i];
      coef[i] *= (y[i][V] + DL[i]) / (1.0 + DL[i]);
      if (dy[i][M] >= 0.0)
         factor[i] = sqr(y[i][M]);
      else
         factor[i] = y[H][P] * y[i][V] * y[i][M];
      factor[i] += (G-1) / G * coef[i] * NUI[i] * y[H][P] * y[i][V];
      factor[i] = sqr(y[i][M]) / factor[i];
      denom[i]  = y[i][V]/G + (G-1)/G * coef[i] * NUI[i] * y[i][V] / y[i][M];
      numer[i]  = NUR[i] * (y[i][T] - WALLTEMP[i]);
      numer[i] += NUI[i] * y[H][P] * dy[i][V] / y[i][M];
      numer[i] *= coef[i] * (G-1) / G;
      numer[i] += y[H][P] * dy[i][V];
   }
   dpres = - RVT * numer[L] * factor[L] - numer[H] * factor[H];
   dpres /= RVT * denom[L] * factor[L] + VDTOT +  denom[H] * factor[H];
   return(dpres);
}

double temp_value(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double temp;

   switch ((enum node) i){
   case H :
   case L : if (y[i][V] == 0.0 || y[i][M] <= 0.0)
               error("temp_value", "zero mass");
            else
               temp = y[H][P] * y[i][V] / y[i][M];
            break;
   default : error("temp_value", "no case matches i");
   }
   return(temp);
}
```

```
double temp_deriv(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double dtemp;

   switch ((enum node) i){
   case H :
   case L : if (y[i][V] == 0.0 || y[i][M] <= 0.0)
               error("temp_deriv", "zero mass");
            dtemp =  dy[H][P] / y[H][P];
    dtemp += dy[i][V] / y[i][V];
    dtemp -= dy[i][M] / y[i][M];
            dtemp *= y[i][T];
            return(dtemp);
   default : error("temp_deriv", "no case matches i");
   }
}

double volm_value(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double angle, volume;

   angle = x + 1.5*PI;
   volume = 0.5 * (2.0 - DEADVOLM[i]);
   switch ((enum node) i){
   case H : volume *= (1.0 + sin(angle - phaselag));
            break;
   case L : volume *= (1.0 + sin(angle));
            break;
   default : error("volm_value","no case matches i");
   }
   volume += DEADVOLM[i];
   return(volume);
}

double volm_deriv(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   double angle, dvolume;

   angle = x + 1.5*PI;
   dvolume = 0.5 * (2.0 - DEADVOLM[i]);
   switch ((enum node) i){
   case H : dvolume *= cos(angle - phaselag);
        return(dvolume);
   case L : dvolume *= cos(angle);
        return(dvolume);
   default : error("volm_deriv", "no case matches i");
   }
}

double work_deriv(int i, int j, double x, double y[NODES][VARS], double dy[NODES][VARS])
{
   switch ((enum node) i){
   case H :
   case L : return(y[H][P] * dy[i][V]);
   default : error("work_deriv", "no case matches i");
   }
}
```

D.6 Integrator

A fourth order Runge-Kutta integrator with adaptive stepsize control is used for the cyclic integration. The code is adapted from *Numerical Recipes in C* by Press, Flannery, Teukolsky and Vetterling. The basic Runge-Kutta algorithm is contained in `rk4`. The subroutine `rkqc` determines the stepsize that satisfies the error criterion. In `odeint`, first initializes the array of variables and continues integration until the end of the cycle is reached. The calculated data points are saved at set intervals.

```
#include "defs.h"

void rk4(double y[NODES][VARS],double dydx[NODES][VARS],int i[],int j[],
   int n,double x,double h,double yout[NODES][VARS]);
void rkqc(double y[NODES][VARS],double dydx[NODES][VARS],int i[],int j[],
   int n,double *x,double htry,double eps,double hmin,
   double yscal[NODES][VARS],double *hdid,double *hnext);
void odeint(int i[],int j[],int nvar,double eps,double hmin,int maxsize);

void rk4(double y[NODES][VARS],double dydx[NODES][VARS],int i[],int j[],int n,
double x,double h,double yout[NODES][VARS])
{
   int a;
   double xh,hh;
   double dym[NODES][VARS],dyt[NODES][VARS],yt[NODES][VARS];

   hh=h*0.5;
   xh=x+hh;
   for (a=1;a<=n;a++){
      yt[i[a]][j[a]]=y[i[a]][j[a]]+hh*dydx[i[a]][j[a]];
      dyt[i[a]][j[a]]=dydx[i[a]][j[a]];
   }
   for (a=1;a<=n;a++) dyt[i[a]][j[a]]=(*df[i[a]][j[a]])(i[a],j[a],xh,yt,dyt);
   for (a=1;a<=n;a++){
      yt[i[a]][j[a]]=y[i[a]][j[a]]+hh*dyt[i[a]][j[a]];
      dym[i[a]][j[a]]=dyt[i[a]][j[a]];
   }
   for (a=1;a<=n;a++) dym[i[a]][j[a]]=(*df[i[a]][j[a]])(i[a],j[a],xh,yt,dym);
   for (a=1;a<=n;a++) {
      yt[i[a]][j[a]]=y[i[a]][j[a]]+h*dym[i[a]][j[a]];
      dym[i[a]][j[a]] += dyt[i[a]][j[a]];
   }
   for (a=1;a<=n;a++) dyt[i[a]][j[a]]=(*df[i[a]][j[a]])(i[a],j[a],x+h,yt,dyt);
   for (a=1;a<=n;a++){
      yout[i[a]][j[a]]=dydx[i[a]][j[a]]+dyt[i[a]][j[a]]+2.0*dym[i[a]][j[a]];
      yout[i[a]][j[a]]*=h/6.0;
      yout[i[a]][j[a]]+=y[i[a]][j[a]];
   }
   /* correct for negative masses */
   for (a=0; a<NODES; a++)
      if (yout[a][M] < 0.001)
         yout[a][M]=0.001;
}
```

```
#define PGROW -0.20
#define PSHRNK -0.25
#define FCOR 0.06666666 /* 1.0/15.0 */
#define SAFETY 0.9
#define ERRCON 6.0e-4

void rkqc(double y[NODES][VARS],double dydx[NODES][VARS],int i[],int j[],
int n,double *x,double htry,double eps,double hmin,
double yscal[NODES][VARS],double *hdid,double *hnext)
{
   int a;
   double xsav,hh,h,temp,errmax;
   double dysav[NODES][VARS],ysav[NODES][VARS],ytemp[NODES][VARS];

   xsav=(*x);
   for (a=1;a<=n;a++){
      ysav[i[a]][j[a]]=y[i[a]][j[a]];
      dysav[i[a]][j[a]]=dydx[i[a]][j[a]];
   }
   h=htry;
   for (;;) {
      hh=0.5*h;
      rk4(ysav,dysav,i,j,n,xsav,hh,ytemp);
      *x=xsav+hh;
      for (a=1;a<=n;a++)
         dydx[i[a]][j[a]]=(*df[i[a]][j[a]])(i[a],j[a],*x,ytemp,dydx);
      rk4(ytemp,dydx,i,j,n,*x,hh,y);
      *x=xsav+h;
      if (*x == xsav) error("rkqc","step size too small");
      rk4(ysav,dysav,i,j,n,xsav,h,ytemp);
      errmax=0.0;
      for (a=1;a<=n;a++) {
         ytemp[i[a]][j[a]]=y[i[a]][j[a]]-ytemp[i[a]][j[a]];
         temp=fabs(ytemp[i[a]][j[a]]/yscal[i[a]][j[a]]);
         if (errmax < temp) errmax=temp;
      }
      errmax /= eps;
      if ((errmax <= 1.0) || (h <= hmin)){
         *hdid=h;
         *hnext=(errmax > ERRCON ? SAFETY*h*exp(PGROW*log(errmax)) : 4.0*h);
         if (*hnext < hmin) *hnext = hmin;
         break;
      }
         h=SAFETY*h*exp(PSHRNK*log(errmax));
         /* if next stepsize to small, set to hmin since hmin limited
            by truncation errors */
         if (h < hmin) h = hmin;
   }
   for (a=1;a<=n;a++)
      y[i[a]][j[a]] += ytemp[i[a]][j[a]]*FCOR;
}

#undef PGROW
#undef PSHRNK
#undef FCOR
#undef SAFETY
#undef ERRCON
```

```
#define MAXSTP 1.0e+5
#define TINY 1.0e-30

void odeint(int i[],int j[],int nvar,double eps,double hmin,
int maxsize)
{
   int nstp,a,ii,jj;
   double dxsav,xsav,x,hnext,hdid,h;
   double y[NODES][VARS], dy[NODES][VARS], yscal[NODES][VARS];

   dxsav=2.0*PI/maxsize;
   h=dxsav;                                    /* initialize stepsize */
   x=0.0;
   xsav=0.0;
   xp[0]=0.0;
   /* initialize arrays that store instantaneous values and derivatives */
   for (ii=0;ii<NODES;ii++)
      for (jj=0;jj<VARS;jj++){
         y[ii][jj]=yp[ii][jj][kount];
         dy[ii][jj]=dyp[ii][jj][kount];
         yp[ii][jj][1]=yp[ii][jj][kount];
         dyp[ii][jj][1]=dyp[ii][jj][kount];
      }
   kount = 1;

   for (nstp=1;nstp<=MAXSTP;nstp++) {
      printf("%17.15f", x);
      for (a=1;a<=nvar;a++)
         dy[i[a]][j[a]]=(*df[i[a]][j[a]])(i[a],j[a],x,y,dy);
      for (a=1;a<=nvar;a++)
         yscal[i[a]][j[a]]=fabs(y[i[a]][j[a]])+fabs(dy[i[a]][j[a]]*h)+TINY;
      if (x+h > 2.0*PI) h=2.0*PI-x;
      rkqc(y,dy,i,j,nvar,&x,h,eps,hmin,yscal,&hdid,&hnext);
      if (x-xsav > dxsav || x >= 2.0*PI)
         if (kount < maxsize-1) {
            xp[++kount]=x;
            for (ii=0;ii<NODES;ii++)
               for (jj=0;jj<VARS;jj++){
                  yp[ii][jj][kount]=y[ii][jj];
                  dyp[ii][jj][kount]=dy[ii][jj];
               }
            xsav=x;
         }
      printf("\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b\b");
      if (x >= 2.0*PI) return;
      if (hnext < hmin) error("odeint","step size too small");
      h=hnext;
   }
   error("odeint","too many steps");
}

#undef MAXSTP
#undef TINY
```

References

Chafe, J.N., 1988, "A Study of Gas Spring Heat Transfer in Reciprocating Cryogenic Machinery," Ph.D. Thesis, Massachusetts Institute of Technology.

Chapra, S.C. and Canale, R.P., 1988, *Numerical Methods for Engineers*, McGraw-Hill Book Company, New York.

Coppage, J.E and London, A.L., 1956,"Heat Transfer and Flow Characteristics of Porous Media," *Chemical Engineering Progress*, Vol. 52, No. 2, pp. 57F-63F.

Cravalho, E.G and Smith, J.L., Jr., 1981, *Engineering Thermodynamics*, Pitman Publishing Inc., Boston.

Finkelstein, T., 1960, "Generalized Thermodynamic Analysis of Stirling Engines," SAE Paper 118B, Society of Automotive Engineers, New York.

Harris, W.S, 1970, "Regenerator Optimization for a Stirling-Cycle Refrigerator," M.S. Thesis, Massachusetts Institute of Technology.

Harris, W.S., Rios, P.A. and Smith, J.L., Jr., 1971, "The Design of Thermal Regenerators for Stirling-Type Refrigerators," *Advances in Cryogenic Engineering*, Vol. 16, pp. 312-323.

Kays, W.M. and Crawford, M.E., 1980, *Convective Heat and Mass Transfer*, 2nd Ed., McGraw-Hill Book Company, New York.

Kays, W.M. and London, A.L., 1984, *Compact Heat Exchangers*, 3rd Ed., McGraw-Hill Book Company, New York.

Kornhauser, A.A., 1989, "Gas-Wall Heat Transfer During Compression and Expansion," Ph.D. Thesis, Massachusetts Institute of Technology.

Lee, K.P., 1983,"A Simplistic Model of Cyclic Heat Transfer Phenomena in Closed Spaces," *Proceedings of the 18th Intersociety Energy Conversion Engineering Conference*, pp. 720-723.

Lienhard, J.H., 1987, *A Heat Transfer Textbook*, 2nd Ed., Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Martini, W.R., 1983, *Stirling Design Manual*, NASA CR-168088, National Aeronautics and Space Administration.

Press, W.H., Flannery, B.P., Teukolsky, S.A. and Vetterling, W.T., 1988, *Numerical Recipes in C*, Cambridge University Press, New York.

Qvale, E.B., 1967, *An Analytical Model of Stirling-Type Engines,* MIT, Ph.D. Thesis.

Qvale, E.B. and Smith, J.L., Jr., 1968, "A Mathematical Model for Steady Operation of Stirling-Type Engines," *Transactions of the ASME, Journal of Engineering for Power*, pp. 45-50.

Qvale, E.B. and Smith, J.L., Jr., 1969, "An Approximate Solution for Thermal Performance of a Stirling-Engine Regenerator," *Transactions of the ASME, Journal of Engineering for Power*, pp. 109-112.

Rios, P.A., 1969"An Analytical and Experimental Investigation of the Stirling Engine," Ph.D. Thesis, Massachusetts Institute of Technology.

Rios, P.A. and Smith, J.L., Jr., 1970, "An Analytical and Experimental Evaluation of the Pressure-Drop Losses in the Stirling Cycle," *Transactions of the ASME, Journal of Engineering for Power*, pp. 182-188.

Rios, P.A., Qvale, E.B. and Smith, J.L., Jr., 1969, "An Analysis of the Stirling-Cycle Refrigerator," *Advances in Cryogenic Engineering*, Vol. 14, pp. 332-442.

Rohsenow, W.M. and Choi H., 1961, *Heat, Mass and Momentum Transfer*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Schmidt, G., 1871, "Theorie der Lehmannschen Calorischen Machine," *Z. Ver. Dtsch. Ing.*, Vol. 15, No. 1, p. 97.

Urieli, I. and Berchowitz, D.M., 1984, *Stirling Cycle Engine Analysis*, Adam Hilgar Ltd., Bristol.

Walker, G., 1980, *Stirling Engines*, Oxford University Press, New York.

West, C.D., 1986, *Principles and Applications of Stirling Engines*, Van Nostrand Rheinhold Company, New York.