

ANALYSIS OF HYBRID LIQUID
DESICCANT COOLING SYSTEMS

by

THOMAS KARL BUSCHULTE

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science
(Chemical Engineering)

at the
UNIVERSITY OF WISCONSIN-MADISON

1984

ANALYSIS OF HYBRID LIQUID DESICCANT COOLING SYSTEMS

THOMAS KARL BUSCHULTE

MASTER OF SCIENCE
(Chemical Engineering)

UNIVERSITY OF WISCONSIN-MADISON

1984

ACKNOWLEDGEMENTS

I would like to thank my advisors, Professor Jack Duffie and Professor Sandy Klein, for their help and encouragement throughout this project. I am particularly indebted to Professor Sandy Klein for the opportunity to participate so closely in the development of this project. The assistance of Professor John Mitchell has also had an important impact on the progress of this work. I am likewise grateful to my fellow graduate students who have been a much appreciated source of ideas, encouragement and support.

The initial year of my stay at the University of Wisconsin-Madison was part of an exchange program organized by Prof. Dr. Ing. Zeitz, Universität Stuttgart, Federal Republic of Germany, and Prof. Dr. W.H. Ray, Department of Chemical Engineering, University of Wisconsin-Madison. Their personal efforts have made this exchange program unique and very successful. The funds for this program were made available by the German taxpayers through the German Academic Exchange Service (DAAD). A grant by the Cusanuswerk, a foundation of the German Catholic Bishops, and funding by the Solar Heating and Cooling Research and Development Branch Office of Conservation and Solar Applications, Department of Energy, provided the financial support to complete this work.

Finally, I would like to thank my parents for their continuous support and encouragement during all of the ups and downs of my stay here in Madison. Herzlichen Dank Euch, Mutti und Vati.

ABSTRACT

In the Science Museum of Virginia in Richmond, VA, a hybrid desiccant dehumidifier-vapor compression heat pump system was installed in addition to the conventional HVAC equipment. Instead of removing moisture in the air by means of low temperature condensing coils, the air is blown through a conditioner chamber. A cold LiCl-water solution absorbs some of the water vapor. The diluted salt solution is heated by a low temperature heat source. In a regenerator chamber, water is desorbed from the solution by return air which is then exhausted.

The conditioner and regenerator are basically two phase contact devices with simultaneous heat and mass transfer. A model based on equilibrium considerations and effectiveness coefficients was developed. A second model employing a finite step integration along the chamber was used to estimate variations of the effectiveness coefficients with varying inlet states.

A simulation model of the HVAC system installed at the Science Museum of Virginia was developed. This model includes the complete liquid desiccant cycle and important components of the conventional HVAC system.

Parametric studies were performed to identify the sensitivity of the overall performance to changes in equipment parameters. The results of steady state simulations were analyzed and the hourly cost of operation were estimated for different regeneration heat sources.

TABLE OF CONTENTS

	<u>Page</u>
Acknowledgements	ii
Abstract	iii
List of Figures	viii
Nomenclature	xi
 1. OPEN CYCLE ABSORPTION SYSTEMS	 1
1.1 Introduction	1
1.2 The Regenerator for Open Cycle Absorption Systems	3
1.3 Conventional Air Conditioning Systems	4
1.4 The Open Cycle Liquid Desiccant System at the Science Museum of Virginia	7
 2. THE HEAT AND MASS EXCHANGER MODELS	 13
2.1 Equilibrium Model	13
2.2 Finite Step Integration Model	21
2.2.1 Implementation in an Algorithm	25
2.2.2 Determination of the Heat and Mass Transfer Coefficients and the Area	30
2.3 Comparison of the Two Models	35

	<u>Page</u>
3. SYSTEM SIMULATION	45
3.1 Physical Property Functions	46
3.1.1 Air Water Vapor Mixtures	46
3.1.2 Lithium Bromide-Water Solutions	47
3.1.3 Lithium Chloride-Water Solutions	47
3.1.3.1 Water Vapor Pressure	47
3.1.3.2 Enthalpy of Lithium Chloride-Water Solutions	48
3.2 TRNSYS Component Models	49
3.2.1 The Heat and Mass Exchanger Model	50
3.2.2 The Heat and Mass Exchanger Sump Model	52
3.2.3 The Heat Exchanger Model	54
3.2.4 The Air Mixer Model	56
3.2.5 The Fan Model	56
3.2.6 The Vapor Compression Heat Pump Model	58
3.2.7 The Vapor Compression Chiller Model	59
3.2.8 The Proportional Integral Controller	63
3.2.9 The Load Model	65
3.3 The System Simulation Model	66

	<u>Page</u>
3.4 Results of Steady State Simulations	69
3.4.1 Variations in the Cooling Water Supply Temperature	73
3.4.2 Variations in the Hot Water Supply Temperature	76
3.4.3 Variations in the Overall Heat Conductance-Area Products of the Solution Cooler and Heater	79
3.4.4 Variations in the Effectiveness of the Heat and Recovery Loop Heat Exchangers	83
3.4.5 Summary	86
4. CONCLUSIONS	87
5. RECOMMENDATIONS	89
REFERENCES	92

APPENDICES

- A Physical Property Functions
 - A.1 Air-Water
 - A.2 Lithium Bromide-Water
 - A.3 Lithium Chloride-Water
- B TRNSYS Component Models
 - B.1 The Heat and Mass Exchanger
 - B.2 The Heat and Mass Exchanger Sump
 - B.3 The Heat Exchanger
 - B.4 The Air Mixer
 - B.5 The Fan
 - B.6 The Vapor Compression Heat Pump
 - B.7 The Vapor Compression Chiller
 - B.8 The Proportional Integral Controller
 - B.9 The Load
- C TRNSYS Deck of the System Model "HVAC System, Science Museum of Virginia"
- D Finite Step Integration Model-Program Listing

LIST OF FIGURES

	<u>Page</u>
Figure 1.1 Dehumidification and cooling of air by partial condensation of the water vapor in the air	6
Figure 1.2 Schematic of the open cycle liquid desiccant system in the Science Museum of Virginia in Richmond, VA	8
Figure 2.1 Graphical representation of the definitions of the effectiveness coefficients	19
Figure 2.2 Heat and mass exchanger element with transfer area dA	22
Figure 2.3 Paths of air and solution states in heat and mass exchanger, conditioner design point	26
Figure 2.4 Paths of air and solution states in heat and mass exchanger, conditioner point, large transfer area	27
Figure 2.5 Paths of air and solution states in heat and mass exchanger, regenerator design point	28
Figure 2.6 Paths of air and solution states in heat and mass exchanger, large transfer area	29
Figure 2.7 The Lewis number as a function of the mean temperature between solution and air	33
Figure 2.8 The mass transfer coefficient as a function of the mean temperature between solution and air	34
Figure 2.9 Fitted effectiveness coefficients versus solution inlet temperature	36
Figure 2.10 Error in transfer rates versus solution inlet temperature holding the effectiveness coefficients constant	37
Figure 2.11 Fitted effectiveness coefficients versus mass flow rate ratio	39

Figure 2.12	Error in transfer rates versus mass flow rate ratio holding the effectiveness factors constant	40
Figure 2.13	Effectiveness coefficients for mass transfer versus mass flow rate ratio as predicted by the finite step integration model and the NTU method	43
Figure 2.14	Effectiveness coefficients for heat transfer versus mass flow rate as predicted by the finite step integration model and the NTU method	44
Figure 3.1	Information flow diagram for the TRNSYS component "heat and mass exchanger"	51
Figure 3.2	Information flow diagram for the TRNSYS component "heat and mass exchanger sump"	53
Figure 3.3	Information flow diagram for the TRNSYS component "heat exchanger"	55
Figure 3.4	Information flow diagram for the TRNSYS component "air mixer"	57
Figure 3.5	Information flow diagram for the TRNSYS component "fan"	57
Figure 3.6	Information flow diagram for the TRNSYS component "vapor compression heat pump"	61
Figure 3.7	Information flow diagram for the TRNSYS component "vapor compression chiller"	61
Figure 3.8	Information flow diagram for the TRNSYS component "proportional integral controller"	64
Figure 3.9	Information flow diagram for the TRNSYS component "load"	64
Figure 3.10	Air and solution states in the liquid desiccant system in the Science Museum of Virginia, design point	67
Figure 3.11	Air and solution states in the liquid desiccant system in the Science Museum of Virginia, as predicted by the TRNSYS system model	68

Figure 3.12	Hourly cost of operation versus cooling water supply temperature for different modes of operation	75
Figure 3.13	Hourly cost of operation versus hot water supply temperature for different modes of operation	78
Figure 3.14	Hourly cost of operation versus UA of the solution cooler for different modes of operation	81
Figure 3.15	Hourly cost of operation versus UA of the solution heater for different modes of operation	82
Figure 3.16	Hourly cost of operation versus heat exchanger effectiveness of the heat recovery loop for different modes of operation	85

NOMENCLATURE

\dot{C}	capacity rate
c_D	drag coefficient of a sphere
c_p	specific heat capacity
COP	coefficient of performance
d	droplet diameter
D	diffusivity
e	controller error
g	gravity constant
h	transfer coefficient
i	enthalpy of air
i_v	heat of vaporization
I	enthalpy of solution
k	heat conductivity
K	controller gain
\dot{m}	mass flow rate
NTU	number of exchange transfer units
Nu	Nusselt number
p_{amb}	ambient pressure
p_w	partial pressure of water vapor
P	Power
Pr	Prandtl number
\dot{Q}	energy flow rate
Re	Reynolds number

Sc	Schmidt number
T	temperature of solution
u	controller action
w	humidity ratio of air
W	humidity ratio of air in equilibrium with solution at (T, ξ)
Y	system state
α	thermal diffusivity
γ	control function
Δ	difference
ϵ	effectiveness coefficient
η	efficiency of a reversible Carnot engine
μ	viscosity
ξ	concentration of solution
ρ	density
τ_I	time constant of integral controller

Subscripts

a	air
c	condenser
e	evaporator
elt	electric
ex	exchanged
h	heat transfer

i	inlet
lat	latent
m	mass transfer
o	outlet
s	solution
sens	sensible
set	set point
w	water

Superscripts

equ	in equilibrium
^	calculated
rev	reversible
•	rate

CHAPTER 1 OPEN CYCLE ABSORPTION SYSTEMS

1.1 Introduction

Closed cycle absorption heat pumps have long been known as useful tools for converting thermal energy from one temperature level to another. They are reliable machines with a large number of applications. Among these are the upgrading of waste heat and the use of heat to generate a heat sink below ambient temperature. Bjurstroem and Raldow [1] present a survey of the history and the various applications of absorption heat pumps.

Generally, absorption heat pumps are designed as closed systems, i.e., both the absorbant and the refrigerant remain in the system permanently. In the early sixties, a team of Russian researchers proposed the application of open absorption cycles for solar cooling [26]. In this system, the condenser is omitted and the refrigerant is released from the desorber to the environment. Naturally, only water can be used as the refrigerant in such a system. Moreover, the other components of the absorbent mixture have to have a very small vapor pressure at operating conditions or else they would evaporate and be carried out of the system. Aqueous solutions of the salts LiCl , LiBr and CaCl_2 are well suited for this purpose.

Since refrigerant is continuously lost from the system, it has to be replaced by make-up water. This water evaporates in a vessel which combines the function of evaporator and absorber. The content

of the vessel is maintained at a low pressure by the salt solution, which absorbs the water vapor. The evaporating water acts as a heat sink at the saturation temperature corresponding to the vessel pressure. The heat needed for vaporization is transferred from the cooling water to the evaporating water via a coil. The diluted solution has to be regenerated to recover its ability to absorb water. An open cycle absorption system is described by Löff et al. [25].

A different design of an open absorption-desorption cycle is a system used for air dehumidification purposes. Both condenser and evaporator are omitted, and the absorber is constructed in such a way as to allow the dehumidification of moist air. As early as 1937, Berestneff [28] outlined the fundamentals of this system. The system consists of a conditioner chamber, where air is dehumidified, and a regenerator chamber, where exhaust air takes on water which evaporates from the diluted solution. Dehumidification systems of this design are being applied where very dry air is required. This type of open cycle absorption system is the subject of this study and is presented in detail in Section 1.4.

1.2 The Regenerator for Open Cycle Absorption Systems

The regenerator or desorber is an important part of any absorption cycle. The diluted absorbent is reconcentrated in the regenerator. The absorbent has to be heated to a temperature at which refrigerant evaporates. The refrigerant is then condensed in the condenser. In refrigeration applications, the heat of condensation is rejected to the environment, typically at 25-35°C. The saturation pressure of water in this temperature range is 3.2-5.6 kPa. The partial pressure of water in air is much lower, e.g., 1.6 kPa for air at 25°C and 50% relative humidity. Therefore, the application of open regenerators allows lower regeneration temperatures than necessary in closed systems.

The fundamental problem in regenerator design is to create a large heat and mass transfer surface between liquid and vapor phases. Various designs have been proposed. Among the most simple are sloped plane-falling film type regenerators. The Russian research team [26] used a blackened roof as a solar driven desorber. Solar energy is absorbed by the black surface and the liquid absorbent in contact with this surface is heated up. Water evaporates from the solution to the surrounding air. Collier [27] developed a model for this combined flat plate solar collector regenerator. However, there are some major problems with this type of regenerator. Among them are corrosion problems, pollution of the solution, difficulties in maintaining a stable film of solution and

very limited transfer surfaces. A modification of this combined solar collector regenerator using a glazed collector was studied by Howell and Shepherd [30].

Another heat and mass exchanger device used as a regenerator is the packed bed column. The advantage of packed bed columns is the large surface of the packing material. Thus, a large heat and mass transfer area can be obtained if the absorbent is distributed properly. Leboeuf [29] studied the application of a packed bed column as a regenerator in an open cycle desiccant cooling system. The drawback of packed bed columns is a large air stream pressure drop.

1.3 Conventional Air Conditioning Systems

The main task of air conditioning is to maintain the air in a space, whether it is a single room or a large building, at a desired state, i.e., at a certain temperature and humidity ratio. Several sources contribute to the sensible portion of a space air conditioning load. Among the most important heat sources are heat conduction through the building envelope, heat released by the lighting system, heat released by people and the energy which is carried into the building with infiltration of air. In addition, there has to be a controlled exchange of the air inside the space with fresh air from outside. This air has to be cooled to the room air state.

The second part of the air conditioning load is called the latent load. It is due to moisture which is released by people and carried into the building by air infiltration. Furthermore, moist ventilation air from outside has to be dehumidified before entering the building.

The latent load is generally much smaller than the sensible load. Nevertheless, it is the cause of the poor overall coefficient of performance, (COP, defined as energy removed per work expended) in conventional air conditioning systems. In these systems, the air to be conditioned is blown through a heat exchanger. Water of typically 4.5 to 9°C is pumped through the heat exchanger piping and cools the air down to its dewpoint temperature. Part of the water vapor in the air condenses, releasing its heat of vaporization. This condensation process is continued until the air reaches the desired humidity ratio. Finally, the dehumidified but cold air has to be reheated to the desired room inlet temperature. Although the energy to reheat the air is generally available from free waste heat, the primary subcooling needs a lot more energy than a thermodynamically optimal process. Moreover, the coefficient of performance (COP) of a chiller, which provides the cooling water, decreases rapidly with decreasing cooling water temperature.

In Figure 1.1 the process of air dehumidification by condensation of water vapor is shown. Air of state 1 is cooled down to its dewpoint and further until it reaches the desired humidity ratio. This cold air has to be reheated to state 2 before it can be supplied to the building.

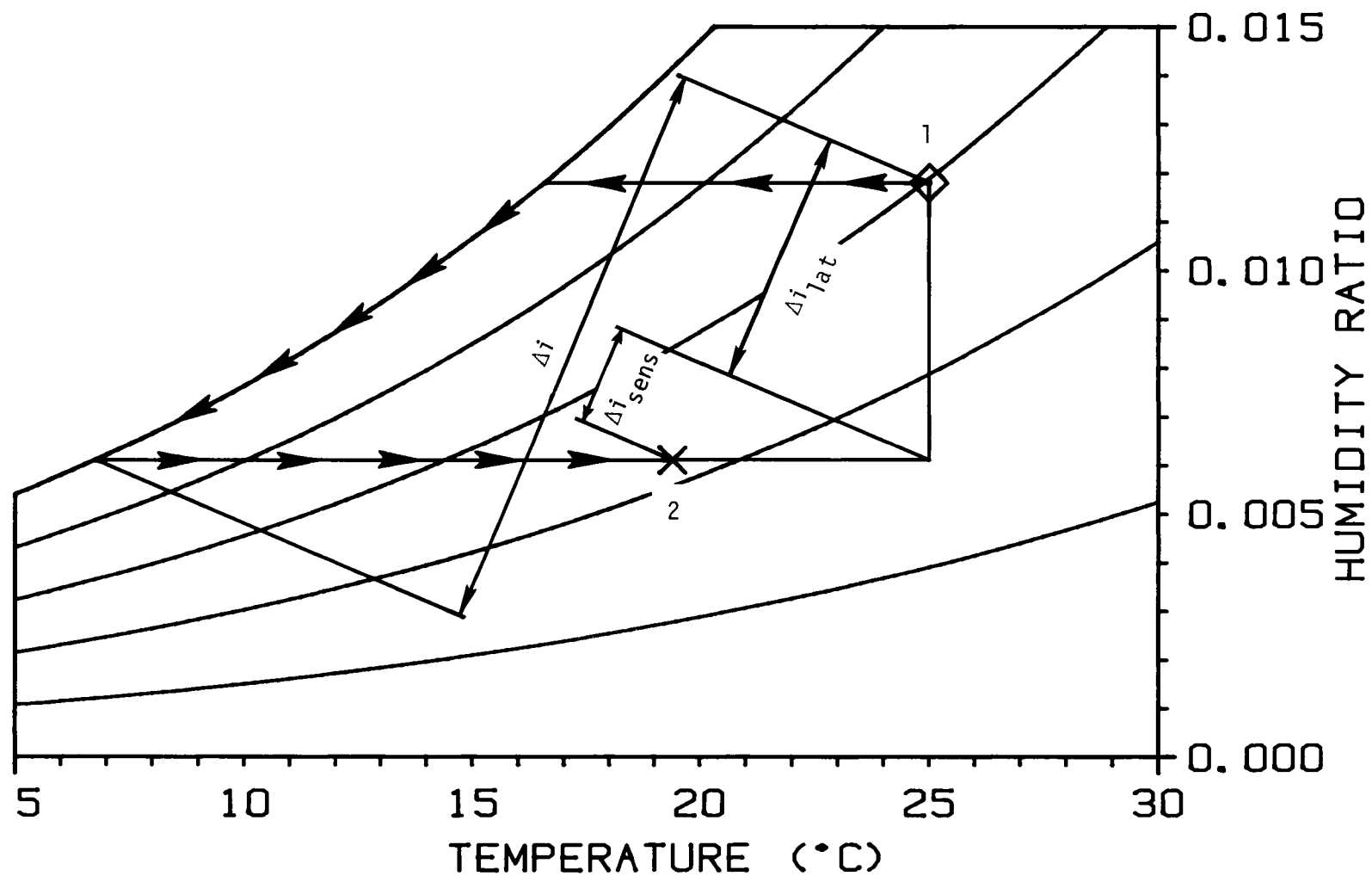


Figure 1.1

Dehumidification and cooling of air by partial condensation of the water vapor in the air

The difference in enthalpy between air at state 1 and the dehumidified air is Δi . However, the net enthalpy difference between the air states 1 and 2 is only the sum of the latent enthalpy difference Δi_{lat} and the sensible enthalpy difference Δi_{sens} . Hence, the enthalpy difference removed during the dehumidification is much larger than the enthalpy difference, which would have to be removed on a direct path from air state 1 to air state 2.

1.4 The Open Cycle Liquid Desiccant System at the Science Museum of Virginia

A thermodynamically optimal process would cool and dehumidify the air only to the extent necessary, using cooling water at the highest temperature possible. This process can be approached by splitting up the air conditioning task into sensible cooling and dehumidification. The sensible cooling can then be accomplished using cooling water at a higher temperature, e.g., 12 to 16°C. Thus not only less energy has to be removed from the air stream but also the chiller operates at a higher COP. The dehumidification is done by equipment specifically designed for this task.

An open absorption-desorption-cycle using a nontoxic salt solution (LiCl - water) can be employed for the air dehumidification task. A schematic of the cycle is shown in Figure 1.2. In a chamber, precooled salt solution flows in countercurrent to the air stream and absorbs water vapor. The diluted solution is pumped to a regenerator, heated and brought into contact with an air stream

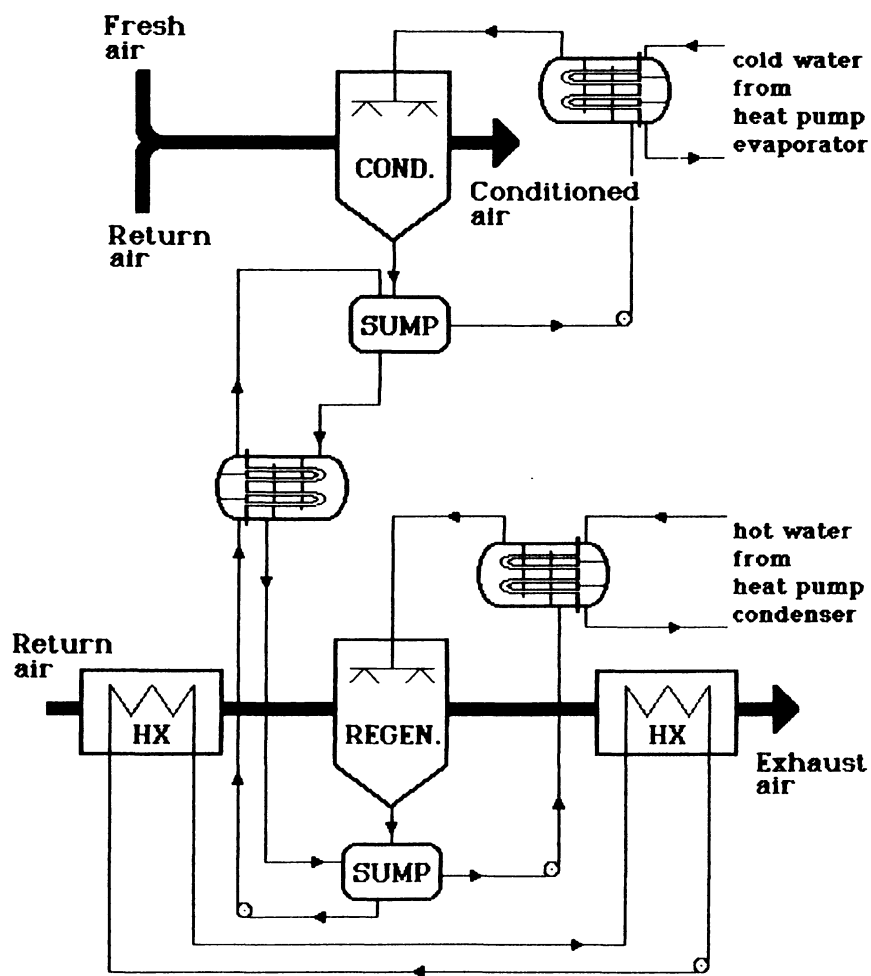


Figure 1.2 Schematic of the open cycle liquid desiccant system in the Science Museum of Virginia in Richmond, VA

returning from the building which then will be exhausted. Water evaporates from the hot solution and the solution becomes more concentrated. The concentrated solution is cooled, pumped to the conditioner and the cycle starts over again.

A liquid desiccant system as described above is installed at the Science Museum of Virginia in Richmond, VA, in addition to the conventional HVAC equipment. Conventional vapor compression machines and the liquid desiccant cycle are combined to a hybrid system. The design of this system and some performance data were presented by Meckler [7,8]. In particular, data for the design point are given. The analysis presented in this paper is based solely on these design point data since no experimental data were available.

The outdoor air state at the design point is 32.8°C and a humidity ratio is 0.0177. The desired room air state is 25°C and 0.0093. The states of the liquid desiccant cycle are shown in Figure 3.10.

Besides the equipment of the desiccant cycle, the system of the Science Museum of Virginia features a number of additional devices. Two 5000 gallon water tanks are installed as heat storage. A heat pump can deliver the heat needed for the regeneration of the salt solution and at the same time meet part of the cooling load. Alternately, a gas cogenerator can produce electricity and regeneration heat. Furthermore, the installation of solar collectors may be considered, as adequate space for the collectors exists and thermal storage is provided in form of the water tanks.

The whole system is set-up so that it can be operated in various modes. Meckler [7] reports six modes of operation. The different modes of operation reflect different ways of providing the heat for the regeneration of the salt solution. The modes considered in this study are

1. operation of the heat pump to simultaneously provide regeneration heat and part of the cooling load ("heat pump" mode).
2. operation of the gas cogenerator to provide the regeneration heat with the cooling load met solely by the chiller ("gas cogeneration" mode).
3. use of solar energy as regeneration heat source with the cooling load met solely by the chiller ("solar" mode).
4. conventional air conditioning by condensation of excess moisture ("chiller" mode).

The simulation model of the system and results of steady state simulations at the design point are presented in Sections 3.3 and 3.4, respectively.

A comparison of the results of the steady state system model with the data given by Meckler in [7] had been intended. Since both sets of data were obtained for the design point, they should be comparable. However, a thorough analysis of the data given by Meckler revealed some apparent inconsistencies.

The total volume flow rate of conditioned air supplied to the building is composed of

19000	SCFM	to the main exhibits and lobby
14000	SCFM	to the planetarium
5000	SCFM	to lower level exhibits
<u>1000</u>	SCFM	to supporting service areas
39000	SCFM	total

A volume flow rate of 12000 SCFM comes from the conditioner, so that 27000 SCFM return air has to be mixed with the dehumidified air. This air is supplied to the building at 66.7°F dry bulb temperature and 60.3 Gr/lb moisture content (state 9) [7], page 197. The air returns from the building with a moisture content of 65 Gr/lb (state 6). Thus, this air stream can meet a latent load of

$$\dot{Q}_{lat} = 39000 \text{ SCFM} \cdot 0.0765 \frac{\text{lb dry air}}{\text{ft}^3} \left(\frac{65 - 60.3}{7000} \right) \frac{\text{lb water}}{\text{lb dry air}}$$

$$1061 \frac{\text{Btu}}{\text{lb water}} = 2125.4 \text{ Btu/min} = 127.5 \text{ MBtu/h} \quad (1.1)$$

However, Meckler [7] (page 196) lists a load of 172.983 MBtu/h internal latent load. There is a discrepancy of 26%, which cannot be explained by round-off errors. However, if state 7 is recalculated from states 5 and 6

$$w_7 = \frac{12000 \text{ SCFM} \cdot 42.6 \text{ GR/lb} + 27000 \text{ SCFM} \cdot 65 \text{ Gr/lb}}{39000 \text{ SCFM}} = 58.1 \text{ Gr/lb} \quad (1.2)$$

and this corrected state is then used for the calculation of the latent load

$$\dot{Q}_{lat} = 39000 \text{ SCFM} \cdot 0.0765 \frac{\text{lb dry air}}{\text{ft}^3} \left(\frac{65 - 58.1}{7000} \right) \frac{\text{lb water}}{\text{lb dry air}}$$

$$1067 \frac{\text{Btu}}{\text{lb water}} = 187.2 \text{ MBtu/h} \quad (1.3)$$

one arrives at a value of 8% higher than the value given by Meckler.

Another questionable point is the coefficient of performance (COP) given for the vapor compression chiller. In his paper, Meckler uses a COP based on evaporator heat flow. For chiller operation between a water leaving temperature of 55°F for the evaporator and 95°F for the condenser, Meckler assumes a COP of 5.2. For operation between 42°F and 95°F, a COP of 3.0 is assumed. This large difference between the two COPs means a large overprediction of the variation of the chiller COP with variation in the operating temperatures.

In this study, a model of the vapor compression chiller based on a Carnot cycle was used. The model accounts for temperature differences across condenser and evaporator. The effectiveness coefficient used to reduce the COP of the Carnot cycle to the COP of the non-ideal chiller was chosen as 0.6. The COP predicted for the operation between 55°F and 95°F is 5.23 and 4.16 for the operation between 42°F and 95°F. These COPs seem to be realistic according to manufacturers data [32]. The model of the vapor compression chiller is treated in detail in Section 3.2.6.

CHAPTER 2 THE HEAT AND MASS EXCHANGER MODELS

The conditioner and the regenerator which are installed in the Museum of Science are basically two-phase contact devices with simultaneous heat and mass transfer. Thus, both can be described by the same model.

A model used in a long term simulation is evaluated many times. Therefore, it is necessary that the algorithm of the model requires the least possible computational expense. A model based on equilibrium considerations and effectiveness factors was developed which meets this requirement. A listing of the FORTRAN source code of the TRNSYS component subroutine can be found in Appendix B.1.

For lack of experimental data, it was necessary to develop a second, more elaborate model. This model employs a finite step integration along the heat and mass exchanger. Combining the results of the two models, it was possible to estimate the variation of the effectiveness coefficients used in the TRNSYS model with variations in inlet states.

2.1 Equilibrium Model

Subsequently, a model using an equilibrium approach and effectiveness factors will be described. Assuming an infinitely long chamber, the equilibrium outlet states of the solution and the air stream can be determined. By means of heat and mass balances the

exchanged heat and mass flow rates are calculated. Finally, the exchanged heat and mass flow rates are corrected for the real chamber dimensions by multiplication with two effectiveness coefficients, one each for the mass and heat exchange.

The calculation of the equilibrium outlet states is based on three mass balances, the overall balance

$$\dot{m}_{si} + \dot{m}_{ai} (1 + w_i) = \dot{m}_{so} + \dot{m}_{ao} (1 + w_o) \quad (2.1)$$

the mass balance for the salt

$$\dot{m}_{si} \xi_i = \dot{m}_{so} \xi_o \quad (2.2)$$

the mass balance for dry air

$$\dot{m}_{ai} = \dot{m}_{ao} \quad (2.3)$$

and the overall energy balance

$$\dot{m}_{si} I_i + \dot{m}_{ai} i_i = \dot{m}_{so} I_o + \dot{m}_{ao} i_o \quad (2.4)$$

In addition to these balances, two equilibrium assumptions are necessary. Table 2.1 shows the possible equilibrium assumptions.

	Equilibrium of temperature	Equilibrium of water vapor pressure
1	at solution inlet	at solution inlet
2	at air inlet	at air inlet
3	at air inlet	at solution inlet
4	at solution inlet	at air inlet
5	air at outlet saturated	at air inlet
6	at air inlet	air at outlet saturated

Table 2.1 Proposed pairs of equilibrium assumptions.

Next, it has to be determined, which one out of the six proposed pairs of equilibria is physically feasible. Four conditions to be fulfilled by the feasible equilibrium are derived subsequently.

The driving force for the mass transfer is the difference in the partial pressures of water vapor in the air and above the solution. Hence the direction of the mass transfer has to be the same as that of the negative gradient in the partial pressure of water vapor. Using the relationship [3]

$$w = 0.62198 \frac{p_w}{p_{amb} - p_w} \quad (2.5)$$

which, for small water vapor pressures, can be approximated by

$$w = 0.62198 \frac{p_w}{p_{amb}} \quad (2.6)$$

and defining the exchanged mass flow rate to be positive in the case

of regeneration, i.e., if water evaporates from the solution into the air, the first feasibility condition can be formulated as:

$$\dot{m}_{ex}(W - w) > 0 \quad (2.7)$$

The driving force for heat transfer is not temperature but enthalpy because there is not only sensible but also latent heat transferred. At any instant of time, for an arbitrary but small area of transfer surface (e.g. droplet surface), the heat and mass transfer can be written as

$$\dot{m}_{ex} = h_m \Delta A (W - w) \quad (2.8)$$

$$\dot{Q}_{ex,sens} = h_h \Delta A (T - t) \quad (2.9)$$

The exchanged energy flow rate can be split into its sensible and latent parts,

$$\dot{Q}_{ex} = \dot{Q}_{ex,sens} + \dot{Q}_{ex,lat} \quad (2.10)$$

The exchanged latent energy flow rate can be expressed

$$\dot{Q}_{ex,lat} = h_m \Delta A (W - w) i_v \quad (2.11)$$

so that

$$\dot{Q}_{ex} = h_h \Delta A (T - t) + h_m \Delta A (W - w) i_v \quad (2.12)$$

By using the Lewis number relationship

$$Le = \frac{h_h}{h_m c_{p,a}} \quad (2.13)$$

Equation (2.12) can be regrouped to

$$\dot{Q}_{ex} = h_m \Delta A (Le c_{p,a} T + i_v W - (Le c_{p,a} t + i_v w)) \quad (2.14)$$

Neglecting the enthalpy term for the liquid water $c_{p,w} t$ and assuming a Lewis number of unity, Equation (2.14) can be expressed as

$$\dot{Q}_{ex} = h_m \Delta A (i^* - i) \quad (2.15)$$

where i^* denotes the enthalpy of air (at T, W) in temperature and partial pressure of water vapor equilibrium with the solution.

The fact that the difference of the driving force ($i^* - i$) can approach zero but cannot switch sign along the heat and mass exchanger is used as second feasibility condition:

$$(i_{in}^* - i_{out})(i_{out}^* - i_{in}) > 0 \quad (2.16)$$

Analogous to the first feasibility condition, the third one can be stated as

$$\dot{Q}_{ex}(i^* - i) > 0 \quad (2.17)$$

Finally, the air at its outlet state may not be supersaturated, which is the fourth feasibility condition.

The last step in the derivation of the model is to account for the finite size of the heat and mass exchanger. The heat and mass flow rates obtained from the equilibrium considerations are multiplied by two effectiveness coefficients, one each for the sensible heat transfer and for the mass flow rate, respectively. In analogy to the effectiveness factor approach for heat exchangers, the effectiveness factors for sensible heat and mass transfer are defined by

$$\Delta i_{sens} = \epsilon_h \Delta i_{sens}^{equ} \quad (2.18)$$

and

$$\Delta w = \epsilon_m \Delta w^{equ} \quad (2.19)$$

A graphic representation of these two definitions is given in Figure 2.1. In the particular case depicted, air leaves an infinitely long chamber in equilibrium with entering solution. The difference in

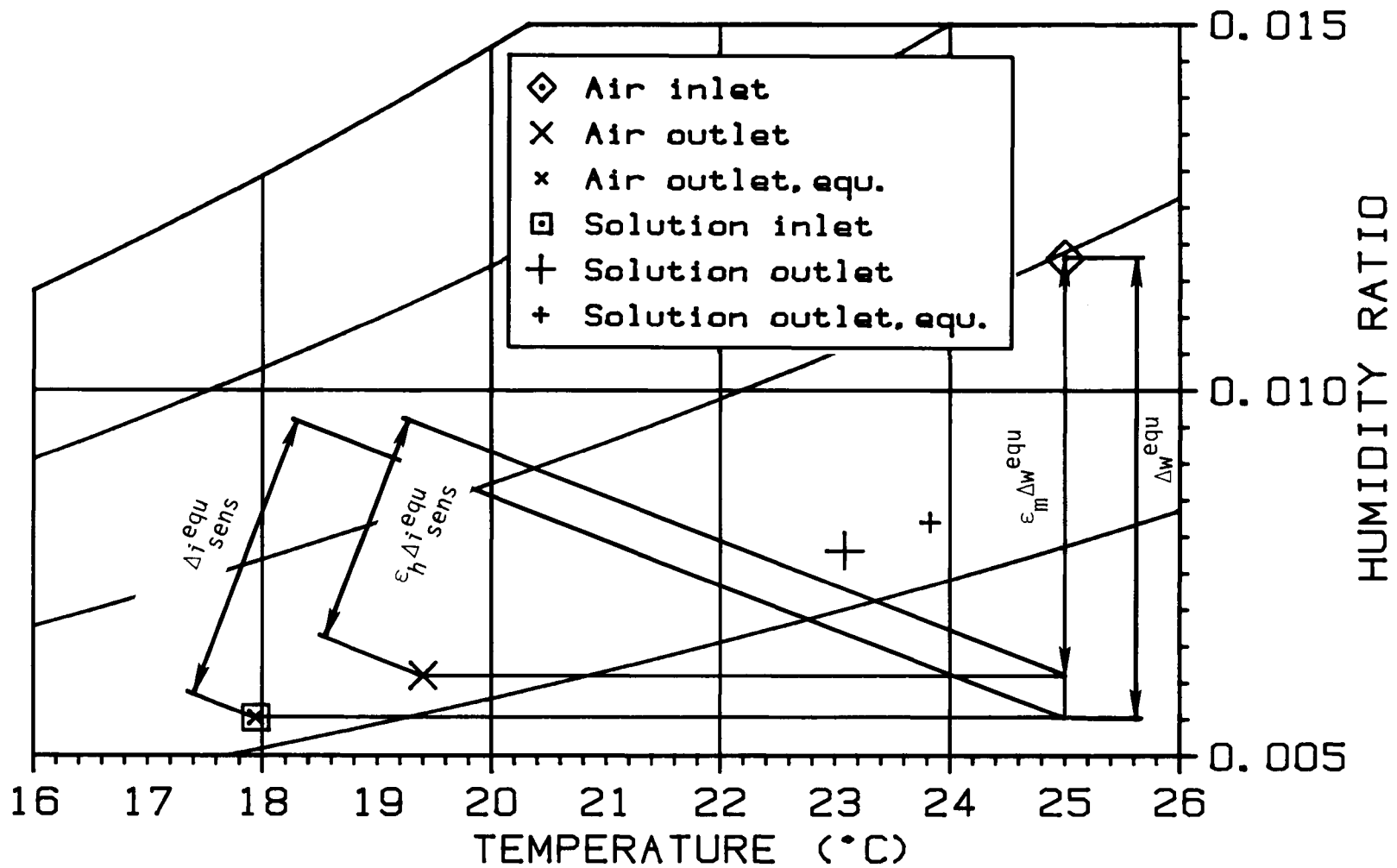


Figure 2.1 Graphical representation of the definitions of the effectiveness coefficients

enthalpy at constant humidity ratio between air inlet and outlet states and the difference in humidity ratio represent the exchanged sensible heat and mass, respectively. The air outlet state for a finite size of a heat and mass exchanger is obtained by multiplying both differences with the appropriate effectiveness factors. The effectiveness coefficients have to be determined empirically from experimental data. The fits for the Richmond data are given in Table 2.2.

	ϵ_h	ϵ_m
CONDITIONER	0.865	0.908
REGENERATOR	0.907	0.840

Table 2.2 Fitted effectiveness coefficients for the design point data.

It is of particular interest to know how much the two effectiveness coefficients vary with variations in the inlet states, the mass flow rates and, eventually, equipment sizes. Unfortunately, no experimental data were available. However, studies using the more elaborate model described in the next section were carried out for this purpose. Once the dependence of the effectiveness coefficients on variations in inlet states is known (or estimated), a simple function using a few parameters (e.g., a spline function) can be fit. The implementation of this function together with the model gives a sufficiently precise model requiring minimal computational expense.

2.2 Finite Step Integration Model

In the previous section, a simple model for the heat and mass exchanger was proposed. It was designed to use minimal computational effort to allow for its application in long term simulations. However, as will be shown later it is applicable only in the neighborhood of its design point. Furthermore, it does not give the intermediate states inside the chamber.

To overcome these drawbacks, a second model was developed. An integration along the length of the chamber allows the calculation of intermediate states. Consider the small element dA in Figure 2.2. The solution mass flow is entering at point 1 and leaving at point 2, whereas the counterflow air stream enters at point 2 and leaves at point 1. In this element, a mass flow $d\dot{m}$ and a heat flow $d\dot{Q}$ are exchanged.

The following assumptions are made:

- one dimensional flow of both phases,
- the liquid phase is well mixed within each element dA ,
- the gas in immediate contact with the solution is in thermodynamic equilibrium with the solution,
- the heat and mass exchanger is adiabatic.

The overall mass balances for this element can be written as

$$\dot{m}_{s2} = \dot{m}_{s1} - d\dot{m} \quad (2.20)$$

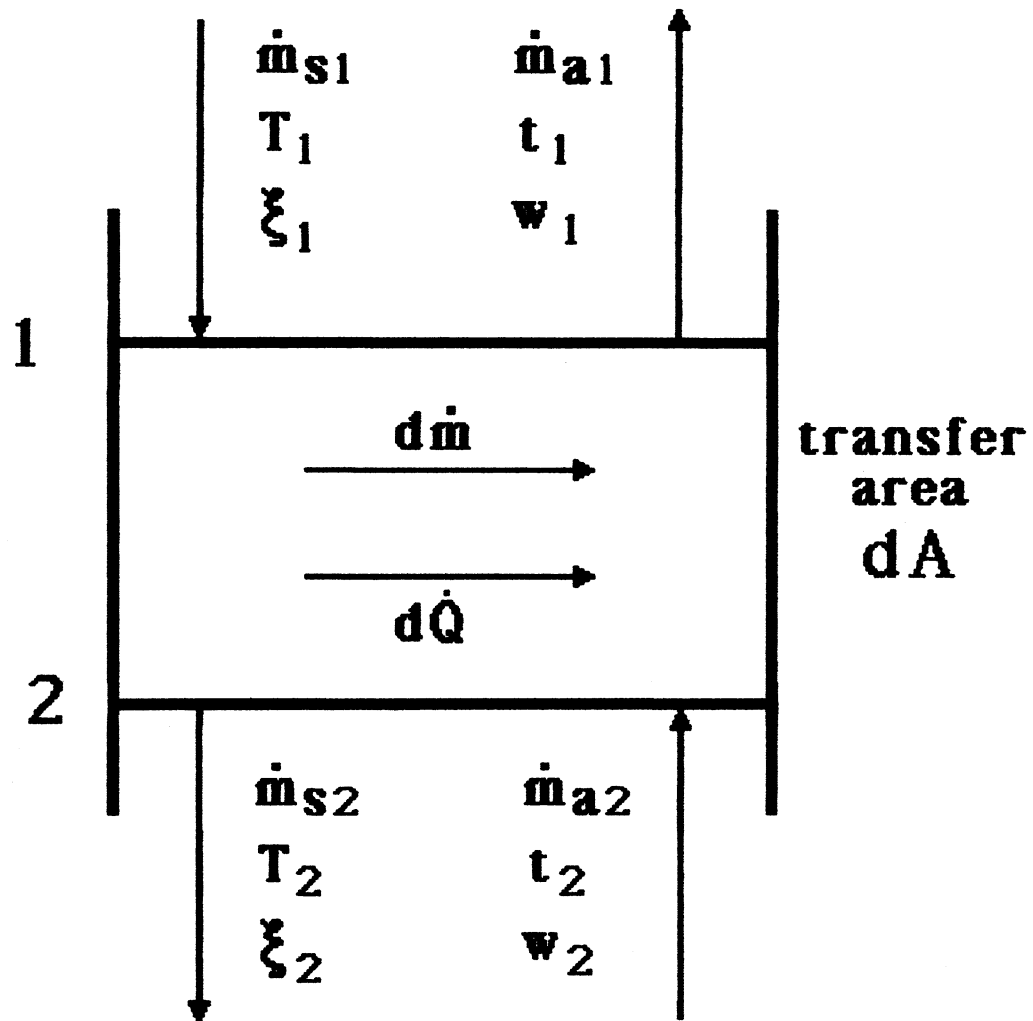


Figure 2.2 Heat and mass exchanger element with transfer area dA

and

$$\dot{m}_{a1} = \dot{m}_{a2} \quad (2.21)$$

Here, \dot{m}_a is constant, since it is the mass flow rate of dry air. In addition, mass balances can be written for the salt

$$\xi_2 \dot{m}_{s1} = \xi_2 \dot{m}_{s1} \quad (2.22)$$

or,

$$\xi_2 = \xi_1 \frac{\dot{m}_{s1}}{\dot{m}_{s1} - d\dot{m}} \quad (2.23)$$

and for the water vapor in the air:

$$w_1 \dot{m}_{a1} = w_2 \dot{m}_{a2} + d\dot{m} \quad (2.24)$$

or, with Equation (2.21)

$$w_1 = w_2 + \frac{d\dot{m}}{\dot{m}_{a2}} \quad (2.25)$$

Furthermore, the two energy balances are

$$\dot{m}_{s2} I_2 = \dot{m}_{s1} I_1 - d\dot{Q} \quad (2.26)$$

for the solution and

$$\dot{m}_a i_1 = \dot{m}_a i_2 + d\dot{Q} \quad (2.27)$$

for the air.

As Peng [16] shows, the heat and mass transfer can be assumed gas-phase controlled. Hence, the transfer equations can be written as

$$d\dot{m} = h_m (W - w) dA \quad (2.28)$$

and

$$d\dot{Q}_{\text{sens}} = h_h (T - t) dA \quad (2.29)$$

where h_m (units: $\text{kg/m}^2\text{-s}$) and h_h (units: $\text{kJ/m}^2\text{-s-K}$) are the mass and heat transfer coefficients, respectively, and dA is the equivalent transfer area of the element in consideration. Using the Lewis number relationship

$$Le = \frac{h_h}{h_m C_{p,a}} \quad (2.30)$$

Equation (2.29) can be written as

$$d\dot{Q}_{\text{sens}} = h_m C_{p,a} Le (T - t) dA \quad (2.31)$$

The latent energy transferred in the element can be expressed as

$$d\dot{Q}_{\text{lat}} = i_v d\dot{m} \quad (2.32)$$

Finally, the total energy transferred results as the sum of the transferred latent (2.32) and sensible (2.31) energy contributions

$$d\dot{Q} = h_m (i_v (W - w) + C_{p,a} Le (T - t)) dA \quad (2.33)$$

2.2.1 Implementation in an Algorithm

The heat and mass exchanger employs a counter current flow scheme. The integration has to be started at either the solution inlet or the air inlet. In either case, the related outlet state of the air or solution, respectively, has to be known in advance. Thus, the algorithm of this model is an iterative one, increasing the computational expense considerably. A routine of the MINPACK [10] package, developed at Argonne National Laboratories, was applied to search for the outlet states which minimize the weighted square error between the computed and the given inlet states:

$$\text{MINIMIZE} \left[\left(\frac{t - \hat{t}}{t} \right)^2 + \left(\frac{w - \hat{w}}{w} \right)^2 \right] \quad (2.34)$$

This algorithm turned out to be very robust with respect to bad guesses as long as inlet and outlet states at the starting point of the integration were not too close together. In this case, it was necessary to start the integration at the other end of the chamber.

The integration was carried out by means of a fourth-order Runge-Kutta algorithm [11]. The stepsize chosen had to be small to avoid that the assumption of constant solution (T,W) and air (t,w) states over each element dA did not cause erroneous results. Typically, 200 elements were needed.

In Figures 2.3 through 2.6, the results of the finite step integration model are presented. Figure 2.3 shows the states of air and solution along their path through the heat and mass exchanger for

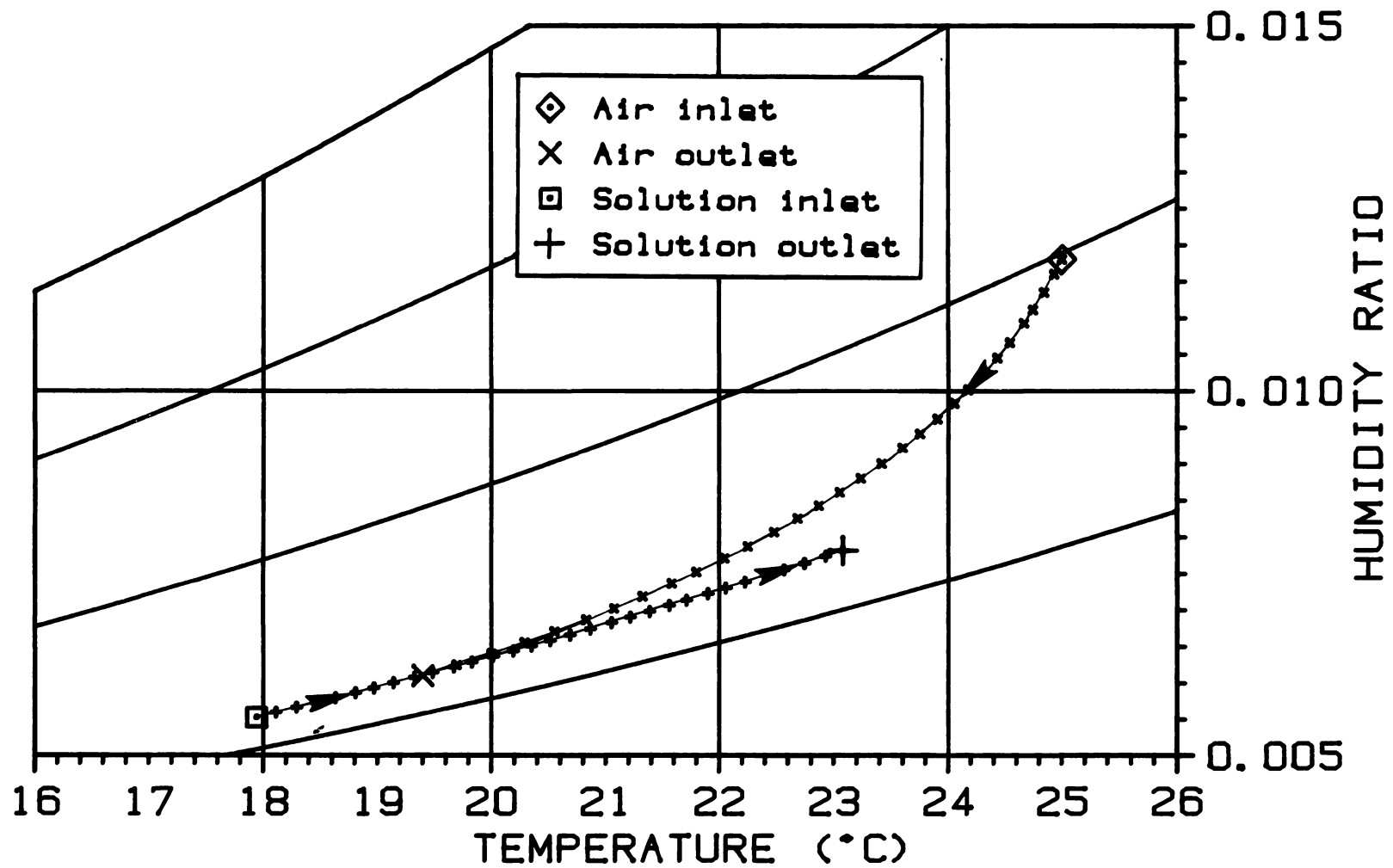


Figure 2.3 Paths of air and solution states in heat and mass exchanger, conditioner design point

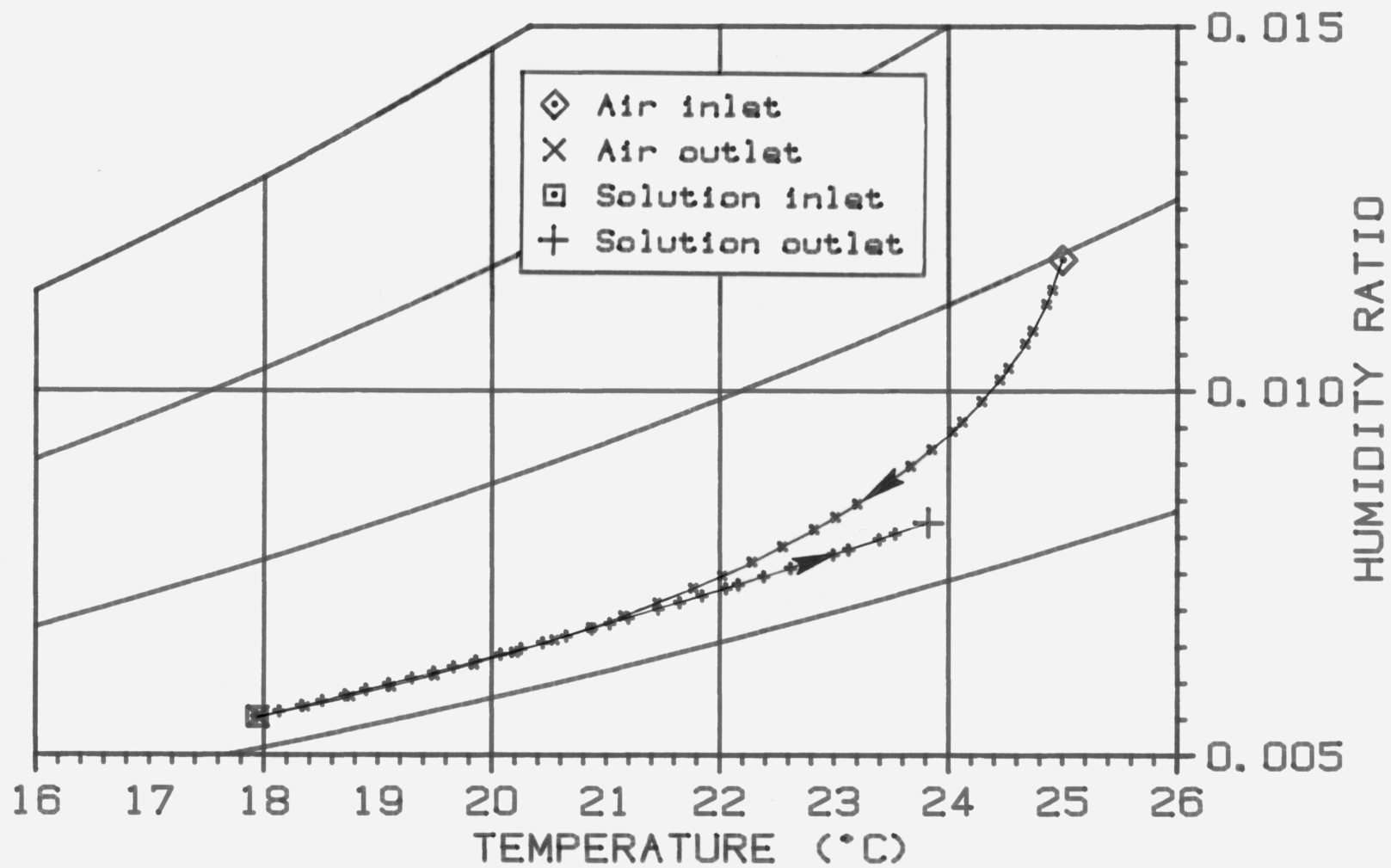


Figure 2.4 Paths of air and solution states in heat and mass exchanger, conditioner point, large transfer area

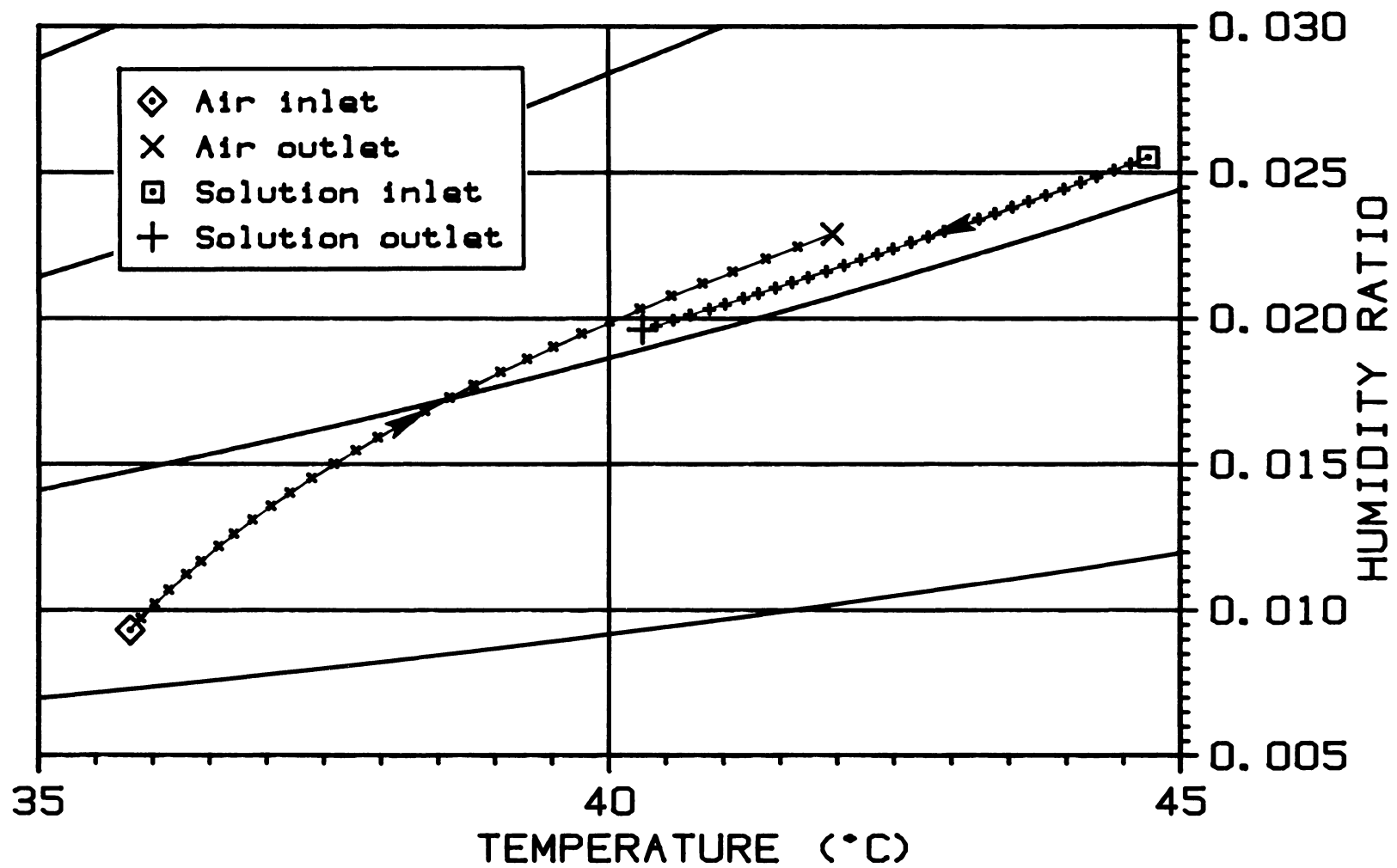


Figure 2.5 Paths of air and solution states in heat and mass exchanger, regenerator design point

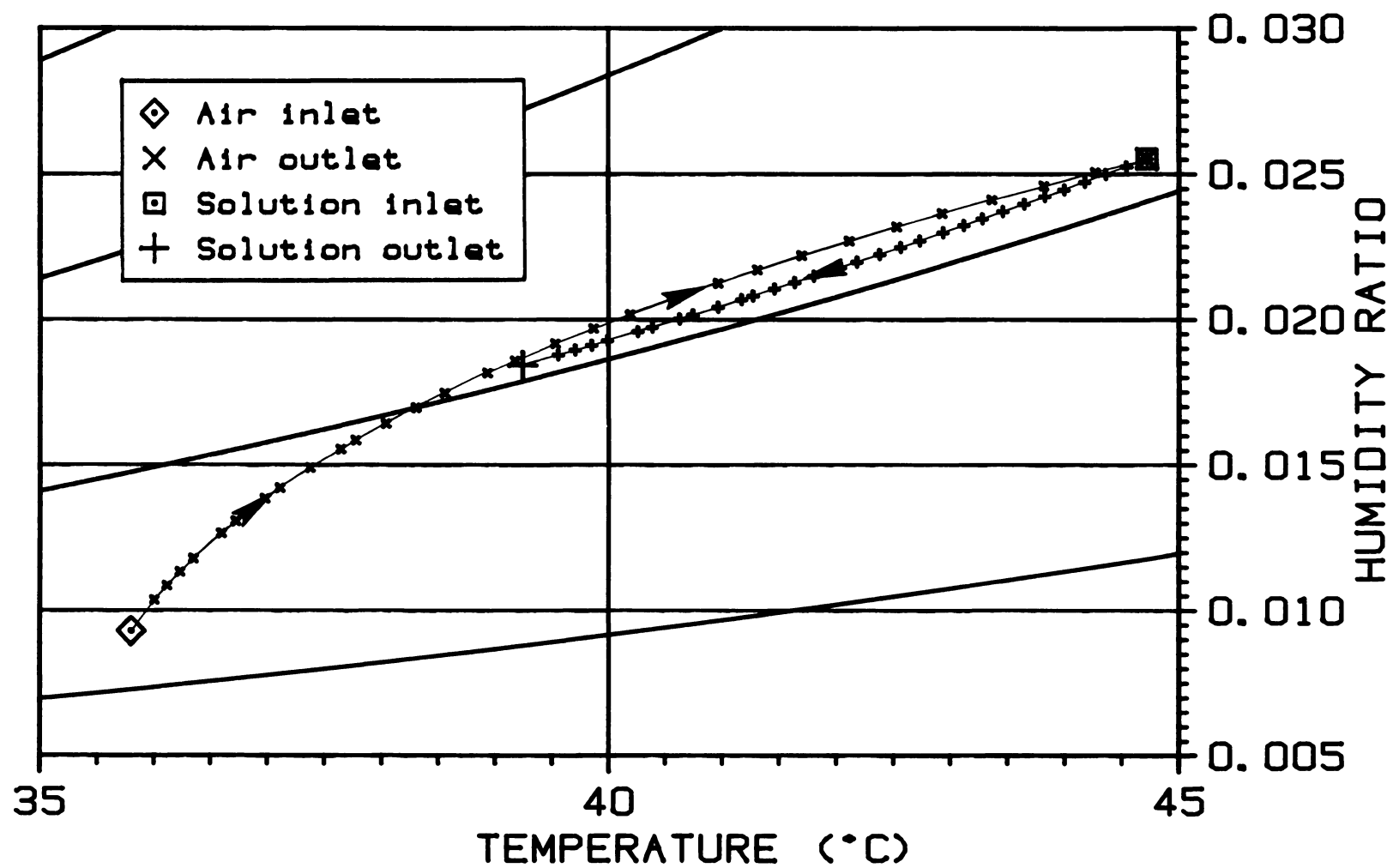


Figure 2.6 Paths of air and solution states in heat and mass exchanger, large transfer area

the Richmond conditioner data. For very large transfer areas, the air leaves in equilibrium with the entering solution as shown in Figure 2.4. Figures 2.5 and 2.6 are the corresponding figures for the given Richmond generator data.

2.2.2 Determination of the Heat and Mass Transfer Coefficients and the Area

There are still two parameters to be determined: the product of the heat transfer coefficient and area $h_h A$ and the product of the mass transfer coefficient and area $h_m A$. As shown in Equation (2.30), the heat transfer coefficient can be replaced by the mass transfer coefficient and the Lewis number. These two parameters represent the degree of freedom of the system. For given inlet states, only one outlet state is free to vary, whereas the second outlet state is determined by the overall mass and energy balances. For given data, the two parameters can be fitted so that the model reproduces one given or measured outlet state.

For the two design points, one each for the conditioner and the regenerator, which are available for the heat and mass exchanger units of the Science Museum of Virginia, the values for the parameters are given in Table 2.3.

	Conditioner	Regenerator
$h_h A$	22	6.9
Le	1.2	2.2

Table 2.3 Fitted Parameters

A correlation can be found to include the dependence of the mass transfer coefficient on the air temperature. Ranz and Marshall [12] proposed, for the evaporation of water from droplets, the dimensionless relations:

$$Nu = 2.0 + 0.6 Pr^{1/3} Re^{1/2} \quad (2.35)$$

and

$$Sh = 2.0 + 0.6 Sc^{1/3} Re^{1/2} \quad (2.36)$$

where the Reynolds number is defined by

$$Re = \frac{vd\rho}{\mu} \quad (2.37)$$

the Prandtl number

$$Pr = \frac{\mu}{\alpha\rho} \quad (2.38)$$

and the Schmidt number

$$Sc = Le Pr \quad (2.39)$$

From a force balance for a sphere, its terminal velocity can be determined as

$$V = \sqrt{\frac{4}{3} g C_D \frac{\rho_s}{\rho_a} d} \quad (2.40)$$

and the Reynold number becomes

$$Re = \sqrt{\frac{4}{3} \frac{g \rho_s \rho_a d^3 C_D}{\mu^2}} \quad (2.41)$$

Combining Equations (2.36), (2.38), (2.39) and (2.41) and the definition of the Sherwood number

$$Sh = \frac{h_m d}{\rho_a D} \quad (2.42)$$

the mass transfer coefficient can be found as

$$h_m = 2.0 \rho_a D d^{-1} + 0.645 D \rho_a^{\frac{11}{12}} Le^{\frac{1}{3}} \alpha^{-\frac{1}{3}} g^{\frac{1}{4}} C_D^{\frac{1}{4}} \rho_s^{\frac{1}{4}} d^{\frac{1}{4}} \mu^{-\frac{1}{6}} \quad (2.43)$$

Air property correlations are readily available in reference [2]. Threlkeld [5] shows the Lewis number as a function of the mean temperature between a wetted surface and an air stream (Figure 2.7). The Lewis number exhibits a very small dependence on the mean temperature over the range of conditions encountered in this liquid desiccant system.

The equivalent mean droplet diameter, d , is heavily dependent on the droplet size distribution, which is not easily measured. The equivalent mean droplet diameter is kept here as a parameter. Hence the mass transfer coefficient has been replaced with another parameter yielding its variation with the mean temperature. In Figure 2.8 the mass transfer coefficient is plotted as a function of

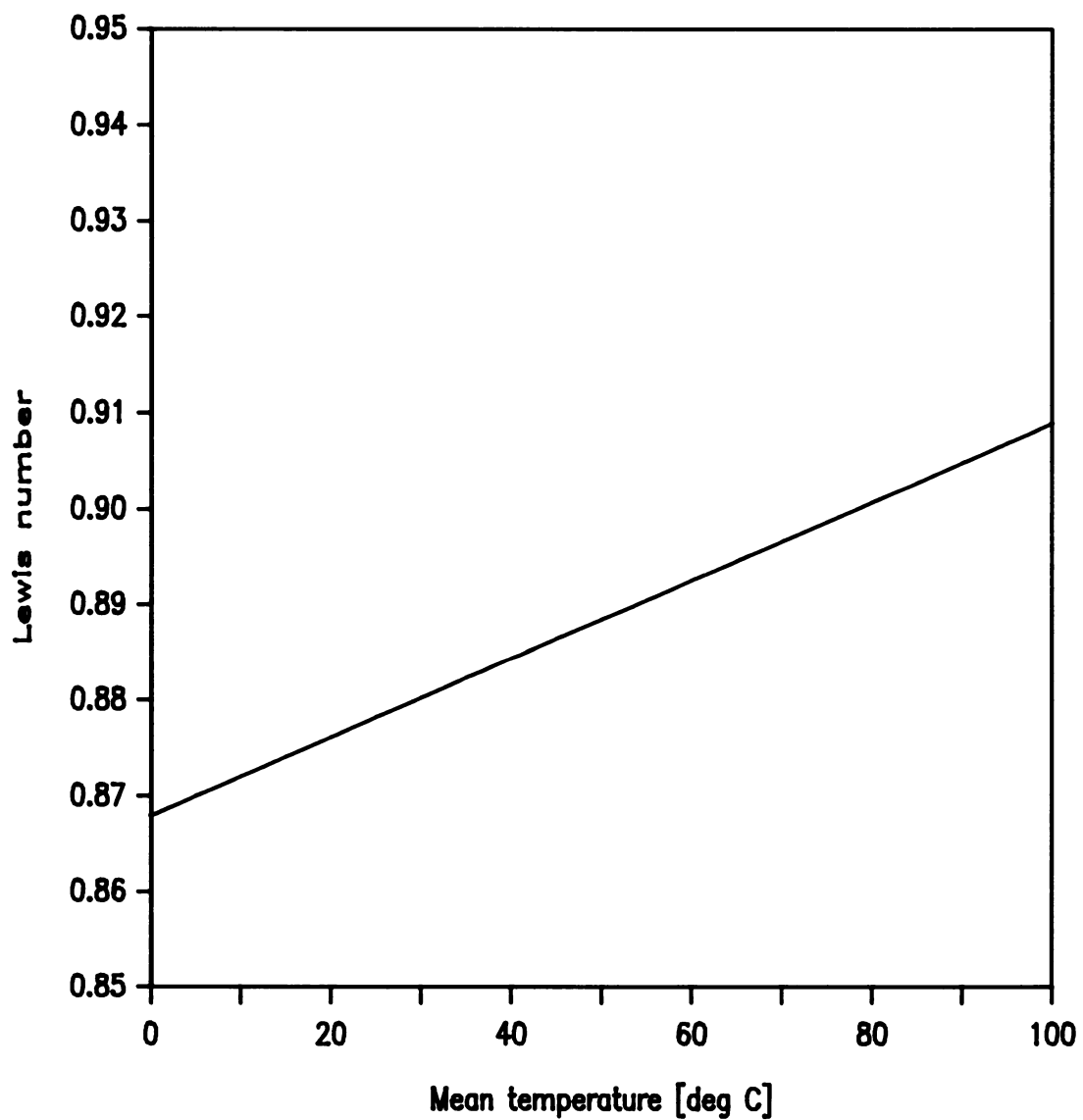


Figure 2.7 The Lewis number as a function of the mean temperature between solution and air

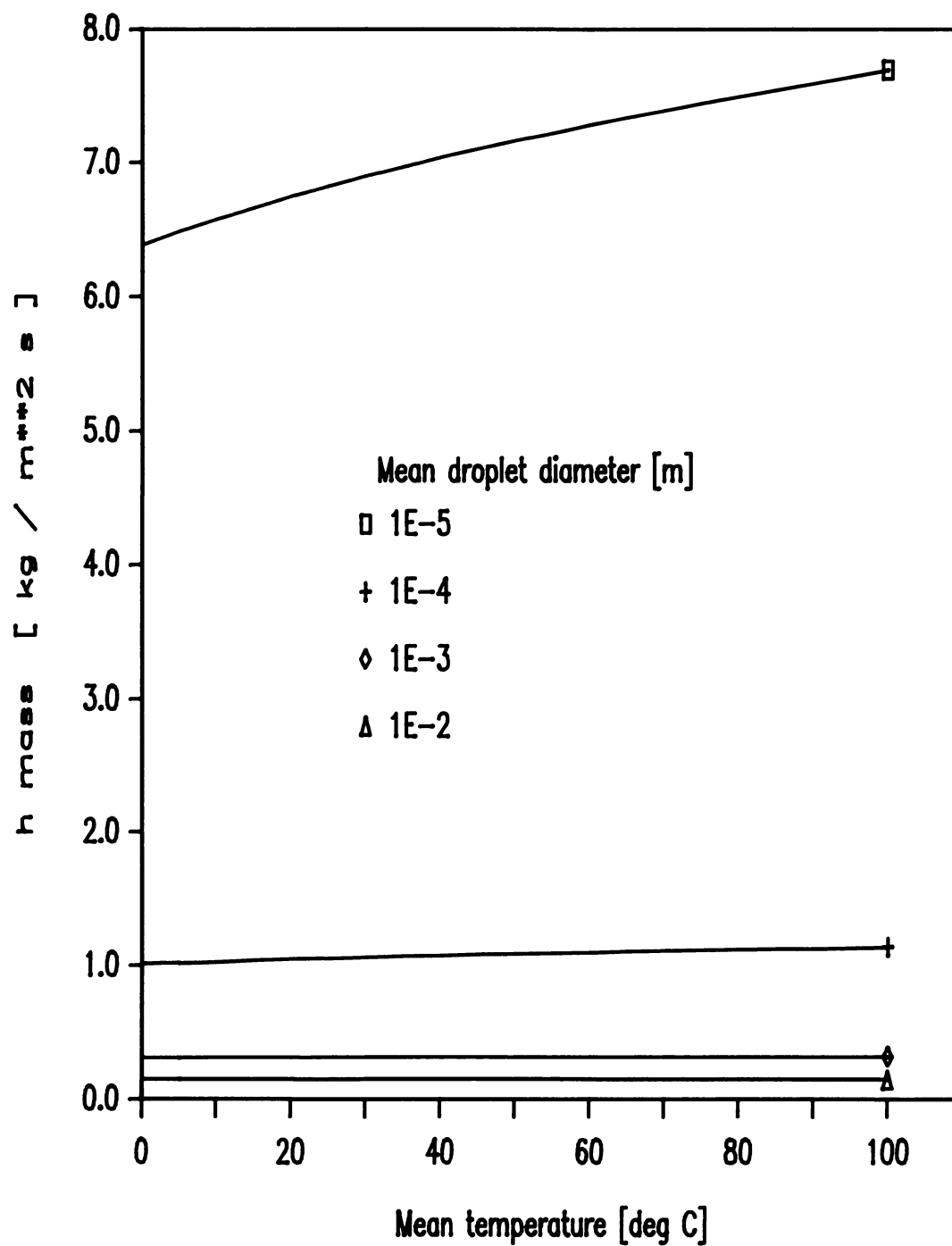


Figure 2.8 The mass transfer coefficient as a function of the mean temperature between solution and air

the mean temperature for various equivalent mean droplet diameters. It is a strong function of the mean droplet diameter, but only a weak function of the temperature. A mean droplet diameter of 10^{-4} is assumed in this study.

2.3 Comparison Of The Two Models

For lack of experimental data it was not possible to verify either of the models presented in the previous section. In particular, it was not possible to determine the variations of the effectiveness factors used in the simple model with varying inlet states. However, a comparison of the two models allowed for a first estimation of those variations.

Two parameters considered to be the most important ones, i.e., the solution inlet temperature and the air mass flow rate, were chosen for the comparison. The solution temperature was the only inlet state which varied in the simulation studies and was hence of particular interest. It was varied $\pm 5^{\circ}\text{C}$ around the design point of 17.94°C for the conditioner and $+12.5^{\circ}\text{C}$ and -7.5°C around the design point of 47.5°C for the regenerator.

The solution and air outlet states were calculated by means of the finite difference model. Subsequently, the effectiveness coefficients for the equilibrium model were fitted to those outlet states. The fitted coefficients plotted versus the solution inlet temperature are shown in Figure 2.9. The errors in the predicted

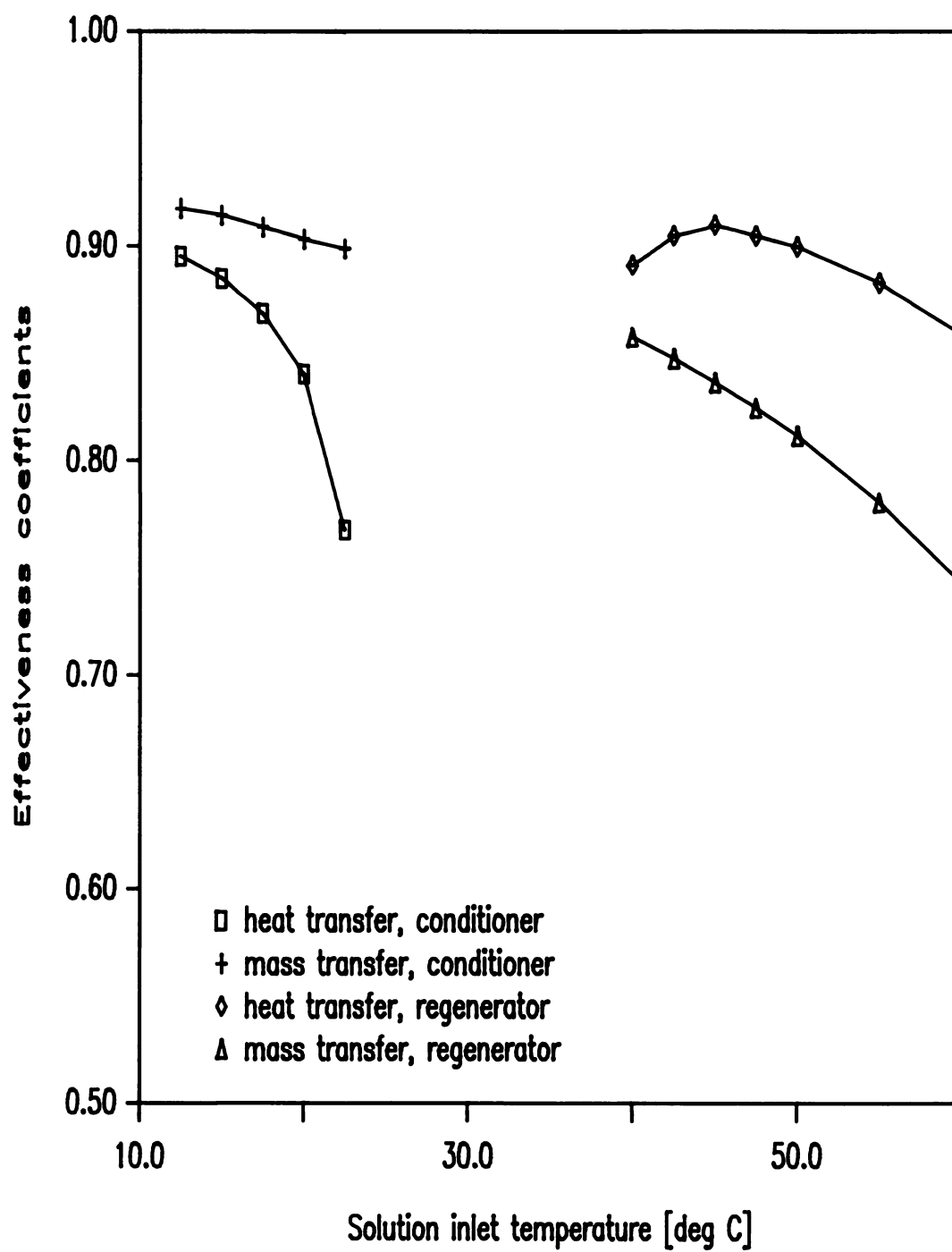


Figure 2.9 Fitted effectiveness coefficients versus solution inlet temperature

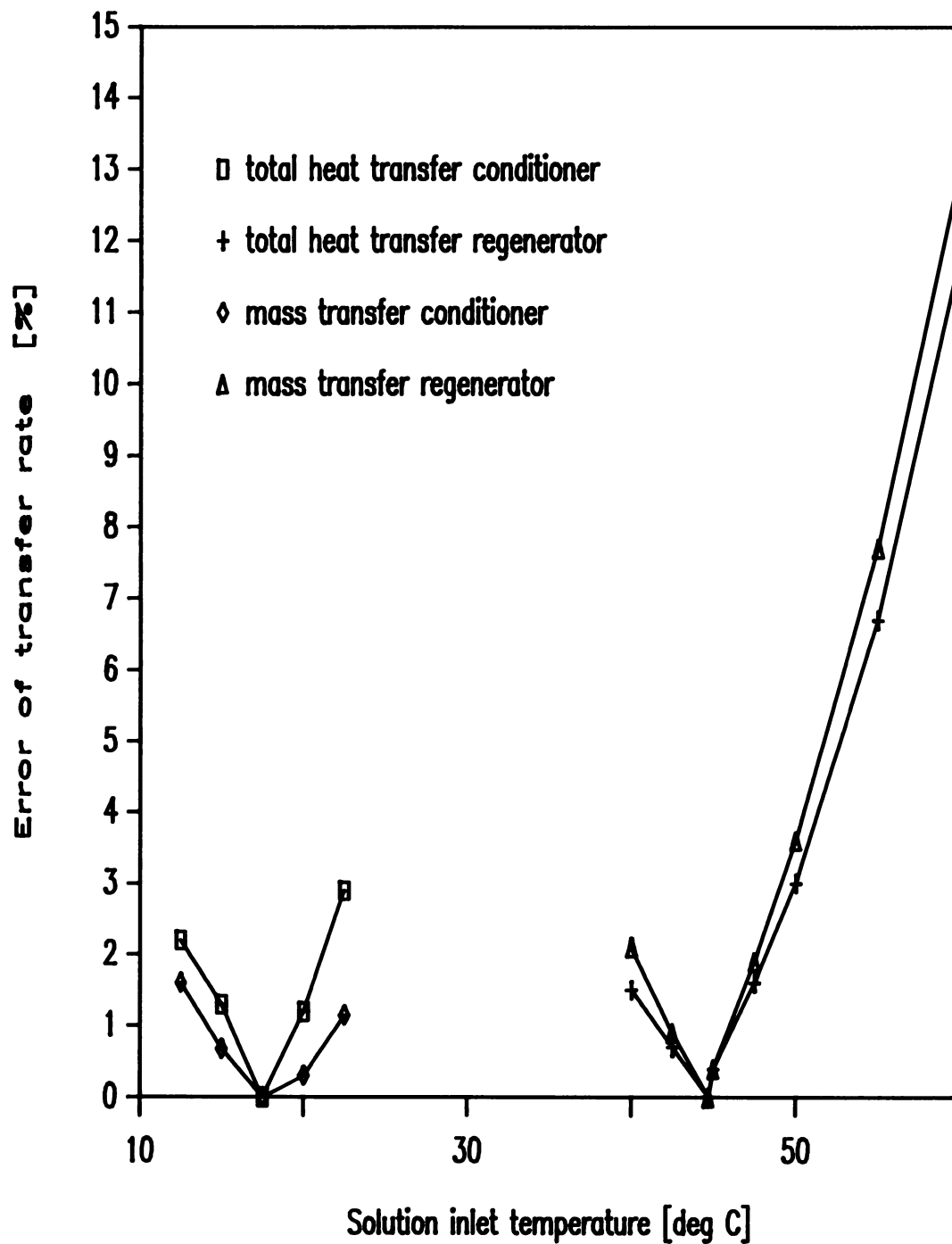


Figure 2.10 Error in transfer rates versus solution inlet temperature holding the effectiveness coefficients constant

transfer rates when the effectiveness coefficients are kept constant (Table 2.2) are presented in Figure 2.10. Within a range of $\pm 5^\circ\text{C}$ around the design point, the errors are less than 5 percent.

The second parameter studied is the mass flow rate ratio. Again the solution and air outlet states were computed by means of the finite difference model. The effectiveness coefficients were then fitted to these outlet states. As can be seen from Figure 2.11, the variations in the effectiveness coefficients are much larger but still follow a similar pattern. In contrast to the variation in the solution inlet temperature, the errors in the transfer rates holding the effectiveness factors constant are much larger (Figure 2.12).

Finally, it was studied whether it is possible to predict the effectiveness coefficients using a ε -NTU approach similar to the one used in heat exchanger design (Kays and London [6]). The capacity rates for heat transfer are given by

$$\dot{C}_{h,a} = \dot{m}_a \left. \frac{\partial i}{\partial t} \right|_i = \dot{m}_a C_{p,a} \quad (2.44)$$

and

$$\dot{C}_{h,s} = \dot{m}_s \left. \frac{\partial I}{\partial T} \right|_i = \dot{m}_s C_{p,s} \quad (2.45)$$

Analogously, the capacity rates for the mass transfer can be written as:

$$\dot{C}_{m,a} = \frac{\partial}{\partial w} (\dot{m}_a w) \Big|_i = \dot{m}_a \quad (2.46)$$

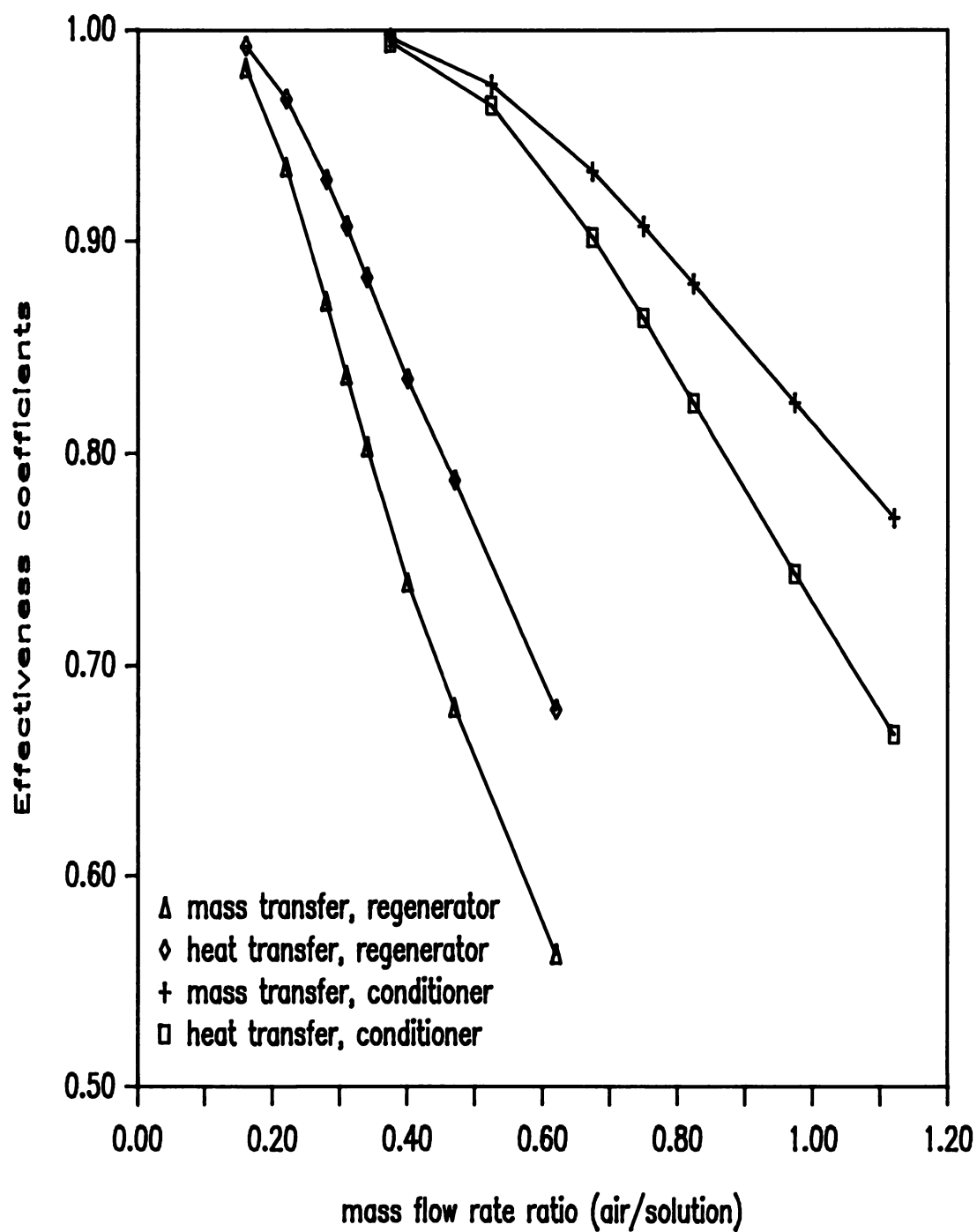


Figure 2.11 Fitted effectiveness coefficients versus mass flow rate ratio

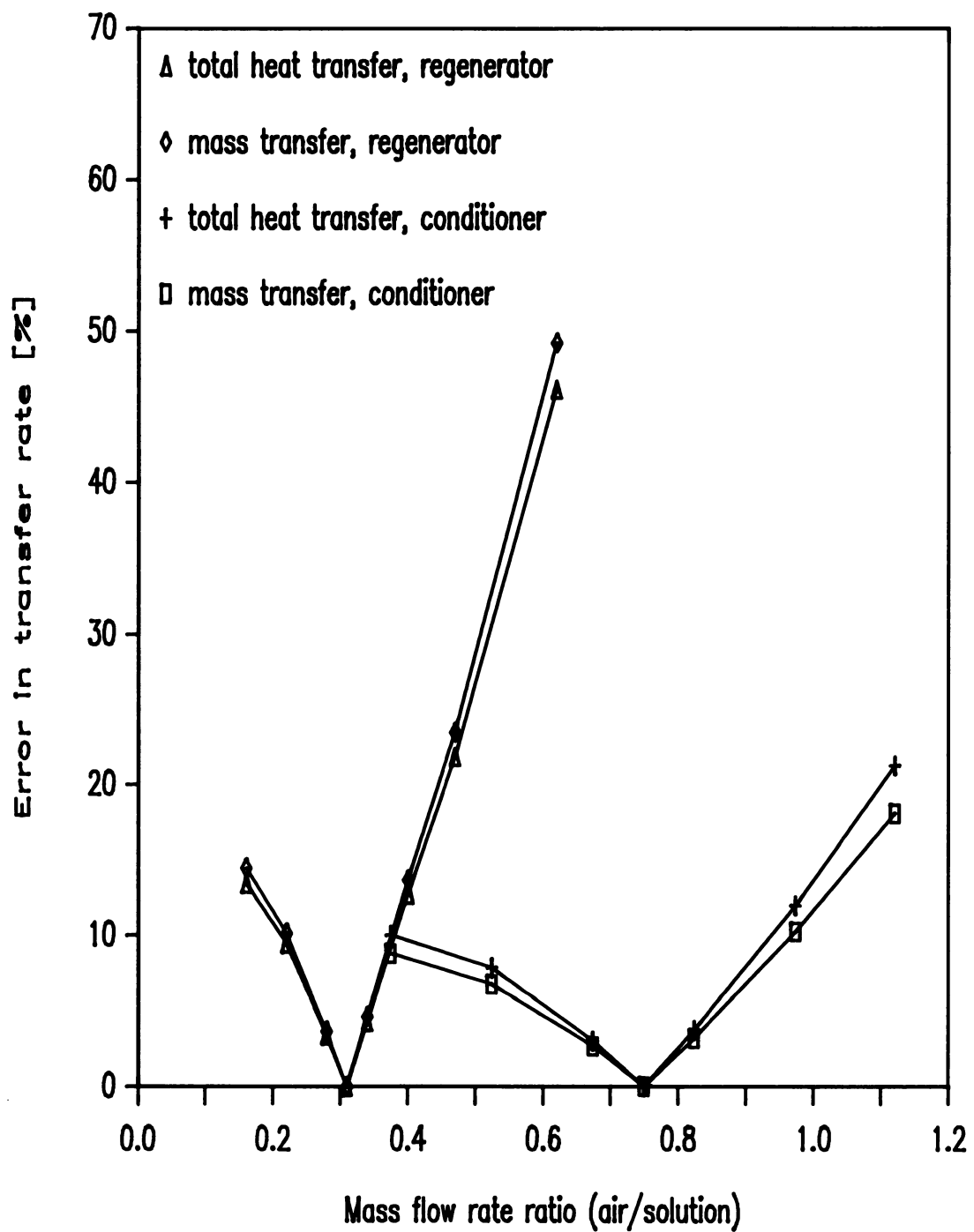


Figure 2.12 Error in transfer rates versus mass flow rate ratio holding the effectiveness factors constant

since $\dot{m}_a w$ is the mass flow rate of water carried by the air, and

$$\dot{C}_{m,s} = \frac{\partial}{\partial w} ((1-\xi) \dot{m}_s) \Big|_i = -\dot{m}_s \frac{\partial \xi}{\partial w} \Big|_i \quad (2.47)$$

Since the humidity ratio of air (w) in equilibrium with solution at (T, ξ) can be calculated via the water vapor pressure correlation, the differential $\partial \xi / \partial w$ can be evaluated. All capacity rates are calculable only for the inlet states because the outlet states are not yet determined.

The number of transfer units NTU can be determined from

$$NTU_h = \frac{h_h A}{C_{h,min}} = \frac{h_m Le C_{p,a} A}{C_{h,min}} \quad (2.48)$$

and

$$NTU_m = \frac{h_m A}{C_{m,min}} \quad (2.49)$$

For h_m and Le , the fits obtained from the finite difference model, were used (Table 2.3).

Finally, the ϵ -NTU relationship for countercurrent flow (as given, for example, by Kays and London [5])

$$\epsilon = \frac{1 - \exp(-NTU (1 - C_{min}/C_{max}))}{1 - (C_{min}/C_{max}) \exp(-NTU (1 - C_{min}/C_{max}))} \quad (2.50)$$

can be used to obtain estimates of the effectiveness coefficients. C_{min} and C_{max} are the appropriate smaller and larger capacity rates respectively. Values for the effectiveness factors obtained from

this procedure and the one using fits to reproduce results from the finite difference model are shown in Figures 2.13 and 2.14. Although each set of two corresponding curves have similar shapes, their values are quite different. Hence, it seems to be more promising to use a curve fit.

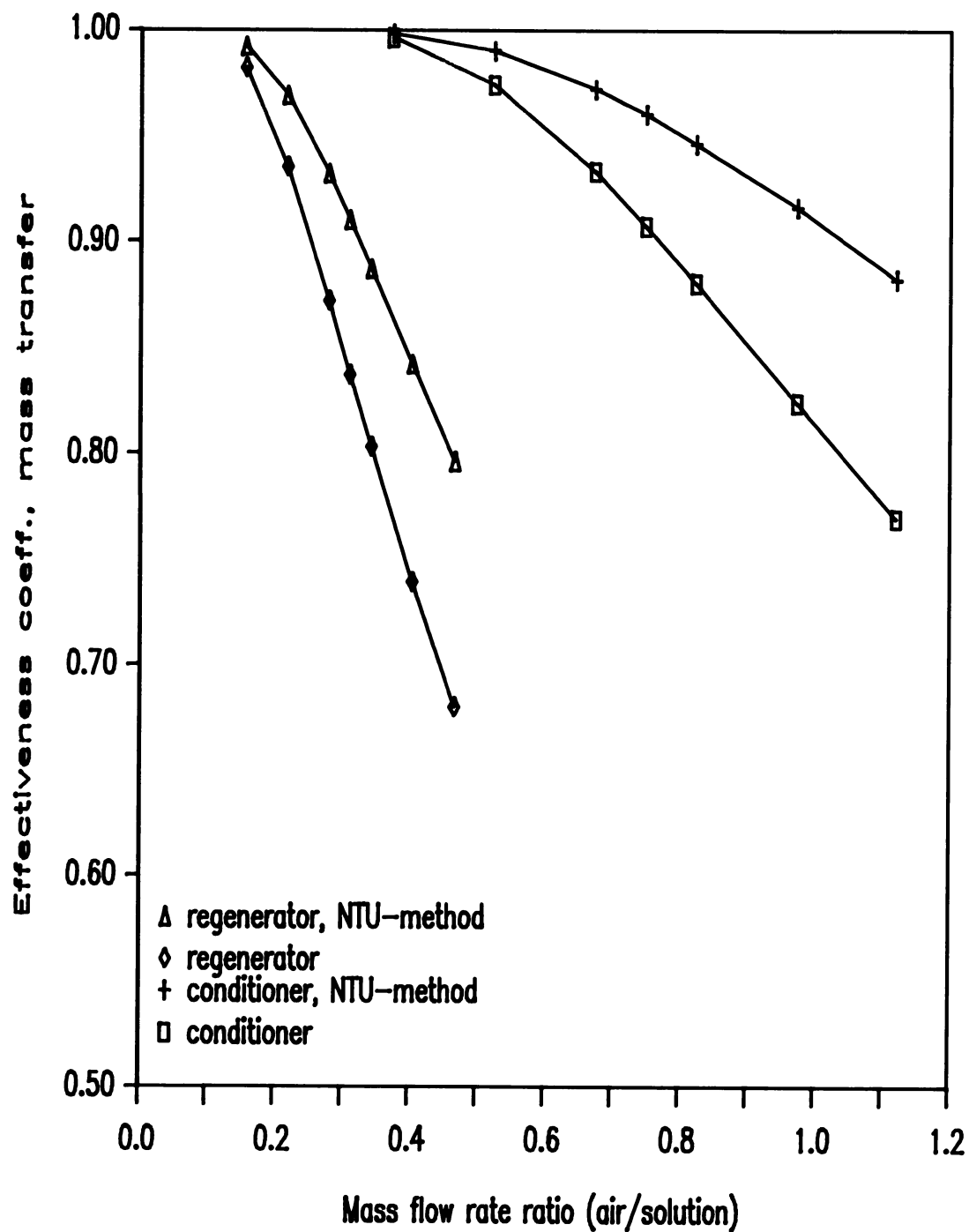


Figure 2.13 Effectiveness coefficients for mass transfer versus mass flow rate ratio as predicted by the finite step integration model and the NTU method

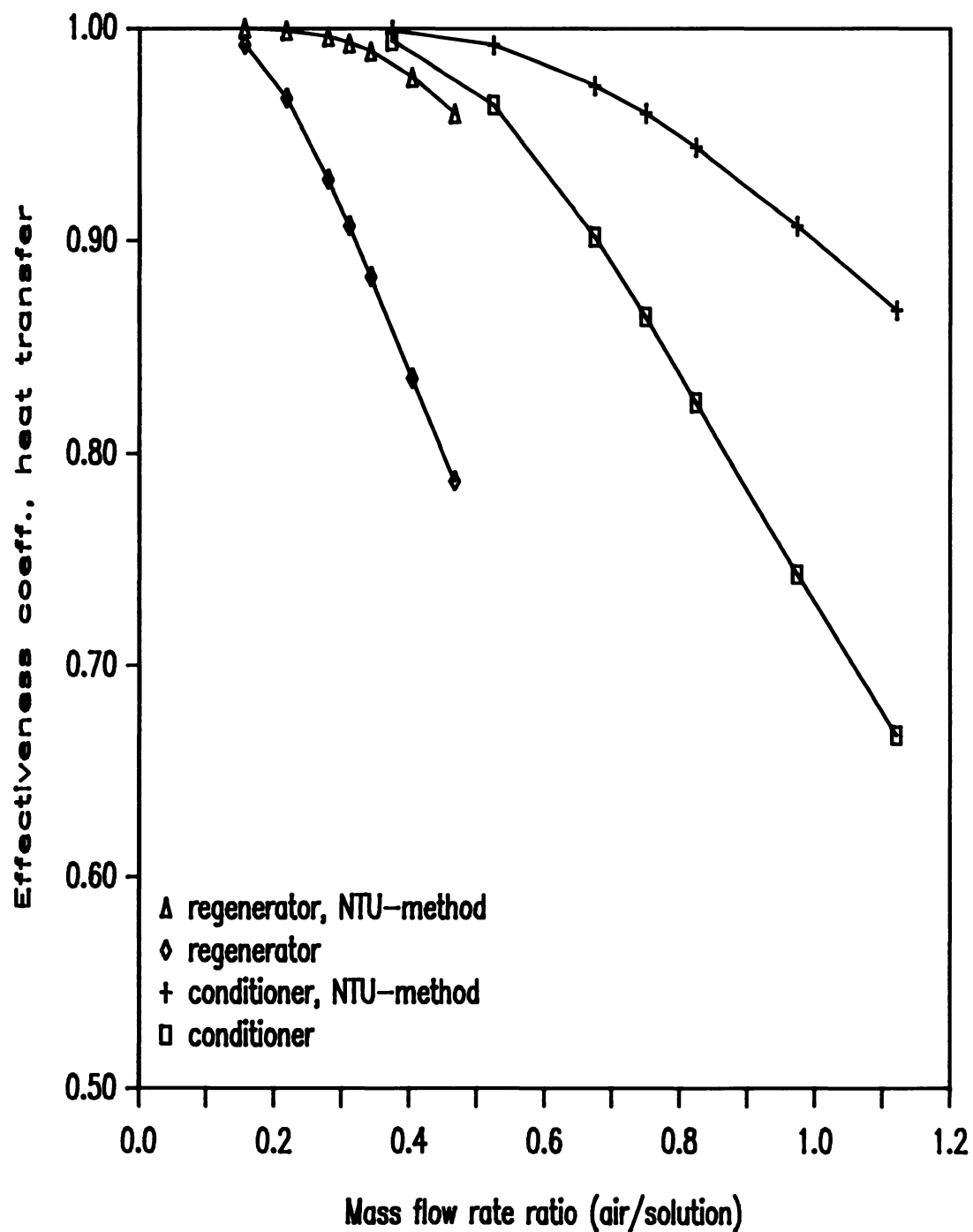


Figure 2.14 Effectiveness coefficients for heat transfer versus mass flow rate as predicted by the finite step integration model and the NTU method

CHAPTER 3 SYSTEM SIMULATION

The transient simulation program TRNSYS [9] was chosen for the system simulation. The modular structure of TRNSYS allows the combination of existing component models and those provided by the user. The standard component library of TRNSYS contains over 50 modules. Nevertheless, it was necessary to develop new components. Most of the new components were developed for this particular application. These components are described in Section 3.2

The simulation involves a number of flows of moist air and of salt solution. The knowledge of the physical properties of these two mixtures is essential for the execution of the simulation. Therefore it was necessary to develop functions which allow the calculation of physical properties such as enthalpy and water vapor pressure. Such property functions were prepared for air-water vapor mixtures and LiBr-water and LiCl-water solutions and are presented in Section 3.1.

In Section 3.3 the simulation model of the HVAC equipment, including the desiccant cycle installed at the Science Museum of Virginia, is described. Simplifications and assumptions made during the process of modeling are stated. The results of a simulation using this model are compared with the data given for the design point.

Finally, results of simulation runs with varying system parameters are presented and analyzed in Section 3.4. In addition, predicted hourly costs of operation for various modes of operation are discussed.

3.1 Physical Property Functions

3.1.1 Air-water vapor mixtures

A function for the enthalpy of moist air can be written as

$$i = c_{p,a} t + w(i_v + i_{w,liq}) \quad (3.1)$$

where i is the specific enthalpy of moist air in kJ/kg dry air. The specific heat capacity of air is 1.003 kJ/kg-°C. The enthalpy of liquid water and the enthalpy of vaporization are functions of water temperature. These functions were obtained as

$$i_{w,liq} = -1145.7 \frac{\text{kJ}}{\text{kg}} + 4.194 \frac{\text{kJ}}{\text{kg}} T \quad (3.2)$$

$$i_v = 3182.1 \frac{\text{kJ}}{\text{kg}} - 2.48 \frac{\text{kJ}}{\text{kgK}} T \quad (3.3)$$

The coefficients were fitted to data given in [22]. A function for the saturation pressure was found as

$$\log_{10} p^{\text{sat}} = 10.094 - 1632.6 \text{ K } \frac{1}{T} - 99377.5 \text{ K}^2 \frac{1}{T^2} \text{ for } T > 275\text{K} \quad (3.4)$$

$$\log_{10} p^{\text{sat}} = 11.160 - 1966.8 \text{ K } \frac{1}{T} - 88060.9 \text{ K}^2 \frac{1}{T^2} \text{ for } T \leq 275\text{K} \quad (3.5)$$

The same reference gives a relation between water vapor pressure and humidity ratio

$$w = 0.622 \frac{P_w}{P_{amb} - P_w} \quad (3.6)$$

3.1.2 Lithium Bromide-Water Solutions

Correlations for the enthalpy of and the partial pressure of water vapor above LiBr-water solutions as functions of solution temperature and concentration are given in the ASHRAE Handbook of Fundamentals [3]. These correlations were implemented. Note, that the coefficients A_3 and B_3 for the vapor pressure correlation in SI units on page 17.142 [3] are missing. Both coefficients can be found from the correlation in English units on page 17.72 as

$$A_3 = 1.97668 \cdot 10^{-5} \quad (3.7)$$

$$B_3 = -7.9509 \cdot 10^{-4} \quad (3.8)$$

To enable the alternate use of both the correlation for LiBr-water solutions and the one for LiCl-water solutions, the corresponding functions were given the same name.

3.1.3 Lithium Chloride-Water Solutions

3.1.3.1 Water Vapor Pressure

Data for the water vapor pressure above a LiCl-water solution were given by Johnson & Molstad [17] for 30, 50 and 70°C, in the CRC Handbook [24] for 100°C and in the Gmehlin Handbook [23] for 18°C and

25°C solution temperature. All these data were curve-fitted to the same function as used in the ASHRAE Handbook of Fundamentals [3] for LiBr-water solutions

$$T_w = A(T_s - 273.15) + B + 273.15 \quad (3.9)$$

where

$$A = 1.000 - 0.1328\xi + 4.822 \cdot 10^{-2}\xi^2 - 0.5076\xi^3 \quad (3.10)$$

and

$$B = -0.4383 + 14.14\xi - 224.5\xi^2 + 123.2\xi^3 \quad (3.11)$$

T_w is the temperature in Kelvin at which pure water has the same vapor pressure as solution of concentration ξ at the temperature T_s .

3.1.3.2 Enthalpy of Lithium Chloride-Water Solutions

The enthalpy of a solution can be written as

$$I = I^{\text{ref}} + \Delta I_{s,e} + \int_{t_{\text{ref}}}^t c_{p,s} dt \quad (3.12)$$

where I^{ref} is an arbitrary reference enthalpy, $\Delta I_{s,e}$ is the integral heat of solution at the reference temperature and $c_{p,s}$ is the specific heat capacity of the solution. Data for the integral heat of solution at 25°C can be found in [19]. These data were curve-fitted to a fourth order polynomial in concentration

$$\Delta I_{s,e} = -0.8759 - 839.9\xi - 61.54\xi^2 + 1978.6\xi^3 \left[\frac{\text{kJ}}{\text{kg}}\right] \quad (3.13)$$

A correlation for the specific heat capacity was introduced by Uemura [20]. Since this correlation is a polynomial, it can be integrated analytically. The reference enthalpy I^{ref} was chosen as 104.75 kJ/kg, so that the enthalpy of the solution at 0% concentration (pure water) and 0°C is 0 kJ/kg.

3.2 TRNSYS Component Models

Most of the existing TRNSYS components are designed to handle only two flow variables, i.e., temperature and mass flow rate. However, in this simulation almost all of the flow streams have a third flow variable associated with them. This third flow variable is the humidity ratio (w) for air streams or the salt concentration (ξ) in the case of salt solution streams. Only cooling or heating water streams can be described by two flow variables. Thus most of the components used in this simulation had to be developed. The newly designed components do not include their own property functions but call the standardized property functions described in Section 3.1. Information flow diagrams similar to those in the TRNSYS manual [9] are shown for each component description. Listings of the FORTRAN source code of all TRNSYS components presented in the following sections can be found in Appendix B.

3.2.1 The Heat and Mass Exchanger Model

The model based on an equilibrium approach and effectiveness coefficients was implemented as a TRNSYS component subroutine. The theory of the heat and mass exchanger was presented in Section 2.1. The results of the comparison of the two models (Section 2.3) were used to obtain functions for the effectiveness coefficients as a function of the solution inlet temperature for the regenerator

$$\epsilon_h = 0.217 + 2.98 \cdot 10^{-2} T_i - 3.22 \cdot 10^{-4} T_i^2 \quad (3.14)$$

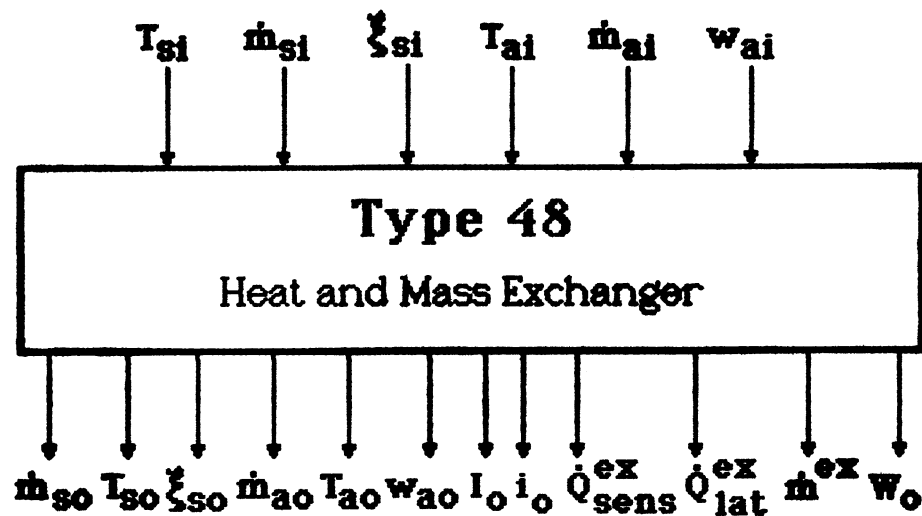
$$\epsilon_m = 0.853 + 3.91 \cdot 10^{-3} T_i - 9.52 \cdot 10^{-5} T_i^2 \quad (3.15)$$

and for the conditioner

$$\epsilon_h = 0.368 + 6.72 \cdot 10^{-2} T_i - 2.19 \cdot 10^{-3} T_i^2 \quad (3.16)$$

$$\epsilon_m = 0.829 + 1.06 \cdot 10^{-2} T_i - 3.35 \cdot 10^{-4} T_i^2 \quad (3.17)$$

The TRNSYS model contains all proposed pairs of equilibria. However, only the first pair, i.e., the equilibrium at the solution inlet (Table 2.1) occurs when using the data for the Science Museum of Virginia. The information flow diagram of the heat and mass exchanger is shown in Figure 3.1.



Parameters:

1. Mode (1 = regenerator,
2 = conditioner)

Figure 3.1 Information flow diagram for the TRNSYS component "heat and mass exchanger"

3.2.2 The Heat and Mass Exchanger Sump Model

Although the heat and mass exchanger and its sump are physically one device, they were modeled separately, since they have different characteristics. The heat and mass exchanger by itself is a heat and mass transfer device with negligible capacitance, whereas the sump has basically both heat and mass capacitance.

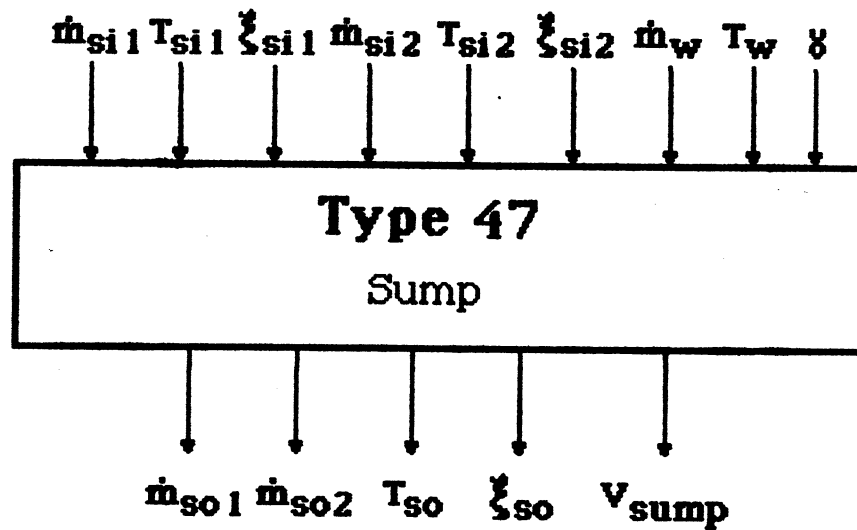
Two solution streams flow into the sump; a large one coming from the spray chamber and a small one coming from the second sump through the solution interchanger. A pump draws a constant volume flow rate from the sump. This leaving solution stream is split into a large stream recycled to the spray chamber and a small one which is pumped through the solution interchanger to the second sump.

The sump has a very small time constant, i.e., it has a small mass capacity compared with the mass flow rate. However, it was necessary to include this model into the simulation. The method of successive substitution used in TRNSYS does not always guarantee convergence. Without the sump model, convergence was not achieved. Due to the small time constant the sump can be considered well mixed. With this assumption, the overall mass balance can be written as

$$\frac{d m_s}{dt} = \sum \dot{m}_{si} - \dot{m}_{so} \quad (3.18)$$

The salt balance

$$\frac{d m_{salt}}{dt} = \sum \dot{m}_{si} \xi_i - \dot{m}_{so} \xi \quad (3.19)$$



Parameters:

1. Total volume flow rate out of sump

Initial values:

1. Solution mass
2. Salt mass
3. Energy (solution mass * enthalpy)

Figure 3.2 Information flow diagram for the TRNSYS component "heat and mass exchanger sump"

with

$$\xi = \frac{m_{salt}}{m_s} \quad (3.20)$$

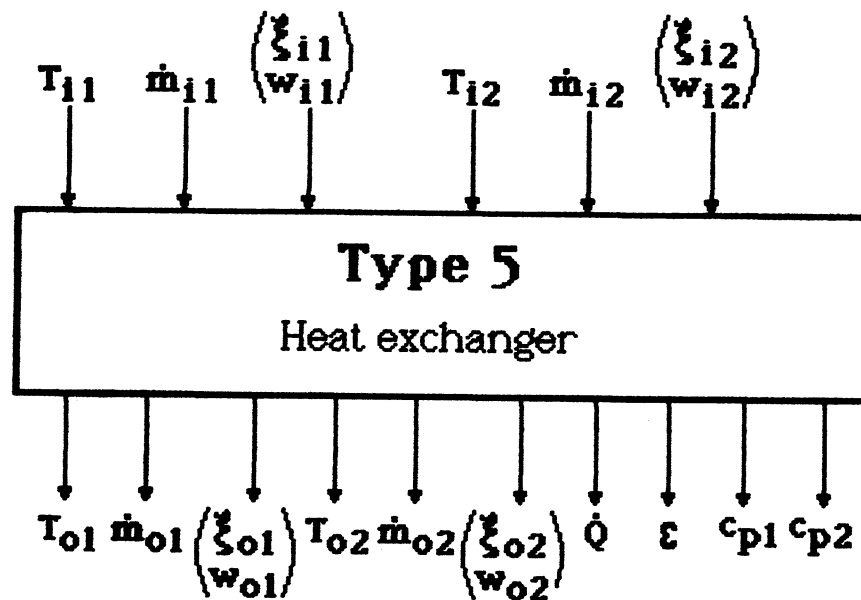
and the energy balance

$$\frac{d(m_s I)}{dt} = \sum \dot{m}_{si} I_i - \dot{m}_{so} I \quad (3.21)$$

complete the sump model. The three differential equations are solved by the differential equation solving routine implemented in TRNSYS [9]. Figure 3.2 shows the information flow diagram of the heat and mass exchanger sump.

3.2.3 The Heat Exchanger Model

The heat exchanger model in TRNSYS [9] was modified to allow for a third flow variable. The assumption of constant heat capacity is not valid for solutions with varying salt concentrations. The appropriate parameters were replaced by an integer code to indicate the type of fluid. On each call of the component, the heat capacities of both flow streams are evaluated as a function of the third flow variable. In the case of pure water or glycol-water solution, the heat capacities are assumed constant at 4.19 kJ/kg-°C and 3.8 kJ/kg-°C, respectively. The information flow diagram of the heat exchanger is shown in Figure 3.3.



Parameters:

1. Mode
2. UA or effectiveness
3. Fluid code 1
4. Fluid code 2

Figure 3.3 Information flow diagram for the TRNSYS component "heat exchanger"

3.2.4 The Air Mixer Model

The adiabatic mixing of two air streams can be described by three balances; one mass balance each for dry air and for water, and an energy balance [3]. The mass balance for dry air is

$$m_{ao} = m_{ai1} + m_{ai2} \quad (3.22)$$

the mass balance for water is

$$m_{ao} w_o = m_{ai1} w_{i1} + m_{ai2} w_{i2} \quad (3.23)$$

and the energy balance is

$$m_{ao} i_o = m_{ai1} i_{i1} + m_{ai2} i_{i2} \quad (3.24)$$

Figure 3.4 shows the information flow diagram of the adiabatic air mixer.

3.2.5 The Fan Model

The fan model was developed to include the increase of the air enthalpy due to a fan. The energy balance

$$\dot{m}_{ao} i_o = \dot{m}_{ai} i_i + P_{elt} \quad (3.25)$$

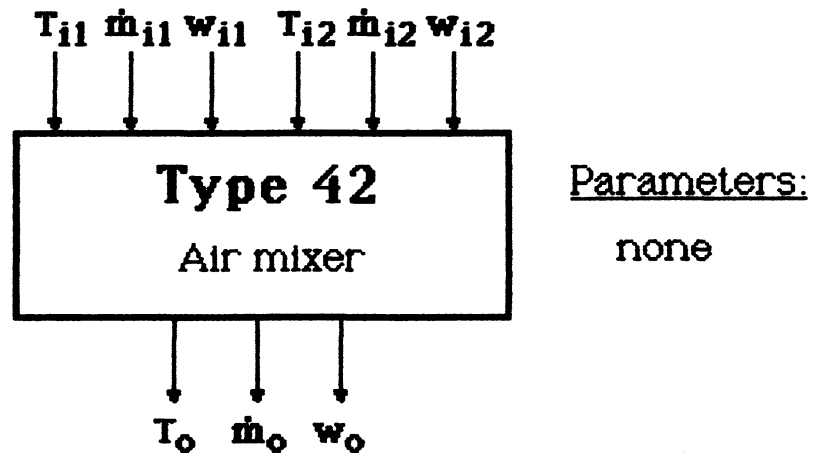


Figure 3.4 Information flow diagram for the TRNSYS component "air mixer"

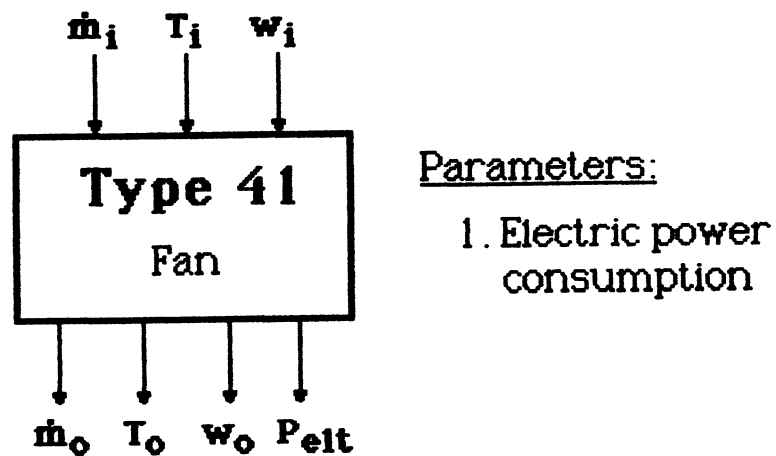


Figure 3.5 Information flow diagram for the TRNSYS component "fan"

allows to calculate the air outlet temperature. Here it is assumed that all work goes into the increase of air enthalpy. The information flow diagram of the fan is shown in Figure 3.5.

3.2.6 The Vapor Compression Heat Pump Model

Manufacturers data [31] for the heat pump installed at the Science Museum were used for a steady state model. The given data for the electric power consumption and heating coefficient of performance (COP) as functions of the temperature of the water leaving the evaporator and the temperature of the water leaving the condenser were curve-fitted

$$P_{elt} = 49.14 - 0.2237 T_{c,o} + 2.095 \cdot 10^{-4} T_{c,o}^2 - 0.2380 T_{e,o} + 2.176 \cdot 10^{-4} T_{e,o}^2 + 4.942 T_{c,o} T_{e,o} \text{ [kW]} \quad (3.26)$$

$$\begin{aligned} \text{COP} = & 0.2789 - 3.341 \cdot 10^{-3} T_{c,o} + 2.095 \cdot 10^{-4} T_{c,o}^2 + 0.0136 T_{e,o} \\ & + 1.580 \cdot 10^{-6} T_{e,o}^2 - 7.086 \cdot 10^{-4} T_{c,o} T_{e,o} \end{aligned} \quad (3.27)$$

It is assumed, that the heat pump always delivers hot water at the desired set point, so that

$$\dot{Q}_c = \dot{m}_c c_{p,w} (T_{c,o} - T_{c,i}) \quad (3.28)$$

The electric power consumption was allowed to fluctuate, so that

$$P_{elt} = \frac{\dot{Q}_c}{COP} \quad (3.29)$$

Approximately 8% of the electric power is lost to the surroundings, so that

$$\dot{Q}_e = P_{elt} (COP - 0.92) \quad (3.30)$$

The model has an iterative nature, since the COP is given as function of outlet temperatures. The MINPACK routine HYBRD1 [10] was used to promote convergence. Figure 3.6 shows the information flow diagram of the vapor compression heat pump.

3.2.7 The Vapor Compression Chiller Model

The manufacturer of the vapor compression chiller installed at the Science Museum of Virginia supplies performance data only for a range of water temperatures at the outlet of the evaporator up to 50°F [32]. Since the chiller is controlled to supply chilled water at 55°F, it was necessary to extrapolate the data provided.

The model is based on the efficiency of a reversible Carnot engine which is defined as

$$\eta = \frac{P}{\dot{Q}_H} = \frac{T_c - T_e}{T_c} \quad (3.31)$$

The coefficient of performance of cooling is defined by

$$\text{COP} = \frac{\dot{Q}_e}{P} \quad (3.32)$$

The COP of reversible Carnot engine can be expressed in terms of the Carnot efficiency

$$\text{COP}^{\text{rev}} = \frac{1 - \eta}{\eta} \quad (3.33)$$

or, in terms of condenser and evaporator temperatures

$$\text{COP}^{\text{rev}} = \frac{T_e}{T_c - T_e} \quad (3.34)$$

Accounting for the temperature difference between the water leaving condenser and evaporator and the condensation and evaporation temperature of the refrigerant, Equation (3.34) can be rewritten as

$$\text{COP}^{\text{rev}} = \frac{T_{e,o} - \Delta T}{(T_{c,o} + \Delta T) - (T_{e,o} - \Delta T)} \quad (3.35)$$

The temperature difference ΔT was chosen as 5°C.

Finally the COP of the irreversible chiller can be found by using an effectiveness factor approach

$$\text{COP} = \epsilon_{\text{COP}} \text{COP}^{\text{rev}} \quad (3.36)$$

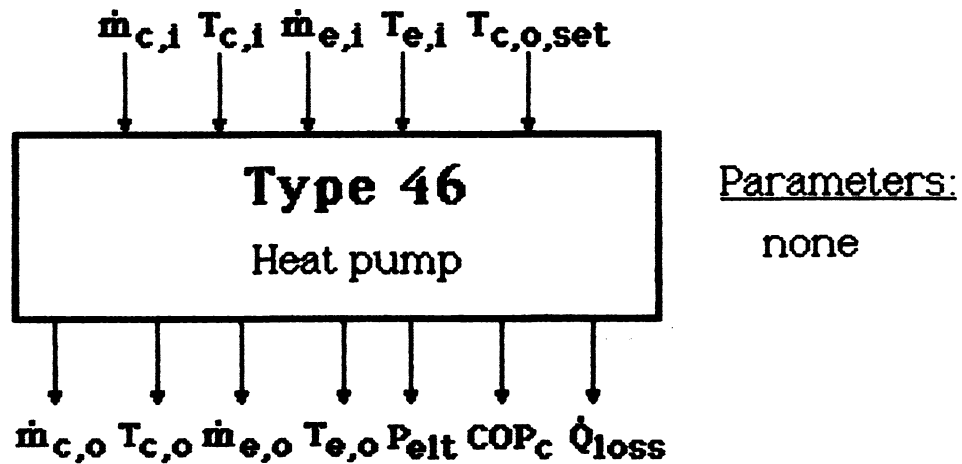


Figure 3.6 Information flow diagram for the TRNSYS component "vapor compression heat pump"

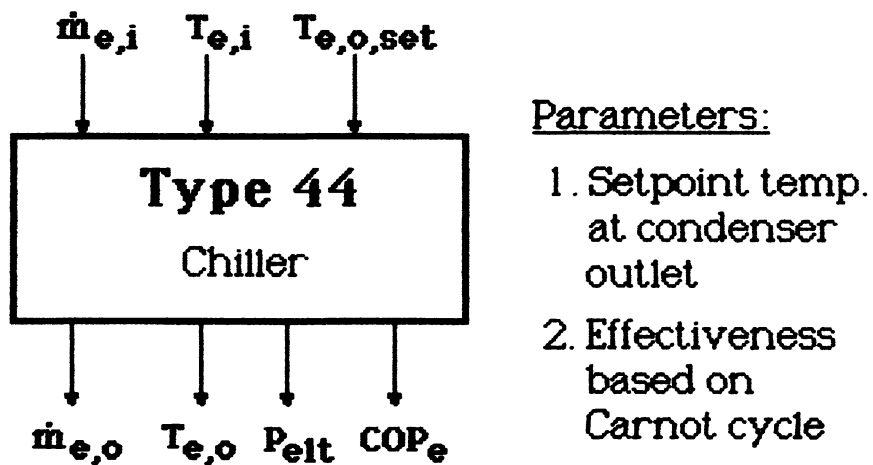


Figure 3.7 Information flow diagram for the TRNSYS component "vapor compression chiller"

The effectiveness factor was taken as 0.60 to reproduce the COP of 5.2 for a chiller operation between 55°F evaporator leaving temperature and 95°F condenser leaving temperature as given by Meckler [7]. For an operation between 42°F and 95°F, the model predicts a COP of 4.16 which is in agreement with the manufacturer's data [32].

It is assumed, that the chiller always provides chilled water at its setpoint, so that

$$\dot{Q}_e = \dot{m}_c c_{p,w} (T_{ei} - T_{eo,set}) \quad (3.37)$$

$$P_{elt} = \frac{\dot{Q}_e}{COP} \quad (3.38)$$

and

$$\dot{Q}_c = (1 + COP) P_{elt} \quad (3.39)$$

The condenser heat is rejected to the environment via a cooling tower. The information flow diagram of the vapor compression chiller is shown in Figure 3.7.

3.2.8 The Proportional Integral Controller

The proportional integral controller was used to maintain the solution volume of the conditioner sump at a constant value (2 m^3). The controller "measured" the sump volume and acted upon the valve which splits up the solution flow rate leaving the sump into the flow which is recycled to the conditioner and the flow which is pumped to the regenerator sump.

A single input, single output (SISO) proportional integral controller was implemented in a quasi-discrete velocity form

$$u_k - u_{k-1} = K((e_k - e_{k-1}) + \frac{\Delta t}{2\tau_i} e_k)) \quad (3.40)$$

where

$$e_k = Y_k - Y_{\text{set}} \quad (3.41)$$

and K is the controller gain, τ_i the integral time constant and Δt the simulation time step. At any time step k the controller output is u_{k-1} which was calculated at the previous time step $k-1$ from the controller error e_{k-1} . To avoid unrealistic controller action, it is limited between its upper and lower limits, u_{max} and u_{min} , respectively. Figure 3.8 shows the information flow diagram of the proportional integral control.

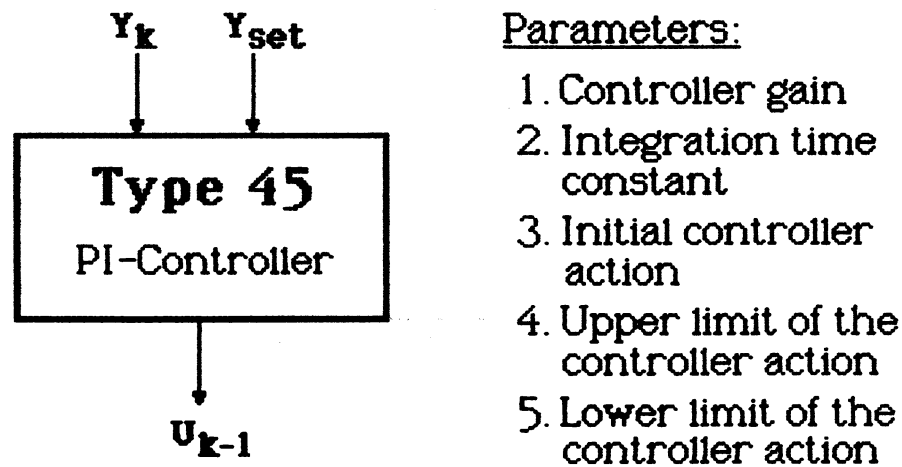


Figure 3.8 Information flow diagram for the TRNSYS component "proportional integral controller"

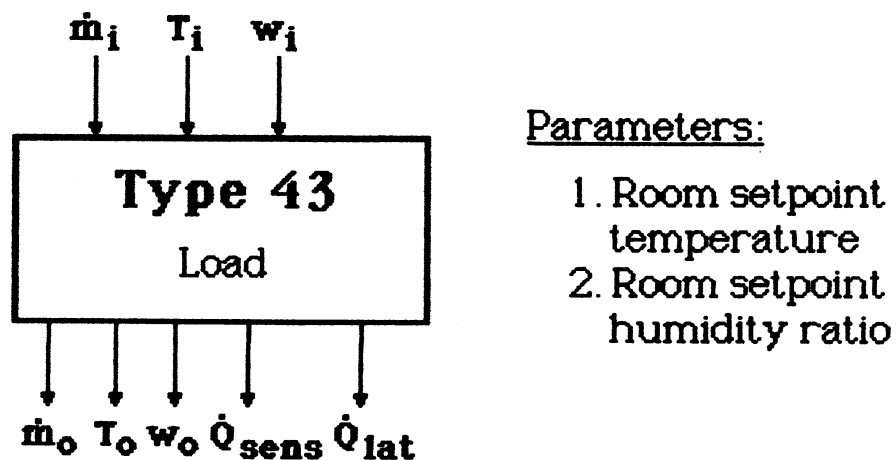


Figure 3.9 Information flow diagram for the TRNSYS component "load"

3.2.9 The Load Model

The load model is not really a model of a load but a routine to calculate the sensible and latent load which could be met given the air inlet state and the desired room air state. The loads, which could be met are

$$\dot{Q} = \dot{m}_a (i_o - i_i) \quad (3.42)$$

$$\dot{Q}_{lat} = \dot{m}_a (w_o - w_i) i_v \quad (3.43)$$

and

$$\dot{Q}_{sens} = \dot{Q} - \dot{Q}_{lat} \quad (3.44)$$

The air outlet state is set to the desired room air state which was 25.0 °C and 0.0093 humidity ratio for the simulation in Section 3.3. The information flow diagram of the load model is shown in Figure 3.9.

3.3 The System Simulation Model

A TRNSYS deck was set up including the dehumidification cycle and major parts of the HVAC equipment installed at the Science Museum of Virginia in Richmond, VA. A listing of the TRNSYS deck is given in Appendix C.

The model of the desiccant cycle was designed closely to the installed equipment. The solution mass flow rate pumped out of the regenerator sump is split up between the flow to the conditioner sump and the recycled flow to the regenerator at a constant ratio of 0.063. The split ratio of the stream leaving the conditioner sump is controlled by a PI controller so as to maintain a constant solution sump level, i.e., constant solution volume in the conditioner sump.

The heat pump is controlled to the set point of the water leaving the condenser. The cooling water flow through the evaporator is controlled to the set point of the water leaving the evaporator. The remaining cooling water flow is pumped to the chiller, which returns it at the cooling water set point temperature.

It was assumed, that the chiller rejects its condenser heat at 95°F and the cooling tower returns water at 85°F. A cooling tower model was not included. Furthermore, the predehumidification coil was not modeled either. For the design point, an air stream of 3.35 kg/s (6000 SCFM) entering at 32.8°C and a humidity ratio of 0.0177 is cooled and dehumidified to 21.1°C and a humidity ratio of 0.0143, thereby releasing 69 kW of heat. This load is taken up by the chiller, with a corresponding chiller electricity demand.

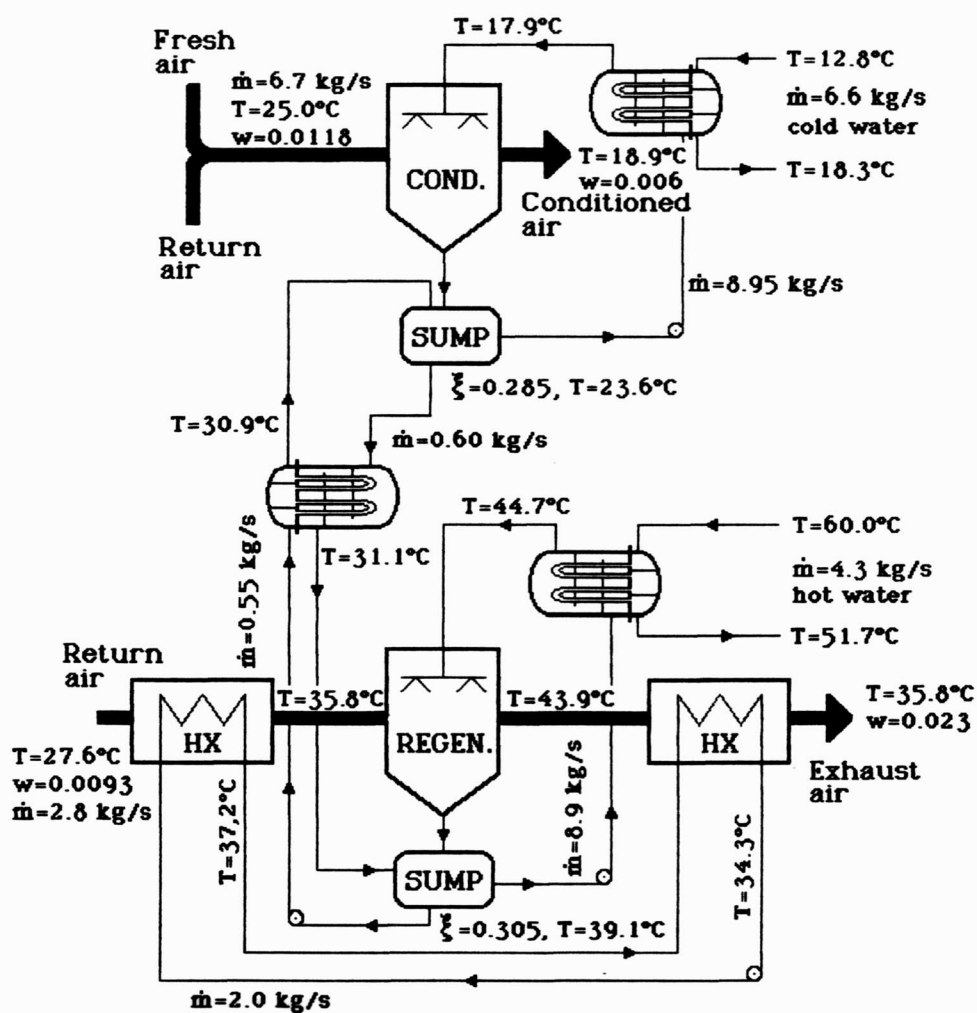


Figure 3.10

Air and solution states in the liquid desiccant system in the Science Museum of Virginia, design point

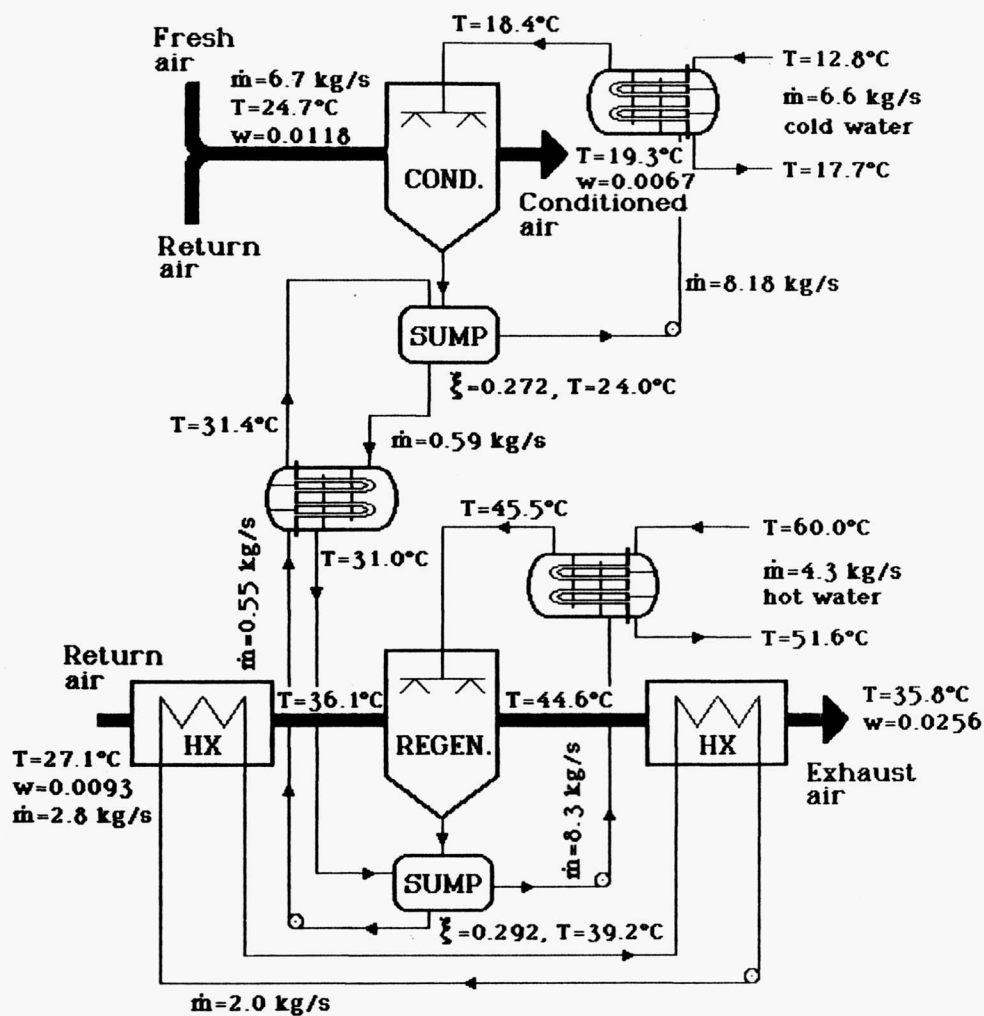


Figure 3.11

Air and solution states in the liquid desiccant system in the Science Museum of Virginia, as predicted by the TRNSYS system model

The existing three air handling units were lumped together as one supply fan, which was modeled without the cooling coil. However, the knowledge of the sensible load and the desired room air state allows the calculation of the cooling coil load. Knowing the cooling coil load and the chiller COP, the chiller electricity demand due to sensible cooling can be readily calculated.

In Figures 3.10 and 3.11, the dehumidification cycle is shown for the states given at the design point and at those predicted by the TRNSYS simulation. Almost all states are equal or very close. It is concluded that the TRNSYS model of the system predicts the system performance and almost all system states satisfactorily. However, the comparison is possible only with the design data, since no experimental data are yet available.

3.4 Results of Steady State Simulations

A number of simulation runs were executed to study the influence of variations in selected variables on the system performance. The results were analyzed to predict the hourly cost of operation of various modes of operation. The analysis is based on the data available for the design point as described in Section 1.4.

Five parameters were varied during the study. Two of these five parameters are of operational nature, i.e., it is probably possible to change these parameters during system operation without alteration of the equipment. These two parameters are the temperatures, at

which cooling water is supplied to the solution cooler and at which hot water is supplied to the solution heater of the desiccant cycle.

The other three parameters were fixed at the design stage. These parameters are the overall heat conductance-area products of the solution cooler and heater and the effectiveness of the heat recovery loop heat exchanger. The study of these parameters yields interesting information on the consequences of the actual design.

The total internal load was assumed to be 178.8 kW, of which one fourth or 44.7 kW is latent load. The outdoor air state was given as 32.8°C temperature and 0.0177 humidity ratio. The desired room air state is 25°C and a humidity ratio of 0.0093.

The total load on the chiller is composed of the heat transferred in the predehumidification coil (69 kW constant), the part of the cooling load of the desiccant cycle which is not met by the heat pump and the total sensible cooling load. The total sensible cooling load is the sum of the internal sensible load and the additional sensible load due to the conditioner. The sensible load due to the conditioner can be either positive or negative, dependent on whether the air leaving the conditioner is warmer or colder than the desired supply air state. In the operation modes "solar energy" and "gas cogenerator", the heat pump is not operating. Thus, the chiller has to provide the total cooling load of the desiccant cycle.

It was assumed that the gas cogenerator converts one third of the energy supplied from gas to electricity. Another third is

utilizable heat and the last third is lost to the surroundings.

The analysis was carried out assuming the price for electric energy was \$0.06 per kW-hr. For the mode "gas cogenerator" electricity to gas price ratios of two and three, i.e., gas energy prices of \$0.03 per kW-hr and \$0.02 per kW-hr were studied.

Furthermore, it is assumed that solar energy could be provided at no extra cost. For a gas price of \$0.02 per kW-hr, the gas cogeneration produces virtually free heat since it produces a dollars worth of electricity with each dollar spent for gas. This means, that the cost of operation for the mode "gas operation" and an electricity to gas price ratio of three is equal to the cost of operation using free solar energy.

The following fan electrical loads were taken into account:

Supply fans	36.8 kW
Return fan	23.0 kW
Conditioner supply fan	7.5 kW
Regenerator supply fan	<u>2.9 kW</u>
Total	70.2 kW

The electrical load consumptions of the pumps were

Chiller pumps	13.4 kW
Solution pumps	2 x 2.25 kW
Heat recovery loop pump	<u>0.75 kW</u>
Total	18.65 kW

From the data in [7] it was concluded that in addition to the chiller the cooling tower fan and pump consume another 12% of the chiller electricity consumption.

For the purpose of comparison, the cost of operation for a conventional air conditioning system was estimated. To meet the load, the air supplied to the building has to be at 19.0°C and 0.0085 humidity ratio. The air has to be cooled to 11.6°C to remove the moisture in excess of the humidity ratio of 0.0085. As for the existing system 85% return air (18.8 kg/s or 33000 SCFM) and 15% outside air (3.3 kg/s or 6000 SCFM) are mixed. This air flow rate of 22.1 kg/s is at 27.1 °C and 0.01059 humidity ratio. The supply fans add 33 kW, raising the temperature to 28.6°C. The heat to be exchanged in the coils is 498.6 kW. The chiller is operating at 5.6°C (42°F) evaporator leaving temperature with a COP of 4.16. Thus the total energy demand for the conventional system is composed of

Chiller	119.9 kW
Cooling tower	14.6 kW
Chiller pumps	13.4 kW
Supply fans	36.8 kW
Return fan	<u>23.0 kW</u>
Total	207.7 kW
at 0.06 \$/kW-hr	12.46 \$/hr

It was assumed that the energy to reheat the air stream from 11.6°C to 19.0°C is available for free from the chiller condenser. The value of 12.46 \$/hr is plotted in Figures 3.12 through 3.16 as a straight line and is not a function of any of the parameters.

3.4.1 Variations in the Cooling Water Supply Temperature

The temperature at which cooling water is supplied to the solution cooler directly determines the solution temperature at the conditioner inlet. It thereby influences the dehumidification and cooling capacity of the solution. At the design point, the cooling water supply temperature is 12.8°C (55°F). The analysis is exhibited in Table 3.1. Figure 3.12 shows a plot of the hourly cost of operation as a function of the cold water supply temperature for different modes of operation. The legends of this and the following figures are:

HP	heat pump mode
Solar	Solar energy mode
Gas cog.	Gas cogeneration mode, electricity to gas price ratio of two
Chiller	Conventional air conditioning with vapor compression chiller only

Both the COP of the chiller and the COP of the heat pump increase with increasing supply temperature. To maintain the same

Cold water supply temperature	°C	16	12.8	10	8	6
P_{elt} heat pump	kW	52.80	45.53	46.64	45.89	45.55
COP of heat pump	--	3.30	3.11	2.96	2.84	2.73
Total sensible load	kW heat	164.25	144.70	132.38	125.44	119.87
P_{elt} chiller	kW	41.45	46.38	50.74	54.05	57.43
COP of chiller	--	5.88	5.23	4.77	4.48	4.22
P_{elt} cooling tower	kW	5.06	5.66	6.19	6.59	7.01
P_{elt} fans and pumps	kW	88.85	88.85	88.85	88.85	88.85
P_{elt} total, "heat pump" mode	kW	188.17	189.40	192.42	195.37	198.82
Cost of operation "heat pump" mode	\$/hr	11.29	11.36	11.55	11.72	11.93
Heating load, solution heater (at 60°C)	kW heat	174.13	151.12	137.86	130.43	124.25
P_{elt} total, "solar" mode	kW	159.31	163.71	168.12	171.59	175.20
Cost of operation "solar" mode	\$/hr	9.56	9.82	10.09	10.30	10.51
Gas used for gas cogenerator	kW	522.92	453.82	413.98	391.68	373.13
Cost of operation "gas cogen." mode Gas price 0.03 \$/kW-hr	\$/hr	14.80	14.73	14.24	14.22	14.25

Table 3.1 Analysis of the system performance for varying cold water supply temperatures.

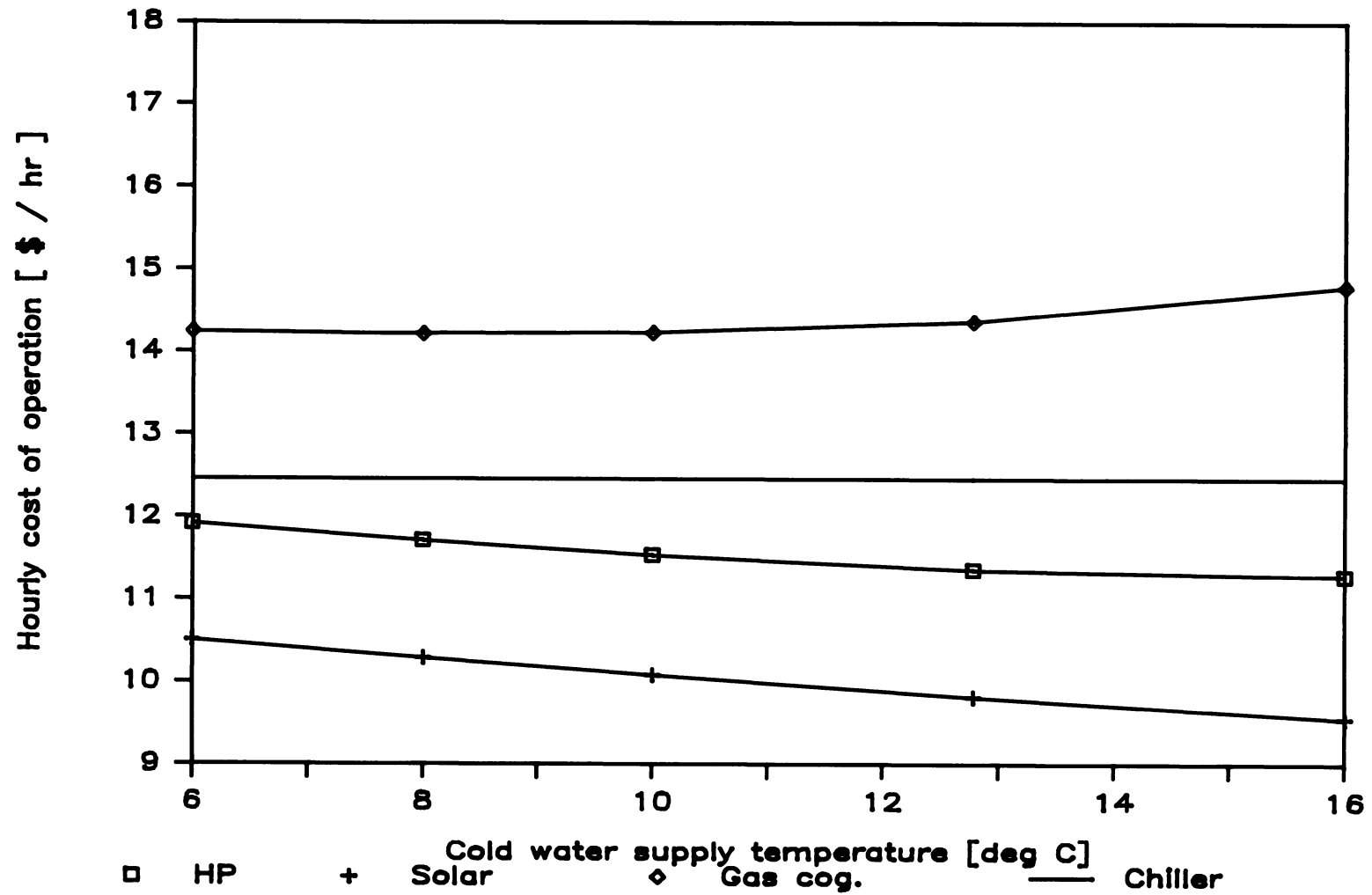


Figure 3.12 Hourly cost of operation versus cooling water supply temperature for different modes of operation

dehumidification potential, the system operates with higher solution concentrations at higher cold water supply temperatures. This in turn causes a higher energy consumption for the regeneration. If free solar energy is used for regeneration, increased energy consumption does not influence the hourly cost of operation. Thus, this cost decreases with increasing cold water supply temperature. For the "gas cogenerator" mode, the increased regeneration energy demand causes the cost of operation to go up as the supply temperature is increased.

Increased COP of the heat pump and increased regeneration energy demand yield a trade-off. The cost of operation decreases with increasing temperature, but almost levels off between the design point (12.8°C) and 16°C . Summarizing, it can be said that the cheaper the regeneration can be provided, the higher the optimal cold water supply temperature.

3.4.2 Variations in the Hot Water Supply Temperature

The analysis of the system performance and the hourly cost of operation for varying hot water supply temperatures is presented in Table 3.2. Figure 3.13 shows the corresponding plot of the cost of operation versus hot water supply temperature for the modes of operation considered in this analysis. The supply temperatures at the design point is 60°C .

The energy needed for the regeneration decreases rapidly as the temperature at which the energy is provided increases. Thus, the

Hot water supply temperature	°C	75	70	65	60	55
P_{elt} heat pump	kW	48.66	47.98	48.07	48.53	51.70
COP of heat pump	--	2.64	2.78	2.94	3.11	3.30
Total sensible load	kW heat	144.95	144.84	144.76	144.70	144.70
P_{elt} chiller	kW	44.80	45.00	45.44	46.36	47.89
COP of chiller	--	5.23	5.23	5.23	5.23	5.23
P_{elt} cooling tower	kW	5.47	5.49	5.54	5.66	5.84
P_{elt} fans and pumps	kW	88.85	88.85	88.85	88.85	88.85
P_{elt} total, "heat pump" mode	kW	187.74	187.30	187.88	189.40	194.29
Cost of operation "heat pump" mode	\$/hr	11.26	11.24	11.27	11.36	11.66
Heating load, solution heater, to be provided at	kW heat °C	128.22 75	133.46 70	141.34 65	151.12 60	170.57 55
P_{elt} total, "solar" mode	kW	157.03	158.51	160.67	163.71	168.97
Cost of operation "solar" mode	\$/hr	9.42	9.51	9.64	9.82	10.14
Gas used for gas cogenerator	kW	385.05	400.77	424.43	453.82	512.22
Cost of operation "gas cogen." mode Gas price 0.03 \$/kW-hr	\$/hr	13.28	13.53	13.89	14.37	15.27

Table 3.2 Analysis of the system performance for varying hot water supply temperatures.

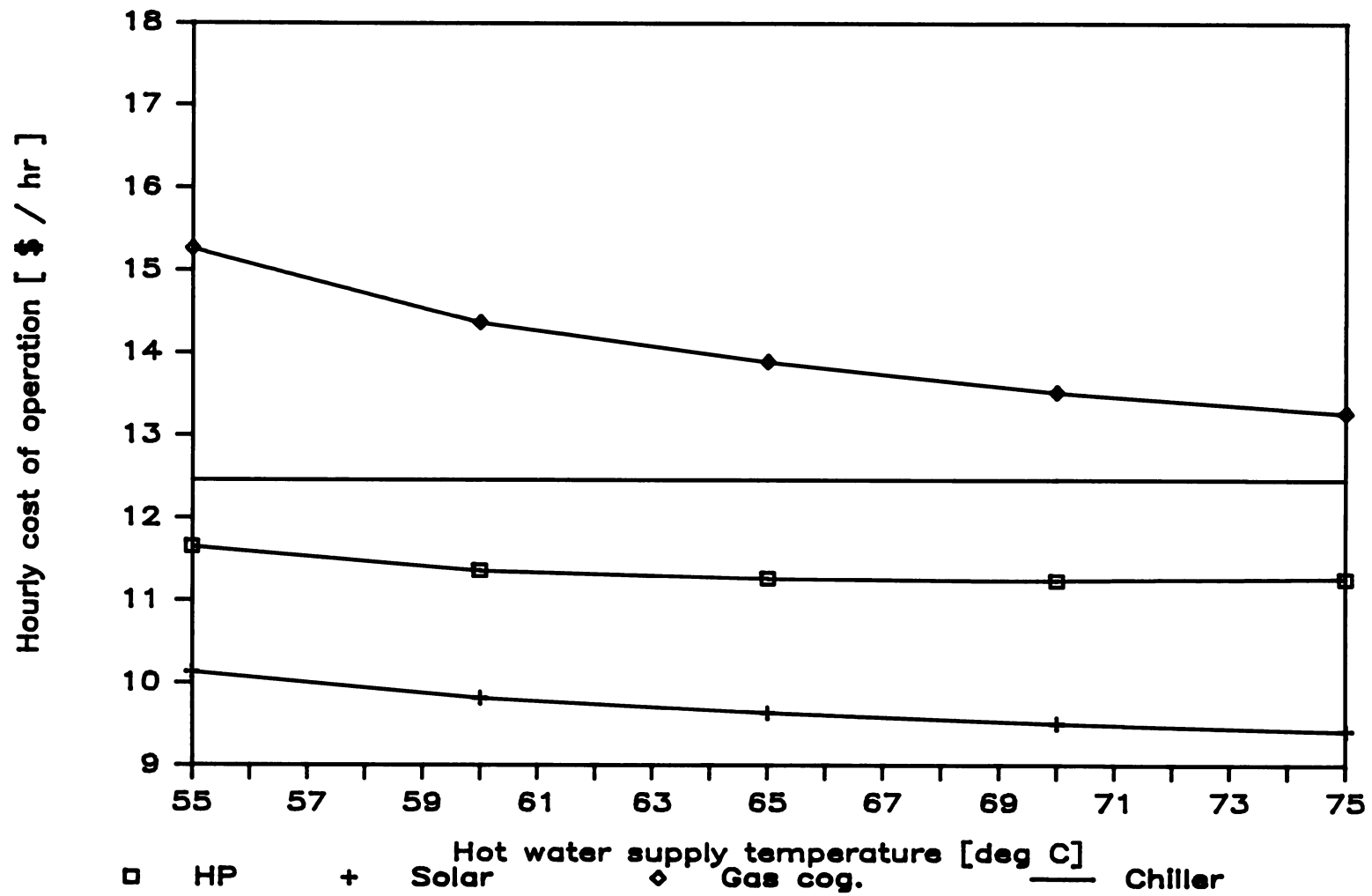


Figure 3.13 Hourly cost of operation versus hot water supply temperature for different modes of operation

cost of operation for the "gas cogenerator" mode drops with increasing supply temperature. Note however, that the analysis does not account for possible variations in the effectiveness of the gas cogenerator with varying water temperature.

The COP of the heat pump degrades sharply as the temperature of water leaving the condenser is increased. Thus, there is an optimal supply temperature of 70°C. The cost of operation for the cost of operation at 75°C might be a little too low, since the heat pump model was used beyond the limits of the curve-fitted function for the COP.

The sensible cooling load due to the desiccant cycle decreases with increasing hot water supply temperature because the system operates with a higher solution concentration. Therefore, the load on the chiller is reduced and hence the cost of operation for the "solar" mode is reduced too. On the other hand, the performance of solar collectors decreases with increasing water temperature due to increased losses. Thus, the optimal temperature of operation for the "solar" mode will depend on the installed solar collector equipment.

3.4.3 Variations in the Overall Heat Conductance-Area Products of the Solution Cooler and Heater

The system performance was analyzed for half and for twice the overall heat conductance-area product (UA) at the design point. The data of the analysis are listed in Table 3.3. Graphs showing the cost of operation as a function of each one of the UA's of the

UA solution heater	MJ/°C-hr	80.860	40.430	20.215	40.430	40.430	40.430
UA solution cooler	MJ/°C-hr	81.600	81.600	81.600	16.320	81.600	40.800
P_{elt} heat pump (COP = 3.11)	kW	43.81	48.53	67.87	43.23	48.53	60.46
Total sensible load	kW heat	144.81	144.70	144.77	129.74	144.70	175.03
P_{elt} chiller (COP = 5.23)	kW	44.45	46.36	52.66	45.77	46.36	47.55
P_{elt} cooling tower	kW	5.42	5.66	6.42	5.58	5.66	5.80
P_{elt} fans and pumps	kW	88.85	88.85	88.85	88.85	88.85	88.85
P_{elt} total, "heat pump" mode	\$/hr	182.55	189.40	215.81	183.43	189.40	202.65
Cost of operation "heat pump" mode	\$/hr	10.95	11.36	12.95	11.01	11.36	12.16
Heating load, solution heater (at 60°C)	kW heat	136.47	151.12	211.35	134.61	151.12	188.26
P_{elt} total, "solar" mode	kW	159.35	163.71	179.88	160.55	163.71	170.65
Cost of operation "solar" mode	\$/hr	9.56	9.82	10.79	9.63	9.82	10.24
Gas used for gas cogenerator	kW	409.82	453.82	634.69	404.22	453.82	565.34
Cost of operation "gas cogen." mode Gas price 0.03 \$/kW-hr	\$/hr	13.67	14.37	17.15	13.68	14.37	15.90

Table 3.3 Analysis of the system performance for varying overall heat conductance-area products of the solution heater and cooler.

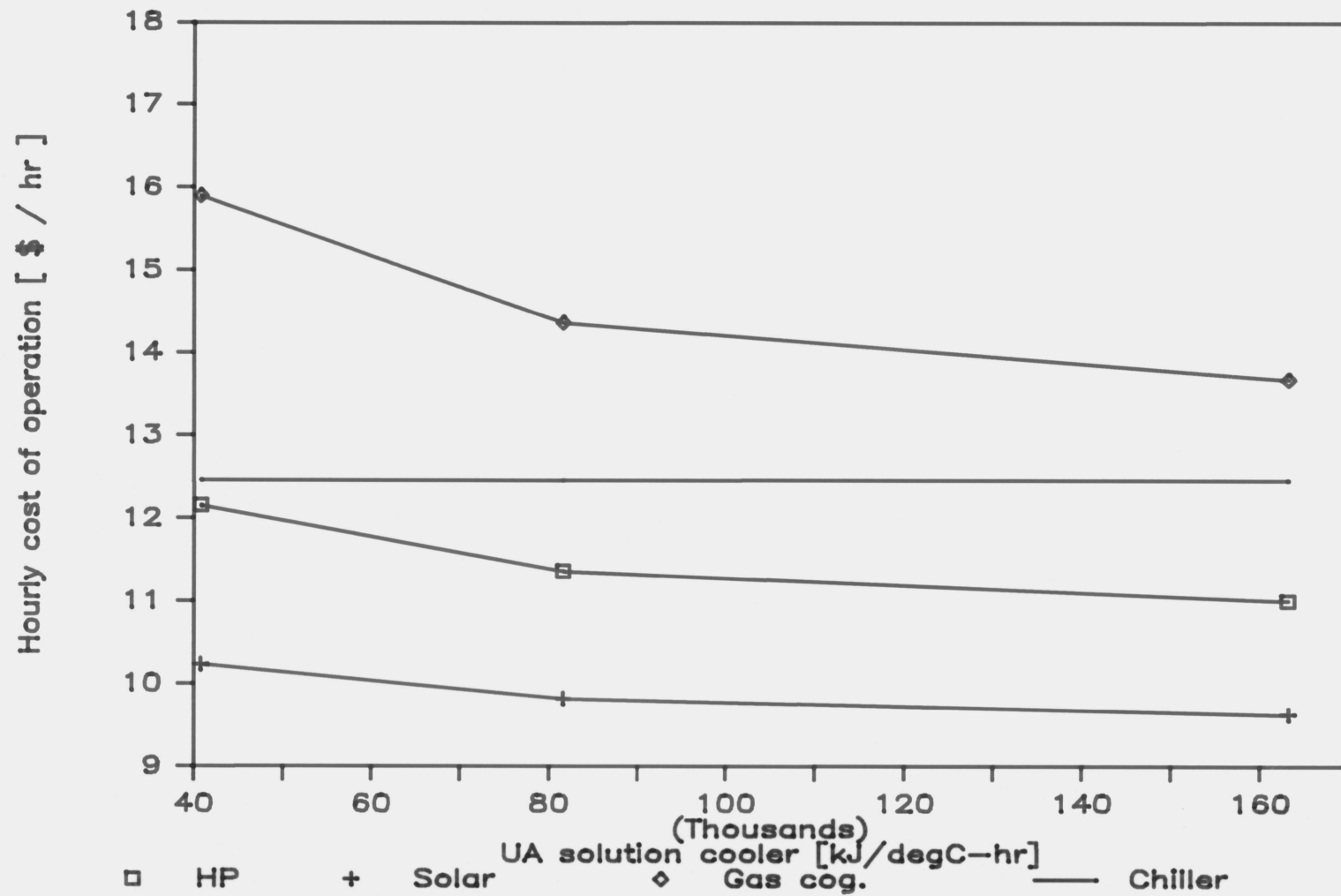


Figure 3.14 Hourly cost of operation versus UA of the solution cooler for different modes of operation

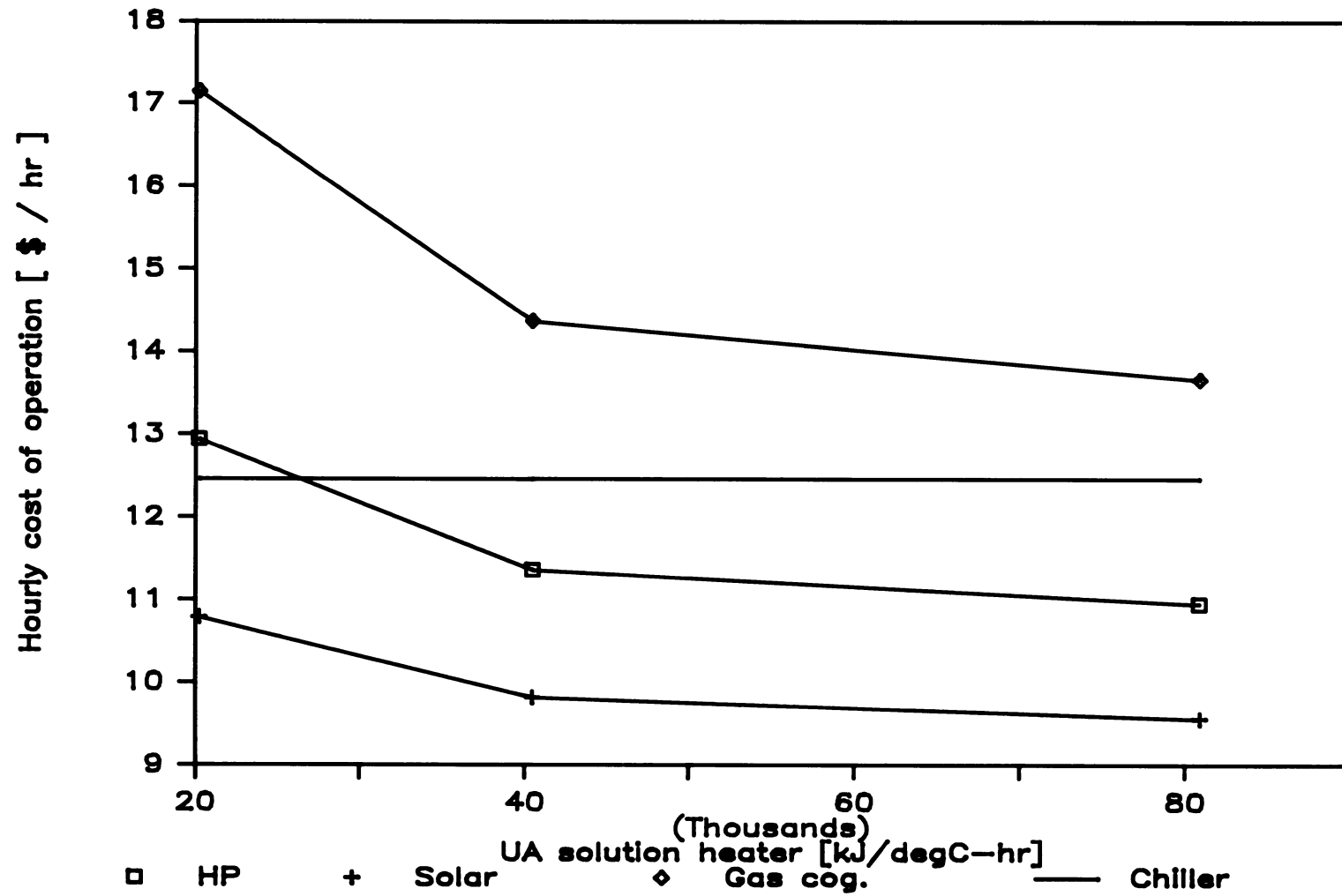


Figure 3.15 Hourly cost of operation versus UA of the solution heater for different modes of operation

solution cooler and heater are given in Figures 3.14 and 3.15, respectively.

As expected, the cost of operation increases dramatically if only half the UA is available, but decreases only slightly if twice the UA of the design point is available. This indicates that the design is appropriate to the problem.

3.4.4 Variations in the Effectiveness of the Heat Recovery Loop Heat Exchangers

The range of heat exchanger effectivenesses studied is zero to unity, which is equivalent to no heat recovery and perfect heat recovery, respectively. The installed heat exchangers have an effectiveness of 0.85. The analysis is exhibited in Table 3.4. Figure 3.16 shows a plot of the hourly cost of operation as a function of the heat exchanger effectiveness for different modes of operation.

The regeneration heat demand decreases with increasing heat exchanger effectiveness. This does not influence the cost of operation for the "solar" mode but causes decreasing costs with increasing heat exchanger effectiveness for the modes "heat pump" and "gas cogenerator". The overall effect of the heat recovery loop is relatively small, since the heat capacity rate of the air flow through the regenerator (2.8 kJ/s) is small compared with that of the solution flow (25.5 kJ/s).

Heat recovery loop heat exchanger effectiveness		1.0	0.85	0.5	0.0
P_{elt} heat pump (COP = 3.11)	kW	46.46	48.53	52.80	57.79
Total sensible load	kW heat	144.74	144.70	144.64	144.55
P_{elt} chiller (COP = 5.23)	kW	46.94	46.36	45.22	43.91
P_{elt} cooling tower	kW	5.73	5.66	5.52	5.36
P_{elt} fans and pumps	kW	88.85	88.85	88.85	88.10
P_{elt} total, "heat pump" mode	kW	188.97	189.40	192.40	195.15
Cost of operation "heat pump" mode	\$/hr	11.28	11.36	11.54	11.71
Heating load, solution heater (at 60°C)	kW heat	144.66	151.12	164.46	179.95
P_{elt} total, "solar" mode	kW	163.38	163.71	164.45	164.56
Cost of operation "solar" mode	\$/hr	9.80	9.82	9.87	9.87
Gas used for gas cogenerator	kW	434.42	453.82	493.88	540.39
Cost of operation "gas cogen." mode Gas price 0.03 \$/kW-hr	\$/hr	14.16	14.37	14.82	15.29

Table 3.4 Analysis of the system performance for varying heat recovery loop heat exchanger effectivenesses.

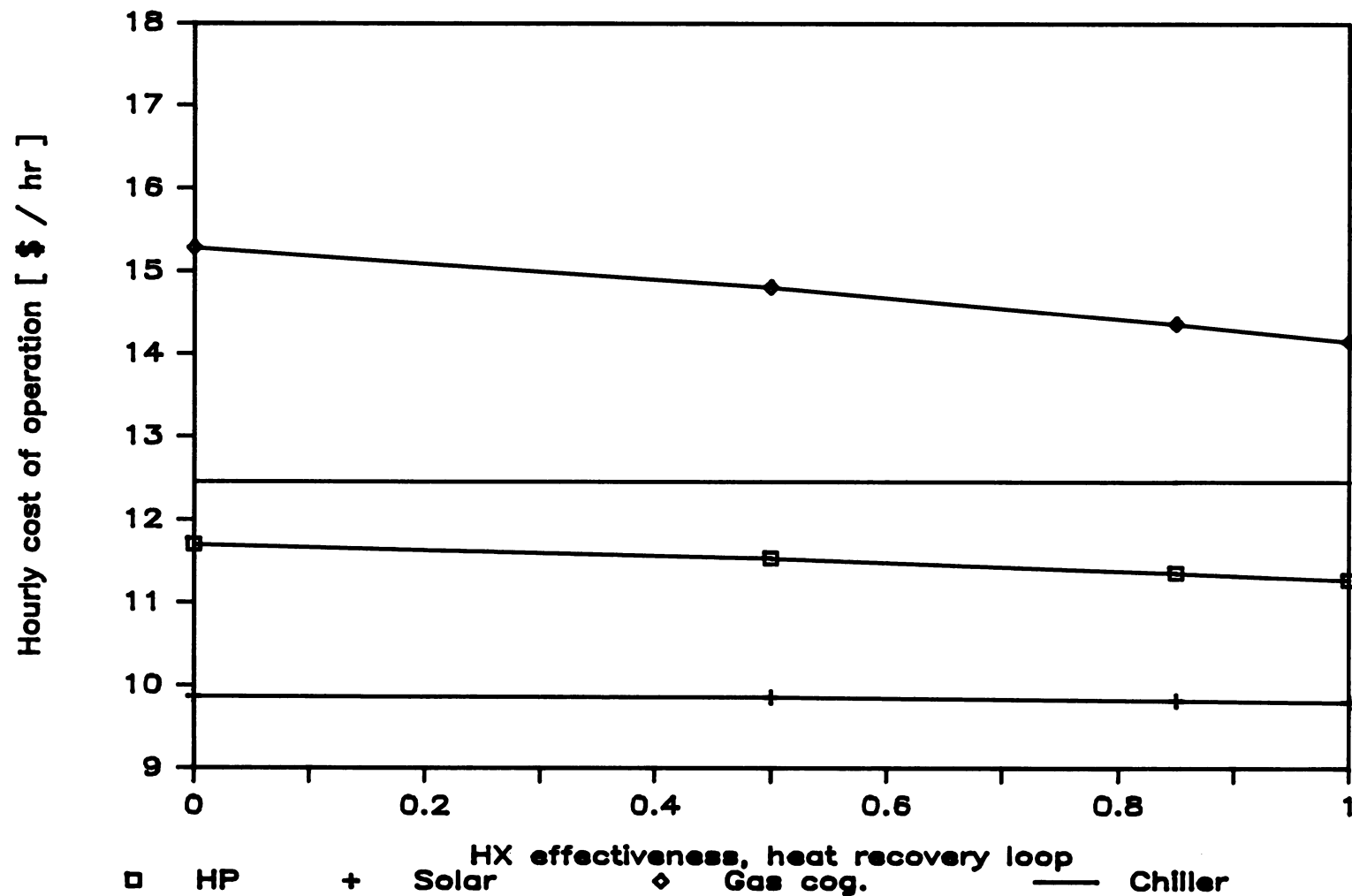


Figure 3.16 Hourly cost of operation versus heat exchanger effectiveness of the heat recovery loop for different modes of operation

3.4.5 Summary

In summary it can be stated that the optimal point at which the hourly cost of operation is minimized depends strongly on the mode of operation. This is due to the fact, that the cost per unit of heat for regeneration depends on the source of the heat, i.e., whether the heat pump, the gas cogenerator or solar collectors are used to provide the heat.

The cost of operation for the gas cogenerator is used as the heat source depends on the electricity to gas price ratio. It can be as inexpensive as solar energy, for a price ratio of three, or it can be even more expensive than the cost of operation for the conventional system, for a price ratio of two.

Generally, it is less expensive to operate the heat pump than a conventional air conditioning system. The use of solar energy yields predicted savings of about \$2/hr at the design point.

CHAPTER 4 CONCLUSIONS

The objective of this study was to provide tools to study liquid desiccant open cycle systems. These tools include models of single components, packages of physical property functions for the working fluids and a model of the HVAC system for the Science Museum of Virginia.

Two models for the conditioner and regenerator were developed. The first model is based on equilibrium considerations and effectiveness coefficients. It requires minimal computational expense and is suitable for use in long term simulations. The second model employs a finite step integration along the path of simultaneous heat and mass exchange. Combining the results of both models, it was possible to develop correlations for the effectiveness coefficients used in the equilibrium model.

A preliminary model of the HVAC equipment, including a liquid desiccant cycle installed at the Science Museum of Virginia, was developed. Only those components which are of direct relevance to the performance of the liquid desiccant cycle and the overall system performance were incorporated into the system model. However, it would be easy to extend the model because of the modular structure of TRNSYS.

Finally, results of steady state simulations using the system model were analyzed. The influence of variations in system parameters such as water supply temperatures and heat exchanger

effectivenesses on the overall system performance were evaluated. The hourly cost of operation for different modes of operation, i.e., different regeneration heat sources, were predicted.

CHAPTER 5 RECOMMENDATIONS

The work presented in this thesis is intended to be the foundation of a number of studies on liquid desiccant systems to be prepared in the future.

The next step should be the updating of the component models and the model of the assembled system. More information on the system has to be gathered and implemented into the simulation. Furthermore, some effort has to be put into the improvement of the convergence behavior of the simulation. In particular, it is recommended to combine the components of the liquid desiccant subsystem together with a convergence routine into one single component. This would enable the development of an optimal convergence scheme for this specific problem.

A very important step is to verify the component models and the system model with experimental data to be taken at the Science Museum of Virginia. Of particular interest is the performance of each component at off-design conditions, i.e., over the whole range of feasible variations in air and solution flow rates and for various modes of operation.

Once the existing models are verified with experimental data, the models will be useful tools for studies of liquid desiccant systems. Long term simulations would allow a thorough analysis of the system performance in varying modes of operation. This analysis would provide the basis on which it would be possible to decide

whether or not to install solar collectors at the Science Museum of Virginia. Furthermore, control strategies could be developed which would allow for an optimal selection of the operating mode, not only as a function of system states, but also as a function of the current energy cost structure.

Another promising field of study might be the development and analysis of alternate systems which include a liquid desiccant cycle. It is, for example, possible to use part of the condenser heat of the chiller to heat up the regeneration air stream, thus reducing the cooling tower load. In addition outside air could be mixed with return air to increase the air flow rate through the regenerator, which in turn might improve the regenerator performance. Another feature worth studying might be the over-dehumidification of air in part-load situations. The higher temperature, at which the regeneration heat has to be provided in this case might be available from a gas cogenerator at no additional cost. Evaporative cooling of the very dry air would reduce the chiller load significantly.

During the work with TRNSYS it was found that the convergence behavior of steady state simulations is very poor. This unsatisfactory convergence behavior is due to the method of successive substitution used by TRNSYS to achieve convergence. Moreover, the application of the existing convergence promoter did not turn out to be helpful. It is suggested, that the existing convergence promoter be redesigned. It may be useful to include a

means to constrain the new estimate of the variable upon which the convergence routine acts.

The differential equation solver built into TRNSYS was designed to handle differential equations describing open systems. Small errors in the integration occur even if very small tolerances (10^{-10}) are specified since the numerical integration algorithm is implemented in single precision variables. For open systems with streams flowing through them these small errors in the integration do not cause any major problems. However, if the system is closed with respect to one or more variables these small errors accumulate during long term simulations to a very large error. The liquid desiccant cycle is closed with respect to the total salt mass and the accumulating error is typically 3% over 6000 time steps. This large error is not acceptable for closed systems, because it may lead to erroneous results. Hence it is suggested that the differential equation solver be modified to operate with double precision variables. This would reduce the error due to integration by some orders of magnitude.

Hopefully, here presented work will be a step in the process of evaluating the prospects of the application of open cycle liquid desiccant systems in large air conditioning systems.

REFERENCES

1. Bjurström, H. and Raldow, W., "The Absorption Process for Heating, Cooling and Energy Storage - A Historical Survey," Energy Research, Vol. 5, pp. 43-59 (1981).
2. Duffie, J.A. and Beckman, W.A., Solar Engineering of Thermal Processes, John Wiley & Sons, (1980).
3. "ASHRAE Handbook, 1981 Fundamentals." Chapter 5 (Psychrometrics) and 17 (Refrigerant Tables and Charts). American Society of Heating, Refrigeration and Air Conditioning Engineers, Atlanta, GA (1981).
4. Berghmans, J., Heat Pump Fundamentals, Proceedings of the NATO Advanced Study Institute on Heat Pump Fundamentals, Espinho, Spain, September, 1980, Martinus Nijhoff (1983).
5. Threlkeld, J.L., Thermal Environmental Engineering, 1st Edition, Prentice-Hall, Englewood Cliffs, NY (1962).
6. Kays, W.M. and London, A.L., Compact Heat Exchangers, 3rd Edition, McGraw-Hill (1984).
7. Meckler, G., "Designing Energy Integrated HVAC Systems Based on Thermodynamic Efficiency (Science Museum, Richmond, VA)." Proceedings of the Designing and Managing Energy Conscious Commercial Buildings Workshop, Denver CO, (1982).
8. Meckler, G., personal communication.
9. Klein, S.A., et. al., "TRNSYS - A Transient Simulation Program," Engineering Experiment Station Report 38-12, University of Wisconsin-Madison (1983).
10. "MINPACK Routines," written at Argonne National Laboratory, Academic Computing Center, University of Wisconsin, Madison (1980).
11. Carnahan, B., Luther, H.A. and Wilkes, J.O., Applied Numerical Methods, John Wiley and Sons (1969).
12. Ranz, W.E. and Marshall, W.R., "Evaporation from Drops," Parts I and II, Chemical Engineering Progress, Vol 48, No. 3 and 4, pp. 141-146 and pp. 173-180 (1952).

13. Furuta, T. et al., "Formation of Crystals on the Surface of Non-Supported Droplet in Drying," in Drying '82, edited by A.S. Mujumdar, Hemisphere Publishing Corporation (1982).
14. Dlouhy, J. and Gauvin, W.H., "Evaporation Rates in Spray Drying," The Canadian Journal of Chemical Engineering, August (1960).
15. Bose, A.K. and Pei, D.C.T., "Evaporation Rates in Spray Drying," The Canadian Journal of Chemical Engineering, December (1964).
16. Peng, C.S.P., "Liquid Absorbent Desiccant Cooling/Dehumidification Systems for Low Grade Thermal Energy Applications," pp. 44-48 Dissertation, University of Texas-Austin (1980).
17. Johnson, E.F. and Molstad, M.C., "Thermodynamic Properties of Aqueous Lithium Chloride Solutions," Journal of Phys. Colloid Chem., Vol. 44, pp. 257-281 (1951).
18. Lange, E. and Dörr, F., "Lösungs - und Verdünnungswärmen von Salzen von grosser Verdünnung bis zur Sättigung. II. Lithiumchlorid.", Zeitschrift für physikalische Chemie, 121 (1926).
19. Landolt-Börnstein: Zahlenwerte und Funktionen aus Physik, Chemie,...., New Series, Vols IV/2 and II/4, Springer Verlag.
20. Uemura, T., "Studies on the Lithium Chloride - Water Absorption Refrigerating Machine" Technology Reports of the Kansai University, Osaka, Japan, Vol. 9, pp. 71-88 (1967).
21. Pollack, R., "The Thermodynamic Properties of Water up to 1200 K and 3000 bar," Dissertation, IUPAC, Imperial College, London.
22. Schmidt, E., "Properties of Water and Steam in SI - Units," Springer Verlag, Berlin, Heidelberg, New York (1979).
23. Gmelins Handbuch der anorganischen Chemie, Vol Lithium, Verlag Chemie (1960).
24. Handbook of Chemistry and Physics, 61st edition, Chemical Rubber Company, (1980).
25. Löf, G.O.G., Rao, S. and Lenz, T., "Open-Cycle Lithium Chloride (Solar) Cooling." Final Report for the Department of Energy, DOE/CS/30206-T1, December (1981).

26. Kakabaev, A. and Khandurdyev, A., "Absorption Solar Refrigeration Unit with Open Regeneration of Solution," *Geliotekhnika*, 5(4), pp. 68-72 (1969).
27. Collier, R.K., "The Analysis and Simulation of an Open Cycle Absorption Refrigeration System," Solar Energy, Vol 23, pp. 357-366, (1979).
28. Berestneff, A.A., "The Open Absorption System," Refrig. Eng., 32, (1937).
29. Leboeuf, C.M., "Solar Driven LiCl Open Cycle Absorption Cooling with Packed Bed Absorbent Reconcentration," Fort Collins, M.S. Thesis, Colorado State University (1979).
30. Howell, J.R. and Shepherd, E., "Parametric Studies of the Performance of Open-Flow Lithium Chloride Regenerators," Final Report, Solar Energy Research Institute, Subcontract No. XP-1-1244 (1982).
31. "McQuay Templifier. Heat Recovery Water Heaters," Catalog 1200, McQuay Group, Minneapolis, MN (1982).
32. "McQuay Centrifugal. Single Compressor Packaged Water Chillers," Catalog 950, McQuay Group, Minneapolis, MN (1983).

APPENDICES

A Physical Property Functions

A.1 Air-Water

A.2 Lithium Bromide-Water

A.3 Lithium Chloride-Water

B TRNSYS Component Models

B.1 The Heat and Mass Exchanger

B.2 The Heat and Mass Exchanger Sump

B.3 The Heat Exchanger

B.4 The Air Mixer

B.5 The Fan

B.6 The Vapor Compression Heat Pump

B.7 The Vapor Compression Chiller

B.8 The Proportional Integral Controller

B.9 The Load

C TRNSYS Deck of the System Model "HVAC System, Science Museum of Virginia"

D Finite Step Integration Model-Program Listing

APPENDIX A.1

[illegible]

APPENDIX A.1

[illegible]

APPENDIX A.1

```

C*                                     *
C*      FUNCTION HWLIQ                                     *
C*                                     *
C*      Calculates H H2O,liqu. as a function of T H2O      *
C*                                     *
C*****
      FUNCTION HWLIQ (TW,LUN,LOF)
      LOGICAL LOF
      DIMENSION HWLI (0:1)
      DATA HWLI / -1145.72495 , 4.19368 /
      HWLIQ = POLY (1,HWLI,TW)
      IF (LOF.AND.((TW.LT.273.15).OR.(TW.GT.393.15))) THEN
      WRITE (LUN,*) ' / '
      WRITE (11,*)
      1 / >>> WARNING STATEMENT FROM FUNCTION HWLIQ /
      WRITE (11,*)
      1 / >>> WARNING ! TEMPERATURE OUT OF RANGE : /
      $ ,TW
      END IF
      RETURN
      END
C*****
C*                                     *
C*      FUNCTION WAPW                                     *
C*                                     *
C*      Calculates W air as a function of P part.,H2O      *
C*      and Pambient                                       *
C*                                     *
C*****
      FUNCTION WAPW (PW,PAMB,LUN,LOF)
      LOGICAL LOF
      WAPW = 0.62198 * PW / (PAMB - PW)
      RETURN
      END
C*****
C*                                     *
C*      FUNCTION PWWA                                     *
C*                                     *
C*      Calculates P part.,H2O as a function of W H2O      *
C*      and Pambient                                       *
C*                                     *
C*****
      FUNCTION PWWA (WA,PAMB,LUN,LOF)
      LOGICAL LOF
      PWWA = PAMB * WA / (WA + 0.62198)
      RETURN
      END
C*****
C*                                     *

```

APPENDIX A.1

```

C*      FUNCTION HATAWA
C*
C*      Calculates H air as a function of T air,dry
C*      and W air
C*
C*****
      FUNCTION HATAWA (TA,WA,LUN,LOF)
      LOGICAL LOF
      HATAWA = 1.003 * (TA - 273.15) + WA
      $* ( HWLIQ(TA,LUN,.FALSE.)
      $      + HWEVAP(TA,LUN,.FALSE.) )
      IF (LOF.AND.((TA.LT.273.15).OR.(TA.GT.393.15))) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
      1 ' >>> WARNING STATEMENT FROM FUNCTION HATAWA '
      WRITE (LUN,*)
      1 ' >>> WARNING ! TEMPERATURE OUT OF RANGE :',TA
      END IF
      RETURN
      END
C*****
C*
C*      FUNCTION WAHATA
C*
C*      Calculates W air as a function of H air
C*      and T air,dry
C*
C*****
      FUNCTION WAHATA (HA,TA,LUN,LOF)
      LOGICAL LOF
      WAHATA = (HA - ( 1.003 * (TA - 273.15))) /
      $      ( HWLIQ(TA,LUN,.FALSE.)
      $      + HWEVAP(TA,LUN,.FALSE.) )
      IF (LOF.AND.((TA.LT.273.15).OR.(TA.GT.393.15))) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
      $ ' >>> WARNING STATEMENT FROM FUNCTION WAHATA '
      WRITE (LUN,*)
      $ ' >>> WARNING ! TEMPERATURE OUT OF RANGE :',TA
      END IF
      RETURN
      END
C*****
C*
C*      FUNCTION TAHAWA
C*
C*      Calculates T air,dry as a function of H air
C*      and W air
C*

```

APPENDIX A.1

```

C*****
      FUNCTION TAHAWA (HA,WA,LUN,LOF)
      LOGICAL LOF ,LOFDEB
      TOL = 0.000001
      TA1 = 290.0
      TA2 = 330.0
      HA2 = HATAWA (TA2,WA,LUN,LOFDEB) - HA
4500  CONTINUE
      HA1 = HATAWA (TA1,WA,LUN,LOFDEB) - HA
      TAN = TA1 - HA1 / (( HA2 - HA1 ) / ( TA2 - TA1 ))
      TA2 = TA1
      TA1 = TAN
      HA2 = HA1
      IF (ERRFU(TA1,TA2,LUN,.FALSE.).GT.TOL) GOTO 4500
C LOOP END
      IF (LOF.AND.((TAN.LT.273.15).OR.(TAN.GT.393.15))) THEN
      WRITE (LUN,*) '
      WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM FUNCTION TAHAWA '
      WRITE (LUN,*)
1 ' >>> WARNING ! TEMPERATURE OUT OF RANGE :',TAN
      END IF
      TAHAWA = TAN
      RETURN
      END
C*****
C*
C*      FUNCTION DATAWA
C*
C*      Calculates RHO dry air as a function of T air
C*      and W air
C*
C*      ASHRAE Equations 26, 9a , page 5.3 ; modified
C*
C*****
      FUNCTION DATAWA (TA,WA,PAMB,LUN,LOF)
      LOGICAL LOF
      RA = 287.055
C      GAS CONSTANT FOR DRY AIR IN [J / KG K]
      DATAWA = PAMB / ( RA * TA * ( 1.0 + 1.6078 * WA ))
      RETURN
      END
C*****
C*
C*      FUNCTION PWTW
C*
C*      Calculates P part.,H2O as a function of T H2O
C*
C*****

```

APPENDIX A.1

```

FUNCTION PWTW (TW,LUN,LOF)
LOGICAL LOF
DIMENSION CPP(0:2,2)
DATA CPP / 10.09434429 , -1632.60428 , -99377.4921 ,
1      11.1604295 , -1966.75688 , -88060.8653 /
IF (LOF.AND.((TW.LT.233.15).OR.(TW.GT.393.15))) THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM FUNCTION PWTW '
WRITE (LUN,*)
1 ' >>> WARNING ! TEMPERATURE OUT OF RANGE :',TW
END IF
IF (TW.GT.275.0) THEN
IOPT = 1
ELSE
IOPT = 2
END IF
OOTW = 1.0 / TW
PP = POLY (2,CPP(0,IOPT),OOTW)
PWTW = 10.0 ** PP
RETURN
END

C*****
C*
C*      FUNCTION TWPW
C*
C*      Calculates T H2O as a function of P part.,H2O
C*
C*****
FUNCTION TWPW (PW,LUN,LOF)
LOGICAL LOF
DIMENSION CPP(0:2,2)
DATA CPP / 10.09434429 , -1632.60428 , -99377.4921 ,
1      11.1604295 , -1966.75688 , -88060.8653 /
IF (PW.GT.700.0) THEN
IOPT = 1
ELSE
IOPT = 2
END IF
TWPW = -2.0*CPP(2,IOPT)/
1 (CPP(1,IOPT)+SQRT(CPP(1,IOPT)**2-4*CPP(2,IOPT)
2 *(CPP(0,IOPT)-LOG10(PW))))
IF (LOF.AND.((TWPW.LT.233.15).OR.(TWPW.GT.393.15)))
1 THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM FUNCTION TWPW '
WRITE (LUN,*)
1 ' >>> WARNING ! TEMPERATURE OUT OF RANGE :',TWPW

```


APPENDIX A.1

```

      END IF
      RETURN
      END
C*****
C*
C*      SUBROUTINE POLY
C*
C*      CALCULATES THE POLYNOM
C*
C*
C*      POLY = AR  $\times X^I$  + AR  $\times X^{I-1}$  + ... + AR  $\times X^0$ 
C*
C*
C*****
      FUNCTION POLY (I,AR,X)
      DIMENSION AR (0:I)
      P = AR (I)
      DO 7010 I701 = I-1,0,-1
      P = P*X+AR(I701)
7010  CONTINUE
      POLY = P
      RETURN
      END
C*****
C*
C*      FUNCTION ERRFU
C*
C*      calculates the relativ error
C*
C*
C*****
      FUNCTION ERRFU (VALNEW,VALOLD,LUN,LOF)
      LOGICAL LOF
      ERRFU = ABS (VALNEW - VALOLD) / VALNEW * 100.0
      IF (LOF) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*) ' ERRFU DEBUGGING STATEMENT'
      WRITE (LUN,*) ' VALNEW',VALNEW,' VALOLD',VALOLD
      WRITE (LUN,*) ' ERRFU',ERRFU
      END IF
      RETURN
      END
C*****
C*
C*      LOGICAL FUNCTION SUPSAT
C*
C*      SUPSAT is true, if the air at T air,dry and W air
C*      is supersaturated
C*
C*
C*****

```

APPENDIX A.1

```

LOGICAL FUNCTION SUPSAT (TAIR,WAIR,PAMB,LUN,LOF)
LOGICAL LOF
PWAIR = PWWA (WAIR,PAMB,LUN,.FALSE.)
PWSAT = PWTW (TAIR,LUN,.FALSE.)
SUPSAT = (PWSAT.LT.PWAIR)
IF (LOF.AND.SUPSAT) THEN
WRITE (LUN,*) ' '
WRITE (LUN,1) TAIR-273.15, WAIR
1  FORMAT (' AIR AT T =',F7.2,' [ DEG C ]'//
$      ' AND W =',F7.3,' IS SUPERSATURATED !')
WRITE (LUN,2) PWAIR, PWSAT
2  FORMAT (' PARTIAL PRESSURE   =',F8.1,' [ PA ]'//
$      ' SATURATION PRESSURE =',F8.1,' [ PA ]')
WRITE (LUN,*) ' '
END IF
RETURN
END

```

APPENDIX A.2

[illegible]

APPENDIX A.2

[illegible]

APPENDIX A.2

```

      TSPWXI = ATP * (TW - 273.15) + BTP + 273.15
      IF (LOF.AND.(XI.LT.0.45.OR.XI.GT.0.7)) THEN
        WRITE (LUN,*) ' / '
        WRITE (LUN,*)
        1 ' >>> WARNING STATEMENT FROM SUBROUTINE TSPWXI '
        WRITE (LUN,6180) XI
6180  FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
        1 ',OUT OF RANGE : ',F5.3)
      END IF
      RETURN
      END

C*****
C*
C*      FUNCTION XITSPW
C*
C*      Calculates XI sol.as a function of T sol.
C*      and P part.,H2O
C*
C*****
      FUNCTION XITSPW (TSS,PW,LUN,LOF)
      LOGICAL LOF
      TOL = 0.00001
      XI1 = 0.5
      XI2 = 0.6
      PW2 = PWTSXI (TSS,XI2,LUN,.FALSE.) - PW
4000  CONTINUE
      PW1 = PWTSXI (TSS,XI1,LUN,.FALSE.) - PW
      XIN = XI1 - PW1 / (( PW2 - PW1 ) / ( XI2 - XI1 ))
      XI2 = XI1
      XI1 = XIN
      PW2 = PW1
      IF (ABS (ERRFU(XI1,XI2,LUN,.FALSE.))>.GT.TOL) GOTO 4000
      XITSPW = XIN
      IF (LOF.AND.(XIN.LT.0.45.OR.XIN.GT.0.7)) THEN
        WRITE (LUN,*) ' / '
        WRITE (LUN,*)
        1 ' >>> WARNING STATEMENT FROM SUBROUTINE XITSPW '
        WRITE (LUN,6170) XIN
6170  FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
        1 ',OUT OF RANGE : ',F5.3)
      END IF
      RETURN
      END

C*****
C*
C*      SUBROUTINE HPOL
C*
C*      CALCULATES THE SUMS  AHP, BHP, CHP
C*
C*

```

APPENDIX A.2

```

C*          4          K          *
C*      AHP = SUM ( AH * XI )      *
C*          0          K          *
C*                                *
C*****
      SUBROUTINE HPOL (XI,AHP,BHP,CHP,LUN,LOF)
      LOGICAL LOF
      DIMENSION AH (0:4), BH (0:4), CH (0:4)
      DATA AH / -2024.33 , 163.309 , -4.88161 ,
$ 6.302948E-002 , -2.913705E-004 /
$ BH / 18.2829 , -1.169175 ,
$ 3.248041E-002 , -4.034184E-004 , 1.8520569E-006 /
$ CH / -3.7008214E-002 , 2.8877666E-003 ,
$ -8.1313015E-005 , 9.9116628E-007 ,
$ -4.4441207E-009 /
      IF (LOF.AND.(XI.LT.0.4.OR.XI.GT.0.7)) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
$ ' >>> WARNING STATEMENT FROM SUBROUTINE HPOL '
      WRITE (LUN,6010) XI
6010  FORMAT (< >>> WARNING ! SOLUTION CONCENTRATION '
$ , 'OUT OF RANGE : ',F5.3)
      END IF
      XII = XI * 100.0
      AHP = POLY (4,AH,XII)
      BHP = POLY (4,BH,XII)
      CHP = POLY (4,CH,XII)
      RETURN
      END
C*****
C*                                *
C*      FUNCTION TSHSXI          *
C*                                *
C*      Calculates T sol. as a function of H sol.      *
C*      and XI sol.          *
C*                                *
C*****
      FUNCTION TSHSXI (HS,XI,LUN,LOF)
      LOGICAL LOF
      CALL HPOL (XI,AHP2,BHP2,CHP2,LUN,LOF)
      TS2 = ((-BHP2 + SQRT(BHP2**2-4.0*CHP2*(AHP2 - HS)))
1 / (2.0 * CHP2))
      TSHSXI = TS2 + 273.15
      IF (LOF.AND.(TS2.LT.15.0.OR.TS2.GT.165.0)) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM FUNCTION TSHSXI '
      WRITE (LUN,*)
1 ' >>> WARNING ! SOLUTION TEMPERATURE OUT OF RANGE : '

```

APPENDIX A.2

```

      2 ,TSHSXI
      END IF
      RETURN
      END
C*****
C*
C*      FUNCTION HSTSXI
C*
C*      Calculates H sol. as a function of T sol.
C*      and XI sol.
C*
C*****
      FUNCTION HSTSXI (TS,XI,LUN,LOF)
      LOGICAL LOF
      DIMENSION HCOEFF (0:2)
      CALL HPOL (XI,HCOEFF(0),HCOEFF(1),HCOEFF(2),LUN,LOF)
      TS1 = TS - 273.15
      IF (LOF.AND.((TS1.LT.15.0).OR.(TS1.GT.165.0))) THEN
      WRITE (LUN,*) '
      WRITE (LUN,*)
      1 ' >>> WARNING STATEMENT FROM FUNCTION HSTSXI '
      WRITE (LUN,*)
      1 ' >>> WARNING ! SOLUTION TEMPERATURE OUT OF RANGE :
      2 ,TS
      END IF
      HSTSXI = POLY (2,HCOEFF,TS1)
      RETURN
      END
C*****
C*
C*      FUNCTION DSTSXI
C*
C*      Calculates RHO sol. as a function of T sol.
C*      and XI sol.
C*
C*****
      FUNCTION DSTSXI (TS,XI,LUN,LOF)
      LOGICAL LOF
      DIMENSION DENSA (0:4), DENSB (0:4), DENSC (0:1)
      DATA DENSA / 1.119705 , 0.805575 , 0.3259097 ,
      $           0.187312904 , 1.16504197 /
      $   DENSB / -4.18781938E-004 , -3.32594749E-004 ,
      $           8.49287599E-004 , -1.78076102E-003 ,
      $           3.80812252E-004 /
      IF (LOF.AND.(TS.LT.273.0R.TS.GT.374)) THEN
      WRITE (LUN,*) '
      WRITE (LUN,*)
      $ ' >>> WARNING STATEMENT FROM FUNCTION DSTSXI '
      WRITE (LUN,*)

```

APPENDIX A.2

```

$ ' >>> WARNING !! SOLUTION TEMPERATURE OUT OF'
$ , ' RANGE : ', TS
END IF
IF (LOF.AND.(XI.LT.0.02.OR.XI.GT.0.65)) THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
$ ' >>> WARNING STATEMENT FROM FUNCTION DSTSXI '
WRITE (LUN,*)
$ ' >>> WARNING !! SOLUTION CONCENTRATION OUT OF'
$ , ' RANGE : ', XI
END IF
DENS (0) = POLY (4,DENSA,XI)
DENS (1) = POLY (4,DENSB,XI)
DSTSXI = POLY (1,DENS,TS) * 1000.0
RETURN
END
C*****
C*
C* SUBROUTINE TPOL
C*
C* CALCULATES THE SUMS ATP, BTP
C*
C*
C*          3          K
C* ATP = SUM ( AT * XI )
C*          0          K
C*
C*****
SUBROUTINE TPOL (XI,ATP,BTP,LUN,LOF)
LOGICAL LOF
DIMENSION AT(0:3) , BT(0:3)
DATA AT / -2.00755 , 0.16976 ,
$ -3.133336E-003 , 1.97668E-005 /
$ BT / 124.937 , -7.7165 ,
$ 1.52286E-001 , -7.9509E-004 /
IF (LOF.AND.(XI.LT.0.45.OR.XI.GT.0.7)) THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
$ ' >>> WARNING STATEMENT FROM SUBROUTINE TPOL '
WRITE (LUN,6110) XI
6110 FORMAT ( ' >>> WARNING ! SOLUTION CONCENTRATION '
$ , ' OUT OF RANGE : ', F5.3)
END IF
XIA = XI * 100.0
ATP = POLY (3,AT,XIA)
BTP = POLY (3,BT,XIA)
RETURN
END

```


APPENDIX A.3

[illegible]

APPENDIX A.3

```

C*****
C$ UNIT CONVENTION $
C$ $
C$ IN THIS PACKAGE OF SUBROUTINES THE SI UNITS $
C$ ARE APPLIED: $
C$ $
C$ TEMPERATURE IN K $
C$ PRESSURE IN PA ( N / M^2 ) $
C$ SPECIFIC ENTHALPY IN KJ / KG SOL. $
C$ DENSITY IN KG / M^3 $
C$ $
C$ CONCENTRATION IN KG SALT / $
C$ KG SOLUTION $
C*****
C* FUNCTION PWTSXI $
C* $
C* CALCULATES P part.,H2O as a function of T sol. $
C* and XI sol. $
C* $
C*****
FUNCTION PWTSXI (TS,XI,LUN,LOF)
LOGICAL LOF
CALL TPOL (XI,ATP,BTP,LUN,.FALSE.)
TW = ATP * (TS - 273.15) + BTP + 273.15
PWTSXI = PWTW (TW,LUN,LOF)
IF (LOF.AND.(XI.LT.0.05.OR.XI.GT.0.45)) THEN
WRITE (LUN,*) '
WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM SUBROUTINE PWTSXI '
WRITE (LUN,10) XI
10 FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION'
1 ', OUT OF RANGE : ',F5.3)
END IF
RETURN
END
C*****
C* FUNCTION TSPWXI $
C* $
C* CALCULATES T sol. as a function of P part.,H2O $
C* and XI sol. $
C* $
C*****
FUNCTION TSPWXI (PW,XI,LUN,LOF)

```

APPENDIX A.3

```

LOGICAL LOF
CALL TPOL (XI,ATP,BTP,LUN,.FALSE.)
TW = TWPW (PW,LUN,LOF)
TSPWXI = (TW-273.15-BTP)/ATP + 273.15
IF (LOF.AND.(XI.LT.0.05.OR.XI.GT.0.45)) THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM SUBROUTINE TSPWXI '
WRITE (LUN,6180) XI
6180 FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
1 , ' OUT OF RANGE : ',F5.3)
END IF
RETURN
END

C*****
C*
C*      FUNCTION XITSPW
C*
C*      Calculates XI sol.as a function of T sol.
C*      and P part.,H2O
C*
C*****

      FUNCTION XITSPW (TSS,PW,LUN,LOF)
      LOGICAL LOF
      TOL = 0.00001
      XI1 = 0.2
      XI2 = 0.4
      PW2 = PWTSXI (TSS,XI2,LUN,.FALSE.) - PW
4000 CONTINUE
      PW1 = PWTSXI (TSS,XI1,LUN,.FALSE.) - PW
      XIN = XI1 - PW1 / (( PW2 - PW1 ) / ( XI2 - XI1 ))
      XI2 = XI1
      XI1 = XIN
      PW2 = PW1
      IF (ABS (ERRFU(XI1,XI2,LUN,.FALSE.))>.TOL) GOTO 4000
      XITSPW = XIN
      IF (LOF.AND.(XIN.LT.0.05.OR.XIN.GT.0.45)) THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
1 ' >>> WARNING STATEMENT FROM SUBROUTINE XITSPW '
WRITE (LUN,6170) XIN
6170 FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
1 , ' OUT OF RANGE : ',F5.3)
END IF
RETURN
END

C*****
C*
C*      FUNCTION ICPDT
C*

```

APPENDIX A.3

```

C*                                     *
C*      CALCULATES THE INTEGRAL      *
C*                                     *
C*      / T=TS                        *
C*      ICPDT =  $\int$  Cp dT              *
C*      / T=25 DEG C                  *
C*                                     *
C*****
      REAL FUNCTION ICPDT (TS,XI,LUN,LOF)
      LOGICAL LOF
      DIMENSION AH (0:2), BH (0:2), CH (0:2), CPCO (0:3)
      DATA AH / 1.0020 , -1.2505 , 0.7575 /
      $      BH / -5.554E-04 , -1.5178E-03 , 6.8248E-03 /
      $      CH / 5.2266E-06 , 3.6623E-06 , -3.8345E-05 /
      IF (LOF.AND.(XI.LT.0.05.OR.XI.GT.0.45)) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
      1 ' >>> WARNING STATEMENT FROM FUNCTION ICPDT '
      WRITE (LUN,6010) XI
6010  FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
      1 , 'OUT OF RANGE : ',F5.3)
      END IF
      IF (LOF.AND.((TS.LT.10.0).OR.(TS.GT.110.0))) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
      1 ' >>> WARNING STATEMENT FROM FUNCTION ICPDT '
      WRITE (LUN,*)
      1 ' >>> WARNING ! SOLUTION TEMPERATURE OUT OF RANGE : '
      2 , TS
      END IF
      AHP = POLY (2,AH,XI)
      BHP = POLY (2,BH,XI)
      CHP = POLY (2,CH,XI)
      CPCO (0) = 0.0
      CPCO (1) = AHP
      CPCO (2) = BHP / 2.0
      CPCO (3) = CHP / 3.0
      ICPDT = 4.19 *
      1 ( POLY (3,CPCO,TS) - POLY (3,CPCO,25.0) )
      RETURN
      END
C*****
C*                                     *
C*      FUNCTION IES                  *
C*                                     *
C*      CALCULATES THE INTEGRAL ENTHALPY OF SOLUTION *
C*                                     *
C*****
      REAL FUNCTION IES (XI,LUN,LOF)

```

APPENDIX A.3

```

LOGICAL LOF
REAL IESCO (0:3)
DATA IESCO / 0.875850824 , 839.866148 ,
$          61.5398937 , -1978.63552 /
IF (LOF.AND.(XI.LT.0.005.OR.XI.GT.0.46)) THEN
WRITE (LUN,*) ' '
WRITE (LUN,*)
1 / >>> WARNING STATEMENT FROM FUNCTION IES '
WRITE (LUN,6010) XI
6010 FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
1 , 'OUT OF RANGE : ',F5.3)
END IF
IES = POLY (3,IESCO,XI)
RETURN
END

C*****
C*
C*      FUNCTION TSHSXI
C*
C*      Calculates T sol. as a function of H sol.
C*      and XI sol.
C*
C*****
      FUNCTION TSHSXI (HS,XI,LUN,LOF)
      LOGICAL LOF
      DATA TOL / 1.0E-05 /
      TS1 = 30.0
      TS2 = 60.0
      HS2 = HSTSXI (TS2,XI,LUN,.FALSE.) - HS
10  CONTINUE
      HS1 = HSTSXI (TS1,XI,LUN,.FALSE.) - HS
      TSN = TS1 - HS1 / ((HS2 - HS1) / (TS2 - TS1))
      TS2 = TS1
      TS1 = TSN
      HS2 = HS1
      IF (ABS(ERRFU(TS1,TS2,LUN,.FALSE.)).GT.TOL) GOTO 10
C-----
C      CALL OF HSTSXI TO GET ERROR MESSAGES !
C-----
      HS1 = HSTSXI (TSN,XI,LUN,LOF)
      TSHSXI = TSN
      RETURN
      END

C*****
C*
C*      FUNCTION HSTSXI
C*
C*      Calculates H sol. as a function of T sol.
C*      and XI sol.
C*

```

APPENDIX A.3

```

C*
C*****
      FUNCTION HSTSXI (TS,XI,LUN,LOF)
      LOGICAL LOF
      REAL ICPDT, IES
      TS1 = TS - 273.15
      HSTSXI = 104.75 - IES(XI,LUN,LOF)
1      + ICPDT(TS1,XI,LUN,LOF)
      RETURN
      END
C*****
C*
C*      FUNCTION DSTSXI
C*
C*      Calculates RHO sol. as a function of T sol.
C*      and XI sol.
C*
C*****
      FUNCTION DSTSXI (TS,XI,LUN,LOF)
      LOGICAL LOF
      DIMENSION DENSA (0:2), DENSB (0:2), DENSC (0:2),
$      DENS (0:2)
      DATA DENSA / 0.767197 , 1.65198915 , -0.37664242 /
$      DENSB / 1.826997E-03 , -7.71858335E-03 ,
$      5.47099245E-03 /
$      DENSC / -3.51677505E-06 , 1.2992472E-05 ,
$      -1.05815404E-05 /
      IF (LOF.AND.(TS.LT.273.OR.TS.GT.374)) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
$ ' >>> WARNING STATEMENT FROM FUNCTION DSTSXI '
      WRITE (LUN,*)
$ ' >>> WARNING !! SOLUTION TEMPERATURE OUT OF'
$ ', RANGE :',TS
      END IF
      IF (LOF.AND.(XI.LT.0.02.OR.XI.GT.0.65)) THEN
      WRITE (LUN,*) ' '
      WRITE (LUN,*)
$ ' >>> WARNING STATEMENT FROM FUNCTION DSTSXI '
      WRITE (LUN,*)
$ ' >>> WARNING !! SOLUTION CONCENTRATION OUT OF'
$ ', RANGE :',XI
      END IF
      DENS (0) = POLY (2,DENSA,XI)
      DENS (1) = POLY (2,DENSB,XI)
      DENS (2) = POLY (2,DENSC,XI)
      DSTSXI = POLY (2,DENS,TS) * 1000.0
      RETURN
      END

```

APPENDIX A.3

```

C*****
C*
C*      SUBROUTINE TPOL
C*
C*      CALCULATES THE SUMS  ATP, BTP
C*
C*
C*      3      K      3      K
C*      ATP = SUM ( AT  * XI ) , BTP = SUM ( BT  * XI )
C*      0      K      0      K
C*
C*****
      SUBROUTINE TPOL (XI,ATP,BTP,LUN,LOF)
      LOGICAL LOF
      DIMENSION AT(0:3) , BT(0:3)
      DATA AT / 1.00011872 , -0.132800828 ,
$ 4.82235441E-02 , -0.507596043 /
$ BT / -0.43831165 , 14.1379014 ,
$ -224.535483 , 123.29564 /
      IF (LOF.AND.(XI.LT.0.05.OR.XI.GT.0.45)) THEN
        WRITE (LUN,*) ' '
        WRITE (LUN,*)
$ ' >>> WARNING STATEMENT FROM SUBROUTINE TPOL '
        WRITE (LUN,6110) XI
6110 FORMAT (' >>> WARNING ! SOLUTION CONCENTRATION '
$ , 'OUT OF RANGE : ',F5.3)
      END IF
      ATP = POLY (3,AT,XI)
      BTP = POLY (3,BT,XI)
      RETURN
      END

```

APPENDIX B.1

[illegible]

APPENDIX B.1

```
C#          9      SENSIBLE HEAT EXCHANGED                      #
C#          10     LATENT HEAT EXCHANGED                        #
C#          11     MASS OF WATER EVAPORATED                     #
C#          12     "HUMIDITY RATIO" AT SOLUTION OUTLET         #
C#                                                     #
C#           PARAMETERS :                                       #
C#                                                     #
C#             1      MODE :                                     #
C#                               1    REGENERATOR               #
C#                               2    CONDITIONER                #
C#                                                     #
C#####
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$
C$   UNIT CONVENTION                                           $
C$
C$   IN THIS COMPONENT THE SI UNITS (MODIFIED) ARE APPLIED :  $
C$
C$       TEMPERATURE ( INTERNAL ) IN      K                    $
C$       TEMPERATURE IN & OUTPUT  IN      DEG C                 $
C$       PRESSURE                   IN      PA ( N / M^2 )        $
C$       SPECIFIC ENTHALPY          IN      KJ / KG SOLUTION      $
C$       SPECIFIC ENTHALPY AIR      IN      KJ / KG DRY AIR       $
C$       MASS FLOW RATE              IN      KG / HOUR            $
C$       ENERGY FLOW RATE          IN      KJ / HOUR            $
C$
C$       SOLUTION CONCENTRATION      IN      KG SALT / KG SOLUTION $
C$       HUMIDITY RATIO              IN      KG WATER / KG AIR    $
C$
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C   SUBROUTINE TYPE 48 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
C   PARAMETER (NI48=6,N048=20,NP48=1,ND48=0
C$           ,NI481=6,N0481=12,NP481=1,ND481=0
C$           )
C   IMPLICIT REAL (M)
C   DIMENSION XIN (NI48), OUT (N048), PAR (NP48), INFO (10)
C   INTEGER    MODE
C   LOGICAL    LERROR, SUPSAT, CONDIT
C   LOGICAL    LOF, LOF1, LOF2, LOF3, LOF4, LOF5, LOF6, LOF7, LOF8
C*****
C   COMMON / IS / TSI, WSI, XISI, TAI, WAI, PAMB
C   COMMON / IF / MDSI, HSI, MDAl, HAl, HASI, QDSI, QDAI
C   COMMON / OS / TSO(6), WSO(6), XISO(6), TAO(6), WAO(6), MDAO
C   COMMON / OF / MDSO(6), HSO(6), HAO(6), HASO(6), QDSO(6), QDAO(6)
C   COMMON / EX / DMEQU(6), DQEQS(6), DQEQL(6)
C   COMMON / CD / ICOND (4,6)
C   COMMON / FL / LOF
C*****
C   DATA      LUN / 11 /
```

APPENDIX B.1

```

LOF = . TRUE.
LOF1 = .FALSE.
LOF2 = .FALSE.
LOF3 = .FALSE.
LOF4 = .FALSE.
LOF5 = .FALSE.
LOF6 = .FALSE.

C-----
C      PRESSURE OF AMBIENT AIR : PAMB
C-----
      PAMB = 101325.0
C-----
C      MODE PARAMETER
C-----
      MODE = INT (PAR(1))
C-----
C      ASSIGNMENT OF THE INLET STATES
C-----
      MDSI = XIN(1)
      TSI = XIN(2) + 273.15
      XISI = XIN(3)
      MDAI = XIN(4)
      TAI = XIN(5) + 273.15
      WAI = XIN(6)
C-----
C      CHECK, WHETHER AIR AT INLET IS SUPERSATURATED
C-----
      IF (SUPSAT(TAI,WAI,PAMB,LUN,.TRUE.)) THEN
        WRITE (LUN,*) ' '
        WRITE (LUN,*) '*** FATAL ERROR : AIR AT INLET IS SUPERSATURATED !'
        WRITE (LUN,*) ' '
        CALL TYPECK (-2,INFO,NI482,NP482,ND482)
        END IF
C-----
C
C      SIMPLE EQUILIBRIUM - EFFECTIVENESS FACTOR MODEL
C
C      ASSUMING :
C
C      - ADIABATIC PROCESS
C      - COUNTERCURRENT FLOW
C
C-----
C:.....
      IF (INFO(7).EQ.-1) THEN
        CALL TYPECK (1,INFO,NI481,NP481,ND481)
        INFO (6) = N0481
        END IF
C:.....

```

APPENDIX B.1

```

C-----
C      MASS BALANCE DRY AIR
C-----
      MDAO  = MDAI
      MDAOUT = MDAO
C-----
C      ENTHALPIES OF THE INLET STATES
C-----
      HSI = HSTSXI (TSI,XISI,LUN,LOF)
      HAI = HATAWA (TAI,WAI,LUN,LOF)
C-----
      WSI = WAPW(PWTSXI(TSI,XISI,LUN,LOF),PAMB,LUN,LOF)
      HASI = HATAWA (TSI,WSI,LUN,LOF)
C-----
C      ENTHALPY FLOW RATES AT THE INLET
C-----
      QDSI = HSI * MDSI
      QDAI = HAI * MDAI
C-----
C      RESET ICOND ARRAY
C-----
      DO 10 I = 1, 6
10     ICOND (I,I) = 0
C-----
      IF (INFO(8).LE.2) THEN
C-----
C      CHECK THE MOST PROBABLE EQUILIBRIUM
C-----
      IF (MDAI/MDSI.LT.1) THEN
        CALL EQUMan (1,LUN,LOF3,LOF4,LOF5)
        IF (CONDIT(1,LUN,LOF4)) THEN
          JEQU = 1
          GOTO 1000
        END IF
      ELSE
        CALL EQUMan (2,LUN,LOF3,LOF4,LOF5)
        IF (CONDIT(2,LUN,LOF4)) THEN
          JEQU = 2
          GOTO 1000
        END IF
      END IF
C-----
C      CHECK THE OLD EQUILIBRIUM
C-----
      ELSE
        JEQU = INT (OUT(19))
        CALL EQUMan (JEQU,LUN,LOF3,LOF4,LOF5)
        IF (CONDIT(JEQU,LUN,LOF4)) GOTO 1000
      END IF

```

```

C-----
C      NEW CALCULATION OF ALL EQUILIBRIA
C-----
      DO 100 JEQU = 1 , 6
      IF (ICOND (1,JEQU).EQ.0) THEN
        CALL EQUMAN (JEQU,LUN,LOF3,LOF4,LOF5)
        IF (CONDIT(JEQU,LUN,LOF4)) GOTO 1000
      END IF
100  CONTINUE
C-----
C      NO FEASIBLE EQUILIBRIUM DETERMINED
C-----
      WRITE (LUN,*) ' '
      WRITE (LUN,*) '*** FATAL ERROR : NO FEASIBLE EQUILIBRIUM !'
      WRITE (LUN,*) ' '
      CALL TYPECK (-2,INFO,NI482,NP482,ND482)
      GOTO 9999
C-----
C      EQUILIBRIUM DETERMINED
C-----
1000  CONTINUE
C-----
C      ASSIGNMENT OF THE EXCHANGED MASS AND ENERGY FOR EQUILIBRIUM
C-----
      DMDEQU = DMEQU (JEQU)
      DQDEQS = DQEQS (JEQU)
C-----
C      CALCULATION OF THE REDUCED (EFFECTIVE) EXCHANGED
C      MASS AND ENERGY
C-----
      EFFHEAT = EFF (TSI-273.15,1,MODE)
      EFFMASS = EFF (TSI-273.15,2,MODE)
      DQDS = DQDEQS * EFFHEAT
      DMD = DMDEQU * EFFMASS
C-----
C      CALCULATION OF THE NEW OUTLET STATES
C-----
      WAOUT = WAI + DMD / MDAI
      HAOI = HAI + DQDS / MDAI
      TAOUT = TAHAWA (HAOI,WAI,LUN,LOF)
      HAOUT = HATAWA (TAOUT,WAOUT,LUN,LOF)
      DQDL = MDAI * (HAOUT - HAOI)
      MDSOUT = MDSI - DMD
      XISOUT = XISI * MDSI / MDSOUT
      HSOUT = (QDSI - DQDS - DQDL) / MDSOUT
      TSOUT = TSHSXI (HSOUT,XISOUT,LUN,LOF)
C-----
C      ADDITIONAL OUTPUT CALCULATION
C      DOCUMENTATION OF W SOL OUT

```

APPENDIX B.1

```

C-----
      PWSOUT = PWTSXI (TSOUT,XISOUT,LUN,LOF)
      WSOUT  = WAPW (PWSOUT,PAMB,LUN,LOF)
C-----
C      ASSIGNMENT OF THE OUTLET STATES
C-----
      OUT (1) = MDSOUT
      OUT (2) = TSOUT - 273.15
      OUT (3) = XISOUT
      OUT (4) = MDAOUT
      OUT (5) = TAOUT - 273.15
      OUT (6) = WAOUT
      OUT (7) = HSOUT
      OUT (8) = HAOUT
      OUT (9) = DQDS
      OUT (10) = DQDL
      OUT (11) = DMD
      OUT (12) = WSOUT
      OUT (19) = FLOAT (JEQU)
C-----
C      CHECK, IF AIR AT OUTLET IS SUPERSATURATED
C-----
      IF (SUPSAT(TAI,WAI,PAMB,LUN,.TRUE.)) THEN
        WRITE (LUN,*) ' '
        WRITE (LUN,*) ' AIR AT OUTLET IS SUPERSATURATED ! '
        WRITE (LUN,*) ' '
      END IF
C-----
C      EXIT LABEL
C-----
9999  CONTINUE
      RETURN
      END
C*****
C*                                          *
C*      REAL FUNCTION EFF (TSI,MODE,NUMBER)  *
C*                                          *
C*      TSI      SOLUTION INLET TEMPERATURE  *
C*      NUMBER   1  SENSIBLE HEAT EXCHANGE EFFECTIVENESS  *
C*               2  MASS          EXCHANGE EFFECTIVENESS  *
C*      MODE     1  REGENERATOR  *
C*               2  CONDITIONER  *
C*                                          *
C*****
      REAL FUNCTION EFF (TSI,NUMBER,MODE)
      DIMENSION COEFF (0:2,2,2)
      DATA      COEFF / 0.2168 , 0.029792 , -0.00032190 ,
1                0.85323 , 0.0039145 , -0.000095238 ,
2                0.36757 , 0.067205 , -0.0021933 ,

```

APPENDIX B.1

```

3          0.82885 , 0.010579 , -0.0003351 /
EFF      = ((COEFF (2,NUMBER,MODE) * TSI)
1          + COEFF (1,NUMBER,MODE)) * TSI + COEFF(0,NUMBER,MODE)
RETURN
END
C*****
C*
C*      SUBROUTINE EQUMAN
C*
C*      CALLS THE DESIRED EQUILIBRIUM CALCULATION SUBROUTINE
C*
C*****
      SUBROUTINE EQUMAN (IEQU,LUN,LOF3,LOF4,LOF5)
      IMPLICIT LOGICAL (L)
      INTEGER LUN
      GOTO (100,200,300,400,500,600) IEQU
100  CONTINUE
      CALL EQUI1 (LUN,LOF3,LOF4,LOF5)
      RETURN
200  CONTINUE
      CALL EQUI2 (LUN,LOF3,LOF4,LOF5)
      RETURN
300  CONTINUE
      CALL EQUI3 (LUN,LOF3,LOF4,LOF5)
      RETURN
400  CONTINUE
      CALL EQUI4 (LUN,LOF3,LOF4,LOF5)
      RETURN
500  CONTINUE
      CALL EQUI5 (LUN,LOF3,LOF4,LOF5)
      RETURN
600  CONTINUE
      CALL EQUI6 (LUN,LOF3,LOF4,LOF5)
      RETURN
      END
C*****
C*
C*      SUBROUTINE CONDIT
C*
C*      FINDS THE FEASIBLE EQUILIBRIUM
C*
C*****
      LOGICAL FUNCTION CONDIT (IEQU,LUN,LOF)
      IMPLICIT LOGICAL (L)
      INTEGER LUN
      COMMON / CD / ICOND (4,6)
C-----
      KCD = 0
      DO 1000 JE = 1,4

```

```

      KCO = KCO + ICOND (JE,IEQU)
1000  CONTINUE
      CONDIR = (KCO,EQ.4)
C-----
      RETURN
      END
C*****
C*
C*      SUBROUTINE EQUI1
C*
C*      EQUILIBRIUM AT THE SOLUTION INLET  ( 1 )
C*
C*****
      SUBROUTINE EQUI1 (LUN,LOF11,LOF12,LOF13)
      LOGICAL SUPSAT
      IMPLICIT REAL (M)
      IMPLICIT LOGICAL (L)
      INTEGER LUN
      COMMON / IS /  TSI, WSI, XISI, TAI, WAI, PAMB
      COMMON / IF /  MDSI, HSI, MDAI, HAI, HASI, QDSI, QDAI
      COMMON / OS /  TSO(6), WSO(6), XISO(6), TAO(6), WAO(6), MDAO
      COMMON / OF /  MDSO(6), HSO(6), HAO(6), HASO(6), QDSO(6), QDAO(6)
      COMMON / EX /  DMEQU(6), DQEQS(6), DQEQL(6)
      COMMON / CD /  ICOND (4,6)
      COMMON / FL /  LOF
C-----
      TAO (1) = TSI
      PWSI = PWTSXI (TSI,XISI,LUN,LOF)
      WAO (1) = WAPW (PWSI,PAMB,LUN,LOF)
      HAO (1) = HATAWA (TAO(1),WAO(1),LUN,LOF)
      QDAO(1) = HAO(1) * MDAO
      DMEQU (1) = MDAI * ( WAO (1) - WAI )
C-----
C      DMEQU IS POSITIVE, IF EVAPORATION  ( REGENERATOR )
C      DMEQU IS NEGATIVE, IF CONDENSATION ( DEHUMIDIFIER )
C-----
C      SOLUTION MASS BALANCE
C-----
      MDSO (1) = MDSI - DMEQU (1)
C-----
C      OUTLET SOLUTION CONCENTRATION
C-----
      XISO (1) = MDSI * XISI / MDSO (1)
C-----
C      OVERALL ENERGY BALANCE
C-----
      QDSO (1) = ( QDSI + QDAI - QDAO (1) )
      HSO (1) = QDSO (1) / MDSO (1)
C-----

```

APPENDIX B.1

```

C      OUTLET SOLUTION TEMPERATURE
C-----
      TSO (1) = TSHSXI (HSO(1),XISO(1),LUN,LOF)
C-----
C      EXCHANGED SENSIBLE HEAT
C-----
      DQEQS (1) = ( HATAWA (TAO(1),WAI,LUN,LOF) - HAI ) * MDAI
C-----
C      EXCHANGED LATENT HEAT
C-----
      DQEQL (1) = QDSI - QDSO (1) - DQEQS (1)
C-----
C      ADDITIONAL OUTPUT CALCULATION
C-----
      WSO (1) = WAPW ( PWT SXI (TSO(1),XISO(1),LUN,LOF), PAMB,LUN,LOF)
C-----
C      FEASABILITY CONDITIONS
C-----
      HASO (1) = HATAWA (TSO(1),WSO(1),LUN,LOF12)
      CD1 = DMEQU(1)*(WSO(1)-WAI)
      CD3 = (HAI-HAO(1))*(DQEQS(1)+DQEQL(1))
      ICOND (1,1) = AINT(SIGN(1.0,CD1))
      ICOND (2,1) = 1
      ICOND (3,1) = AINT(SIGN(1.0,CD3))
      IF (SUPSAT(TAO(1),WAO(1),PAMB,LUN,LOF12)) THEN
        ICOND (4,1) = -1
      ELSE
        ICOND (4,1) = 1
      END IF
C-----
      RETURN
      END

```


APPENDIX B.2

```

C*****
C*
C*
C*      TRNSYS  COMPONENT  47  >  SUMP  <
C*
C*
C*****
C///////////////////////////////////////////////////////////////////
C%                               % %                               %
C%                               % %      UPDATED  VERSION      %
C%  VERSION :  09-20-1984  # 1  % %                               %
C%                               % %                               %
C%                               % %                               %
C///////////////////////////////////////////////////////////////////
C#####
C#
C#
C#      CONFIGURATION
C#
C#      I      I
C#  MDSI1 [ XIN(1) ]      I      I      MDSI2 [ XIN(4) ]
C#  TSI1  [ XIN(2) ]      I      I      TSI2  [ XIN(5) ]
C#  XISI1 [ XIN(3) ]      I      I      XISI2 [ XIN(6) ]
C#
C#      I      I
C#      I      I      MDH20 [ XIN(7) ]      I
C#      I      I      TH20  [ XIN(8) ]      I
C#      I      I
C#      V      V
C#
C#
C#      :-----:
C#      :-----:
C#      :-----:
C#      :-----:
C#
C#      I
C#      V      VDSOL [ PAR(2) ]
C#      I
C#      -----|>I<|-----
C#  MDS03 [ OUT(1) ]      I      I      I
C#  TSO   [ OUT(3) ]      I      ==      I
C#  XISO  [ OUT(4) ]      I      GAMMA [ XIN(9) ]      I
C#      I
C#      V      MDS04 [ OUT(2) ]      I
C#      TS04  [ OUT(3) ]      I
C#      XISO4 [ OUT(4) ]      I
C#
C#      V
C#
C#      WHERE :  MDS03  =  GAMMA * VDSOL * RHOSOL
C#
C#      THIS IS A MODEL OF AN ADIABATIC SOLUTION SUMP
C#      WITH TANK VOLUME
C#

```

APPENDIX B.2

```
C# ADDITIONAL OUTPUTS : #  
C# VOLUME [ OUT(5) ] #  
C# PARAMETERS : #  
C# 1 VDSOL VOLUME FLOW RATE AT OUTLET (CONST) #  
#####  
C$ $$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ $  
C$ UNIT CONVENTION $  
C$ IN THIS COMPONENT THE SI UNITS (MODIFEID) ARE APPLIED : $  
C$ TEMPERATURE ( INTERNAL ) IN K $  
C$ TEMPERATURE IN & OUTPUT IN DEG C $  
C$ SPECIFIC ENTHALPY IN KJ / KG SOL. $  
C$ MASS IN KG $  
C$ MASS FLOW RATE IN KG / HOUR $  
C$ ENERGY FLOW RATE IN KJ / HOUR $  
C$ VOLUME IN M^3 $  
C$ SOLUTION CONCENTRATION IN KG SALT/KG SOL.$  
C$ $  
C$ SUBROUTINE TYPE47 (TIME,XIN,OUT,T,DTDT,PAR,INFO)$  
PARAMETER (NI47=9,N047=5,NP47=1,ND47=3)  
IMPLICIT REAL (M)  
LOGICAL LOF  
DIMENSION XIN (NI47), OUT (N047), T (ND47),  
1 DTDT (ND47), PAR (NP47), INFO (10)  
DATA LUN / 3 / LOF / .TRUE. /  
-----  
C ASSIGNMENT OF THE INLET STATES  
C-----  
MDSI1 = XIN(1)  
TSI1 = XIN(2) + 273.15  
XISI1 = XIN(3)  
MDSI2 = XIN(4)  
TSI2 = XIN(5) + 273.15  
XISI2 = XIN(6)  
MDH2O = XIN(8)  
TH2O = XIN(8) + 273.15  
GAMMA = XIN(9)
```

APPENDIX B.2

```

C      ASSIGNMENT OF THE PARAMETER
C-----
      VDSOL = PAR (1)
      IF (INFO(7).EQ.-1) THEN
        CALL TYPECK (1,INFO,NI47,NP47,ND47)
        INFO (6) = N047
      END IF
C-----
C      GET TANK STATES
C-----
      MSOL = T (1)
      MSALT = T (2)
      QSOL = T (3)
C-----
C      CALCULATE DERIVED STATES
C-----
      HSOL = QSOL / MSOL
      XISOL = MSALT / MSOL
      TSOL = TSHSXI (HSOL,XISOL,LUN,LOF)
      RHOSOL = DSTSXI (TSOL,XISOL,LUN,LOF)
      VOLUME = MSOL / RHOSOL
C-----
C      CALCULATE OUTLET MASS FLOW RATE
C-----
      MDSOL = VDSOL * RHOSOL
C-----
C      CALCULATE DERIVATIVES
C-----
C      1 : SOLUTION MASS BALANCE
C      2 : SALT MASS BALANCE
C      3 : ENERGY BALANCE
C-----
      DTDT (1) = MDSI1 + MDSI2 + MDH20 - MDSOL
      DTDT (2) = MDSI1 * XISI1 + MDSI2 * XISI2
1      - MDSOL * XISOL
      DTDT (3) = MDSI1 * HSTSXI (TSI1,XISI1,LUN,LOF)
1      + MDSI2 * HSTSXI (TSI2,XISI2,LUN,LOF)
2      + MDH20 * 4.19 * (TH20-273.15)
C-----
C      ENTHALPY OF WATER CONSISTENT
C      WITH ENTHALPY OF SOLUTION
C      ( PER DEFINITION ENTHALPY OF
C      PURE WATER AT 0.0 DEGREE C
C      = 0.0 KJ / KG )
C-----
3      - MDSOL * HSTSXI (TSOL,XISOL,LUN,LOF)
C-----
C      LIMIT CONTROL FUNCTION
C-----

```

APPENDIX B.2

```

      IF (GAMMA.GT.1.0)  GAMMA = 1.0
      IF (GAMMA.LT.0.0)  GAMMA = 0.0
      IF (GAMMA.LT.0.0.OR.GAMMA.GT.1.0)
1      WRITE (LUN,*) ' TYPE47 : GAMMA OUT OF BOUNDS :',
2      GAMMA
C-----
C      MASS FLOW RATE 3
C-----
      MDS03 = GAMMA * MDSOL
C-----
C      MASS FLOW RATE 4
C-----
      MDS04 = ( 1.0 - GAMMA ) * MDSOL
C-----
C      ASSIGNMENT OF THE OUTLET STATES
C-----
      OUT (1)  = MDS03
      OUT (2)  = MDS04
      OUT (3)  = TSOL - 273.15
      OUT (4)  = XISOL
      OUT (5)  = VOLUME
C-----
C      EXIT LABEL
C-----
9999  CONTINUE
      RETURN
      END

```

APPENDIX B.3

```

C*****
C*
C*
C*
C*      TRNSYS  COMPONENT  5  >  SENSIBLE  HEAT  EXCHANGER  <
C*
C*
C*
C*****
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C%                                % %                                %
C%                                % %      UPDATED  VERSION      %
C%      VERSION :  09-20-1984  # 1  % %                                %
C%                                % %                                %
C%                                % %                                %
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C#####
C#
C#      THIS IS A MODIFIED VERSION OF THE TRNSYS COMPONENT 5
C#
C#      IT USES VARIABLE SPECIFIC HEAT CAPACITIES
C#      FOR VARIOUS FLUIDS (SALT SOLUTIONS, AIR WATER MIXTURES)
C#
C#      FLUID CODES :
C#          1      SALT - WATER SOLUTIONS
C#          2      AIR - WATER MIXTURES
C#          3      30 % GLYCOL - WATER SOLUTION
C#          4      PURE WATER
C#
C#      PARAMETERS  4 :
C#          1      MODE
C#          2      UA  OR EFFECTIVENESS
C#          3      FLUID CODE HOT SIDE
C#          4      FLUID CODE COLD SIDE
C#
C#      INPUTS  6 :
C#          1      INLET TEMPERATURE      HOT SIDE
C#          2      INLET MASS FLOW RATE    HOT SIDE
C#          3      INLET CONCENTRATION     HOT SIDE
C#          4      INLET TEMPERATURE      COLD SIDE
C#          5      INLET MASS FLOW RATE    COLD SIDE
C#          6      INLET CONCENTRATION     COLD SIDE
C#
C#      OUTPUTS  8 :
C#          1      OUTLET TEMPERATURE      HOT SIDE
C#          2      OUTLET MASS FLOW RATE    HOT SIDE
C#

```

APPENDIX B.3

```

C#          3      OUTLET CONCENTRATION   HOT SIDE      #
C#          4      OUTLET TEMPERATURE    COLD SIDE      #
C#          5      OUTLET MASS FLOW RATE  COLD SIDE      #
C#          6      OUTLET CONCENTRATION   COLD SIDE      #
C#          7      TOTAL HEAT TRANSFER RATE              #
C#          8      HEAT EXCHANGE EFFECTIVENESS           #
C#          9      SPECIFIC HEAT CAPACITY  HOT SIDE      #
C#         10      SPECIFIC HEAT CAPACITY  COLD SIDE      #
C#
C#
C#      MODIFIED BY :      THOMAS K BUSCHULTE            #
C#
C#####
C      SUBROUTINE TYPE5 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
C-----
C  THIS ROUTINE SIMULATES A SENSIBLE HEAT EXCHANGER, GIVING OUTLET
C  TEMPERATURES AND FLOWRATES OF HOT AND COLD STREAMS. MODES 1,2,3,
C  AND 4 SIGNIFY PARALLEL, COUNTERFLOW, CROSS FLOW, AND CONSTANT
C  EFFECTIVENESS MODES RESPECTIVELY. FOR MODE 4, THE HEAT EXCHANGER
C  EFFECTIVENESS MUST BE SUPPLIED AS A PARAMETER.
C
C  UA-OVERALL TRANSFER COEFF PER UNIT TEMP DIFFERENCE, CPH-SPECIFIC
C  HEAT OF HOT SIDE FLUID, CPC-SPECIFIC HEAT OF COLD SIDE FLUID
C  FLWH-HOT SIDE FLOW RATE, TCI-COLD SIDE INLET TEMP, FLWC-COLD SIDE
C  FLOW RATE
C-----
C      DIMENSION XIN(6),PAR(4),OUT(20),INFO(10)
C      IF (INFO(7).GE.0) GO TO 1
C-----
C      FIRST CALL OF SIMULATION
C-----
C      INFO(6)=8
C      INFO(9)=0
C      CALL TYPECK(1,INFO,6,4,0)
C      MODE = IFIX (PAR(1))
C      IFLH = IFIX (PAR(3))
C      IFLC = IFIX (PAR(4))
C      IF ((MODE.LT.1.OR.MODE.GT.4).OR.((MODE.EQ.4).AND.
C$      ((IFLH.LT.1).OR.(IFLH.GT.4)).OR.
C$      ((IFLC.LT.1).OR.(IFLC.GT.4)))) THEN
C      CALL TYPECK(4,INFO,0,0,0)
C      RETURN
C      END IF
C-----
C      SET PARAMETER AND INPUT VARIABLES
C-----
1      CONTINUE
C      MODE = IFIX (PAR(1))
C      UA = PAR(2)

```

APPENDIX B.3

```

IFLH  = IFIX (PAR(3))
IFLC  = IFIX (PAR(4))
IF (MODE.EQ.4) EFF=PAR(2)
THI   = XIN(1)
FLWH  = XIN(2)
CONCFH = XIN(3)
TCI   = XIN(4)
FLWC  = XIN(5)
CONCFC = XIN(6)
-----
C      CALCULATION OF THE CP'S
C-----
      CPH = CPFUN (IFLH,THI-1.0,CONCFH)
      CPC = CPFUN (IFLC,TCI+1.0,CONCFC)
C-----
C      CALCULATE MINIMUM AND MAXIMUM CAPACITY RATES
C-----
      CH=CPH*FLWH
      CC=CPC*FLWC
      CMAX = AMAX1(CC,CH)
      CMIN = AMIN1(CC,CH)
      IF (CMIN .LE. 0.) GO TO 98
      IF (MODE.EQ.4) GO TO 40
C-----
C      MODES 1-3
C-----
      RAT=CMIN/CMAX
      UC=UA/CMIN
      EFF=1.0-EXP(-UC)
      IF((CMIN/CMAX) .LE. 0.01) GO TO 38
      GO TO (10,20,30), MODE
C-----
C      PARALLEL FLOW
C-----
10    EFF=(1.0-EXP(-UC*(1.0+RAT)))/(1.0+RAT)
      GO TO 38
C-----
C      COUNTER FLOW
C-----
20    CHECK=ABS(1.0-RAT)
      IF(CHECK .LT. .01) GO TO 25
      EFF=(1.0-EXP(-UC*(1.0-RAT)))/(1.0-RAT*EXP(-UC*(1.0-RAT)))
      GO TO 38
25    EFF=UC/(UC+1.0)
      GO TO 38
C-----
C      CROSSFLOW, HOT SIDE UNMIXED
C-----
30    GAM=1.0-EXP(-UC*RAT)

```

```

      EFF=1.0-EXP(-GAM/RAT)
      IF(CMAX .EQ. CH) GO TO 38
      GAM=1.0-EXP(-UC)
      EFF=(1.0-EXP(-GAM*RAT))/RAT
38    THO=THI-EFF*(CMIN/CH)*(THI-TCI)
      TCO=EFF*(CMIN/CC)*(THI-TCI)+TCI
      QT=EFF*CMIN*(THI-TCI)
      GO TO 88

C-----
C      MODE 4
C-----
40    QMAX=CMIN*(THI-TCI)
      QT=EFF*QMAX
      THO=THI-QT/CH
      TCO=TCI+QT/CC

C-----
C      SET OUTPUTS --
C      THO-OUTLET TEMP ON HOT SIDE,
C      TCO-OUTLET TEMP ON COLD SIDE, QT-TOTAL
C      INSTANTANEOUS ENERGY TRANSFER ACROSS EXCHANGER,
C      EFF-EFFECTIVENESS
C-----
88    OUT(1)=THO
      OUT(2)=FLWH
      OUT(3)=CONCFH
      OUT(4)=TCO
      OUT(5)=FLWC
      OUT(6)=CONCFC
      OUT(7)=QT
      OUT(8)=EFF
      OUT(9)=CPH
      OUT(10)=CPC
      RETURN

C-----
C      MINIMUM CAPACITY RATE IS .LE. 0.
C-----
98    OUT(1)=THI
      OUT(2)=FLWH
      OUT(3)=CONCFH
      OUT(4)=TCI
      OUT(5)=FLWC
      OUT(6)=CONCFC
      OUT(7)=0.0
      OUT(8)=0.0
      OUT(9)=CPH
      OUT(10)=CPC
      RETURN
      END
C*****

```


APPENDIX B.3

```

C*                                     *
C*      REAL FUNCTION CPFUN                                     *
C*                                     *
C*      CALCULATES THE SPECIFIC HEAT CAPACITY OF A FLUID      *
C*                                     *
C*****
      REAL FUNCTION CPFUN (IOPT,TFL,CONCFL)
C-----
C      IOPT      FLUID CODE ( SEE COMPONENT SUBROUTINE )
C      TFL       TEMPERATURE OF FLUID
C      CONCFL    SOLUTION CONCENTRATION OR
C                HUMIDITY RATIO          RESPECTIVELY
C-----
      LOGICAL LOF
      DATA LOF / .TRUE. / LUN / 11 /
      GOTO (10,20,30,40) IOPT
10    CONTINUE
      CPFUN = (HSTSXI(TFL+274.15,CONCFL,LUN,LOF) -
$        HSTSXI(TFL+272.15,CONCFL,LUN,LOF)) / 2.0
      RETURN
20    CONTINUE
      CPFUN = (HATAWA(TFL+274.15,CONCFL,LUN,LOF) -
$        HATAWA(TFL+272.15,CONCFL,LUN,LOF)) / 2.0
      RETURN
30    CONTINUE
      CPFUN = 3.80
      RETURN
40    CONTINUE
      CPFUN = 4.19
      RETURN
      END

```

APPENDIX B.4

```

C*****
C*
C*
C*      TRNSYS  COMPONENT  42  > AIR MIXER <
C*
C*
C*****
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C%                               % %                               %
C%                               % %      UPDATED  VERSION      %
C%  VERSION :  10-24-1984  # 1  % %                               %
C%                               % %                               %
C%                               % %                               %
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C#####
C#
C#      THIS IS A SIMPLE MODEL OF A FIXED LOAD
C#
C#      INPUTS      6 :
C#
C#          1      M DOT AIR IN 1
C#          2      T AIR IN 1
C#          3      W AIR IN 1
C#          4      M DOT AIR IN 2
C#          5      T AIR IN 2
C#          6      W AIR IN 2
C#
C#      OUTPUTS      3 :
C#
C#          1      M DOT AIR OUT
C#          2      T AIR OUT
C#          3      W AIR OUT
C#
C#      PARAMETERS  0 :
C#
C#      PROGRAMED BY :      THOMAS K BUSCHULTE
C#
C#####
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$
C$      UNIT CONVENTION
C$
C$      IN THIS COMPONENT THE SI UNITS (MODIFIED)
C$      ARE APPLIED :
C$
C$          TEMPERATURE IN & OUTPUT  IN  DEG C
C$          TEMPERATURE INTERNAL      IN  K
C$          SPECIFIC ENTHALPY          IN  KJ/KG DRY AIR
C$          MASS FLOW RATE              IN  KG / HOUR
C$

```

APPENDIX B.4

```

C$          HUMIDITY RATIO          IN   KG WATER /   $
C$                                     KG DRY AIR   $
C$                                     $
C$*****
      SUBROUTINE TYPE42 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
      IMPLICIT REAL (M)
      LOGICAL LOF
      DIMENSION XIN(6), OUT(20), INFO(10)
      COMMON / SIM / TIME0,TFINAL,DELT
      DATA LUN / 3 / LOF / .TRUE. / PAMB / 101325.0 /
C-----
C      FIRST CALL OF SIMULATION
C-----
      IF (INFO(7).EQ.-1) THEN
        INFO(6)=3
        CALL TYPECK (1,INFO,6,0,0)
      END IF
C-----
C      SET INPUT VARIABLES AND PARAMETERS
C-----
1      CONTINUE
      MDAIN1 = XIN (1)
      TAIN1 = XIN (2) + 273.15
      WAIN1 = XIN (3)
      MDAIN2 = XIN (4)
      TAIN2 = XIN (5) + 273.15
      WAIN2 = XIN (6)
C-----
C      CALCULATE OUTLET STATE
C-----
      MDAOUT = MDAIN1 + MDAIN2
      WAOUT = (WAIN1 * MDAIN1 + WAIN2 * MDAIN2) / MDAOUT
      HAIN1 = HATAWA (TAIN1,WAIN1,LUN,.TRUE.)
      HAIN2 = HATAWA (TAIN2,WAIN2,LUN,.TRUE.)
      QDOT = MDAIN1 * HAIN1 + MDAIN2 * HAIN2
      HAOUT = QDOT / MDAOUT
      TAOUT = TAHAWA (HAOUT,WAOUT,LUN,.TRUE.)
C-----
C      ASSIGN OUTPUTS
C-----
      OUT (1) = MDAOUT
      OUT (2) = TAOUT - 273.15
      OUT (3) = WAOUT
      RETURN
C-----
C      END OF TYPE42
C-----
      END

```

APPENDIX B.5

```

C*****
C*
C*
C*      TRNSYS  COMPONENT  41  > FAN <
C*
C*
C*****
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C%                                % %                                %
C%                                % %      UPDATED  VERSION      %
C%  VERSION :  10-27-1984  # 1  % %                                %
C%                                % %                                %
C%                                % %                                %
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C#####
C#
C#      THIS IS A SIMPLE MODEL OF A FIXED LOAD
C#
C#      INPUTS      3 :
C#
C#          1      M DOT AIR IN
C#          2      T AIR IN
C#          3      W AIR IN
C#
C#      OUTPUTS     4 :
C#
C#          1      M DOT AIR OUT
C#          2      T AIR OUT
C#          3      W AIR OUT
C#          4      ELECTRIC POWER CONSUMPT.[KW]
C#
C#      PARAMETERS  1 :
C#
C#          1      ELECTRIC POWER CONSUMPT.[KW]
C#
C#      PROGRAMED BY :      THOMAS K BUSCHULTE
C#
C#####
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$
C$      UNIT CONVENTION
C$
C$      IN THIS COMPONENT THE SI UNITS (MODIFIED)
C$      ARE APPLIED :
C$
C$          TEMPERATURE IN & OUTPUT  IN  DEG C
C$          TEMPERATURE INTERNAL     IN  K
C$          HUMIDITY RATIO             IN  KG WATER /
C$                                     KG DRY AIR
C$

```

APPENDIX B.5

```

C$          SPECIFIC ENTHALPY              IN    KJ/KG DRY AIR    $
C$          MASS FLOW RATE                 IN    KG / HOUR      $
C$          ELECTRIC POWER                  IN    KW = KJ / SEC   $
C$                                           $
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
      SUBROUTINE TYPE41 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
      IMPLICIT REAL (M)
      LOGICAL LOF
      DIMENSION XIN(3), OUT(20), INFO(10)
      COMMON / SIM / TIME0,TFINAL,DELT
      DATA LUN / 3 / LOF / .TRUE. / PAMB / 101325.0 /
C-----
C          FIRST CALL OF SIMULATION
C-----
      IF (INFO(7).EQ.-1) THEN
        INFO(6)=4
        CALL TYPECK (1,INFO,3,1,0)
      END IF
C-----
C          SET INPUT VARIABLES AND PARAMETERS
C-----
1      CONTINUE
      MDAIN = XIN (1)
      TAIN  = XIN (2) + 273.15
      WAIN  = XIN (3)
      PELT  = PAR
C-----
C          CALCULATE OUTLET STATE
C-----
      MDAOUT = MDAIN
      WAOUT  = WAIN
      HAIN   = HATAWA (TAIN,WAIN,LUN,.TRUE.)
      HAOUT  = HAIN + (PELT * 3600.0 / MDAIN)
      TAOUT  = TAHAWA (HAOUT,WAOUT,LUN,.TRUE.)
C-----
C          ASSIGN OUTPUTS
C-----
      OUT (1) = MDAOUT
      OUT (2) = TAOUT - 273.15
      OUT (3) = WAOUT
      OUT (4) = PELT
      RETURN
C-----
C          END OF TYPE41
C-----
      END

```

APPENDIX B.6

```

C*****
C*
C*
C*   TRNSYS COMPONENT 46 > VAPOR COMPRESSION HEAT PUMP <
C*
C*
C*****
C////////////////////////////////////////////////////////////////////
C%                               % %                               %
C%                               % %   UPDATED   VERSION   %
C%  VERSION :   10-30-1984   # 1   % %                               %
C%                               % %                               %
C%                               % %                               %
C////////////////////////////////////////////////////////////////////
C#####
C#
C#   THIS IS A STEADY STATE MODEL
C#   IT USES CURVE FITS OF DATA OF THE HEAT PUMP
C#
C#   MCQUAY TEMPLIFIER TPB-060B
C#
C#   INPUTS       5 :
C#
C#           1      M DOT CONDENSER
C#           2      T IN CONDENSER
C#           3      M DOT EVAPORATOR
C#           4      T IN EVAPORATOR
C#           5      T OUT CONDENSER SET
C#
C#   OUTPUTS      7 :
C#
C#           1      M DOT CONDENSER
C#           2      T OUT CONDENSER
C#           3      M DOT EVAPORATOR
C#           4      T OUT EVAPORATOR
C#           5      ELECTRIC POWER CONSUMPTION
C#           6      COEFFICIENT OF PERFORMANCE
C#           7      Q DOT LOSS
C#
C#   PROGRAMED BY :   THOMAS K BUSCHULTE
C#
C#####
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C$
C$   UNIT CONVENTION
C$
C$   IN THIS COMPONENT THE SI UNITS (MODIFIED)
C$   ARE APPLIED :
C$

```

APPENDIX B.6

```

C$          TEMPERATURE ( FUNCTION ) IN      DEG F      $
C$          TEMPERATURE IN & OUTPUT  IN      DEG C      $
C$          SPECIFIC ENTHALPY          IN      KJ / KG WATER $
C$          MASS FLOW RATE              IN      KG / HOUR  $
C$          ENERGY FLOW RATE          IN      KJ / HOUR  $
C$          POWER                      IN      KW = KJ / SEC $
C$                                     $
C$*****$
      SUBROUTINE TYPE46 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
      EXTERNAL SUB46
      IMPLICIT REAL (M)
      IMPLICIT DOUBLE PRECISION (Z)
      DIMENSION XIN(5), OUT(20), INFO(10),
1          ZX (2), ZFVEC (2), ZWORK (19)
      COMMON / COM46 / MDC, TCI, MDE, TEI, TCOSET
      COMMON / SIM   / TIME0,TFINAL,DELT
      DATA LUNW / 6 / ETA / 0.92 /

C-----
C          FIRST CALL OF SIMULATION
C-----
      IF (INFO(7).EQ.-1) THEN
          INFO(6)=7
          INFO(9)=0
          CALL TYPECK (1,INFO,5,0,0)
          TEO = TEI - 5.0
          OUT (4) = TEO
          CAPFAC = 1.0
          OUT (19) = CAPFAC
      END IF

C-----
C          SET INPUT VARIABLES
C-----
1      CONTINUE
      MDC      = XIN (1)
      TCI      = XIN (2)
      MDE      = XIN (3)
      TEI      = XIN (4)
      TCOSET   = XIN (5)

C-----
C          FIRST GUESSES FOR OUTLET TEMPERATURES
C-----
      TEO      = OUT (4)
      CAPFAC   = OUT (19)
      ZX (1)   = CAPFAC
      ZX (2)   = TEO
      TOL      = 1.0E-03

C-----
C          CALL OF MINPACK ROUTINE HYBRD1
C-----

```

APPENDIX B.6

```

      CALL HYBRD1 (SUB46,2,ZX,ZFVEC,TOL,INFOMP,ZWORK,19)
C-----
C      ASSIGNMENT OF THE OUTLET TEMPERATURES
C      RESULTS OF HYBRD1 CALL
C-----
      CAPFAC = ZX (1)
      TEO    = ZX (2)
      TCO    = TCOSET
C-----
C      TEST FOR ERRONEOUS CONDITIONS
C-----
      IF (INFOMP.NE.1) THEN
        WRITE (LUNW,*) / *** TYPE46 ERROR MESSAGE :/
        WRITE (LUNW,*)
1      / MINPACK ROUTINE HYBRD1 INDICATES/
        WRITE (LUNW,*) / UNSUCCESSFUL CALCULATION !/
        WRITE (LUNW,*) / INFO =',INFO
      END IF
C-----
      IF (TIME.GE.TFINAL) THEN
        DTC = TCO - TCI
        IF (DTC.LT.3.00.OR.DTC.GT.11.00) THEN
          WRITE (LUNW,*) / *** TYPE46 ERROR MESSAGE :/
          WRITE (LUNW,*)
1        / TEMPERATURE RISE AT CONDENSER/
          WRITE (LUNW,*) / OUT OF RANGE !/
          WRITE (LUNW,*) / DELTA T COND =',DTC
        END IF
C-----
        DTE = TEI - TEO
        IF (DTE.LT.5.00.OR.DTC.GT.11.00) THEN
          WRITE (LUNW,*) / *** TYPE46 ERROR MESSAGE :/
          WRITE (LUNW,*)
1        / TEMPERATURE RISE AT EVAPORATOR/
          WRITE (LUNW,*) / OUT OF RANGE !/
          WRITE (LUNW,*) / DELTA T EVAP =',DTE
        END IF
C-----
C      80 DEG F < T COND OUT < 165 DEG F
C-----
      IF (TCO.LT.26.7.OR.TCO.GT.73.9) THEN
        WRITE (LUNW,*) / *** TYPE46 ERROR MESSAGE :/
        WRITE (LUNW,*)
1      / OUTLET TEMPERATURE AT CONDENSER/
        WRITE (LUNW,*) / OUT OF RANGE !/
        WRITE (LUNW,*) / T COND OUT =',TCO
      END IF
C-----
C      40 DEG F < T EVAP OUT < 110 DEG F

```



```

C-----
      IF (TEO.LT.4.4.OR.TEO.GT.43.3) THEN
        WRITE (LUNW,*) ' *** TYPE46 ERROR MESSAGE : '
        WRITE (LUNW,*)
      1    '      OUTLET TEMPERATURE AT EVAPORATOR'
        WRITE (LUNW,*) '      OUT OF RANGE !'
        WRITE (LUNW,*) '      T EVAP OUT =',TEO
      END IF
    END IF
C-----
C      CALCULATE COP, PELT AND QDLOSS
C-----
      COP      = FCT46 (TDEGF(TCO),TDEGF(TEO),1)
      PELT     = CAPFAC * FCT46 (TDEGF(TCO),TDEGF(TEO),2)
      QDLOSS   = PELT * (1.0-ETA)
C-----
C      ASSIGN OUTPUTS
C-----
      OUT (1) = MDC
      OUT (2) = TCO
      OUT (3) = MDE
      OUT (4) = TEO
      OUT (5) = PELT
      OUT (6) = COP
      OUT (7) = QDLOSS
      OUT (19)= CAPFAC
      RETURN
C-----
C      END OF TYPE46
C-----
      END
C*****
C*
C*      SUBROUTINE SUB46
C*
C*      IS CALLED BY THE MINPACK ROUTINE HYBRD1
C*
C*****
      SUBROUTINE SUB46 (NDUM,ZX,ZFVEC,IFLAG)
      IMPLICIT REAL (M)
      DOUBLE PRECISION ZX(2), ZFVEC(2)
      COMMON / COM46 / MDC, TCI, MDE, TEI, TCOSET
      DATA CPH20 / 4.19 / ETA / 0.92 /
C-----
      TCO      = TCOSET
      CAPFAC   = ZX (1)
      TEO      = ZX (2)
      TCOF     = TDEGF (TCO)
      TEOF     = TDEGF (TEO)

```

```

      PELT      = FCT46 (TCOF,TEOF,2)
C-----
C      CONVERT PELT FROM KJ/S TO KJ/HOUR
C-----
      PELT      = PELT * 3600.
      COP       = FCT46 (TCOF,TEOF,1)
      QDE       = CPH20 * MDE * (TEI - TEO)
      QDC       = CPH20 * MDC * (TCO - TCI)
      PELBAR    = ETA * PELT * CAPFAC
      ZFVEC (1) = QDE - QDC + PELBAR
      ZFVEC (2) = QDC - ( PELT * COP * CAPFAC )
      RETURN
C-----
C      END OF SUB46
C-----
      END
C*****
C*                                     *
C*      REAL FUNCTION FCT46                                     *
C*                                     *
C*      CALCULATES :                                             *
C*                                     *
C*      FCT46 = C1 + C2 * TCO + C3 * TCO2 + C4 * TEO           *
C*                                     *
C*      + C5 * TEO2 + C6 * TCO * TEO                           *
C*                                     *
C*      TCO, TEO IN DEGREE FAHRENHEIT !                          *
C*                                     *
C*      INFO = 1 : CALCULATE C O P                               *
C*      2 : CALCULATE                                           *
C*      ELECTRIC POWER CONSUMPTION                               *
C*                                     *
C*****
      REAL FUNCTION FCT46 (TCO,TEO,IOPT)
      DIMENSION COEFF (6,2)
C-----
C      COEFFICIENTS FOR THE COP
C-----
      DATA COEFF / 2.789E-01 , -3.341E-03 , 8.025E-05 ,
1      1.30567E-01 , 1.58E-06 , -7.0858E-04 ,
C-----
C      COEFFICIENTS FOR THE ELECTRIC POWER CONSUMPTION
C      PELT IN KW !
C-----
      2      4.9140E+01 , -2.2372E-01 , 2.095E-04 ,
      3      -2.3797E-01 , 2.176E-04 , 4.9423E-03 /
C-----
      FCT46 = COEFF (1,IOPT)
1      + ((TCO*COEFF(3,IOPT))+COEFF(2,IOPT))*TCO

```

APPENDIX B.6

```

      2      + ( (TE0*COEFF(5,IOPT))+COEFF(4,IOPT))*TE0
      3      + COEFF(6,IOPT)*TC0*TE0
      RETURN
C-----
C      END OF FCT46
C-----
      END
C*****
C*
C*      REAL FUNCTION TDEGF
C*
C*      CALCULATES T IN DEG F AS A FUNCTION OF T IN DEG C *
C*
C*****
      REAL FUNCTION TDEGF (TDEGC)
C-----
      TDEGF = ( TDEGC * 1.8 ) + 32.0
      RETURN
C-----
C      END OF TDEGF
C-----
      END

```

APPENDIX B.6

```

C-----
      IF (TEO.LT.4.4.OR.TEO.GT.43.3) THEN
        WRITE (LUNW,*) ' *** TYPE46 ERROR MESSAGE : '
        WRITE (LUNW,*)
1       '      OUTLET TEMPERATURE AT EVAPORATOR'
        WRITE (LUNW,*) '      OUT OF RANGE ! '
        WRITE (LUNW,*) '      T EVAP OUT =',TEO
      END IF
    END IF
C-----
C      CALCULATE COP, PELT AND QDLOSS
C-----
      COP      = FCT46 (TDEGF(TCO),TDEGF(TEO),1)
      PELT     = CAPFAC * FCT46 (TDEGF(TCO),TDEGF(TEO),2)
      QDLOSS   = PELT * (1.0-ETA)
C-----
C      ASSIGN OUTPUTS
C-----
      OUT (1) = MDC
      OUT (2) = TCO
      OUT (3) = MDE
      OUT (4) = TEO
      OUT (5) = PELT
      OUT (6) = COP
      OUT (7) = QDLOSS
      OUT (19)= CAPFAC
      RETURN
C-----
C      END OF TYPE46
C-----
      END
C*****
C*
C*      SUBROUTINE SUB46
C*
C*      IS CALLED BY THE MINPACK ROUTINE HYBRD1
C*
C*****
      SUBROUTINE SUB46 (NDUM,ZX,ZFVEC,IFLAG)
      IMPLICIT REAL (M)
      DOUBLE PRECISION ZX(2), ZFVEC(2)
      COMMON / COM46 / MDC, TCI, MDE, TEI, TCOSET
      DATA CPH20 / 4.19 / ETA / 0.92 /
C-----
      TCO      = TCOSET
      CAPFAC   = ZX (1)
      TEO      = ZX (2)
      TCOF     = TDEGF (TCO)
      TEOF     = TDEGF (TEO)

```

```

      PELT      = FCT46 (TCOF,TEOF,2)
C-----
C      CONVERT PELT FROM KJ/S TO KJ/HOUR
C-----
      PELT      = PELT * 3600.
      COP       = FCT46 (TCOF,TEOF,1)
      QDE       = CPH20 * MDE * (TEI - TEO)
      QDC       = CPH20 * MDC * (TCO - TCI)
      PELBAR    = ETA * PELT * CAPFAC
      ZFVEC (1) = QDE - QDC + PELBAR
      ZFVEC (2) = QDC - ( PELT * COP * CAPFAC )
      RETURN
C-----
C      END OF SUB46
C-----
      END
C*****
C*
C*      REAL FUNCTION FCT46
C*
C*      CALCULATES :
C*
C*      FCT46 = C1 + C2 * TCO + C3 * TCO2 + C4 * TEO
C*
C*      + C5 * TEO2 + C6 * TCO * TEO
C*
C*      TCO, TEO IN DEGREE FAHRENHEIT !
C*
C*      INFO = 1 : CALCULATE C O P
C*      2 : CALCULATE
C*      ELECTRIC POWER CONSUMPTION
C*
C*****
      REAL FUNCTION FCT46 (TCO,TEO,IOPT)
      DIMENSION COEFF (6,2)
C-----
C      COEFFICIENTS FOR THE COP
C-----
      DATA COEFF / 2.789E-01 , -3.341E-03 , 8.025E-05 ,
1          1.30567E-01 , 1.58E-06 , -7.0858E-04 ,
C-----
C      COEFFICIENTS FOR THE ELECTRIC POWER CONSUMPTION
C      PELT IN KW !
C-----
      2          4.9140E+01 , -2.2372E-01 , 2.095E-04 ,
      3          -2.3797E-01 , 2.176E-04 , 4.9423E-03 /
C-----
      FCT46 = COEFF (1,IOPT)
1          + ((TCO*COEFF(3,IOPT))+COEFF(2,IOPT))*TCO

```

APPENDIX B.6

```

      2      + ((TEO*COEFF(5,IOPT))+COEFF(4,IOPT))*TEO
      3      + COEFF(6,IOPT)*TCO*TEO
      RETURN
C-----
C      END OF FCT46
C-----
      END
C*****
C*                                          *
C*      REAL FUNCTION TDEGF                      *
C*                                          *
C*      CALCULATES T IN DEG F AS A FUNCTION OF T IN DEG C *
C*                                          *
C*****
      REAL FUNCTION TDEGF (TDEGC)
C-----
      TDEGF = ( TDEGC * 1.8 ) + 32.0
      RETURN
C-----
C      END OF TDEGF
C-----
      END

```

APPENDIX B.7

```

C*****
C*
C*
C*      TRNSYS  COMPONENT  44  > VAPOR COMPRESSION CHILLER <
C*
C*
C*****
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C%                                % %                                %
C%                                % %      UPDATED VERSION          %
C%      VERSION :  10-16-1984  # 1  % %                                %
C%                                % %                                %
C%                                % %                                %
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C#####
C#
C#      THIS IS A MODEL OF A VAPOR COMPRESSION CHILLER
C#      BASED ON AN CARNOT EFFICIENCY APPROACH
C#
C#      IDEAL          T EVAP
C#      COP           =  -----
C#      EVAP          T COND - T EVAP
C#
C#      IDEAL
C#      COP           =  EFF *  COP
C#      EVAP          EVAP
C#
C#      INPUTS      3 :
C#
C#          1      M DOT EVAP IN
C#          2      T EVAP IN
C#          3      T EVAP OUT SET
C#
C#      OUTPUTS     4 :
C#
C#          1      M DOT EVAP OUT
C#          2      T EVAP OUT
C#          3      P ELECTRIC
C#          4      COP
C#
C#      PARAMETERS  2 :
C#
C#          1      T COND OUT SET
C#          2      EFFICIENCY
C#
C#      PROGRAMED BY :  THOMAS K BUSCHULTE
C#
C#####

```

APPENDIX B.7

```
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$  
C$UNIT CONVENTION$  
C$IN THIS COMPONENT THE SI UNITS (MODIFIED) ARE APPLIED : $  
C$TEMPERATURE IN & OUTPUT IN DEG C$  
C$TEMPERATURE INTERNAL IN K$  
C$SPECIFIC ENTHALPY IN KJ / KG DRY AIR$  
C$MASS FLOW RATE IN KG / HOUR$  
C$POWER IN KW = KJ / SEC$  
C$  
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$  
SUBROUTINE TYPE44 (TIME,XIN,OUT,T,DTDT,PAR,INFO)  
IMPLICIT REAL (M)  
LOGICAL LOF  
DIMENSION XIN(3), OUT(20), PAR (2), INFO(10)  
COMMON / SIM / TIME0,TFINAL,DELT  
DATA LUN / 6 / LOF / .TRUE. / PAMB / 101325.0 /  
  
-----  
C FIRST CALL OF SIMULATION  
-----  
IF (INFO(7).EQ.-1) THEN  
INFO(6)=4  
CALL TYPECK (1,INFO,3,2,0)  
END IF  
  
-----  
C SET INPUT VARIABLES AND PARAMETERS  
-----  
1 CONTINUE  
MDEVIN = XIN (1)  
TEVIN = XIN (2) + 273.15  
TEVSET = XIN (3) + 273.15  
TCOSET = PAR (1) + 273.15  
EFF = PAR (2)  
  
-----  
C CALCULATE OUTLET STATE  
-----  
MDEVOU = MDEVIN  
TEVOUT = TEVSET  
TC = TCOSET + 5.0  
TE = TEVSET - 5.0  
  
-----  
C COP BASED ON EVAPORATION ENERGY !  
C IF EFF = 0.6, COP (55 DEG F) = 5.2 (RICHMOND ASSUMPTION)  
-----  
COPID = TE / (TC - TE)  
COP = EFF * COPID  
WELT = MDEVOU * 4.19 * (TEVIN-TEVOUT) / COP
```


APPENDIX B.7

```
      PELT  = WELT / 3600.0
C-----
C      ASSIGN OUTPUTS
C-----
      OUT (1) = MDEV0U
      OUT (2) = TEVOUT - 273.15
      OUT (3) = PELT
      OUT (4) = COP
      RETURN
C-----
C      END OF TYPE44
C-----
      END
```

APPENDIX B.8

```

C*****
C*
C*
C*      TRNSYS  COMPONENT  45  >  PROPORTIONAL - INTEGRAL CONTROLLER <
C*
C*
C*****
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C%                                     % %                                     %
C%                                     % %      UPDATED  VERSION          %
C%      VERSION :  10-28-1984  # 1  % %                                     %
C%                                     % %                                     %
C%                                     % %                                     %
C%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
C#####
C#
C#      "VELOCITY FORM" OF THE PI CONTROLLER :
C#
C#                                     DELTA T
C#      U  - U    = K * ( ( EPS  - EPS  ) + ----- * EPS  )
C#        K    K-1      K      K-1      2 * TAUI      K
C#
C#      EPS  = Y  - Y
C#        K    K    SET
C#
C#      U  = PAR (3)
C#      0
C#
C#      Y  = FUNCTION ( U  ) (BY OTHER TRNSYS COMPONENTS)
C#        K              K-1
C#
C#      INPUTS      2 :
C#
C#          1      YK      SYSTEM STATE AT TIME STEP K
C#          2      Y SET  SYSTEM STATE SET POINT
C#
C#      OUTPUTS     1 :
C#
C#          USER  ----> 1      UKM1I CONTROLLER ACTION AT TIME STEP K-1#
C#                               LIMITED BY UMIN AND UMAX
C#          STORAGE  2      UKM1 CONTROLLER ACTION AT TIME STEP K-1 #
C#          .        3      EPSKM1 CONTROLLER ERROR AT TIME STEP K-1 #
C#          .
C#          .        4      UKLI CONTROLLER ACTION AT TIME STEP K
C#                               LIMITED BY UMIN AND UMAX
C#          .        5      UK  CONTROLLER ACTION AT TIME STEP K
C#          .        6      EPSK CONTROLLER ERROR AT TIME STEP K
C#          .        7      Y SET SYSTEM STATE SET POINT
C#
C#      PARAMETERS  5 :

```

APPENDIX B.8

```

C#                                     #
C#          1      K      (CONTROLLER GAIN)      #
C#          2      TAU1 (INTEGRATION TIME CONSTANT) [HOURS] #
C#          3      U0      (INITIAL CONTROLLER ACTION)      #
C#          4      UMIN (MINIMUM CONTROLLER ACTION)      #
C#          5      UMAX (MAXIMUM CONTROLLER ACTION)      #
C#                                     #
C#          PROGRAMED BY :      THOMAS K BUSCHULTE      #
C#                                     #
C#####
C      SUBROUTINE TYPE45 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
C      IMPLICIT REAL (K)
C      DIMENSION XIN(2), OUT(20), PAR(5), INFO(10)
C      COMMON      / SIM / TIME0, TFINAL, DELT
C-----
C      SET INPUT VARIABLES AND PARAMETERS
C-----
1      CONTINUE
      YK      = XIN (1)
      YSET     = XIN (2)
      K       = PAR (1)
      TAU1    = PAR (2)
      U0      = PAR (3)
      UMIN    = PAR (4)
      UMAX    = PAR (5)
C-----
C      FIRST CALL OF SIMULATION
C-----
      IF (INFO(7).EQ.-1) THEN
C      INFO(6) = 1
      INFO(6) = 7
      CALL TYPECK (1,INFO,2,5,0)
      UK      = U0
      UKLI    = UK
      UKM1    = U0
      UKM1LI  = UKM1
      EPSK    = 0.0
      EPSKM1  = 0.0
      IF (UK.GT.UMAX) THEN
          UKLI = UMAX
          UKLIM1 = UMAX
      END IF
      IF (UK.LT.UMIN) THEN
          UKLI = UMIN
          UKLIM1 = UMIN
      END IF
      ELSE
C-----
C      REASSIGN THE VALUES OF TIME STEP K - 1

```

```

C-----
      IF (INFO(7).EQ.0) THEN
        OUT (1) = OUT (4)
        OUT (2) = OUT (5)
        OUT (3) = OUT (6)
      END IF
C-----
C      GET VALUES OF LAST TIME STEP
C-----
      UKM1LI = OUT (1)
      UKM1    = OUT (2)
      EPSKM1  = OUT (3)
C-----
C      CONTROL ERROR
C-----
      EPSK = YK - YSET
C-----
C      CONTROLLER ACTION
C-----
      UK = UKM1 + K * ((EPSK - EPSKM1) + (DELT * EPSK)
1          / (2.0 * TAU1))
C-----
C      LIMIT CONTROLLER ACTION
C-----
      UKLI = UK
      IF (UK.GT.UMAX) UKLI = UMAX
      IF (UK.LT.UMIN) UKLI = UMIN
      END IF
C-----
C      ASSIGN OUTPUTS
C-----
      OUT (1) = UKM1LI
      OUT (2) = UKM1
      OUT (3) = EPSKM1
      OUT (4) = UKLI
      OUT (5) = UK
      OUT (6) = EPSK
      OUT (7) = YSET
      RETURN
C-----
C      END OF TYPE45
C-----
      END

```

APPENDIX B.9

```

C*****
C*
C*
C*      TRNSYS  COMPONENT  43  >  LOAD  <
C*
C*
C*****
C*****
C*****
C%                               % %                               %
C%                               % %                               %
C%      VERSION :  10-30-1984  # 1  % %                               %
C%                               % %                               %
C%                               % %                               %
C*****
C#####
C#
C#      THIS IS A SIMPLE MODEL OF A FIXED LOAD
C#
C#      INPUTS      3 :
C#
C#          1      M DOT AIR IN
C#          2      T AIR IN
C#          3      W AIR IN
C#
C#      OUTPUTS     5 :
C#
C#          1      M DOT AIR OUT
C#          2      T AIR OUT
C#          3      W AIR OUT
C#          4      SENSIBLE LOAD
C#          5      LATENT LOAD
C#
C#      PARAMETERS  2 :
C#
C#          1      T OUT SET
C#          2      W SET OUT
C#
C#      PROGRAMED BY :      THOMAS K BUSCHULTE
C#
C#####
C$
C$      UNIT CONVENTION
C$
C$      IN THIS COMPONENT THE SI UNITS (MODIFIED) ARE APPLIED :
C$
C$      TEMPERATURE IN & OUTPUT  IN  DEG C
C$      TEMPERATURE INTERNAL      IN  K
C$      SPECIFIC ENTHALPY          IN  KJ / KG DRY AIR
C$

```

```

C$          MASS FLOW RATE          IN    KG / HOUR          $
C$                                                                    $
C*****
      SUBROUTINE TYPE43 (TIME,XIN,OUT,T,DTDT,PAR,INFO)
      IMPLICIT REAL (M)
      LOGICAL LOF
      DIMENSION XIN(3), OUT(20), PAR (2), INFO(10)
      COMMON / SIM    / TIME0,TFINAL,DELT
      DATA      LUN / 6 / LOF / .TRUE. / PAMB / 101325.0 /
C-----
C      FIRST CALL OF SIMULATION
C-----
      IF (INFO(7).EQ.-1) THEN
        INFO(6)=5
        CALL TYPECK (1,INFO,3,2,0)
      END IF
C-----
C      SET INPUT VARIABLES AND PARAMETERS
C-----
1      CONTINUE
      MDAIN = XIN (1)
      TAIN  = XIN (2) + 273.15
      WAIN  = XIN (3)
      TAOUT = PAR (1) + 273.15
      WAOUT = PAR (2)
CCC      WRITE (3,*) ' LOAD : MDAIN =',MDAIN,' TAIN =',TAIN
CCC      WRITE (3,*) ' LOAD : WAIN =',WAIN
C-----
C      CALCULATE OUTLET STATE
C-----
      MDAOUT = MDAIN
      HAIN   = HATAWA (TAIN,WAIN,3,.TRUE.)
      HAIWAO = HATAWA (TAIN,WAOUT,3,.TRUE.)
      HAOUT  = HATAWA (TAOUT,WAOUT,3,.TRUE.)
      QDLAT  = MDAIN * (HAIWAO - HAIN) / 3600.0
      QDSENS = MDAIN * (HAOUT - HAIWAO) / 3600.0
C-----
C      ASSIGN OUTPUTS
C-----
      OUT (1) = MDAOUT
      OUT (2) = TAOUT - 273.15
      OUT (3) = WAOUT
      OUT (4) = QDSENS
      OUT (5) = QDLAT
      RETURN
C-----
C      END OF TYPE43
C-----
      END

```

```

***** 12 - 17 - 84 *****
*
*   TRNSYS DECK  > HVAC SYSTEM, SCIENCE MUSEUM OF VIRGINIA <
*
*   OUTPUT COMPONENTS ARE NOT INCLUDED
*
*****

```

```

*
*   WIDTH 72

```

```

*
*   CONSTANTS 18

```

```

    TST   = 0.0
    TEND  = 100.0
    DTI   = 1 / 60
    TRON  = TEN
    TOFF  = TEN + DTI
    PRON  = TST
    DTPR  = 50.0
    DPLO  = 120 * DTI
    MDC   = 23.76E+03
    MDH   = 15.48E+03
    TCI   = 12.78
    THI   = 60.0
    UAH   = 40430.
    UAC   = 81600.
    MDG   = 7056.
    EHR   = 0.85
    TAI   = 21.10
    WAI   = 0.0143

```

```

*
*   TST      SIMULATION START TIME
*   TEND     SIMULATION END TIME
*   DTI      TIME STEP
*   TRON     TRACE START TIME
*   TOFF     TRACE END TIME
*   PRON     PRINTER ON TIME
*   DTPR     PRINT TIME STEP
*   DPLO     PLOT TIME STEP
*   MDC      MASS FLOW RATE COOLER , LIQUID ( H2O )
*   TCI      TEMPERATURE OF THE COOLER, LIQUID AT INLET ( H2O )
*   MDH      MASS FLOW RATE HEATER, LIQUID ( H2O )
*   THI      TEMPERATURE OF THE HEATER, LIQUID AT INLET ( H2O )
*   UAH      UA OF SOLUTION HEATER
*   UAC      UA OF SOLUTION COOLER
*   MDG      MASS FLOW RATE OF GLYCOL IN HEAT RECUPERATION LOOP
*   EHR      EFFECIENCY OF HEAT RECOVERY LOOP HX
*   TAI      AIR INTAKE TEMPERATURE
*   WAI      AIR INTAKE HUMIDITY RATIO
*

```

```

*
* SIMULATION TST TEND DTI
*
* TOLERANCES 0.0000000001 0.00001
* INTEGRATION CONVERGENCE
*
* LIMITS 500 10 499
* MAX ITER MAX WARN TRACE START
*
*
* UNIT DIRECTORY
*
* UNIT TYPE COMMENT
*
* DESORBER CYCLE (REGENERATOR)
*
* 10 48 SPRAY CHAMBER REGENERATOR, DESORBER
* 11 47 SUMP OF THE REGENERATOR
* 12 5 HEATEXCHANGER SOLUTION HEATER
* 14 11 FLOW DIVERTER VALVE FOR HEATING WATER
* 15 11 TEE-PIECE FOR HEATING WATER
* 16 5 HEATEXCHANGER EXHAUST AIR HEATER
* 17 5 HEATEXCHANGER EXHAUST AIR COOLER
*
* ABSORBER CYCLE (CONDITIONER)
*
* 20 48 SPRAY CHAMBER CONDITIONER, ABSORBER
* 21 47 SUMP OF THE CONDITIONER
* 22 5 HEATEXCHANGER SOLUTION COOLER
* 24 11 FLOW DIVERTER VALVE FOR COOLING WATER
* 25 11 TEE-PIECE FOR COOLING WATER
* 26 45 PI CONTROLLER CONTR. SUMP DIVERTER VALVE
* 27 5 HEATEXCHANGER SOLUTION INTERCHANGER
*
* HEAT PUMP, CHILLER AND AUXILLIARY EQUIPMENT
*
* 30 46 HEAT PUMP
* 32 45 PI CONTROLLER CONTR. VALVE 34
* 33 44 CHILLER (AT 55 DEG F)
* 34 11 FLOW DIVERTER VALVE HP / CHILLER DIVERSION
* 35 11 TEE-PIECE HP / CHILLER TEE-PIECE
* EVAPORATOR SIDE
*
* FANS, AIRMIXERS
*
* 44 41 RETURN FAN
* 45 42 SUPPLY AIR MIXER
* 46 41 SUPPLY FAN
* 47 42 CONDITIONER AIR MIXER

```



```

*      48      41  CONDITIONER FAN
*      49      41  REGENERATOR FAN
*      50      43  LOAD

```

```

*

```

```

*-----

```

```

*

```

```

*      DESORBER CYCLE

```

```

*

```

```

*-----

```

```

*

```

```

UNIT 16 TYPE 5 EXHAUST AIR HEATER

```

```

PARAMETERS 4

```

```

4          EHR          3          2
MODE      EFFECT      FLUID HOT  FLUID COLD

```

```

*

```

```

INPUTS 6

```

```

17,4  0,0  0,0  49,2  49,1  49,3

```

```

*

```

```

T HOT IN      MD HOT IN      CONC HOT IN  T COLD IN      MD COLD IN

```

```

*

```

```

CONC COLD IN

```

```

37.2 ,      MDG ,      0.30 ,      27.6 ,      9972.0 ,
0.0093

```

```

*

```

```

*-----

```

```

*

```

```

UNIT 10 TYPE 48 SPRAYCHAMBER ( REGENERATOR )

```

```

PARAMETERS 1

```

```

1

```

```

*

```

```

MODE

```

```

INPUTS 6

```

```

12,5  12,4  12,6  16,5  16,4  16,6

```

```

*

```

```

MDSI      TSI      XISI      MDAI      TAI

```

```

*

```

```

WAI

```

```

32.04E+03  44.72 ,      0.305 ,      9972.0 ,      35.8 ,
0.0093

```

```

*

```

```

*-----

```

```

*

```

```

UNIT 11 TYPE 47 SUMP OF REGENERATOR (10)

```

```

PARAMETERS 1

```

```

27.2

```

```

*

```

```

VOLUME FLOW RATE OUT

```

```

*

```

```

INPUTS 9

```

```

10,1  10,2  10,3  27,5  27,4  27,6  0,0  0,0  0,0

```

```

*

```

```

MDSOL1IN  TSOL1IN  XISOL1IN  MDSOL2IN  TSOL2IN

```

```

*

```

```

XISOL2IN  MDH20IN  TH20IN  GAMMA

```

```

32.04E+03  40.4 ,      0.306 ,      0.6 ,      31.1
0.285 ,      0.0 ,      20.0 ,      0.0618

```

```

*

```

```

DERIVATIVES 3

```

```

2356.0 ,      718.6 ,      -145.836E+03
*  M SOL INIT  M SALT INIT  Q SOL INIT
*
*-----
*
UNIT 14 TYPE 11 FLOW DIVERTER, VALVE FOR HEATING WATER
PARAMETERS 1
2
*  MODE
INPUTS 3
30,2 30,1 0,0
*  T IN      M DOT IN      GAMMA
THI ,      MDH ,      1.0
*
*-----
*
UNIT 12 TYPE 5 HEATEXCHANGER ( SOLUTION HEATER )
PARAMETERS 4
2          UAH          4          1
*  MODE      UA      FLUID HOT  FLUID COLD
INPUTS 6
14,3 14,4 0,0 11,3 11,2 11,4
*  T HOT IN  MD HOT IN  CONC HOT IN  T COLD IN  MD COLD IN
*  CONC COLD IN
THI ,      MDH ,      0.0 ,      39.1 ,      32.04E+03
0.305
*
*-----
*
UNIT 15 TYPE 11 TEE - PIECE, HEATING WATER
PARAMETERS 1
1
*  MODE
INPUTS 4
14,1 14,2 12,1 12,2
*  T IN 1    M DOT IN 1    T IN 2      M DOT IN 2
60.0 ,      0.0 ,      51.67 ,      MDH
*
*-----
*
UNIT 17 TYPE 5 EXHAUST AIR COOLER
PARAMETERS 4
4          EHR          2          3
*  MODE      EFFECT      FLUID HOT  FLUID COLD
INPUTS 6
10,5 10,4 10,6 16,1 16,2 16,3
*  T HOT IN  MD HOT IN  CONC HOT IN  T COLD IN  MD COLD IN
*  CONC COLD IN
43.9 ,      9972.0 ,      0.023 ,      34.3 ,      MDG ,

```

0.30

```

*
*-----
*
*      ABSORBER CYCLE
*-----
*
*      UNIT 20  TYPE 48  SPRAYCHAMBER ( CONDITIONER )
*      PARAMETERS 1
*      2
*      MODE
*      INPUTS 6
*      22,2  22,1  22,3  48,1  48,2  48,3
*      MDSI      TSI      XISI      MDAI      TAI
*      WAI
*      32.27E+03 ,  18.0 ,      0.285 ,      24.12E+03 ,  25.0 ,
*      0.0118
*
*-----
*
*      UNIT 21  TYPE 47  SUMP OF CONDITIONER (20)
*      PARAMETERS 1
*      27.2
*      VOLUME FLOW RATE OUT
*
*      INPUTS 9
*      20,1  20,2  20,3  27,2  27,1  27,3  0,0  0,0  26,1
*      MDSOL1IN  TSOL1IN  XISOL1IN  MDSOL2IN  TSOL2IN
*      XISOL2IN  MDH20IN  TH20IN  GAMMA
*      32.5E+03  22.9 ,      0.284 ,      1980.0 ,      30.9 ,
*      0.305 ,      0.0 ,      20.0 ,      0.063
*
*      DERIVATIVES 3
*      2367.0 ,      675.0 ,      -243.500E+03
*      M SOL INIT  M SALT INIT  Q SOL INIT
*
*-----
*
*      UNIT 24  TYPE 11  FLOW DIVERTER, VALVE FOR COOLING WATER
*      PARAMETERS 1
*      2
*      MODE
*      INPUTS 3
*      35,1  35,2  0,0
*      T IN      M DOT IN      GAMMA
*      TCI ,      MDC ,      1.0
*
*-----

```

```

*
UNIT 22 TYPE 5 HEATEXCHANGER ( SOLUTION COOLER )
PARAMETERS 4
2          UAC          1          4
*  MODE      UA          FLUID HOT   FLUID COLD
  INPUTS 6
21,3 21,2 21,4 24,3 24,4 0,0
*  T HOT IN   MD HOT IN   CONC HOT IN T COLD IN   MD COLD IN
*  CONC COLD IN
23.6 ,      32.22E+03 , 0.285 ,      TCI ,      MDC
0.0

```

```

*
-----
*

```

```

UNIT 25 TYPE 11 TEE - PIECE, HEATING WATER
PARAMETERS 1
1
*  MODE
  INPUTS 4
24,1 24,2 22,4 22,5
*  T IN 1      M DOT IN 1   T IN 2      M DOT IN 2
*  TCI ,      0.0 ,      18.33 ,      MDC

```

```

*
-----
*

```

```

UNIT 26 TYPE 45 PI CONTROLLER CONTROLS SOLUTION DIVERTER
                                VALVE AS A FUNCTION OF
                                THE SUMP LEVEL (= VOLUME)
PARAMETERS 5
0.10      0.5      0.063      0.0      0.1
*  K - GAIN  TAU I   U 0      U MIN    U MAX
  INPUTS 2
21,5 0,0
*  SUMP VOL.  SETPOINT
2.0 ,      2.0

```

```

*
-----
*

```

```

UNIT 27 TYPE 5 SOLUTION HEATEXCHANGER
PARAMETERS 4
2          5688.0      1          1
*  MODE      UA          FLUID HOT   FLUID COLD
  INPUTS 6
11,3 11,1 11,4 21,3 21,1 21,4
*  T HOT IN   MD HOT IN   CONC HOT IN T COLD IN   MD COLD IN
*  CONC COLD IN
39.1 ,      1980.0 ,      0.285 ,      23.6 ,      2160.0 ,
0.305

```

```

*

```

```

*-----
*
*      HEAT PUMP AND CHILLER, AUXILLIARY EQUIPMENT
*
*-----
*
*      UNIT 32  TYPE 45  PI CONTROLLER  CONTROLS VALVE 34
*                                     AS A FUNCTION OF
*                                     THE HP EVAPORATOR LEAVING TEMP
*
*      PARAMETERS 5
*      0.02      0.01      0.318      0.0      1.0
*      K - GAIN  TAU I      U 0      U MIN      U MAX
*      INPUTS 2
*      30,4  0,0
*      T COND OUT  SETPOINT
*      TCI ,      TCI
*
*-----
*
*      UNIT 34  TYPE 11  FLOW DIVERTER, HP EVAPORATOR / CHILLER DIVERTER
*      PARAMETERS 1
*      2
*      MODE
*      INPUTS 3
*      25,1  0,0  32,1
*      T IN      M DOT IN      GAMMA
*      18.33 ,      MDC ,      0.318
*
*-----
*
*      UNIT 30  TYPE 46  HEAT PUMP
*      PARAMETERS 0
*      INPUTS 5
*      0,0  15,1  34,2  34,1  0,0
*      M DOT COND  T IN COND  M DOT EVAP  T IN EVAP  T OUT COND SET
*      MDH ,      51.67 ,      16195.0 ,      18.33 ,      THI
*
*-----
*
*      UNIT 33  TYPE 44  CHILLER
*      PARAMETERS 2
*      35.0      0.6
*      T CO OUT SET EFFECT.
*      INPUTS 3
*      34,4  34,3  0,0
*      M DOT EV IN  T EV IN      T EV OUT SET
*      7570.0 ,      18.33 ,      TCI
*
*-----

```

```

*
UNIT 35 TYPE 11 TEE - PIECE, HP EVAPORATOR / CHILLER TEE-PIECE
PARAMETERS 1
1
*
MODE
INPUTS 4
30,4 30,3 33,2 33,1
*
T IN 1      M DOT IN 1    T IN 2      M DOT IN 2
12.78 ,      16195.0 ,    12.78 ,      7570.0
*
-----
*
UNIT 45 TYPE 42 SUPPLY AIR MIXER
PARAMETERS 0
INPUTS 6
20,4 20,5 20,6 0,0 44,2 44,3
*
M DOTAIRIN1 T AIR IN 1    W AIR IN 1    MDOTAIRIN2  T AIR IN 2
*
W AIR IN 2
24120.0 ,    18.89 ,      0.0061      55393.0      26.61
0.00929
*
-----
*
UNIT 46 TYPE 41 SUPPLY FAN
PARAMETERS 1
33.0
*
P ELT [KW]
INPUTS 3
45,1 45,2 45,3
*
M DOT AIR IN T AIR IN    W AIR IN
79513.0 ,    26.44 ,      0.00861
*
-----
*
UNIT 47 TYPE 42 CONDITIONER AIR MIXER
PARAMETERS 0
INPUTS 6
0,0 0,0 0,0 0,0 44,2 44,3
*
M DOTAIRIN1 T AIR IN 1    W AIR IN 1    MDOTAIRIN2  T AIR IN 2
*
W AIR IN 2
12060.0 ,    TAI ,      WAI ,      12060.0      26.61
0.00929
*
-----
*
UNIT 48 TYPE 41 CONDITIONER FAN
PARAMETERS 1
7.5
*
P ELT [KW]

```

```

      INPUTS 3
      47,1 47,2 47,3
*    M DOT AIR IN T AIR IN      W AIR IN
      24120.0 ,    25.00 ,    0.0118

```

```

*

```

```

*-----

```

```

*

```

```

      UNIT 49 TYPE 41 REGENERATOR FAN
      PARAMETERS 1
      2.9
*    P ELT [KW]
      INPUTS 3
      0,0 44,2 44,3
*    M DOT AIR IN T AIR IN      W AIR IN
      9970.0 ,    26.60 ,    0.00929

```

```

*

```

```

*-----

```

```

*

```

```

      UNIT 50 TYPE 43 LOAD OF BUILDING
      PARAMETERS 2
      25.0      0.0093
*    T ROOM SET W ROOM SET
      INPUTS 3
      46,1 46,2 46,3
*    M DOT AIR IN T AIR IN      W AIR IN
      79513.0 ,    19.28 ,    0.00861

```

```

*

```

```

*-----

```

```

*

```

```

      UNIT 44 TYPE 41 RETURN FAN
      PARAMETERS 1
      23.0
*    P ELT [KW]
      INPUTS 3
      0,0 50,2 50,3
*    M DOT AIR IN T AIR IN      W AIR IN
      77475.0 ,    25.61 ,    0.00929

```

```

*

```

```

*-----

```

```

*

```

```

END

```

```

C*****
C*
C*
C*      FINITE STEP INTEGRATION MODEL
C*
C*      BY :   THOMAS KARL BUSCHULTE
C*
C*      MINPACK ROUTINES NECESSARY : LMDIF1, DMPAR
C*
C*      THE ROUTINES  GETINT, GTREAL, YESNO AND PLTDOT
C*      ARE PART OF THE PSYCHCHART PLOT PROGRAM "PSYCHY"
C*
C*****
C////////////////////////////////////////////////////////////////////
C%                               % %                               %
C%                               % %      UPDATED VERSION        %
C%  VERSION :   09-28-1984   # 1   % %                               %
C%                               % %                               %
C%                               % %                               %
C////////////////////////////////////////////////////////////////////
      PROGRAM MAIN
      IMPLICIT LOGICAL (L)
      INTEGER  LUNW, LUNR, LUNP, LUNF
      COMMON / LUNITS / LUNR,LUNW,LUNP,LUNF,LOF
      CALL FSIP (.FALSE.,LERROR,LEXIT)
      STOP
      END
      BLOCK DATA
C*****
C*
C*      BLOCK DATA
C*
C*****
      IMPLICIT  LOGICAL (L)
      IMPLICIT  REAL (M)
      IMPLICIT  DOUBLE PRECISION (Z)
      INTEGER   LUND
      REAL      LEWIS
      COMMON / SOLIN / MDSI,QDSI,TSI,XISI,WSI,MDSALT
      COMMON / AIRIN / MDAI,QDAI,TAI,WAI
      COMMON / CONTRL / DAREA,LEWIS,IOPT,LUND,NPRINT,NSTEP,
1      NCALL,IVIOL,LPLO,LPRINT,LDEBUG
      DATA     MDSI / 8.95 / TSI / 17.94 / XISI / 0.285 /
      DATA     MDAI / 6.70 / TAI / 25.00 / WAI / 0.0118 /
      DATA     NPRINT / 0 / NSTEP / 250 / LEWIS / 0.868 /
      DATA     LDEBUG / .TRUE. / LUND / 3 / IOPT / 1 /
C-----
C      END OF BLOCK DATA
C-----

```



```

      END
      SUBROUTINE FSIP (LPLOT,LERROR,LEXIT)
C*****
C*
C*      SUBROUTINE FSIP
C*
C*      LPLOT      LOGICAL "PLOT DESIRED" FLAG
C*      LERROR     LOGICAL "ERROR DETECTED" FLAG
C*      LEXIT      LOGICAL "EXIT DESIRED" FLAG
C*
C*****
      IMPLICIT LOGICAL (L)
      IMPLICIT REAL (M)
      IMPLICIT DOUBLE PRECISION (Z)
      EXTERNAL HMEXCH, CALL48
      INTEGER LUNW, LUNR, LUNP, LUNF, LUND
      DIMENSION ZX(2), ZXSAVE (2), ZFVEC(2), IWORK(2),
1          ZWORK(16)
      REAL LEWIS
      DOUBLE PRECISION DPMPAR
      CHARACTER*75 CHDUM
      COMMON / LUNITS / LUNR,LUNW,LUNP,LUNF,LOF
      COMMON / SOLIN / MDSI,QDSI,TSI,XISI,WSI,MDSALT
      COMMON / AIRIN / MDAI,QDAI,TAI,WAI
      COMMON / SOLOUT / TSO, XISO, WSO
      COMMON / AIROUT / TAO, WAO
      COMMON / CONTRL / DAREA,LEWIS,IOPT,LUND,NPRINT,NSTEP,
1          NCALL,IVIOL,LPLO,LPRINT,LDEBUG
      DATA TAOG / 18.89 / WAOG / 0.0061 /
      DATA TSOG / 23.18 / XISOG / 0.2838 /
      DATA AREA / 22.0 / PAMB / 101325.0 /
      DATA IWORK / 2*0 / ZWORK / 16*0.0 /
C-----
C      INITIAL WELCOME
C-----
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (2,1,1,CHDUM)
      CALL PRISIP (3,7,38,CHDUM)
      CALL PRISIP (2,1,1,CHDUM)
C-----
C      INPUT OF INLET STATES
C-----
100  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,20,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,110) MDSI
110  FORMAT (' I',F7.2,' J')
      CALL GTREAL (MDSI,0.0,10000.0,.FALSE.,.TRUE.,LBACK,

```

```

1          LEXIT)
  IF (LEXIT) GOTO 9999
  IF (LBACK) GOTO 100
200  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,21,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,210) TSI
210  FORMAT (' I',F6.2,' J')
      CALL GTREAL (TSI,0.0,200.0,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 100
300  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,22,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,310) XISI
310  FORMAT (' I',F6.3,' J')
      CALL GTREAL (XISI,0.0,0.80,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 200
400  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,23,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,410) MDAI
410  FORMAT (' I',F7.2,' J')
      CALL GTREAL (MDAI,0.0,10000.0,.FALSE.,.TRUE.,LBACK,
1          LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 300
500  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,24,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,510) TAI
510  FORMAT (' I',F6.2,' J')
      CALL GTREAL (TAI,0.0,100.0,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 400
600  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,25,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,610) WAI
610  FORMAT (' I',F7.4,' J')
      CALL GTREAL (WAI,0.0,0.1,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 500

```

```

      INEXT = 2
      GOTO 3000
C-----
C      INPUT OF THE WAY OF INTEGRATION
C-----
700   CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,34,CHDUM)
      CALL PRISIP (3,2,35,CHDUM)
      CALL PRISIP (3,2,36,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,710) IOPT
710   FORMAT (' I',I2,' J')
      CALL GETINT (IOPT,1,2,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 600
      IF (IOPT.EQ.1) THEN
C-----
C      INPUT OF THE GUESS OF THE AIR OUTLET STATE
C-----
1000  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,26,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,1010) TAOG
1010  FORMAT (' I',F6.2,' J')
      CALL GTREAL (TAOG,0.0,100.0,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 700
      ZX (1) = TAOG
1100  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,27,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,1110) WAOG
1110  FORMAT (' I',F7.4,' J')
      CALL GTREAL (WAOG,0.0,0.1,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 1000
      ZX (2) = WAOG
      ELSE
C-----
C      INPUT OF THE GUESS OF THE SOLUTION OUTLET STATE
C-----
1500  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,32,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,1010) TSOG

```

```

CALL GTREAL (TSOG,0.0,200.0,.TRUE.,.TRUE.,LBACK,LEXIT)
IF (LEXIT) GOTO 9999
IF (LBACK) GOTO 700
ZX (1) = TSOG
TSO = TSOG
1600 CONTINUE
CALL PRISIP (1,1,1,CHDUM)
CALL PRISIP (3,3,33,CHDUM)
CALL PRISIP (1,1,1,CHDUM)
WRITE (LUNW,1110) XISOG
CALL GTREAL (XISOG,0.0,0.8,.TRUE.,.TRUE.,LBACK,LEXIT)
IF (LEXIT) GOTO 9999
IF (LBACK) GOTO 1500
ZX (2) = XISOG
END IF
INEXT = 3
GOTO 3000
C-----
C      INPUT OF THE INTEGRATION CONTROL PARAMETER
C-----
2000 CONTINUE
CALL PRISIP (1,1,1,CHDUM)
CALL PRISIP (3,3,28,CHDUM)
CALL PRISIP (1,1,1,CHDUM)
WRITE (LUNW,2010) NSTEP
2010 FORMAT (' I',I6,' J')
CALL GETINT (NSTEP,0,100000,.TRUE.,.TRUE.,LBACK,LEXIT)
IF (LEXIT) GOTO 9999
IF (LBACK) GOTO 1100
2100 CONTINUE
CALL PRISIP (1,1,1,CHDUM)
CALL PRISIP (3,3,31,CHDUM)
CALL PRISIP (1,1,1,CHDUM)
WRITE (LUNW,2110) NPRINT
2110 FORMAT (' I',I6,' J')
CALL GETINT (NPRINT,0,1000,.TRUE.,.TRUE.,LBACK,LEXIT)
IF (LEXIT) GOTO 9999
IF (LBACK) GOTO 2000
2200 CONTINUE
CALL PRISIP (1,1,1,CHDUM)
CALL PRISIP (3,3,29,CHDUM)
CALL PRISIP (1,1,1,CHDUM)
WRITE (LUNW,2210) AREA
2210 FORMAT (' I',F10.4,' J')
CALL GTREAL (AREA,0.0,100000.0,.FALSE.,.TRUE.,LBACK,
1      LEXIT)
IF (LEXIT) GOTO 9999
IF (LBACK) GOTO 2100
2300 CONTINUE

```

```

      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,30,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,2310) LEWIS
2310  FORMAT (' [',F6.2,' ]')
      CALL GTREAL (LEWIS,0.0,100000.0,.TRUE.,.TRUE.,LBACK,
1      LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 2200
      INEXT = 4
      GOTO 3000

C-----
C      END OF INPUT
C-----
C      MAIN MENU
C-----
3000  CONTINUE
      IADD = 0
      IF (LPLOT) IADD = 1
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,9,42,CHDUM)
      CALL PRISIP (2,1,1,CHDUM)
      CALL PRISIP (3,2,43+IADD,CHDUM)
      CALL PRISIP (3,2,45,CHDUM)
      CALL PRISIP (3,2,46,CHDUM)
      CALL PRISIP (3,2,47,CHDUM)
      CALL PRISIP (3,2,48,CHDUM)
      CALL PRISIP (3,2,49+IADD,CHDUM)
      CALL PRISIP (3,2,51,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,3100) INEXT
3100  FORMAT (' [ ',I1,' ]')
      CALL GETINT (INEXT,0,6,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT.OR.INEXT.EQ.0) RETURN
      GOTO (100,700,2000,4000,5000,6000) INEXT

C-----
C      DO INTEGRATION, SEARCH FOR OUTLET STATES
C-----
4000  CONTINUE
C-----
C      INITIAL CALCULATIONS
C-----
      FAREA = AREA
      NFSTEP = NSTEP
      MDSALT = MDSI * XISI
      WSI = WAPW (PWTSXI(TSI+273.15,XISI,LUND,LOF),PAMB,
1      LUND,LOF)
      QDSI = MDSI * HSTSXI (TSI+273.15,XISI,LUND,LOF)
      QDAI = MDAI * HATAWA (TAI+273.15,WAI,LUND,LOF)

```

```

LPLO = .FALSE.
LITERA = .FALSE.
LBOT = .FALSE.
LFIRST = .TRUE.
ICOUNT = 1
IF (IOPT.EQ.1) THEN
    ZX (1) = TAOG
    ZX (2) = WAOG
ELSE
    ZX (1) = TSOG
    ZX (2) = XISOG
END IF
ZXSAVE (1) = ZX (1)
ZXSAVE (2) = ZX (2)
C-----
C          TOL MAY BE MAXIMAL 3.0E-06
C-----
C          TOL = 10.0 * DSQRT (DPMPAR(1))
C-----
C          CALL OF MINPACK ROUTINE LMDIF1
C-----
4100 CONTINUE
NCALL = 0
IVIOL = 0
IF (AREA.LT.FAREA/500) THEN
    CALL PRISIP (1,1,1,CHDUM)
    CALL PRISIP (3,6,55,CHDUM)
    CALL PRISIP (3,2,40,CHDUM)
    CALL PRISIP (3,2,41,CHDUM)
    CALL BEEP ()
    AREA = FAREA
    NSTEP = NFSTEP
    GOTO 3000
END IF
IF (AREA.GT.FAREA) AREA = FAREA
NSTEP = MAX0 (NINT(FLOAT(NFSTEP) * AREA / FAREA),
1           NFSTEP/10,100)
DAREA = AREA / NSTEP
ZX (1) = ZXSAVE (1)
ZX (2) = ZXSAVE (2)
WRITE (3,*) ' : ::::::::::::::::::::::::::::::::::::'
1,' : :::::::::::::::::::::::'
WRITE (3,*) ' CALL OF HMEXCH WITH AREA =' ,AREA,
1           ' NSTEP =' ,NSTEP
WRITE (3,*) ' : ::::::::::::::::::::::::::::::::::::'
1,' : :::::::::::::::::::::::'
CALL LMDIF1 (HMEXCH,2,2,ZX,ZFVEC,TOL,INFO,IWORK,
1           ZWORK,16)
C-----

```

```

C      EVALUATE RESULTS
C-----
      IF (INFO.EQ.-1) THEN
        LFIRST = .TRUE.
        IF (LITERA) THEN
          IF (LBOT) ICOUNT = ICOUNT * 2
          IF (ICOUNT .GT.128) THEN
            CALL PRISIP (1,1,1,CHDUM)
            CALL PRISIP (3,6,55,CHDUM)
            CALL PRISIP (3,2,40,CHDUM)
            CALL PRISIP (3,2,41,CHDUM)
            CALL BEEP ( )
            AREA = FAREA
            NSTEP = NFSTEP
            GOTO 3000
          END IF
          AREA = AREA * (1.500 ** (-1.00/FLOAT(ICOUNT)))
          GOTO 4100
        ELSE
          CALL PRISIP (1,1,1,CHDUM)
          CALL PRISIP (3,6,39,CHDUM)
          CALL PRISIP (3,2,40,CHDUM)
          CALL PRISIP (3,2,41,CHDUM)
          CALL PRISIP (1,1,1,CHDUM)
          CALL PRISIP (3,3,54,CHDUM)
          CALL PRISIP (1,1,1,CHDUM)
          CALL YESNO (LITERA,LBACK,LEXIT)
          IF (LEXIT) RETURN
          IF (LITERA) THEN
            AREA = AREA * 0.6666666666666666
            GOTO 4100
          ELSE
            AREA = FAREA
            NSTEP = NFSTEP
            GOTO 3000
          END IF
        END IF
      ELSE
C-----
C      INTEGRATION SUCESSFUL
C-----
        ZXSAVE (1) = ZX (1)
        ZXSAVE (2) = ZX (2)
        IF (.NOT.LBOT) LBOT = .TRUE.
        QDAO = MDAI * HATAWA (TA0+273.15,WA0,LUND,LOF)
        DQD = QDAO - QDAI
        DMD = (WA0 - WAI) * MDAI
        IF (IOPT.EQ.1) THEN
          TA0 = ZX (1)

```

```

        WAO = ZX (2)
    ELSE
        TSO = ZX (1)
        XISO = ZX (2)
        WSO = WAPW (PWTSXI(TSO+273.15,XISO,LUND,LOF)
1          ,PAMB,LUND,LOF)
    END IF
    IF (LFIRST) THEN
        LFIRST = .FALSE.
    ELSE
        IF (ICOUNT.GT.2) ICOUNT = ICOUNT / 2
    END IF
    IF (FAREA-AREA.LT.0.0001*FAREA) THEN
        WRITE (LUNW,*) ' '
        WRITE (LUNW,*)
1        ' FINAL RESULT (INFO =',INFO,' ) : '
        WRITE (LUNW,*) ' TAO =',TAO,' WAO =',WAO
        WRITE (LUNW,*) ' TSO =',TSO,' WSO =',WSO,
1        ' XISO =',XISO
        WRITE (LUNW,*) ' EXCHANGED MASS =',DMD,
1        ' EXCHANGED TOTAL HEAT =',DQD
        IF (LITERA) CALL BEEP ( )
C-----
C      UPDATE GUESSES = RESULTS ?
C-----
        CALL PRISIP (1,1,1,CHDUM)
        CALL PRISIP (3,2,37,CHDUM)
        CALL PRISIP (1,1,1,CHDUM)
        CALL YESNO (LYES,LBACK,LEXIT)
        IF (LEXIT) RETURN
        IF (LYES) THEN
C-----
C      UPDATE GUESSES = RESULTS !
C-----
            TAOG = TAO
            WAOG = WAO
            TSOG = TSO
            XISOG = XISO
        END IF
        AREA = FAREA
        NSTEP = NFSTEP
        GOTO 3000
    END IF
    AREA = AREA * (1.500 ** (1.00/FLOAT(ICOUNT)))
C-----
C      UPDATE GUESSES = RESULTS !
C-----
        TAOG = TAO
        WAOG = WAO

```



```

        TSOG = TSO
        XISOG = XISO
        GOTO 4100
    END IF
    GOTO 3000
C-----
C      SINGLE INTEGRATION
C-----
5000  CONTINUE
      DAREA = AREA / NSTEP
      MDSALT = MDSI * XISI
      WSI = WAPW (PWTSXI (TSI+273.15,XISI,LUND,LOF),PAMB,
1      LUND,LOF)
      QDSI = MDSI * HSTSXI (TSI+273.15,XISI,LUND,LOF)
      QDAI = MDAI * HATAWA (TAI+273.15,WAI,LUND,LOF)
      IF (LPLO) THEN
        CALL PLTDOT (TAI,WAI,0,5,3,LERROR)
        IF (LERROR) RETURN
        CALL PLTDOT (TSI,WSI,0,3,3,LERROR)
        IF (LERROR) RETURN
      END IF
      LPLO = LPLO
      NCALL = 0
      CALL HMEXCH (2,2,ZX,ZFVEC,IFLAG)
      GOTO 3000
C-----
C      CALL OF LMDIF1 / TYPE48
C-----
6000  CONTINUE
6100  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,52,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,6110) TAO
6110  FORMAT (' [',F7.3,' ]')
      CALL GTREAL (TAO,0.0,100.0,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 3000
6200  CONTINUE
      CALL PRISIP (1,1,1,CHDUM)
      CALL PRISIP (3,3,53,CHDUM)
      CALL PRISIP (1,1,1,CHDUM)
      WRITE (LUNW,6210) WAO
6210  FORMAT (' [',F9.6,' ]')
      CALL GTREAL (WAO,0.0,0.1,.TRUE.,.TRUE.,LBACK,LEXIT)
      IF (LEXIT) GOTO 9999
      IF (LBACK) GOTO 6100
      ZX (1) = 0.7
      ZX (2) = 0.7

```

```

      TOL = 1.0E-05
      CALL LMDIF1 (CALL48,2,2,ZX,ZFVEC,TOL,INFO,IWORK,
1              ZWORK,16)
      WRITE (LUNW,6800) ZX (1), ZX (2)
6800  FORMAT ('0 RESULT :  EFFHEAT =',F6.4,
1          '  EFFMASS =',F6.4/)
      GOTO 3000
9999  CONTINUE
      RETURN
C-----
C      END OF FSIP
C-----
      END
      SUBROUTINE HMEXCH (NDUM1,NDUM2,ZX,ZFVEC,IFLAG)
C*****
C*
C*  SUBROUTINE HMEXCH
C*
C*  IS CALLED BY THE MINPACK ROUTINE  LMDIF1
C*
C*****
      IMPLICIT LOGICAL (L)
      IMPLICIT REAL (M)
      IMPLICIT DOUBLE PRECISION (Z)
      INTEGER LUNW, LUNR, LUNP, LUNF, LUND
      REAL LEWIS
      LOGICAL SUPSAT
      CHARACTER*75 CHDUM
      DIMENSION ZX(2), ZFVEC(2)
      COMMON / LUNITS / LUNR,LUNW,LUNP,LUNF,LOF
      COMMON / SOLIN / MDSI,QDSI,TSI,XISI,WSI,MDSALT
      COMMON / AIRIN / MDAI,QDAI,TAI,WAI
      COMMON / SOLOUT / TSO,XISO,WSO
      COMMON / AIROUT / TAO,WAO
      COMMON / CONTRL / DAREA,LEWIS,IOPT,LUND,NPRINT,NSTEP,
1          NCALL,IVIOL,LPLO,LPRINT,LDEBUG
1  DATA PAMB / 101325.0 / CPAIR / 1.0076 /
1  NCALL / 0 /
C-----
      NCALL = NCALL + 1
      IF (LDEBUG) THEN
          WRITE (LUNW,*) ' '
          WRITE (LUNW,*) ' HMEXCH CALL #',NCALL,' WITH : '
      END IF
C-----
C      INITIAL CALCULATIONS
C      IOPT = 1 :  START INTEGRATION AT SOLUTION INLET
C                  GUESS AIR OUTLET
C-----

```

```

      IF (IOPT.EQ.1) THEN
        TSOL = TSI
        XISOL = XISI
        WSOL = WSI
        TAO = ZX (1)
        WAO = ZX (2)
        TAIR = TAO
        WAIR = WAO
        QDAOG = MDAI * HATAWA (TAIR+273.15,WAIR,LUND,LOF)
        ZMDSOL = MDSI
        ZQDSOL = QDSI
        ZMDAIR = MDAI * (1.0 + WAIR)
        ZQDAIR = QDAOG
        FACTOR = 1.0
        IF (LDEBUG) WRITE (LUNW,*) ' TAOG =',TAO,
1                                     ' WAOG =',WAO
      ELSE
C-----
C          IOPT = 2 :  START INTEGRATION AT AIR INLET
C          GUESS SOLUTION OUTLET
C-----
        TSO = ZX (1)
        XISO = ZX (2)
        WSO = WAPW (PWTSXI(TSO+273.15,XISO,LUND,LOF)
1                ,PAMB,LUND,LOF)
        TSOL = TSO
        XISOL = XISO
        WSOL = WSO
        TAIR = TAI
        WAIR = WAI
        MDSOG = MDSALT / XISOL
        QDSOG = MDSOG * HSTSXI(TSOL+273.15,XISOL,LUND,LOF)
        ZMDSOL = MDSOG
        ZQDSOL = QDSOG
        ZMDAIR = MDAI * (1.0 + WAI)
        ZQDAIR = QDAI
        FACTOR = -1.0
        IF (LDEBUG) WRITE (LUNW,*) ' TSOG =',TSOL,
1                                     ' WSOG =',WSOL
      END IF
C-----
C          PLOT OUTLET POINT (AT START OF INTEGRATION)
C-----
      IF (LPLO) THEN
        IF (IOPT.EQ.1) THEN
          CALL PLTDOT (TAIR,WAIR,0,2,3,LERROR)
        ELSE
          CALL PLTDOT (TSOL,WSOL,0,4,3,LERROR)
        END IF
      END IF

```

```

      IF (LERROR) RETURN
      END IF
C-----
C      CALCULATE PLOT CONTROL PARAMETERS
C-----
C      WRITE (3,*) ' TSO ',TSO,' TSI ',TSI,' NPRINT ',NPRINT
      IF (NPRINT.GT.0) THEN
        TDS = (TSO - TSI) / FLOAT (NPRINT)
      ELSE
        TDS = TSO - TSI
      END IF
      TPLO = TSOL + (TDS * FACTOR)
C      WRITE (3,*) ' TDS ',TDS,' TPLO ',TPLO
C-----
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C      START OF "STEP" SUMMATION LOOP
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
      LPRINT = LDEBUG
      LFIRST = .TRUE.
      DO 8000 N = 1,NSTEP
      IF (NPRINT.EQ.0) THEN
        LDEB = .FALSE.
      ELSE
        LDEB = (LPRINT.AND.(MOD(N,(NSTEP/NPRINT)).EQ.0))
      END IF
C-----
C      EXCHANGED HEAT AND MASS FLOW RATES PER STEP
C-----
      DELHA = HMASS (TSOL) * DAREA
      IF (SUPSAT(TAIR+273.15,WAIR,PAMB,LUNW,.FALSE.)) THEN
        IF (NCALL.GT.4) THEN
          IFLAG = -1
          RETURN
        END IF
        DELMDS = 0.0
        DELQDS = DELHA * (LEWIS * CPAIR * (TAIR-TSOL))
        IF (LPRINT.AND.LFIRST) THEN
          WRITE (LUNW,*) ' / '
          WRITE (LUNW,*) ' > SATURATION OF THE AIR < '
          WRITE (LUNW,*) ' / '
          LFIRST = .FALSE.
        END IF
      ELSE
        DELMDS = DELHA * (WAIR - WSOL)
        DELQDS = DELHA * (HWEVAP(TAIR+273.15,LUND,LOF)
1          * (WAIR-WSOL)
2          + LEWIS * CPAIR * (TAIR-TSOL))
      END IF
      ZMDSOL = ZMDSOL + (FACTOR * DELMDS)

```

```

      ZQDSOL = ZQDSOL + (FACTOR * DELQDS)
      HSOL = ZQDSOL / ZMDSOL
      XISOL = MDSALT / ZMDSOL
      TSOL = TSHSXI (HSOL,XISOL,LUND,LOF) - 273.15
      WSOL = WAPW (PWT SXI(TSOL+273.15,XISOL,LUND,LOF),
1      PAMB,LUND,LOF)
      IF (IOPT.EQ.1) THEN
        WAIR = WAO + (ZMDSOL - MDSI) / MDAI
        HAIR = (QDAOG + ZQDSOL - QDSI) / MDAI
      ELSE
        WAIR = WAI + (ZMDSOL - MDSOG) / MDAI
        HAIR = (QDAI + ZQDSOL - QDSOG) / MDAI
      END IF
      TAIR = TAHAWA (HAIR,WAIR,LUND,LOF) - 273.15
      ZMDAIR = MDAI * (1.0 + WAIR)
      ZQDAIR = MDAI * HAIR
C-----
C      DOCUMENTATION
C-----
      IF (LDEB) THEN
        WRITE (LUND,6100) TSOL,WSOL,XISOL,TAIR,WAIR
6100      FORMAT (' TSOL:',F7.2,' WSOL:',F8.5,
1              ' XISOL:',F7.4/
2              ' TAIR:',F7.2,' WAIR:',F8.5/)
      END IF
C-----
C      PLOT INTERMEDIATE STATES IF DESIRED
C-----
      IF ((LPLO).AND.
1      ((IOPT.EQ.1.AND.((TSOL.GT.TPLO.AND.TDS.GT.0.0).OR.
2      (TSOL.LT.TPLO.AND.TDS.LT.0.0)))
3      .OR.
4      (IOPT.EQ.2.AND.((TSOL.LT.TPLO.AND.TDS.GT.0.0).OR.
5      (TSOL.GT.TPLO.AND.TDS.LT.0.0))))))
6      THEN
        CALL PLTDOT (TSOL,WSOL,0,4,1,LERROR)
        IF (LERROR) RETURN
        CALL PLTDOT (TAIR,WAIR,0,2,1,LERROR)
        IF (LERROR) RETURN
        TPLO = TPLO + (TDS * NINT(FACTOR))
      END IF
C-----
C      CHECK FOR BOUNDARY VIOLATIONS
C-----
      IF ((TSOL.LT.10.0).OR.(TSOL.GT.120.0).OR.
1      (TAIR.LT.0.0).OR.(TAIR.GT.120.0).OR.
2      (WSOL.LT.0.0).OR.(WSOL.GT.0.1).OR.
3      (WAIR.LT.0.0).OR.(WAIR.GT.0.1)) THEN
        IVIOL = IVIOL + 1

```

```

      IF (IVIOL.EQ.2) IFLAG = -1
      GOTO 9000
    END IF
8000  CONTINUE
C-----
C      END OF "STEP" SUMMATION LOOP
C-----
C-----
C      "CURRENT" OUTLET STATES
C      (AS RESULT OF THIS INTEGRATION)
C-----
      IF (IOPT.EQ.1) THEN
        TSO = TSOL
        WSO = WSOL
        XISO = XISOL
      ELSE
        TAO = TAIR
        WAO = WAIR
      END IF
C-----
C      DOCUMENT FINAL RESULTS
C-----
      IF (LPRINT) THEN
        IF (IOPT.EQ.1) THEN
          WRITE (LUND,7100) TSOL,WSOL,XISOL,TAIR,WAIR,
1             TAI,WAI
7100  FORMAT ('  TSO :',F7.2,' WSO :',F8.5,
1          '      XISO:',F7.4/
2          '  TAIC:',F7.2,' WAIC:',F8.5/
3          '  TAI :',F7.2,' WAI :',F8.5/)
        ELSE
          WRITE (LUND,7200) TSOL,WSOL,XISOL,TSI,WSI,
1             XISI,TAIR,WAIR
7200  FORMAT ('  TSIC:',F7.2,' WSIC:',F8.5,
1          '      XISIC:',F7.4/
2          '  TSI :',F7.2,' WSI :',F8.5,
3          '      XISI: ',F7.4/
4          '  TAO :',F7.2,' WAO :',F8.5/)
        END IF
      END IF
9000  CONTINUE
      IF (IOPT.EQ.1) THEN
        ZFVEC (1) = ERRFU (TAI,TAIR,LUNW,.FALSE.)
        ZFVEC (2) = ERRFU (WAI,WAIR,LUNW,.FALSE.)
      ELSE
        ZFVEC (1) = ERRFU (TSI,TSOL,LUNW,.FALSE.)
        ZFVEC (2) = ERRFU (WSI,WSOL,LUNW,.FALSE.)
      END IF
      IF (LPLO) THEN

```

```

      IF (IOPT.EQ.1) THEN
        CALL PLTDOT (TSOL,WSOL,0,4,3,LERROR)
      ELSE
        CALL PLTDOT (TAIR,WAIR,0,2,3,LERROR)
      END IF
      IF (LERROR) RETURN
    END IF
9999  CONTINUE
      RETURN
C-----
C      END OF HMEXCH
C-----

      END
      SUBROUTINE CALL48 (IDUM1,IDUM2,ZX,ZVEC,IFLAG)
C*****
C*
C*  SUBROUTINE CALL48
C*
C*  IS CALLED BY THE MINPACK ROUTINE LMDIF1
C*  AND CALLS THE TRNSYS MODEL TYPE48
C*
C*****
      IMPLICIT LOGICAL (L)
      IMPLICIT REAL (M)
      IMPLICIT DOUBLE PRECISION (Z)
      DIMENSION ZX (2), ZVEC (2), XIN (6), OUT (20),
1      PAR (4), INFO (10)
      INTEGER LUNW, LUNR, LUNP, LUNF
      COMMON / LUNITS / LUNR,LUNW,LUNP,LUNF,LOF
      COMMON / SOLIN / MDSI, QDSI, TSI, XISI, WSI, MDSALT
      COMMON / AIRIN / MDAI, QDAI, TAI, WAI
      COMMON / SOLOUT / TSO, XISO, WSO
      COMMON / AIROUT / TAO, WAO
C-----
      INFO (7) = 0
      XIN (1) = MDSI
      XIN (2) = TSI
      XIN (3) = XISI
      XIN (4) = MDAI
      XIN (5) = TAI
      XIN (6) = WAI
      PAR (1) = 1.0
      PAR (2) = ZX (1)
      PAR (3) = ZX (2)
      PAR (4) = 11.0
      WRITE (LUNW,*) ' CALL OF TYPE 48 WITH : '
      WRITE (LUNW,*) ' EFFHEAT =',PAR(2),' EFFMASS =',
1      PAR(3)
      CALL TYPE48 (1.0,XIN,OUT,T,DTDT,PAR,INFO)

```

```

ZVEC (1) = ABS (OUT(5) - TAO) / TAO
ZVEC (2) = ABS (OUT(6) - WAO) / WAO
RETURN

```

```

C-----
C      END OF CALL48
C-----

```

```

      END
      SUBROUTINE PRISIP (IOPT,IMESS,ITEXT,CHLIN)

```

```

C*****

```

```

C*

```

```

C*      SUBROUTINE PRISIP

```

```

C*

```

```

C*      + IOPT      OPTION PARAMETER

```

```

C*          = 1      PRINT FRAME LINE      >-----<

```

```

C*          = 2      PRINT SPACE LINE      >

```

```

C*          = 3      PRINT PREDEFINED TEXT >      TEXT

```

```

C*          = 4      PRINT SUPPLIED TEXT   >      TEXT

```

```

C*

```

```

C*      + IMESS      MESSAGE NUMBER

```

```

C*      + ITEXT      TEXT      NUMBER ( FOR PREDEFINED TEXT )

```

```

C*      + CHLIN      CHARACTER STRING CONTAINING THE SUPPLIED

```

```

C*                  TEXT ( CHARACTER*75 )

```

```

C*

```

```

C*      + SPECIFY THESE ON CALLING

```

```

C*

```

```

C*****

```

```

C*

```

```

C*      PRISIP PRINTS ONE LINE AT A TIME ON THE SCREEN

```

```

C*

```

```

C*****

```

```

      PARAMETER (IMMAX=10 , ITMAX=60)

```

```

      IMPLICIT LOGICAL (L)

```

```

      INTEGER      LUNW, LUNR, LUNP, LUNF, LUND

```

```

      CHARACTER*10 CHMESS (IMMAX)

```

```

      CHARACTER*63 CHTEXT (ITMAX)

```

```

      CHARACTER*75 CHLIN

```

```

      COMMON / LUNITS / LUNR,LUNW,LUNP,LUNF,LOF

```

```

C-----

```

```

C      ASSIGNMENT OF THE MESSAGES AND TEXTS

```

```

C-----

```

```

      DATA CHMESS (1) / '-----' /

```

```

      DATA CHMESS (2) / '      : ' /

```

```

      DATA CHMESS (3) / ' INPUT : ' /

```

```

      DATA CHMESS (4) / ' HINT  : ' /

```

```

      DATA CHMESS (5) / ' ERROR : ' /

```

```

      DATA CHMESS (6) / 'WARNING : ' /

```

```

      DATA CHMESS (7) / 'WELCOME : ' /

```

```

      DATA CHMESS (8) / 'BYE BYE : ' /

```

```

      DATA CHMESS (9) / ' MENU  : ' /

```



```

DATA CHMESS (10) / ' ' /
C-----
DATA CHTEXT ( 1) /
$'-----',
$'-----'//
DATA CHTEXT ( 2) /
$' //
DATA CHTEXT ( 9) /
$'....1....1....1....2....1....3....1....4..',
$'..1....5....1....6... '//
DATA CHTEXT ( 11) /
$' you must enter at least one character '//
DATA CHTEXT ( 12) /
$' value out of range '//
DATA CHTEXT ( 13) /
$' Hit RETURN to continue '//
DATA CHTEXT ( 20) /
$' Solution mass flow rate at inlet '//
DATA CHTEXT ( 21) /
$' Solution temperature at inlet '//
DATA CHTEXT ( 22) /
$' Solution concentration at inlet '//
DATA CHTEXT ( 23) /
$' Air mass flow rate at inlet '//
DATA CHTEXT ( 24) /
$' Air temperature at inlet '//
DATA CHTEXT ( 25) /
$' Air humidity ratio at inlet '//
DATA CHTEXT ( 26) /
$' Air temperature at outlet (guess) '//
DATA CHTEXT ( 27) /
$' Air humidity ratio at outlet (guess) '//
DATA CHTEXT ( 28) /
$' number of "integration" (summation) steps '//
DATA CHTEXT ( 29) /
$' total contact area in spray chamber '//
DATA CHTEXT ( 30) /
$' Lewis number '//
DATA CHTEXT ( 31) /
$' number of documentation print-outs (plo',
$'tted points) '//
DATA CHTEXT ( 32) /
$' Solution temperature at outlet (guess) '//
DATA CHTEXT ( 33) /
$' Solution concentration at outlet (guess) '//
DATA CHTEXT ( 34) /
$' Which way do you want to integrate ? '//
DATA CHTEXT ( 35) /
$' 1 start at solution inlet (guess ai',

```

```

$r outlet states) //
DATA CTEXT ( 36) /
$'      2      start at air inlet (guess solutio',
$n outlet states) //
DATA CTEXT ( 37) /
$'      Do you want to use these results as new',
$' guesses ? [Y/N] //
DATA CTEXT ( 38) /
$'      to      F S I P      (Finite Step Integrati',
$'on Program) //
DATA CTEXT ( 39) /
$'      The integration hit a boundary ! //
DATA CTEXT ( 40) /
$'      The result may be erroneous ! //
DATA CTEXT ( 41) /
$'      Try new initial guesses . //
DATA CTEXT ( 42) /
$'      Options currently available : //
DATA CTEXT ( 43) /
$'      0      EXIT from program //
DATA CTEXT ( 44) /
$'      0      Return to PSYCHY //
DATA CTEXT ( 45) /
$'      1      Redefine inlet states //
DATA CTEXT ( 46) /
$'      2      Redefine guesses of outlet states //
DATA CTEXT ( 47) /
$'      3      Redefine integration parameters //
DATA CTEXT ( 48) /
$'      4      Search for "exact" outlet states //
DATA CTEXT ( 49) /
$'      5      Do a single integration //
DATA CTEXT ( 50) /
$'      5      Do a single integration and p',
$'lot the results //
DATA CTEXT ( 51) /
$'      6      Call TYPE48 and search for th',
$'e EFF parameters //
DATA CTEXT ( 52) /
$'      Air temperature at outlet //
DATA CTEXT ( 53) /
$'      Air humidity ratio at outlet //
DATA CTEXT ( 54) /
$'      Do you want to continue (start iteratio',
$'n) ? [Y/N] //
DATA CTEXT ( 55) /
$'      The iteration was not sucessful ! //

```

C-----
C WRITING OF THE LINE

```

C-----
      IF (IOPT.GT.4.OR.IOPT.LT.1) RETURN
      GOTO (100,200,300,400) IOPT
      RETURN
100    CONTINUE
      WRITE (LUNW,150)
150    FORMAT (' > ',77(' - '), '< ')
      RETURN
200    CONTINUE
      WRITE (LUNW,250) CHMESS (2)
250    FORMAT (' > ',A10,66(' '), '< ')
      RETURN
300    CONTINUE
      IF (IMESS.LT.1.OR.IMESS.GT.IMMAX) RETURN
      IF (ITEXT.LT.1.OR.ITEXT.GT.ITMAX) RETURN
      WRITE (LUNW,350) CHMESS (IMESS), CHTEXT (ITEXT)
350    FORMAT (' > ',A10,A63, ' < ')
      RETURN
400    CONTINUE
      WRITE (LUNW,450) CHLIN
450    FORMAT (' > ',A75, ' < ')
      RETURN
C-----
C      END OF PRISIP
C-----

      END
      REAL FUNCTION HMASS (TAIR,TSOL,XISOL)
C*****
C*
C*      REAL FUNCTION HMASS
C*
C*      + TAIR      LOCAL AIR TEMPERATURE (DEG C)
C*      + TSOL      LOCAL SOLUTION TEMPERATURE (DEG C)
C*      + XISOL     LOCAL SOLUTION CONCENTRATION
C*                  (KG SALT/KG SOLUTION)
C*
C*      + SPECIFY THESE ON CALLING
C*****
C*
C*      HMASS CALCULATES THE MASS TRANSFER COEFFICIENT AS
C*      A FUNCTION OF AIR AND SOLUTION TEMPERATURE AND
C*      SOLUTION CONCENTRATION
C*
C*****
      REAL MU
      DATA GEARTH / 1.770 /
C-----
C      GEARTH = G ** 1 / 4 [M/S**2]
C-----

```

```

      DATA CSUBD / 0.83 /
C-----
C      CSUBD = C SUB D ** 1 / 4
C      DRAG COEFFICIENT OF A SPHERE FOR LAMINAR FLOW
C-----
      DATA DDROPL / 1.0E-04 /
C-----
C      DDROPL = D DROPLET [M]
C      MEAN DIAMETER OF DROPLETS IN THE SPRAY
C      (ASSUMPTION)
C-----
      DATA RHOEXP / 0.9166667 /
C-----
C      RHOEXP = 11 / 12
C      EXPONENT
C-----
C      FUNCTIONS
C-----
      DIFFU (T) = (2166 + 14.85 * T) * 1.0E-08
      RHO (T) = ((1.0E-05 * T) - 4.4607E-03) * T + 1.29064
      ALPHA (T) = ((1.90476219E-10 * T) + 1.27142853E-07)
1      * T + 1.859524E-05
      MU (T) = 4.5833E-08 * T + 1.72E-05
C-----
      TMEAN = (TAIR + TSOL) / 2.0
      HMASS = 2.0 * RHO (TMEAN) * DIFFU (TMEAN) / DDROPL
2      + 0.645 * LEWIS (TAIR,TSOL) ** 0.33333
3      * RHO (TMEAN) ** RHOEXP
4      * DSTSXI(TSOL+273.15,XISOL,3,.FALSE.)
5      ** 0.250
6      * GEARTH * CSUBD * DIFFU (TMEAN) /
7      ( DDROPL ** 0.250
8      * MU (TMEAN) ** 0.16666667
9      * ALPHA (TMEAN) ** 0.33333 )
      RETURN
C-----
C      END OF HMASS
C-----
      END
      REAL FUNCTION LEWIS (TAIR,TSOL)
C*****
C*
C*      REAL FUNCTION LEWIS
C*
C*      + TAIR      LOCAL AIR TEMPERATURE (DEG C)
C*      + TSOL      LOCAL SOLUTION TEMPERATURE (DEG C)
C*
C*      + SPECIFY THESE ON CALLING
C*****

```

APPENDIX D

```

C*                                     *
C*      LEWIS CALCULATES THE LEWIS NUMBER AS      *
C*      A FUNCTION OF AIR AND SOLUTION TEMPERATURE *
C*                                     *
C*****
      TMEAN = (TAIR + TSOL) / 2.0
      LEWIS = 0.868 + 4.09671E-04 * TMEAN
      IF (TMEAN.LT.-17.8.OR.TMEAN.GT.149.0) THEN
        WRITE (3,*) ' '
        WRITE (3,*) ' >>> LEWIS FUNCTION OUT OF RANGE,'
1      WRITE (3,*) ' T MEAN =',TMEAN
        WRITE (3,*) ' '
      END IF
      RETURN
C-----
C      END OF LEWIS
C-----
      END

```