MODELING AND SIMULATION OF MEE SYSTEM WITH PRODUCT AND CONDENSATE FLASH

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree

of

MASTER OF TECHNOLOGY

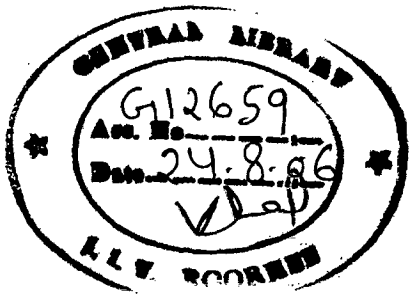
in

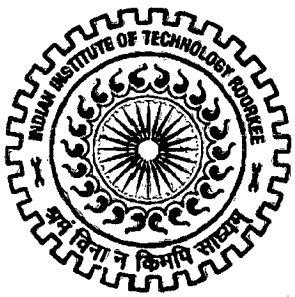
CHEMICAL ENGINEERING

(With Specialization in Industrial Pollution Abatement)

By

BRAJESH KR. TRIPATHI





DEPARTMENT OF CHEMICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE - 247 667 (INDIA)

JUNE, 2006

CANDIDATE'S DECLARATION

I hereby declare that the work which is being presented in the dissertation entitled "**MODELING AND SIMULATION OF MEE SYSTEM WITH PRODUCT AND CONDENSATE FLASH**" in partial fulfilment of the requirements for the award of the degree of **Master of Technology in Chemical Engineering** with specialization in **Industrial Pollution Abatement** submitted in the Department of Chemical Engineering, Indian Institute of Technology, Roorkee is an authentic record of my own work carried out during the period from **July 2005 to June 2006**, under the supervision of **Dr. Ravindra Bhargava**.

The matter embodied in this dissertation work has not been submitted by me for the award of any other degree.

Place: IIT ROORKEE

Date 30 June, 2006

Tripathi

(BRAJESH KR. TRIPATHI)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

R Bhargava 30.6.06

(DR. RAVINDRA BHARGAVA)
Assistant Professor
Department of Chemical Engineering
Indian Institute of Technology Roorkee

ABSTRACT

In the present work, a steady state model of sextuple effect backward feed evaporator system including heat recovery by product and condensate flash has been developed. Flashing can be incorporated for product flash only, alternate condensate flash only, series condensate only or the combination of these. And all these are compared to achieve efficient heat recovery system.

Quasi Newton method is used to solve the nonlinear model equations. Overall Heat Transfer Coefficient (OHTC) is improved using Gudmundson's model for Long Tube Vertical evaporators. In the model, for BPR determination TAPPI correlation has been used. For determining most of the liquor properties Regestad correlations are used.

Results for different flash arrangements are compared and it is concluded that a system with both product and series condensate flash arrangement give the most efficient heat recovery system.

Although the program thus developed is used for multiple effect evaporator system of pulp and paper industry, nevertheless it can also be applied in many other process industries, with suitable modification in calculation method of OHTC.

ACKNOWLEDGEMENT

I am greatly indebted to my guide **Dr. Ravindra Bhargava** for his kind support and guidance during the entire course of this dissertation work. His co-operation and in-depth knowledge have made my work possible.

I would like to thank **Dr. Shri Chand**, professor and head of the department, and **Dr. I.M.Mishra,** professor, Department of Chemical Engineering, IIT Roorkee, for providing various facilities during the course of this dissertation work.

Also I would like to thank Gopal, Ramesh, Sumit and all my colleagues who spared their valuable time in the hours of need in my present study.

Last but not the least, it is all owed to the blessings of my parents, and God that I have come up with this work in due time.

(BRAJESH KR. TRIPATHI)

CONTENTS

LIST OF FIGURES

NOMENCLATURE

A or AREA= heat transfer area of evaporator or effect, m^2

BPR = boiling point rise, °C

C = flow rate of condensate leaving each effect, kg/s, or total solids concentration, %

C_p= specific heat, J/(kg.K)

C_μ= viscosity correction factor

C_T = temperature correction factor

e = error of tolerance

F= feed rate, kg/s

FD = liquor feed per unit cross sectional area, kg/(s.m^2)

g = a function defined by $g = f/(F\lambda_0)$

h = specific enthalpy of the liquor, J/kg

H = specific enthalpy of the vapor, J/kg

K = thermal conductivity, W/(m.K)

L = liquor flow rate from the effect, kg/s

l = fractional liquor flow rate defined by l = L/F

M = flash vapor flow rate, kg/s

m = fractional flash vapor flow rate defined by m = M/F

q = heat flux, W/m^2

S_e = steam economy

t = boiling temperature of the liquor, °C

T = saturation temperature of water at pressure P, °C

T_0 = steam temperature

w = fractional temperature of the effect defined by w = T/To

T_{max} = maximum temperature, °C

TS = total solids concentration, %

U = overall heat transfer coefficient, W/(m^2.K)

V_s= vapor flow rate from the effect, kg/s

v_s = fractional steam flow rate defined by $v_s = V_s/F$

X = mass fraction of solute in the liquor

Subscripts

s = steam

f = feed

i = effect number (i=l, 2... n)

Greek letters

μ = viscosity, cp

ρ = density, kg/m^3

λ = latent heat of vaporization, J/kg

1. INTRODUCTION

Chemical industries widely use evaporators to concentrate weak solutions and recover the desired components. It is used in industries like paper, sugar, and caustic soda to concentrate black liquor, sugar cane juice, and caustic soda solutions, respectively. In pulp and paper plant Kraft pulping process is widely used pulp manufacturing process in all over the world due to its adaptability to all types of fibrous material, strong pulp and paper manufactured and development of the modern chemical recovery process.

In the kraft process wood chips and the white liquor (sodium sulphide and sodium hydroxide) are charged in a digester. The digester is maintained in the temperature range of 160-180 °C. The cooked chips together with the spent liquor, called black liquor, are then discharged to a blow tank and then to washers where they are separated. The wood pulp is sent for its further processing to obtain paper. Part of the black liquor is recycled to the digester and the remainder is concentrated to recover the chemicals.

The weak black liquor contains dissolved chemicals ranging from 12% to 20%. So this liquor is concentrated to about 45% to 50%. The evaporation or concentration of black liquor is normally carried out in a multiple effect evaporator system using low-pressure steam where a series of evaporators are operated at different pressures so that the vapor from one evaporator body becomes the steam supply to the next evaporator body. This improves high steam economy.

The black liquor exhibits rapid increase in viscosity with increase in concentration and decrease in temperature. Consequently a backward feed multiple effect evaporator system is amongst the best feeding arrangements. The evaporation process consumes maximum energy in a manufacturing operation. It is reported that in cellulosic fiber and filament industries in India, the consumption of steam in evaporation process comes to about 10 to 17.5 tonne per tonne of paper production. The black liquor multiple effect evaporators also consume around 24 % to 30% of the total steam energy required in a large paper mill. Therefore it arises the need to study of the means of decreasing energy consumption or enhancing the performance of the evaporator system. In order to achieve this, it would be desirable to have a simulation model that can be used to predict the performance of the

evaporation process while considering different types of flashing for the recovery of energy.

The efficiency of the evaporator is measured by the steam economy. It depends on number of effects, feed temperature, steam pressure; feed concentration, product concentration, liquor and condensate flash, and flow sequence, thus the effect of these parameters on steam economy must be investigated.

For analyzing the performance and developing the model of a plant or an individual process unit the collection and analysis of data from processes are important means. Mathematical model of a plant is not only useful for day-to-day performance evaluation, but also for optimization and control of the process.

Based on above consideration, present work has following objectives:

1) To formulate a mathematical model for sextuple effect evaporator system with backward feed incorporating product and condensate flash.
2) For simulating the model, develop a computer program using C++ programming language.
3) To arrive at an efficient system, determine steam economy for different types of flashing arrangement.

2. LITERATURE REVIEW

The modeling of multiple effect evaporators has been carried out by several researchers. For many years the approach for the design of evaporator *has* been the trial and error method, as presented by **Geankoplis** [26], **Bremford** [12], etc. In this method, the temperature difference between steam chest and the boiling liquid in the evaporator is first estimated. The design calculation is then performed repeatedly until some restriction on the design parameters is met. The initial guesses must be good ones for this method to converge easily. Furthermore, for large number of effects, the method becomes more tedious.

2.1 Evaporator house Modeling:

Stewart and **Beveridge** [3] they presented a generalized cascade algorithm for the simulation of multiple effect evaporator systems with backward feed arrangement. The algorithm is based on the simultaneous solution of linearised forms of the effect model. The cascade algorithm they developed belongs to a general class of method for solving non-linear problems known as quasi-Newton method.

Radovic et al [10] they have given a mathematical model for computer design and analysis of a five effect evaporator system, commonly used in sugar industry. Their model consists of four equations for each effect. For deriving the model equations they used enthalpy balance, heat transfer rate, phase equilibrium relationship, and mass balance equation. There model give rise to a set of 18 linear algebraic equations in 18 unknowns, and in the case of analysis of operation of an existing system gives a set of 20 equations in 20 unknown. They solved the proposed model by the use of Newton Raphson method by making simulation program in FORTRAN, they done their calculation in two modes. The first one is, they calculated the steam consumption, heat transfer surface, distribution of temperature, composition, and mass flow rates. The second one they done are to calculate all the necessary process parameters of an existing industrial evaporator system.

Ayangbile Okeke and **Beveridge** [4] they give a generalized cascade algorithm for steady state simulation of multiple effect evaporator system. Their algorithm is capable of handling any feed arrangement, which includes liquor and condensate flash units and feed preheating unit. They tested their algorithm for a number of evaporator systems for different feed arrangements, including the conventional ones-Forward, backward and mixed feed system. It proved to be very powerful

Lambert and **Joye** [8] they give the procedure which is useful in the design of multiple effect evaporator systems, there algorithm reduces the series of nonlinear algebraic equations that govern the evaporator system to a linear form and solves them iteratively by a linear technique, e.g., Gaussian elimination. They used boiling point rise and enthalpy relationships, which are obtained by curve fitting or interpolation. For a given number of stages, the calculation procedure used by them computes design variables such

as area (or area ratios between effects), externally supplied steam rate, stage temperatures and flows, etc.

Hillenbrand and Westerberg [6] developed the model to compute the utility consumption for multiple effect evaporator systems that exchange sensible heat with steam on both inside and outside the evaporation system. A Modified Grand Composite Curve (MGCC) is developed to plot the temperature for placing an evaporator Vs the maximum amount of such sensible heat that can be exchanged. The simplified model and MGCC are then used to find out the approximate best temperature at which to place a single evaporator effect.

Agarwal [7] carried out a theoretical study on energy conservation in multiple effect evaporator system, using **the** of material **balance, energy balance, heat transfer rate,** and boiling point rise in individual effects of an N- effect evaporators. He developed 4N nonlinear simultaneous algebraic equation for quadruple, quintuple and triple effect evaporators. He compared the values predicted by the model with that of plant values.

Goel [5] carried out the simulation studies of multiple effect evaporator used for the concentration of Kraft black liquor. The multiple effects consist of six effect and condensate flash tank. Equations based on the material and energy balance and heat transfer rate are developed. Gudmundson model is used for the computation of over all heat transfer coefficient .17 set of nonlinear non linear algebraic equations are developed and solve by using Broydon's method. He studied the effect of operating variable on steam economy, steam consumption and heating surface area of multiple effect evaporator.

Zain and **S. Kumar** [2] they carried out work on simulation of a triple effect evaporator used for concentrating the caustic soda solution. They develop empirical correlations for the enthalpies of steam condensate, and caustic soda solution. They shared their computational experience, which they encountered during the simulation of triple effect evaporator. It was found that numerical solution of model equations of a triple effect evaporator system by Newton-Raphson method should be done using double precision arithmetic in order obtain the converged solution. They also proposed a new arrangement of model equation, which reduces the number of equations.

Dangwal, Maheshwari and **Azad Veer** [11] studies the changes occur in evaporator plant after retrofitting and installing a free flow falling film evaporator in a mill. Comparing the same production level before and after retrofit of evaporation plant in LTV and addition of FFFF evaporator they found that reduction of L.P steam consumption per day of about 100 T. power saving of 1500 kWh/d and higher concentration being achieved from FFFF. Downtime in evaporator plant was reduced by 75% due to lesser cleaning time of evaporator tubes.

Mengistu [9] developed a steady state model of sextuple effect backward feed black liquor evaporator including heat recovery by flashing. 18 non-linear algebraic equations were developed and by using globally convergent method equation were solved. He compared the obtained result with the reported work of others and proposed that the evaporators with series condensate and product flashing arrangement have the best steam economy.

Ray and Singh [13] carried out study to design sextuple effect black liquor evaporator system for paper industry. They solved twelve nonlinear simultaneous equation based on study state mass and energy balances, heat transfer rate, equilibrium relationships and some physico-chemical / physico thermal properties of liquor. They used Newton-Raphson-Jacobian matrix method and method of gauss elimination as a numerical technique to solve the problem. They developed a generalized algorithm for the simulation of multiple effect evaporator system with backward feed arrangement.

Singh & et al [1] they performed a study on black liquor sextuple effect evaporator system for backward feed sequence which involves splitting of feed. They compared energy gain and steam economy when the flash from condensate and product are used with no splitting of feed in backward feed sequence. They developed a mathematical model based on the set of 17 non-linear simultaneous equation for steady state mass and energy balance, enthalpy balance heat transfer rate equations coupled with models for overall heat transfer coefficients and physico thermal properties of black liquor including boiling point rise calculations. They solved system of non-linear equations by using Newton-Raphson method with Jacobian matrix and method of Gauss elimination with partial pivoting with the aid of Hilbert norms. The procedure they adopted gave rapid convergence and estimates the energy consumption more accurately and efficiently.

2.2 Physico-chemo & thermal properties

Characteristics of the solids present in black liquor ensure efficient and continuous operation of the evaporator. The black liquor is a complex mixture of organic and inorganic chemicals. The proportion of organic compounds is in the range of 50-70 % depending upon the chemical composition of the species used, digester operating condition and pulp yield. The typical chemical composition of black liquor is given below.

Black liquor composition

Organic Compounds
Alkali lignin and thiolignin, Isoaccharinic acid, Low molecular weight polysaccharides Resin and fatty acid soaps, and Sugars

Inorganic Compounds	kg/m^3
Sodium hydroxide	4-8
Sodium sulphide	6-12
Sodium carbonate	6-15
Sodium thiosulphate	1-2
Sodium polysulphides	small
Sodium sulphate	0.5-1
Elemental sulphur	small
Sodium sulphite	small

The weak black liquor after digestion and washing steps is at 50 - 90°C with a pH of 10.5-13.0 and 12-15 % total dissolved solids concentration.

The properties of black liquor are necessary for the design, operation and simulation of the black liquor multiple effect evaporator. The major physico-thermal properties used for engineering design of the evaporator equipment are viscosity, thermal conductivity, specific heat capacity, density, boiling point rise and surface tension.

These properties would depend upon the nature and concentration of the liquor and are temperature dependent. Enormous volume of data has been reported about these properties for different type of liquors. It is generally found that data are proprietary in nature and strongly influenced by the type of raw materials, and pulping process parameters like temperature and concentration [21]. Thus, the equations and data used for the simulation purpose must carefully be selected and checked.

To derive a mathematical model for the evaporator system the physico thermal properties plays a great role, properties such as specific heat of the liquor, density of the liquor, boiling point elevation are very important.

Zaman, McNally and Fricke [15] they have studied vapor-liquid equilibrium and boiling point elevation of slash pine Kraft black liquors over a wide range of solid concentrations (up to 85%). It was found that boiling point elevation of black liquors is pressure dependent, and this dependency is more significant at higher solid concentrations. The boiling point elevation data at different solids contents (at fixed pressure) were correlated to the dissolved solids in black liquor as:

$$BPR=K*S/(1-S)$$

Where, S is the solid mass fraction and S/ (1-S) is the ratio of solids to water.
They studied that due to the solubility limit of some of the salts in black liquor, a change in the slope of the boiling point elevation as a function of the dissolved solids was observed at a concentration of around 65% solids. They have developed an empirical method to describe the boiling point elevation of black liquor as a function of pressure and solid mass fraction. They have correlated the boiling point elevation of slash pine black liquor quantitatively to the pulping variables, using different statistical procedure. The models predicted by them can be applied to determine the boiling point rise (and boiling point) of slash pine black liquors at processing conditions from the knowledge of pulping variables.

Boiling point elevation

The volatility of black liquor may be measured in terms of vapor pressure lowering or boiling point elevation. The boiling point elevation rise (BPR) is defined as the difference between the boiling point of black liquor and that of pure water at the same pressure
BPR is smaller at lower concentration and increase with increase in solids content. Pressure influences BPR to a small extent.
The **TAPPI** correlation for BPR in ^0F is given by:

$$BPR=41.4(C/100 + 0.1)^2$$

Ray, Bansal and Rao [14] they performed various experiments on the transport properties on Indian black liquors and found out that the specific gravity increases linearly with solid contents. They suggested that black liquor exhibits non Newtonian behavior while being handled in multiple evaporators particularly at higher solid contents. Based on experiments they found that the specific gravity increases linearly with an increase in the concentration of the total solids of black liquor obtained from different evaporators. They found that effect of temperature is marginal.

Viscosity

Viscosity is one of the most important properties controlling pressure drop, heat and mass transfer rates and mixing rates. The high viscosity of strong black liquor can adversely affect heat transfer rates. The viscosity is usually reduced by preheating to a temperature of 110 - 120 °C.
Gudmundson has given correlation for the determination of viscosity as a function of temperature and solids concentration based on the measured data of Kobe and McCormack.

$$\mu = \exp (a + b(C) + c(C)^2 + d(C)^3)$$

Where

$a=0.4717 - 0.02472(T) + 0.7059* 10^{-5} (T^2)$
$b=-0.06973- 0.5452*10^{-3}(T) + 0.1656*10^{-5}(T^2)$
$c=0.002046 + 0.3183*10^{-4}(T)-0.9761*10^{-7}(T^2)$
$d=0.5793*10^{-4} -0.6129*10^{-6}(T) + 0.1837*10^{-8}(T^2)$
C= total solids concentration in %
T= temperature in °C
μ = viscosity in cp

Density

Density varies with species of raw material, concentration and temperature. For wood black liquor the density is expressed as kg/m^3 in terms of temperature T (in ^0C), and total solids concentration % (C) by **Regestad** relation,

$\rho = 1007 + 6(C) - 0.495(T)$

Density of spent liquor is reported in terms of specific gravity referred to the density of water at a reference temperature. The specific gravity is measured in terms Degree Twaddell (Tw)

$Tw=200*(G-1.0)$

Where, G is specific gravity

Specific heat capacity

Specific heat of black liquor decreases with increase in total solids content and increase in temperature.
Specific heat (C_p) many researchers' proposed different correlations for specific heat

Harvin and Brown gave a correlation for a Kraft black liquor with an inorganic content at 29% of the solid the equation is given as:
C_p=l .246+14.4*e (-3)*(T)-(0.34342-l 1.52*e (-4)*(T))*S

Kobe and Sorenson also give the correlation in the range of 25 to 95 ^0C the relation is as follows:
C_p=0.98-0.52(S)

Regestad has also proposed a similar correlation:
C_p= (1.0-0.0054*S) (4187)
Where C_p= specific heat capacity, J/kg/ ^0C

Thermal conductivity

Thermal conductivity decreases almost linearly with increase in temperature and increase in concentration.

Harvin and Brown have reported data for thermal conductivity of black liquor
$k = [0.504-0.282(C) +1.35*e^{-3}T] *1.163$
Where k = thermal conductivity, W/(m*K)

Properties of water

In addition to the properties of black liquor, properties of water substance like enthalpy of saturated steam, enthalpy of saturated water, latent heat of vaporization and latent heat of condensation is required for simulation model. These properties are correlated to a polynomial function using the data from steam table. The polynomials are adjusted using MS - EXCEL 97 with a correlation factor greater than 0.99.

Liquor enthalpy h_{Li} is calculated using Regestad correlation for specific heat capacity as:

$$h_{Li} = [4187\text{-}22.6098 * X_f / l_i] * T_{Li}$$

Enthalpy of saturated water

$h_s = 1803.8 - 4209.8(T)$

Enthalpy of saturated steam

$H = 2497500 + 1993.5(T) - 2.2(T^2)$

Latent heat of vaporization of water

$\lambda = 2499000 - 2208.2(T) - 2.2(T^2)$

2.3 Parameter Estimation:

The mathematical models usually represent the parameters with physical significance. The true value of parameter in nature is required for a true and correct model, because of the generally imprecise nature of measurements we can never hope to determine the true value with absolute certainty. Also due to the random nature of the error in measurements the value of parameter that best fits one series of measurements differs from the value that fits another series. However, we can look for procedure to obtain values of the parameter that not only fit the data well, but also come on the average fairly close to the true value and do not vary excessively from one set of experiments to the next and also it can estimate the parameters (known & unknown) properly, it can also considered those data which are missing or incomplete data.

Thus parameter estimation is a procedure used to analyze the data and to produce a reliable set of data that is, one that approximate real (true) values. As the real values are not known, this means that we can only judge data reliability on the basis of how realistic and consistent the data are. Several researchers have carried out the study on parameter estimation of steady state processes.

Graham and Stevenson [16] studied the effects of experimental error on the values of the parameter estimates of a general four-parameter kinetic model, which was investigated by conducting a series of hypothetical experiments at different levels of an imposed independent normal error. The parameters used were obtained from the chlorination of niobium oxy chloride with phosgene. The values of the parameters estimated from the hypothetical data varied from the true values according to the magnitude of the experimental error and the particular parameter in the model. The

effects of such errors on the convergence of a sequential experimental design were reflected in uniform parameter convergence for small errors and a lack of uniform convergence for large error values. However, predicted values of the dependent variable, obtained from the kinetic model using the final parameter estimates, were within the limits of the imposed experimental error at all levels investigated.

Crowe [17] presented a theoretical approach for solving the problem of reconciling flow measurement data for the nonlinear case. He used matrix projection method and constructed two projection matrices in order to decompose the problem in to three sub problems to be solved in sequence. The first matrix eliminates all unmeasured component flow rates and concentration from the equations; the second one then removes the unmeasured total flow rates. The unmeasured variables are iteratively determined starting with the guessed values of unmeasured flow rates.

Banvolgyi, et al [18] they give the procedure for the identification of a steady state computer model of a multistage countercurrent evaporator, by fitting the model to set of observed input and selected output variables by the least square principle. Principal component analysis is used in order to rank the parameter according to their influence on input and output relation.

Crowe [19] gave a new model, derived for recursive prediction of the changes of the objective function, and of the statistical tests for the measurements, which would result from the deletion of suspect measurements. An algorithm for the detection of suspect sets of gross errors, whose deletion leads to acceptable values of all statistical tests and process flow rates, is proposed and illustrated.

Aldrich et al. [20] commented that the reliability of the data, which characterize the behavior of a plant, is critical to the effective monitoring and improvement off plant performance. It is thus essential that gross errors in these data, which can arise from measurement problems or inadequate mathematical models, are detected and eliminated before the performance of the plant is evaluated .they, applied another type of neural nets, such as probabilistic neural nets (PNN) learning vector quantization neural net (LVQNN) for the detection of gross errors on nonlinearity constrained system and compared with back propagation neural net they proposed that other neural nets are superior than back propagation neural nets.

2.4 Solution Technique:

Gudmundson's model [24]

The overall heat transfer coefficient in the individual effects Ui of the multiple effect evaporator are obtained from Gudmundson's model. The algorithm described below is shown in the flow chart subsequently.

Algorithm

Step 1: determine physical properties μ_m, C_p, ρ, k based on liquor concentration and temperature.

Step 2: determine liquor feed rate per unit cross sectional area, FD, kg/m^2s.

Step 3: determine the heat flux $q(W/m^2)$ based on evaporation rate from a body.

Step 4: determine the values of the correction factor C_μ and C_T

$C_\mu = P_6+P_7\mu_m+P_8\mu_m^2+P_9\mu_m^3$ for $O<\mu_m<15$ (1)

$= P_{10}+P_{11}\log(\mu_m)$ (2)

$C_T = P_{10}+P_{11}t+P_{12}t^2+P_{13}t^3$ (3)

Step 5: determine the value of T_{max}

$T_{max} = P_{22} + \{P_{23}+P_{24}\} FD + (P_{25}+P_{26}t)\log(\mu_m) + P_{27}t$ (4)

Step 6: compute bound1 and bound2

bound1 $= P_{19}+P_{20}\log(\mu_m) + P_{21}t$ (5)

bound2 $= P_{28}+P_{29}\log(\mu_m) + P_{30}t$ (6)

(for both cases $\mu_m = 1.0$ if $\mu_m<1.0$)

Step 7: determine the value of C_F and C

$Z = 1 + P_{16}\text{Tanh}((q-\text{boundl})/P_{17})\, T_{sub}^{P18}$ (7)

$T_{sub} = T_{max} - T_{in}$ ($T_{sub} = 0.0$ if $T_{sub} < 0.0$)

$C_F = Z + (P_{14}+P_{15}Z)FD$ (8)

$C = C_\mu C_T C_F$ (9)

Step 8: if the liquid feed is superheated, extra vapor is flashed off in the tube which is equivalent to a certain increase in the heat flux. The increase in the heat flux is:

$\Delta q = T_{sup}\, C_p\, FD \prod D^2/4.C_{eff}/1.321$ (10)

the superheating T_{sup} is defined by

$T_{sup} = T_{in} - T_{max}$ (11)

The efficiency factor C_{eff} shows the efficiency of the flashed vapor in comparison with the corresponding increase of the heat flux.
It usually lies between 0 and 1, and generally taken to be 0.61.

Step 9: determine the overall heat transfer coefficient

Case 1: if q > bound2

$U = P_1 C^{P2}(P_3 q + P_4 q^2 + P_5 q^3)$ (12)

for q>20000 , take q=20000

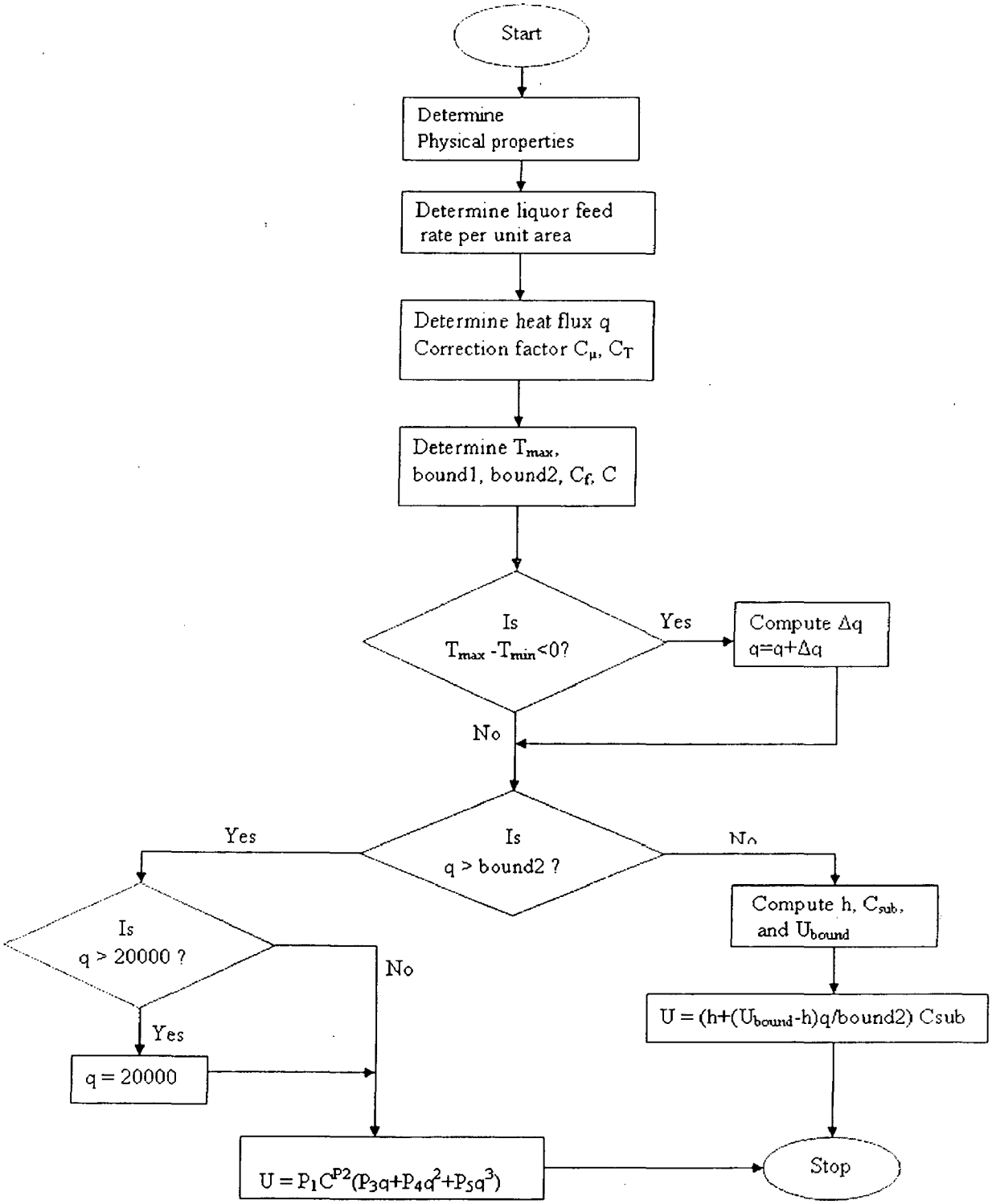


Figure 1: Gudmundson's model flowchart

Case2: if q < bound2

$$U = (h + (U_{bound} - h)\, q / bound2)\, C_{sub} \quad (13)$$

h can be computed by **Dittus-Boelter** correlations

$$h = 0.023 Re^{0.8} Pr^{0.4} k/D \quad (14)$$

$$Csub = 1 + P_{31} T_{sub} ((bound2 - q)/bound2)^2 \quad (15)$$

Ubound is the value of U with the existing parameter values but with q assumed to be equal to bound2

$$U_{bound} = P_1 C^{P2} (P_3 (bound2) + P_4 (bound2)^2 + P_5 (bound2)^3) \quad (16)$$

$T_{sub} = 0.0$ if $Tsub < 0.0$

Values of the constants P_1 to P_{31} as used in Gudmundson's model are computed from experimental data.

Holland [22] proposed the Broyden method for solving the system of non linear equation for the simulation of multiple effect evaporators.

J Lee and W. K. Lee [21] proposed the least square updating formula for use in quasi Newton method to solve systems of nonlinear equations they proposed two updating schemes the method of exponential weighted least square to quasi Newton and Newton iteration equation respectively. And the proposed method is compared with Broyden method and found there method are more reliable than Broyden's method.

Paloschi [23] presented a hybrid algorithm that combines a Newton iteration process with the continuation code P1TCON. The proposed algorithm is tested by using standard sets of example 36 different problems combined with different scaling conditions and initial points total 327 problems solved .the algorithm solves more than 98% of the problem.

3. MODEL FORMULATION

Number of researchers has done the mathematical modeling of black liquor multiple effect evaporator of backward feed. Different types of evaporation system have been suggested for improving the performance of the evaporator. Methods like condensate flashing, splitting of feed and splitting of steam have been suggested to improve the thermal efficiency of the evaporators.

In this portion a mathematical model is formulated for sextuple effect evaporator with condensate and product flashing.

3.1 Detailed Model formulation for sextuple effect backward feed evaporator with product and condensate flash

The schematic flow diagram is shown in Fig. 3.1 below.

For the illustration purpose, the model equations for the first effect, product flash tank, first condensate flash tank, second effect and second condensate flash tank of the evaporator are derived hereunder.

3.1.1 Material and energy balances for the first effect

1. Total material balance equations

$$L_2 = L_1 + V_1 \qquad (3.1)$$

2. Solute material balance

$$L_2X_2=L_1X_1 \qquad (3.2)$$

3. Enthalpy balance

$$V_s\lambda o + L_2h_2 = L_1h_1 + V_1H_1 \qquad (3.3)$$

4. Heat transfer equation

$$V_s\lambda o = U_1A_1 (T_0-t_1) \qquad (3.4)$$

The boiling temperature of the liquor differs from the saturation temperature by the boiling point rise (BPR):

$$t_1 = T_1 + BPR_1 \qquad (3.5)$$

substitution of values of V_1 from Eq. (3.1) into Eq. (3.3) and t_1 from Eq. (3.5) into Eq. (3.4) and rearranging Eq. (3.3) gives

$$V_s\lambda o + L_2h_2 = L_1h_1 + (L_2-L_1) H_1 + L_2h_1 - L_2h_1$$

$$V_s\lambda o + L_2 (h_2 -h_1) = (L_2 -L_1) (H_1 - h_1) \qquad (3.6)$$

$$V_s\lambda o =U_1A_1 (T_0-T_1-BPR_1) \qquad (3.7)$$

Functional notations f_1 and f_2 are defined as follows:

$$f_1 = V_s\lambda o + L_2 (h_2-h_1) - (L_2-L_1) (H_1 - h_1) \qquad (3.8)$$

$$f_2 = V_s\lambda o -U_1A_1 (To-T_1-BPR_1) \qquad (3.9)$$

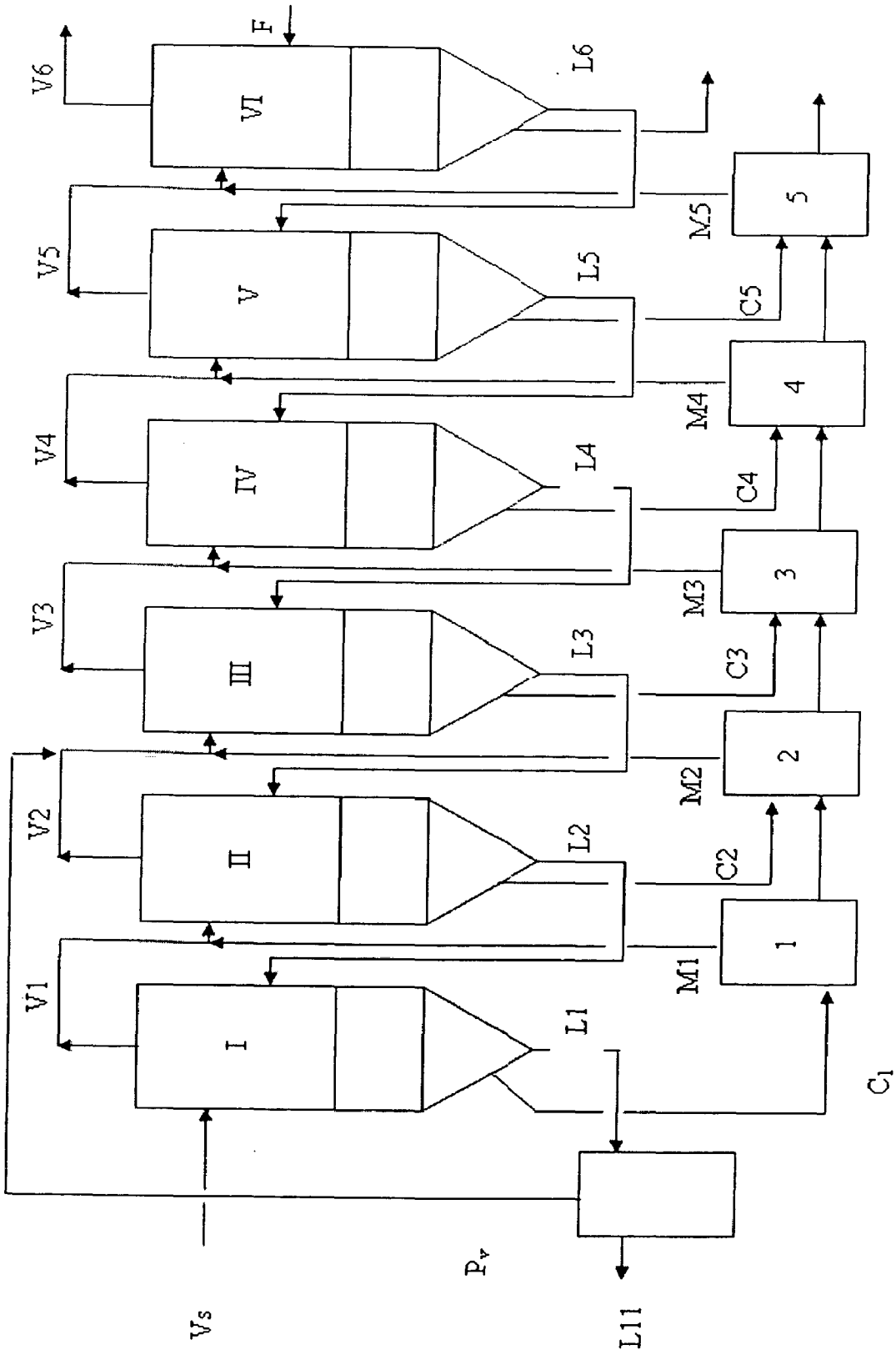


Figure 2: Sextuple effect backward feed evaporator with product and series condensate flash

3.1.2 Material and energy balances for the product flash tank

1. Total material balance equations

$$L_I = L_{II} + P_v \quad (3.10)$$

2. Solute material balance

$$L_I X_I = L_{II} X_{II} \quad (3.11)$$

3. Enthalpy balance

$$L_I h_I = L_{II} h_{II} + P_v H_2 \quad (3.12)$$

Substituting for P_v in Eq. (3.12) from Eq. (3.10) and rearranging.

$$L_I h_I = L_{II} h_{II} + (L_I - L_{II})\, H2 \quad (3.13)$$

$$L_I (h_I - H_2) = L_{II} (h_{II} - H_2) \quad (3.14)$$

Let a functional notation f_3 be defined as

$$f_3 = L_I (h_I - H_2) - L_{II} (h_{II} - H_2) \quad (3.15)$$

3.1.3 Material and energy balances for the first condensate flash tank

1. Total material balance equations

$$C_I = M_I + C_{II} \quad (3.16)$$

2. Enthalpy balance

$$C_I h_{so} = M_I H_I + C_{II} h_{sI} \quad (3.17)$$

Combining the two equations in one form,

$$C_I h_{so} = M_I H_I + (C_I - M_I)\, h_{sI} \quad (3.18)$$

But $C_I = V_s$, hence

$$V_s h_{so} = M_I H_I + (V_s - M_I)\, h_{sI}$$

$$V_s (h_{so} - h_{sI}) = M_I (H_I - h_{sI})$$

$$V_s (h_{so} - h_{sI}) = M_I \lambda_I$$

Where $\lambda_I = Hi - h_{sI}$ is the latent heat of vaporization of water at pressure P_I

Defining f_4

$$f_4 = V_s (h_{s0} - h_{sI}) - M_I \lambda_I \quad (3.19)$$

3.1.4 Material and energy balances for the second effect

1. Total material balance equations

$$L_3 = L_2 + V_2 \quad (3.20)$$

2. Solute material balance

$$L_3 X_3 = L_2 X_2 \quad (3.21)$$

3. Enthalpy balance

$$L_3 h_3 + (M_I + V_I)(\lambda_I + C_{pv}\, BPR_I) = L_2 h_2 + V_2 H_2 \quad (3.22)$$

4. Heat transfer equation

$$(M_I + V_I)(\lambda_I + C_{pv}\, BPR_I) = U_2 A_2 (t_I - 1_2) \quad (3.23)$$

Where the boiling temperature of the liquor t_I and t_2 is given by

$$t_I = T_I + BPR,$$

$$t_2 = T_2 + BPR_2$$

Let us define $\lambda_{I3} = \lambda + C_{pv}\, BPR$

Rearranging Eq.(3.22) by solving for V_1 from Eq.(3.1) and for V_2 from Eq.(3.20) and rearranging

$L_3h_3 + (L_2-L_1+ M_1)\lambda_{B1} = L_2h_2 + (L_3-L_2)H_2$

$L_3h_3 - L_3h_2 + (L_2 - L_1+ M_1)\lambda_{B1} = L_2h_2 - L_3h_2+ (L_3 - L_2)H_2$

$L_3(h_3-h_2) + (L_2 -L_1+M_1)\lambda_{B1} = (L_3-L_2)(H_2-h_2)$

But $H-h = \lambda + C_pv\ BPR = \lambda_B$

Defining f_5

$$f_5 = L_3(h_3-h_2) + (L_2 -L_1 + M_1)\lambda_{B1} - (L_3-L_2)\lambda_{B2} \quad (3.24)$$

From Eq. (3 23), doing similar simplification, function f_6 can be defined as follows:

$$f_6 = (L_2 -L_1 +M_1)\lambda_{B1} - U_2A_2(T_1 + BPR_1 - T_2 - BPR_2) \quad (3.25)$$

3.1.5 Material and energy balances for the second condensate flash tank

1. Total material balance equations

$$C_2+C_{11} = M_2+C_{22} \quad (3.26)$$

2. Enthalpy balance

$$(C_2 + C_{11})h_{s1} = M_2H_2 + C_{22}h_{s2} \quad (3.27)$$

Combining the two equations in one form,

$(C_2 +C_{11})h_{s1} = M_2H_2 + (C_2+C_{11}-M_2)h_{s2}$

Substituting,

$C_2 = V_1+M_1 = L_2 - L_1 + M_1$, and $C_{11} = C_1 - M_1 = V_s - M_1$

We get,

$(L_2- L_1 + M_1 + V_s - M_1)h_{s1} = M_2H_2 + (L_2 - L_1 + M_1 + V_s - M_1 - M_2)h_{s2}$

$(L_2 - L_1 + V_s)h_{s1} = M_2H_2 + (L_2- L_1 + V_s - M_2)h_{s2}$

$(L2 - L_1 + V_s)(h_{s1} - h_{s2}) = M_2(H_2-h_{s2})$

$(L2 - L_1 + V_s)(h_{s1} - h_{s2}) = M_2\lambda_2$

f_7 is now defined as:

$$f_7 = (L_2 - L_1 + V_s)(h_{s1} - h_{s2}) - M_2\lambda_2 \quad (3.28)$$

The following scaling procedure has been applied to reduce the magnitude of the terms appearing in the functional equations and matrices for the computational purposes,
All temperatures are expressed as a fraction of the steam temperature T_o, all flow rates as a fraction of feed rate F, and the area in the fraction form of A/(50F).

Defining $w_i = T_i / To$ $\quad a_i = A_i/(50F)$ $\quad v_s = V_s/F$

$l_i = L_i/F$ $\quad m_i = M_i/F$

Each functional equation is divided by the product $F\lambda o$ and the new functional relation is denoted by g where

$g=f/(F\lambda o)$

By the assumption of equal heat transfer area, we have

$a_1 = a_2 = a_3 = a_4 = a_5 = a_6 = a$

With this scaling procedure, Esq. (3.8), (3.9), (3.15), (3.19), (3.24), (3.25), and (3 28) are rewritten in the following reduced form:

First effect

$g_1 = v_s + l_2 (h_2-h_1)/ \lambda o - (l_2 - l_1) \lambda_{B1} / \lambda o$ (3.29)

$g_2 = v_s - 50U_1a / \lambda o (To - To\ w_1 - BPR_1)$ (3.30)

Product flash tank

$g_3 = l_1 (h_1 - H_2) / \lambda o - m_1 \lambda_1 / \lambda o$ (3.31)

First condensate flash tank

$g_4 = v_s (h_{s0} - h_{s1}) / \lambda o - m_1 \lambda_1 / \lambda o$ (3.32)

Second effect

$g_5 = l_3 (h3 - h_2)/ \lambda o + (l_2 - l_1 + m_1) \lambda_{B1}/ \lambda o - (l_3 - l_2) \lambda_{B2} / \lambda o$ (3.33)

$g_6 = (l_2 - l_1 + v_s) \lambda_{B1}/ \lambda o - 50U_2a/ \lambda o (Tow_1 + BPR_1 - Tow_2 - BPR_2)$ (3.34)

Second condensate flash tank

$g_7 = (l_2 - l_1 + v_s) (h_{s1} - h_{s2}) / \lambda o - m_2 \lambda_2 / \lambda o$ (3.35)

Similarly for the subsequent effects and condensate flash tanks, the following relations can be obtained:

Third effect

$g_8 = l_4 (h_4-h_3)/ \lambda o + (l_3 - l_2 + m_2 + l_1 - l_{11}) \lambda_{B2} / \lambda o - (l_4 - l_3) \lambda_{B3} / \lambda o$ (3.36)

$g_9 = (l_3 - l_2 + m_2 + l_1 - l_{11}) \lambda_{B2}/\lambda_0 - 50U_3a/\lambda o (T_o w_2 + BPR_2 - Tow_3 - BPR_3)$ (3.37)

Third condensate flash tank

$g_{10} = (l_3 - l_{11} + v_s) (h_{s2}-h_{s3}) / \lambda o - m_3 \lambda_3 / \lambda o$ (3.38)

Fourth effect

$g_{11} = l_5(h_5-h_4)/ \lambda o + (l_4 - l_3 + m_3) \lambda_{B3} / \lambda o - (l_5 - l_4) \lambda_{B4}/ \lambda o$ (3.39)

$g_{12} = (l_4 - l_3 + m_3) \lambda_{B3}/ \lambda o - 50U_4a/ \lambda o (T_o w_3 + BPR_3 - T_o w_4 - BPR_4)$ (3.40)

Fourth condensate flash tank

$g_{13} = (l_4 - l_3 + v_s) (h_{s3} - h_{s4})/ \lambda o - m_4 \lambda_4 / \lambda o$ (3.41)

Fifth effect

$g_{14} = l_6 (h6 - h_5) / \lambda o + (l_5 - l_4 + m_4) \lambda_{B4}/ \lambda o - (l_6 - l_5) \lambda_{B5}/ \lambda o$ (3.42)

$g_{15} = (l_5 - l_4 + m_4) \lambda_{B4}/ \lambda o - 50U_5a/ \lambda o (T_o w_4 + BPR_4 - T_o w_5 - BPR_5)$ (3.43)

Fifth condensate flash tank

$g_{16} = (l_5 - l_{11} + v_s) (h_{s4}-h_{s5})/\lambda o - m_5 \lambda_5 / \lambda o$ (3.44)

Sixth effect

$g_{17} = l_7 (h_7-h_6)/ \lambda o + (l_6 - l_5 + m_5) \lambda_{B5}/ \lambda o - (l_7 - l_6) \lambda_{B6}/ \lambda o$ (3.45)

$$g_{18} = (l_6 - l_5 + m_5)\,\lambda_{B5}/\lambda o - 50U_6a/\lambda o\,(T_o w_5 + BPR_5 - T_o w_6 - BPR_6) \quad (3.46)$$

The concentration in each effeci is calculated from the component material balance for the solute:

$$X_f = liXi \quad (3.47)$$

3.2 Solution Procedure:

We have 18 nonlinear equations in the model developed for the system in section 3.2. The different variables involved in the model equations are classified below.

a) Design variables:

Feed rate	$F\ (L_7)$	Feed concentration	$X_f\ (X_7)$
Feed temperature	$Tf\ (T_7)$	Last effect sat. temperature	T_6
Product concentration	X_{11}	Steam temperature	T_s

b) Specified variables: $U_1, U_2, U_3, U_4, U_5, U_6$

c) Unknowns variables: $v_s, l_1, l_2, l_3, l_4, l_5, l_6$
w_1, w_2, w_3, w_4, w_5
m_1, m_2, m_3, m_4, m_5
$x_1, x_2, x_3, x_4, x_5, x_6,$
and a

But x is related to l by Eq. (3.47), hence there are exactly eighteen unknowns. So we have 18 nonlinear equations and 18 unknowns. These can be solved simultaneously by nonlinear solving method.

Newton -Raphson method is quite often used to solve the system of nonlinear equations. This method converges only if it starts from an initial estimate of the variables close enough to the solution. Furthermore it requires the evaluation of Jacobian matrix at each iteration which can be computationally expensive for many practical problems.

Quasi Newton Method

The globally convergent method of QN-least square updating is used to solve the nonlinear equations more effectively and reliably. This method is tested and found to be more reliable and efficient than the well-known Broyden method.

Algorithm

1. Approximate the Jacobian matrix of a set of nonlinear equations F(X) by a matrix B satisfying

$$Y_k = B_{k+1}S_k \quad (3.48)$$

Where

$$Y_k = F(X_{k+1}) - F(X_k)$$

$Sk = X_{k+1} - X_k$

2. Update X and an approximation to the Jacobian matrix

$X_{k+1} = X_k - B^{-1}_k F(X_k)$ (3.49)

$B_{k+1} = B_k + (Y_k - B_k S_k) S^T_k P_k / (S^T_k P_k S_k)$ (3.50)

$Bo = F'(Xo)$

$P_{k+1} = P_k - P_k S_k S^T_k P_k / (\alpha^{k+3} + S^T_k P_k Sk)$ (3.51)

$Po = I$

Where $0<\alpha<1$,

The initial estimate of the Jacobian matrix Bo is calculated using the forward finite-difference approximation. The initial Po is the identity matrix I.

3. Check $\sum_{n}^{1} F^2(X)$ approaches zero

This expression goes to zero only if each of the functional relation Fi (X) goes to zero.

For the given case n=18, the step size h for the approximation of the Jacobian matrix is taken 0.00001. The simulation program is written in C++ based on the methodology shown in the flowchart shown in figure 6.

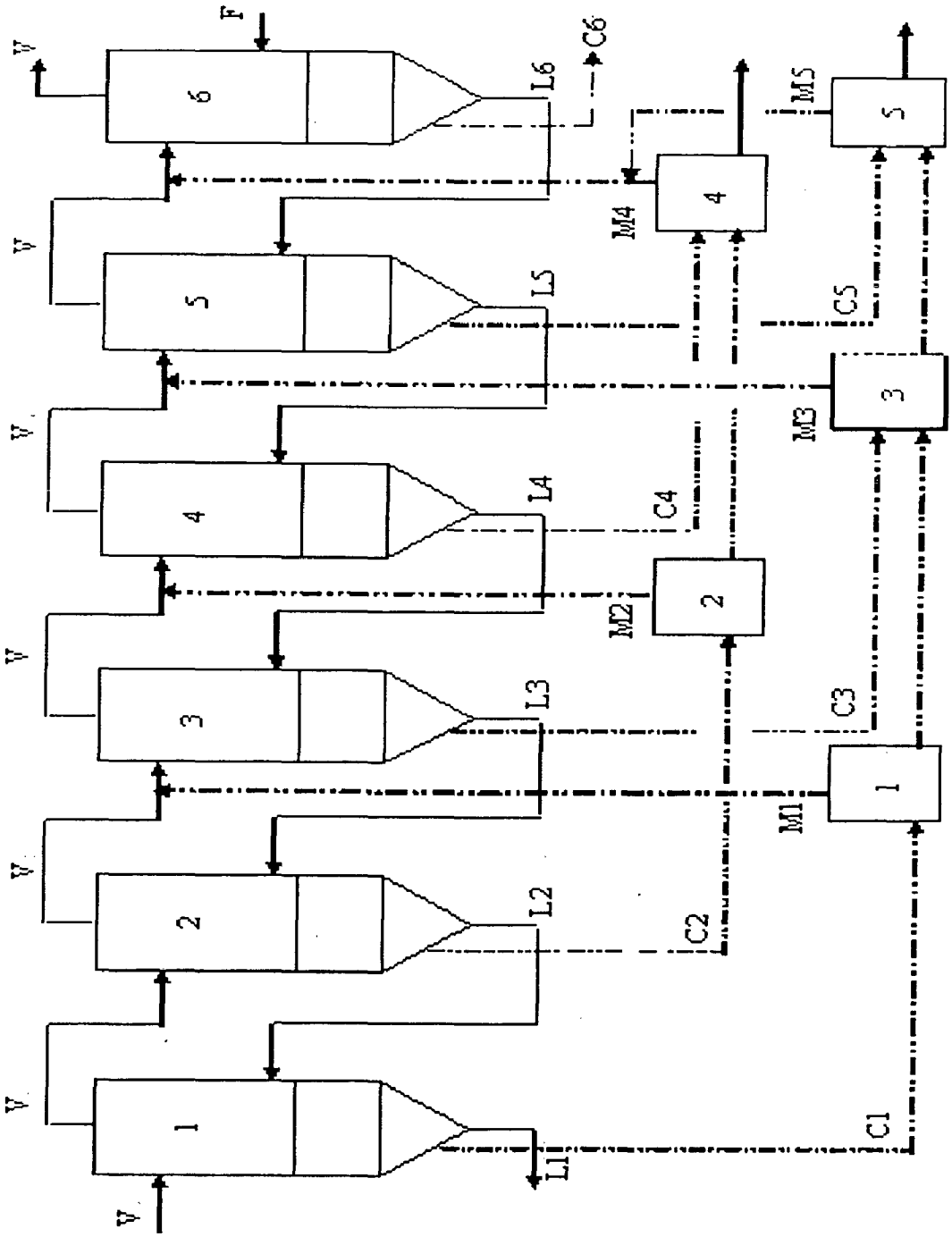


Figure 3: Evaporator system with alternate condensate flash

3.3 Model for sextuple effect backward feed evaporator with alternate condensate flashing

First effect

$g_1 = v_s + l_2(h_2-h_1)/\lambda o - (l_2-l_1)\,\lambda B_1 / \lambda o$ (1)

$g_2 = v_s - 50U_1a/\lambda o\,(To - To\,w_1 - BPR_1)$ (2)

First condensate flash tank

$g_3 = v_s\,(h_{s0} - h_{s1})/\lambda o - m_1\,\lambda_2 / \lambda o$ (3)

Second effect

$g_4 = l_3(h_3-h_2)/\lambda o + (l_2 - l_1)\,\lambda B_1/\lambda o - (l_3-l_2)\,\lambda B_2/\lambda o$ (4)

$g_5 = (l_2 - l_1)\,\lambda B_1/\lambda o - 50U_2a/\lambda o\,(Tow_1 + BPR_1 - To\,w_2 - BPR_2)$ (5)

Second condensate flash tank

$g_6 = (l_2 - l_1)\,(h_{s1}-h_{s3})/\lambda o - m_2\,\lambda_3/\lambda o$ (6)

Third effect

$g_7 = l_4(h_4-h_3)/\lambda o + (l_3 - l_2 + m_1)\,\lambda B_2/\lambda o - (l_4-l_3)\,\lambda B_3 / \lambda o$ (7)

$g_8 = (l_3 - l_2 + m_1)\,\lambda B_2/\lambda o - 50U_3a/\lambda o\,(Tow_2 + BPR_2 - To\,w_3 - BPR_3)$ (8)

Third condensate flash tank

$g_9 = (l_3 - l_2 + v_s)(h_{s2}-h_{s4})/\lambda o - m_3\,\lambda_4/\lambda o$ (9)

Fourth effect

$g_{10} = l_5(h_5-h_4)/\lambda o + (l_4 - l_3 + m_2)\,\lambda B_3/\lambda o - (l_5-l_4)\,\lambda B_4/\lambda o$ (10)

$g_{11} = (l_4 - l_3 + m_2)\,\lambda B_3/\lambda o - 50U_4a/\lambda o\,(T_o w_3 + BPR_3 - To\,w_4 - BPR_4)$ (11)

Fourth condensate flash tank

$g_{12} = (l_4 - l_3 + l_2 - l_1)\,(h_{s3}-h_{s5})/\lambda o - m_4\lambda_5/\lambda o$ (12)

Fifth effect

$g_{13} = l_6\,(h_6-h_5)/\lambda o + (l_5 - l_4 + m_3)\,\lambda B_4/\lambda o - (l_6-l_5)\,\lambda B_5/\lambda o$ (13)

$g_{14} = (l_5 - l_4 + m_3)\,\lambda B_4/\lambda o - 50U_5a/\lambda o\,(Tow_4 + BPR_4 - Tow_5 - BPR_5)$ (14)

Fifth condensate flash tank

$g_{15} = (l_5 - l_4 + l_3 - l_2 + v_s)\,(h_{,4} - h_{s5})/\lambda o - m_5\,\lambda_5/\lambda o$ (15)

Sixth effect

$g_{16} = l_7\,(h_7 - h_6)/\lambda o + (l_6 - l_5 + m_4 + m_5)\,\lambda B_5/\lambda o - (l_7 - l_6)\,\lambda B_6 / \lambda o$ (16)

$g_{17} = (l_6 - l_5 + m_4 + m_5)\,\lambda B_5/\lambda o - 50U_6a/\lambda o\,(T_o w_5 + BPR_5 - To\,w_6 - BPR_6)$ (17)

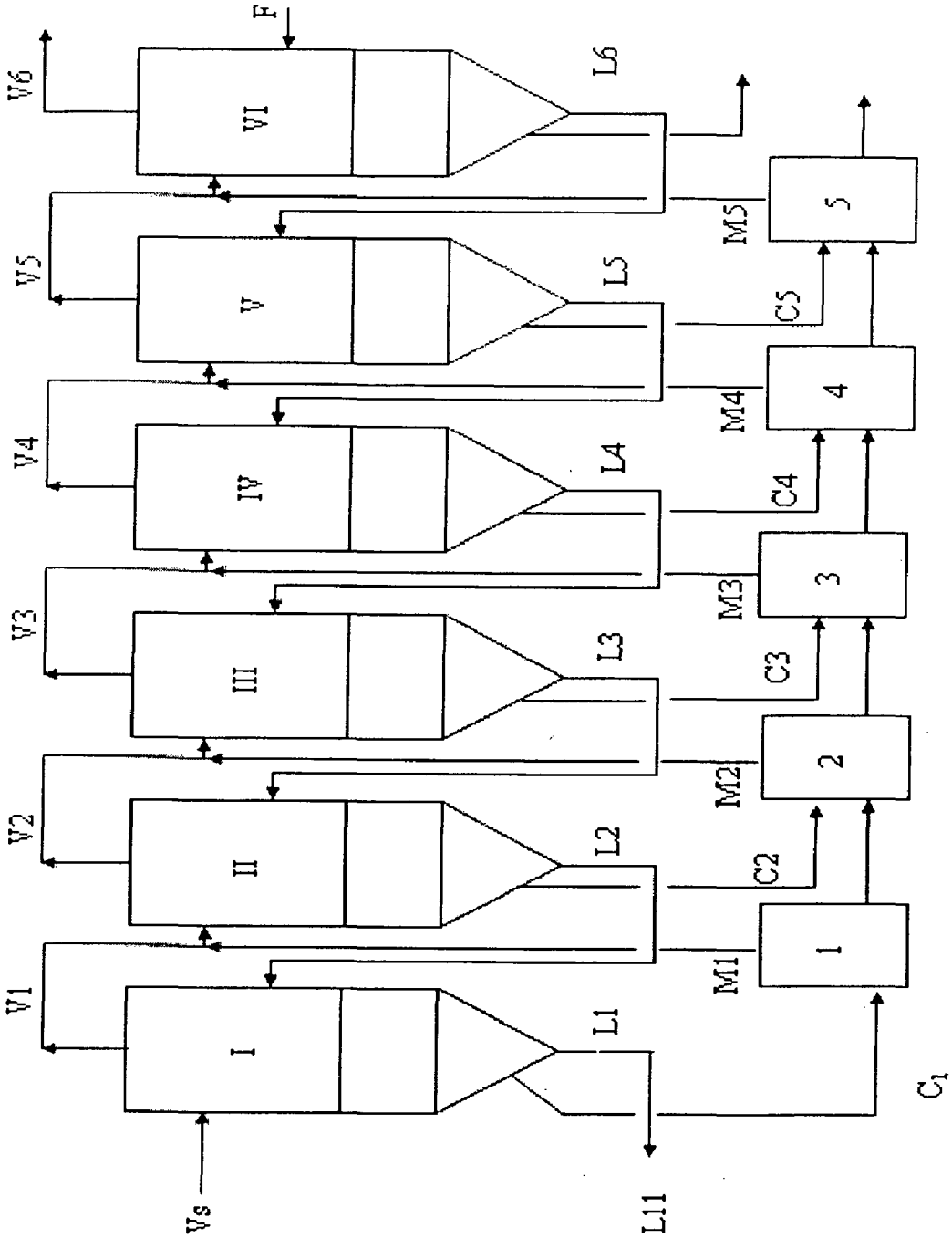


Figure 4: Evaporator system with series condensate flash

3.4 Model for sextuple effect backward feed evaporator with series condensate flashing

First effect

$g_1 = v_s + l_2(h_2-h_1)/\lambda o - (l_2-l_1)\lambda B_1 / \lambda o$ (1)

$g_2 = v_s - 50U_1a/\lambda o\ (To - To\ w_1 - BPR_1)$ (2)

First condensate flash tank

$g_3 = v_s\ (h_{s0} - h_{s1})/\lambda o - m_1\ \lambda_1 / \lambda o$ (3)

Second effect

$g_4 = l_3(h_3-h_2)/\lambda o + (l_2 - l_1 + m_1)\lambda B_1/\lambda o - (l_3-l_2)\lambda B_2/\lambda o$ (4)

$g_5 = (l_2 - l_1 + m_1)\lambda B_1/\lambda o - 50U_2a/\lambda o\ (Tow_1 + BPR_1 - To\ w_2 - BPR_2)$ (5)

Second condensate flash tank

$g_6 = (l_2 - l_1 + v_s)\ (h_{s1}-h_{s2})/\lambda o - m_2\ \lambda_2/\lambda o$ (6)

Third effect

$g_7 = l_4(h_4-h_3)/\lambda o + (l_3 - l_2 + m_2)\lambda B_2/\lambda o - (l_4-l_3)\lambda B_3 / \lambda o$ (7)

$g_8 = (l_3 - l_2 + m_2)\lambda B_2/\lambda o - 50U_3a/\lambda o\ (Tow_2 + BPR_2 - To\ w_3 - BPR_3)$ (8)

Third condensate flash tank

$g_9 = (l_3 - l_1 + v_s)(h_{s2}-h_{s3})/\lambda o - m_3\ \lambda_3/\lambda o$ (9)

Fourth effect

$g_{10} = l_5(h_5-h_4)/\lambda o + (l_4 - l_3 + m_3)\lambda B_3/\lambda o - (l_5-l_4)\lambda B_4/\lambda o$ (10)

$g_{11} = (l_4 - l_3 + m_3)\lambda B_3/\lambda o - 50U_4a/\lambda o\ (T_o w_3 + BPR_3 - To\ w_4 - BPR_4)$ (11)

Fourth condensate flash tank

$g_{12} = (l_4 - l_1 + v_s)\ (h_{s3}-h_{s4})/\lambda o - m_4\lambda_4/\lambda o$ (12)

Fifth effect

$g_{13} = l_6\ (h_6-h_5)/\lambda o + (l_5 - l_4 + m_4)\lambda B_4/\lambda o - (l_6-l_5)\lambda B_5/\lambda o$ (13)

$g_{14} = (l_5 - l_4 + m_4)\lambda B_4/\lambda o - 50U_5a/\lambda o\ (Tow_4 + BPR_4 - Tow_5 - BPR_5)$ (14)

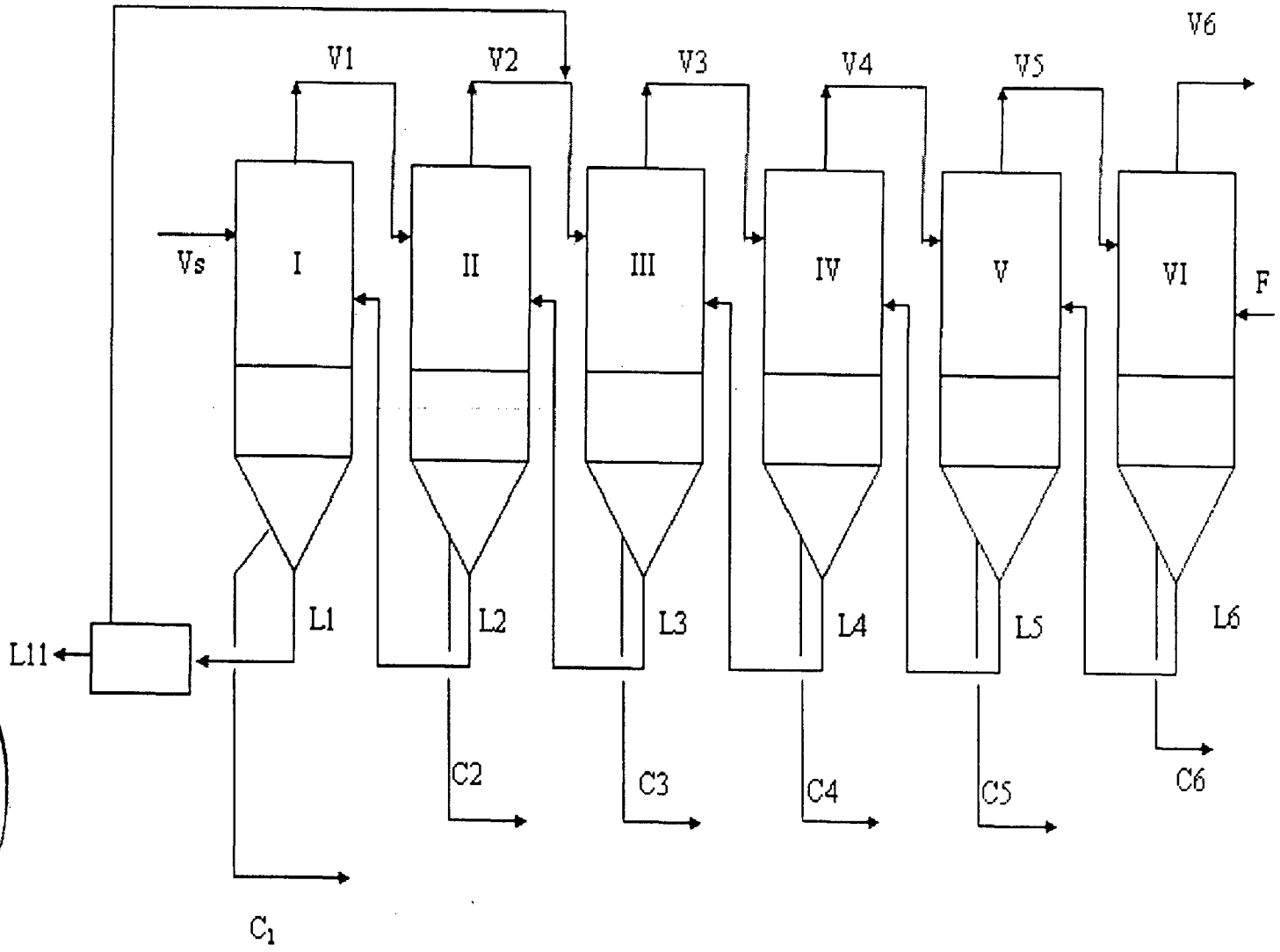
Fifth condensate flash tank

$g_{15} = (l_5 - l_1 + v_s)\ (h_{,4} - h_{s1})/\lambda o - m_5\ \lambda_5/\lambda o$ (15)

Sixth effect

$g_{16} = l_7\ (h_7 - h_6)/\lambda o + (l_6 - l_5 + m_5)\lambda B_5/\lambda o - (l_7 - l_6)\lambda B_6 / \lambda o$ (16)

$g_{17} = (l_6 - l_5 + m_5)\lambda B_5/\lambda o - 50U_6a/\lambda o\ (T_o w_5 + BPR_5 - To\ w_6 - BPR_6)$ (17)



CENTRAL LIBRARY
G12659

Figure 5: Evaporator system with product flash

3.5 Model for sextuple effect backward feed evaporator with product flashing

First effect

$g_1 = v_s + l_2(h_2-h_1)/\lambda o - (l_2-l_1)\,\lambda B_1 / \lambda o$ (1)

$g_2 = v_s - 50U_1a/\,\lambda o\,(To - To\,w_1 - BPR_1)$ (2)

Product flash tank

$g_3 = l_1(h_1-H_2)/\lambda_0 - l_{11}(h_{11}-H_2)/\,\lambda_0$ (3)

Second effect

$g_4 = l_3(h_3-h_2)/\lambda o + (l_2 - l_1)\,\lambda B_1/\,\lambda o - (l_3-l_2)\,\lambda B_2/\,\lambda o$ (4)

$g_5 = (l_2 - l_1)\,\lambda B_1/\,\lambda o - 50U_2a/\,\lambda o\,(Tow_1 + BPR_1 - To\,w_2 - BPR_2)$ (5)

Third effect

$g_6 = l_4(h_4-h_3)/\,\lambda o + (l_3 - l_2 + l_1 - l_{11})\,\lambda B_2/\,\lambda o - (l_4-l_3)\,\lambda B_3 / \lambda o$ (6)

$g_7 = (l_3 - l_2 + l_1 - l_{11})\,\lambda B_2/\,\lambda o - 50U_3a/\,\lambda o\,(T_0w_2 + BPR_2 - To\,w_3 - BPR_3)$ (7)

Fourth effect

$g_8 = l_5(h_5-h_4)/\,\lambda o + (l_4 - l_3)\,\lambda B_3/\,\lambda o - (l_5-l_4)\,\lambda B_4/\,\lambda o$ (8)

$g_9 = (l_4 - l_3)\,\lambda B_3/\,\lambda o - 50U_4a/\,\lambda o\,(T_ow_3 + BPR_3 - To\,w_4 - BPR_4)$ (9)

Fifth effect

$g_{10} = l_6\,(h_6-h_5)/\,\lambda o + (l_5 - l_4)\,\lambda B_4/\,\lambda o - (l_6-l_5)\,\lambda B_5/\,\lambda o$ (10)

$g_{11} = (l_5 - l_4)\,\lambda B_4/\,\lambda o - 50U_5a/\,\lambda o\,(Tow_4 + BPR_4 - Tow_5 - BPR_5)$ (11)

Sixth effect

$g_{12} = l_7\,(h_7 - h_6)/\,\lambda o + (l_6 - l_5)\,\lambda B_5/\,\lambda o - (l_7 - l_6)\,\lambda B_6 / \lambda o$ (12)

$g_{13} = (l_6 - l_5)\,\lambda B_5/\,\lambda o - 50U_6a/\,\lambda o\,(T_ow_5 + BPR_5 - To\,w_6 - BPR_6)$ (13)

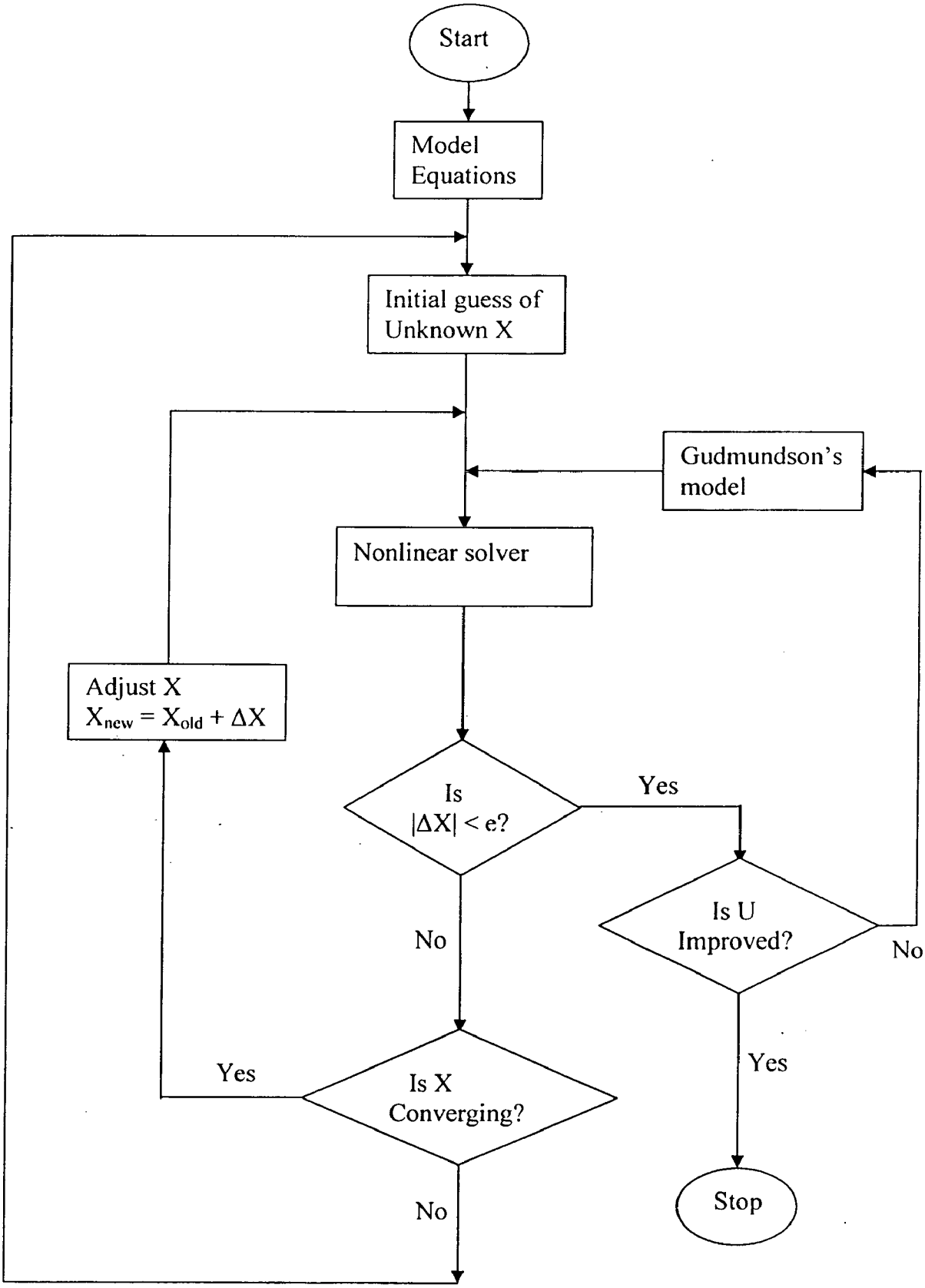


Figure 6: Simulation Flowchart

4. RESULTS AND DISCUSSION

The simulation program developed for the detailed model discussed in chapter 3 is tested with the available data reported in the literature.

Comparison of simulation and reported data of Kern [25].

Parameter		**Evaporator Body**					
		1	2	3	4	5	6
Steam Flow	Kern	2.522					
kg/s	SIM	2.574					
Liquor temp	Kern	126	111	97	82	69	54
^{0}C	SIM	123	115	100	75	65	54
Solid %	Kern	38.9	29.8	24.7	21.6	18.7	20
(X)	SIM	39.7	29.5	24.2	21.2	18.7	20
Area	Kern	300	300	300	300	300	300
m^2	SIM	295	295	295	295	295	295

Further study has been carried out to see the effects of input parameters on the steam economy to arrive at best heat recovery system.

Figure 7 and 8 illustrate the effect of steam temperature on steam economy at different feed conditions (i.e. temperature, concentration)

Figure 9, 10 and 11 illustrate the effect of steam temperature on steam economy at different types of flashing arrangements.

Figure 7: Effect of steam temperature on steam economy with feed temperature as parameter

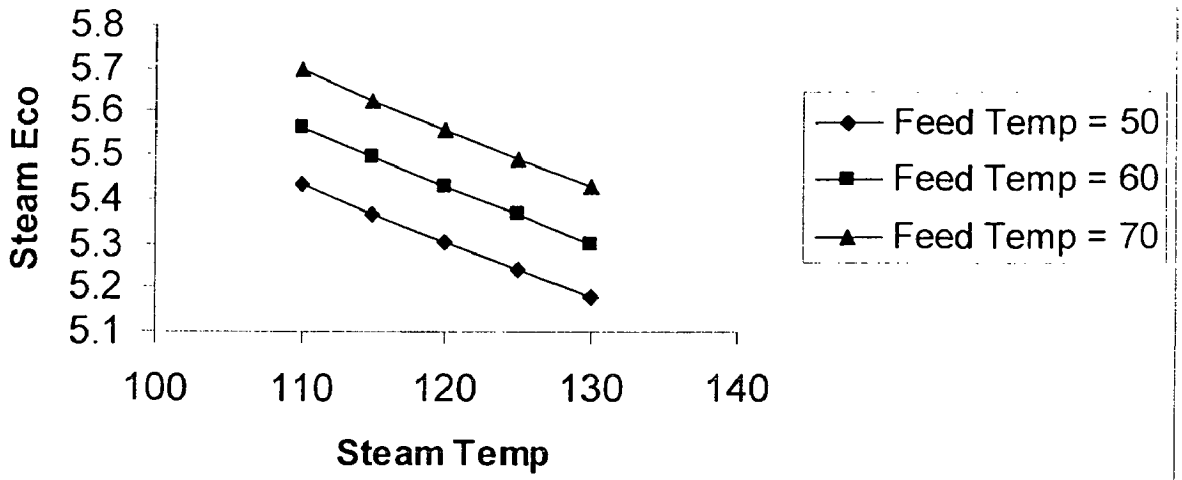


Figure 8: Effect of steam temperature on steam economy with feed concentration as parameter

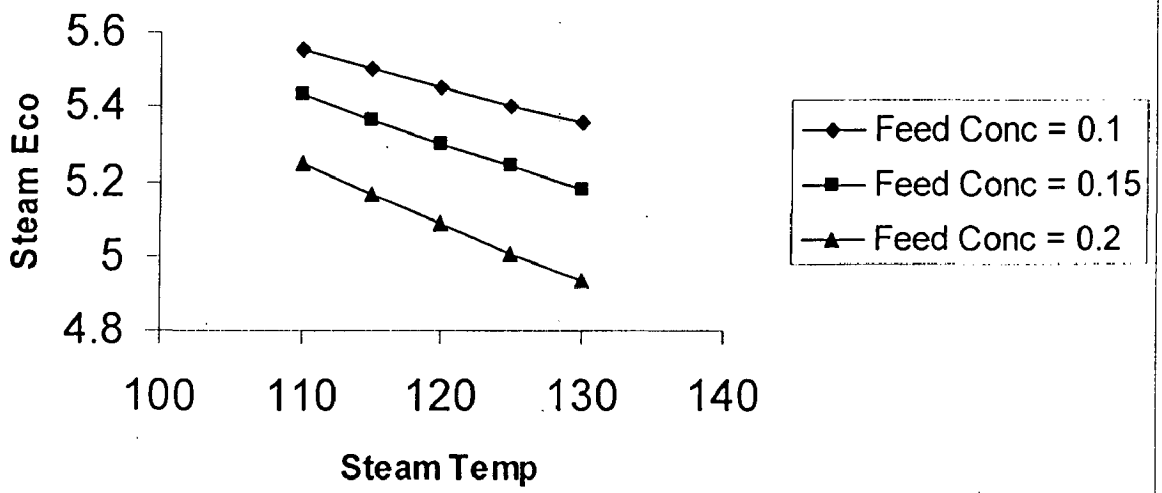


Figure 9: Comparison of system with product and without product flash

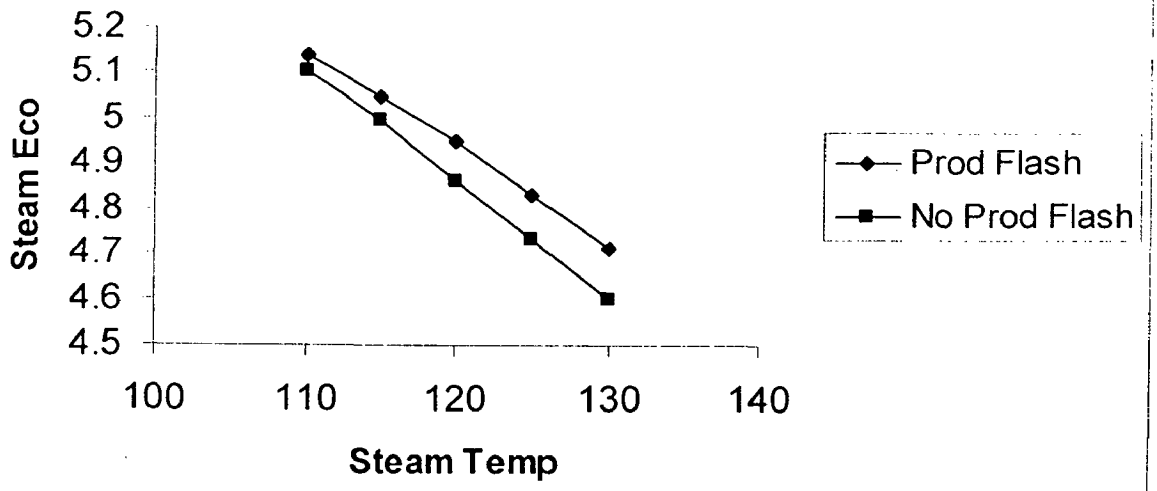


Figure 10: Comparison of system with series and with alternate condensate flash

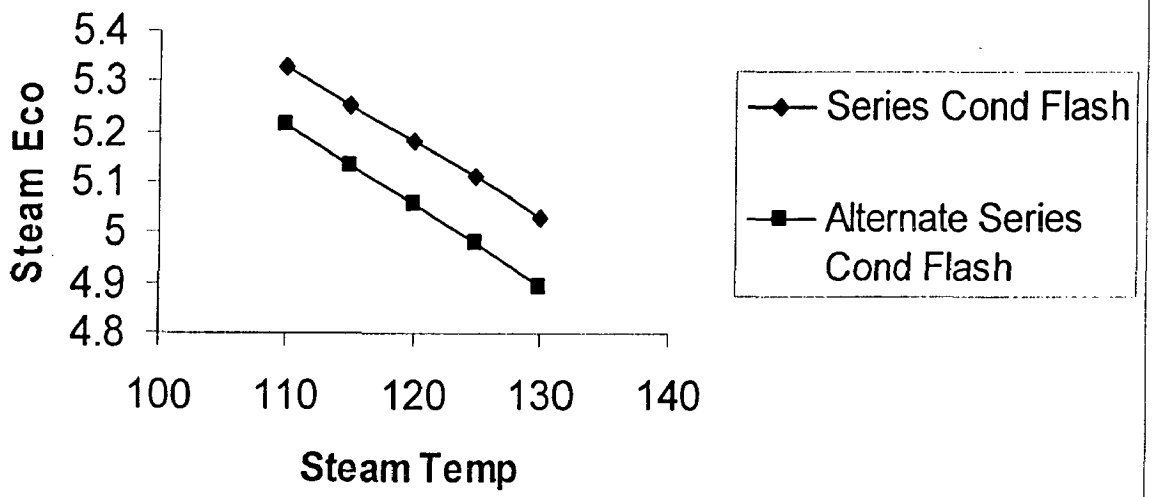
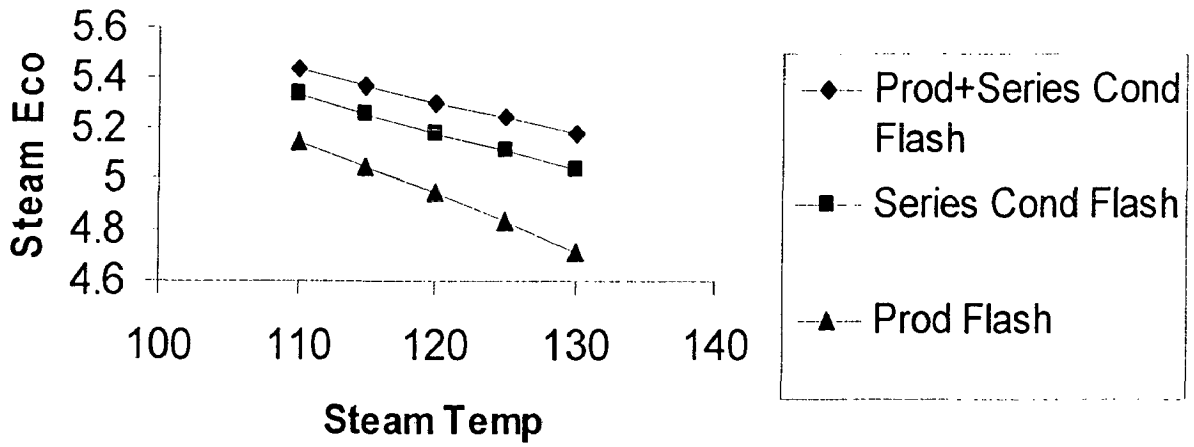


Figure 11: Comparison of system with product, with series condensate and with combined product plus series condensate flash



5. CONCLUSIONS AND RECOMMENDATIONS

CONCLUSIONS:

The values of unknown variables obtained from the simulation runs are found to be very close that reported in reference books e.g. Kern. The slight difference observed can be attributed to the quality of physico-thermal data of the liquor. Furthermore the following conclusions can be drawn from the work performed:

- The solution method used in this work gives stable convergence characteristic.
- Combined product and series condensate flashing arrangement should be employed as a heat recovery system to achieve the best steam economy.
- The simulation program generated can be used for any type of heat recovery arrangement and for any type of feeding sequence.

RECOMMENDATIONS:

On the basis of above discussion future work should focus on:

- The computer program development, in which user has the choice of giving all the following inputs:
  - Number of effects
  - Feeding sequence
  - Arrangement of preheater system
  - Arrangement of heat recovery system
- To validate the model the data collection should be done with better accuracy and reliability.

REFERENCES

1. Singh, P., Ray, A.K. Gupta, P.K., Sharma, N.K. and Mohanty B., "Design of Black Liquor Multiple Effect Evaporator System-Effect of Splitting of Feed and Flash Utilization", IPPTA Conventional Issue, pp. 83-96, (2001).
2. Zain O.S. and Kumar S., "Simulation of a Multiple Effect Evaporator for Concentrating Caustic Soda Solution - Computational Aspects," J. chem. Engg of Japan, Vol.29, No.5, (1996).
3. Stewart G., Beveridge G.S.G., "Steady State Cascade Simulation in Multiple Effect Evaporation", Comp. Chem. Engg., Vol. 1, pp. 3-9, (1977).
4. Ayangible, W.O., Okeke, E.O., and Beveridge, G.S.G., "Generalised Steady-Stale Cascade Simulation Algorithm in Multiple-Effect Evaporation", Computer Chem. Engg., Vol. 8, No 3- 4, pp. 235-242, (1984).
5. Goel, R.K., M.E thesis, "Simulation of Multiple Effect Evaporator with Provision for Flash, University of Roorkee, (1995).
6. Hillenbrand. J.B and Westerberg A.W," The synthesis of Multiple-tiffed Evaporator systems using minimum utility insights-I. A cascaded Heat Representation," Computer Chem.Engg.,Vol. 12, No 7, pp. 611-624, (1988).
7. Agarwal, V.K. Ph.D. Thesis, "Energy conservation in multiple effect evaporator system", University of Roorkee, Roorkee, 1992.
8. Lambert, Richard N., Donald, Joye, D. and Koko, Frank W. (1987). "Design Calculation for Multiple-Effect Evaporators. 1. Linear Method", Ind. Engg. Chem. Res., Vol. 26, No. I, pp. 100-104.

9. Mengistu, T. T., M.E Thesis, "Simulation of Evaporation System in Pulp and Paper Plant," University of Roorkee, (1999).
10. Radovic, L.R. et. al., "Computer Design and Analysis of Operation of a Multiple - Effect Evaporator System in the Sugar Industry, Ind. Engg. Chem. Process Des. Dev., Vol. 18, No.2, (1979).
11. Dangwal, P., Maheshwari, Y.K. and Ajad V.," Retrofit of Evaporator Plant and Free Flow Falling Film Evaporator at Star Paper Mill Ltd.", IPPTA, Vol. 10, No. 2, pp. 33-41, (June 1998).
12. Bremford D. J.; Muler-Steinhagen H.M., Multiple Effect Evaporator Performance for Black Liquor, Appita J., 47(4), 320, Sept. 1996.
13. Ray, A.K. and Singh, Pitam, "Simulation of Multiple Effect Evaporator for Black Liquor Concentration", IPPTA, Vol. 2, No, 3. Pp. 53-63, (Sept. 2000).
14. Ray, A.K., Bansal M.C., Rao, N.J., "Transport Properties of Indian Black Liquor", Pulp and Paper Canada, 90:9, (1989).
15. Zaman, A.A., McNally, T.W. and Fricke, "Pressure and Boiling Point Elevation of Slash Pine Black Liquor Predictive Models with Statistical Approach", Ind. Engg. Chem. Res., Vol. 37, No.1, (1988).
16. Graham, R.J., and Stevenson, D.F., "Effects of Experimental Error on Parameter Estimation and Convergence of a Sequential Experimental Design", Ind. Engg. Chem. Process Design Development, Vol. 11, No. 2, pp. 164-169, (1972).
17. Crowe, CM., "Reconciliation of Process Flow Rates by Matrix Projection", Part II, AIChE J, Vol. 32, No. 4, pp. 616-622, (1986).
18. Banvolgyi G. et. al., "Estimation of Influential Parameters in a Steady State Evaporator Model: The Principal component Approach", Comp. Chem. Engg., Vol. 12, No. 2/3, pp 117-122, (1988).

19. Crowe, C.M., "Recursive Identification of Gross Errors in Liner Data Reconciliation", AlChE J., Vol. 34, No. 4, pp. 541-550, (1988).
20. Aldrich, C and Deventer, S.J.," comparison of different artificial neural nets for detection and location of gross errors in process systems," Ind. Engg. Chem. Res., Vol.34, No. 1, (1995).
21. Lee J. and Lee W. K., Least Squares Updates for Quassi-Newton Methods in Process Design and Simulation, Computers Chem. Engg. Vol 13, No. 6, pp 717-723, 1989.
22. Holland, CD. ," Fundamentals and Modeling of separation processes", Prenlice-Hall inc., Engle wood cliffs, New Jersey, (1975).
23. Paloschi, J.R., "A Hybrid Continuation Algorithm to Solve Algebraic Nonlinear Equations", Computer Chem. Engg., Vol. 18, Suppl., pp S201-S209, (1999).
24. Gudmundson C., "Heat transfer in Industrial Black Liquor Evaporator Plants, part II", Sevensk. Papperstidn, Vol. 75, No. 22, pp 901-908 (Dec. 1972).
25. Kern D.Q., Process Heat Transfer, International ed. McGraw-Hill, 2003.
26. Geankoplis, C.J.,Transport Processes and Unit Operations, 4th ed., Prentice Hall, 2003.

APPENDIX: COMPUTER PROGRAM

Variables

F = liquor feed rate (kg/s)

X = concentration of effect

X_f = liquor feed concentration

L = liquor flow rate from each effect (kg/s)

l = fractional liquor flow rate from evaporator effect

LII= product flow rate (kg/s)

lII = fractional product flow rate

T = temperature of effect (°C)

T_0 = steam temperature (°C)

T_f = liquor feed temperature (°C)

w = fractional temperature of each effect

V_S = steam flow rate (kg/s)

v_S = fractional steam flow rate

U = overall heat transfer coefficient (W/m^2 °C)

AREA = heat transfer area of each effect (m^2)

a = fractional heat transfer area of evaporator effect

M = flow rate of vapor flashed from the condensate (kg/s)

m = fractional flow rate of flash vapor from the condensate

Functions

mee ()

This function is called constructor, and used for carrying out automatic initialization of the Jacobian matrix.

get_input_data()

This function receives both the design variables (Ts, T6, Tf, F, Xf, X11, Ts) and initial estimate of unknown variables (area, liquor flow rate of each effect, steam supply flow

rate, condensate flash vapor flow rate, temperature of each effect except the last effect) to start the computation.

scale_down()

This function scales down the variables for the purpose of computational stability.

scale_up()

This function stores the magnitude of the reduced variables for reporting the result.

initial_U()

Initializes overall heat transfer coefficients.

func()

Computes the functional value of the model equations.

Jacob()

Forms the Jacobian matrix.

lu()

Solves the set of linear equations of the Jacobian matrix.

error()

Returns error to stop the iteration.

New_P()

Updates the steps towards the solutions.

New_J()

Updates the Jacobian matrix towards the solutions.

Compute_U()

Computes overall heat transfer coefficient using Gudmundson's empirical model.

gi_mass_entalpy()

These functions provide model equations based on material balance.
(i=1, 2, 3, and 4)

gi_heat_transfer()

These functions provide model equations based on energy balance. (i=1, 2, 3, and 4)

g_flash_product()

This function provides model equations for the product flash tanks.

gi_flash_condensate()

These functions provide model equations for the condensate flash tanks. (i=1, 2)

hs()

Returns enthalpy of condensed steam at given temperature.

H()

Returns enthalpy of steam at given temperature.

lanS()

Returns latent heat of condensation of saturated steam at a given temperature.

v()

Returns enthalpy of condensed steam at given temperature.

k()

Returns thermal conductivity of black liquor at given temperature and concentration.

Cp()

Returns specific heat of black liquor at given concentration.

den()

Returns density of black liquor at given concentration.

BPR()

Returns boiling point rise of black liquor at a given concentration.

```
// Simulation of Sextuple Effect backword feed evaporator system
// including product and condensate flash


#include<iostream.h>
#include<conio.h>
#include<math.h>
#include<fstream.h>
#include<stdlib.h>


const int N=9; const int P=20;

class mee
{
 private:
                double L[N], l[N], M[N], m[N], T[N], w[N], X[N], Area[N], f[P];
                double T0, Tf, F, L11,l11, lan0;
                double Vs, vs, Xf, X11, AREA, AREA2;
                double A[P][P], a, B[P], U[N], UU[N], S[P],  J[P][P];
                double x[P], p[P], PP[P][P], lan_prev[P];
 public:
                mee( );
                void get_input_data ( );
                void scale_down( );
                void scale_up( );
                void Jacob ( );
                void new_value( );
                void save_res ( );
                void heat_trans_area ( );
                void initial_U ( );
                void func ( );
                void compute_U ( );
                void report_U ( );
                void step_size( );
                void lu ( );
                void new_J ( );
                void new_P ( int );
                double error();

                double g1_mass_enthalpy();
                double g1_heat_transfer();
                double g1_flash_condensate();

                double g2_mass_enthalpy();
                double g2_heat_transfer();
                double g2_flash_condensate(int);

                double g3_mass_enthalpy();
                double g3_heat_transfer();
```

```
                double g4_mass_enthalpy(int);
                double g4_heat_transfer(int);

                double g_flash_product();

                double t(double,double);
                double BPR(double);
                double h(double,double);
                double hs(double);
                double H(double);
                double lanS(double);
                double lanB(double,double);
                long double v(double,double);
                double k(double,double);
                double Cp(double);
                double den(double,double);


};

void mee::mee()
{
        for( int i=1; i<=P; ++i)
        for( int j=1; j<=P; j++)
        {
         A[i][j] = 0.0;
         J[i][j] = 0.0;
        }
}

void mee :: get_input_data()
{
char file[20], str[40];
cout<<endl;

 // design data
 cout<<"\nenter input file name   "; cin>> file;
 ifstream inn(file);
 inn>>str;  inn>>L[7]; cout<<"\nl[7]="<<L[7];
 inn>>str;  inn>>X[7]; cout<<"\nx[7]="<<X[7];
 inn>>str;  inn>>X11;  cout<<"\nx11="<<X11;
 inn>>str;  inn>>T[7]; cout<<"\nt[7]="<<T[7];
 inn>>str;  inn>>T0;   cout<<"\nT0="<<T0;
 inn>>str;  inn>>T[6]; cout<<"\nT[6]="<<T[6];
 F=L[7];    Xf=X[7];        Tf=T[7];
 T[0]=T0;   L11=F*Xf/X11;     lan0=lanS(1);

 // intial guesses for starting the calculation
 inn>>str;  inn>>Vs;   cout<<"\nVs="<<Vs;
 inn>>str;
 for(int i=1; i<=6; i++)
```

```
        {
         inn>>L[i]; cout<<"\nL["<<i<<"]="<<L[i];
         inn>>str;
        }
  for(i=1;i<=5;i++)
        {
         inn>>T[i]; cout<<"\nT["<<i<<"]="<<T[i];
         inn>>str;
        }
  for( i=1; i<=5; i++)
        {
         inn>>M[i]; cout<<"\nM["<<i<<"]="<<M[i];
         inn>>str;
        }
        inn>>AREA;
        cout<<"\nArea="<<AREA;
}

void mee :: scale_down()
{
        vs=Vs/F;        x[1]=vs;        a=AREA/(50*F);
        l11=L11/F;      x[13]=a;        w[0]= T0/T0;
        w[6]=T[6]/T0; w[7]=T[7]/T[0]; l[7]=L[7]/F;
        for(int i=1; i<=18; i++)
        for(int j=1; j<=18; j++)
        PP[i][j]=1;
        for(i=1; i<=5; i++)
        {
        w[i]=T[i]/T0;   x[i+7]=w[i];
        }
        for(i=1; i<=6; i++)
        {
        l[i]=L[i]/F;    x[i+1]=l[i];
        }
        for(i=1;i<=5; i++)
        {
        m[i]=M[i]/F;    x[i+13]=m[i];
        }
}

void mee::scale_up()
{
        vs=x[1];        Vs=vs*F;    a=x[13];        AREA=a*50.0*F;
        L11=l11*F;      T[6]=w[6]*T0; T[7]=w[7]*T0;
        for(int i=1; i<=5; i++)
        {
        w[i]=x[i+7];    T[i]=w[i]*T0;
        }
        for (i=1; i<=6; i++)
        {
        l[i]=x[i+1];    L[i]=l[i]*F;
```

```
        }
        for(i=1;i<=5;i++)
        {
        m[i]=x[i+13];   M[i]=m[i]*F;
        }
}

void mee::initial_U()
{
        U[1]=1488;      U[2]=1675;      U[3]=1431;
        U[4]=1425;      U[5]=1255;      U[6]=1255;
}

double mee::error()
{
int const n=18;
long double sum=0.0, mse;
func();
for(int i=1; i<=n; i++)
sum+=pow(f[i],2);
mse = sum/n;
return mse;
}

double mee::g1_mass_enthalpy()
{
double f;
vs=x[1];
for(int i=1; i<=6;  i++)
l[i]=x[i+1];
for(i=1; i<=5; i++)
w[i]= x[i+7];
a=x[13];
for(i=1; i<=5; i++)
m[i]=x[i+13];

f=l[2]*(h(l[2],w[2])-h(l[1],w[1]))/lan0 + vs-(l[2]-l[1])*lanB(l[1],w[1])/lan0;
return f;
}

double mee::g2_mass_enthalpy()
{
double f;
vs=x[1];
for (int i=1; i<=6; i++)
l[i]=x[i+1];
for(i=1; i<=5; i++)
w[i]=x[i+7];
a=x[13];
for(i=1;i<=5;i++)
m[i]=x[i+13];
```

```
f = l[3]*(h(l[3],w[3])-h(l[2],w[2]))/lan0 + (l[2]-l[1]+m[1])*lanS(w[1])/lan0-(l[3]-
l[2])*lanB(l[2],w[2])/lan0;
return f;
}

double mee::g3_mass_enthalpy()
{
double f;
 vs=x[1];
 for(int i=1; i<=6; i++)
l[i]=x[i+1];
for(i=1; i<=5; i++)
w[i]=x[i+7];
a=x[13];
 for(i=1;i<=5;i++)
m[i]=x[i+13];

f = l[4]*(h(l[4],w[4])-h(l[3],w[3]))/lan0 + (l[3]-l[2]+m[2]+l[1]-l11)*lanS(w[2])/lan0 -
(l[4]-l[3])*lanB(l[3],w[3])/lan0;
return f;
}

double mee::g4_mass_enthalpy(int j)
{
double f;
vs=x[1];
for(int i=1; i<=6; i++)
l[i]=x[i+1];
for(i=1; i<=5; i++)
w[i]=x[i+7];
a=x[13];
for (i=1; i<=5; i++)
 m[i]=x[i+13];

f = l[j]*(h(l[j],w[j])-h(l[j-1],w[j-1]))/lan0 + (l[j-1]-l[j-2]+m[j-2])*lanS(w[j-2])/lan0-(l[j]-l[j-
1])*lanB(l[j-1],w[j-1])/lan0;
return f;
}

double mee::g1_heat_transfer()
{
double f;
vs=x[1];
for (int i=1; i<=6; i++)
l[i]=x[i+1];
for(i=1; i<=5; i++)
w[i]=x[i+7];
a=x[13];
for(i=1; i<=5; i++)
m[i]=x[i+13];
```

```
f =vs-50*U[1]*a/lan0*(T0-w[1]*T0-BPR(l[1]));
return f;
}

double mee::g2_heat_transfer()
{
double f;
 vs=x[1];
 for(int i=1; i<=6; i++)
l[i]=x[i+1];
 for(i= 1 ;i<=5;i++)
w[i]=x[i+7];
 a=x[13];
 for (i=1; i<=5; i++)
m[i]= x[i+13];

f =(l[2]-l[1]+m[1])*lanS(w[1])/lan0-50*U[2]*a/lan0*(w[1]*T0+BPR(l[1])-w[2]*T0-BPR(l[2]));
return f;
}

double mee::g3_heat_transfer()
 {
double f;
 vs=x[1];
 for(int i=1; i<=6; i++)
l[i]=x[i+1];
 for(i= 1 ;i<=5;i++)
w[i]=x[i+7];
 a=x[13];
 for (i=1; i<=5; i++)
m[i]= x[i+13];

f = (l[3]-l[2]+m[2]+l[1]-111)*lanS(w[2])/lan0 - 50*U[3]*a/lan0*(w[2]*T0+BPR(l[2]) - w[3]*T0 -
BPR(l[3]));
return f;
}

double mee::g4_heat_transfer(int j)
{
double f;
 vs=x[1];
 for(int i=1; i<=6; i++)
l[i]=x[i+1];
 for(i= 1 ;i<=5;i++)
w[i]=x[i+7];
 a=x[13];
 for (i=1; i<=5; i++)
m[i]= x[i+13];

f = (l[j]-l[j-1]+m[j-1])*lanS(w[j-1])/lan0-50*U[j]*a/lan0*(w[j-1]*T0+BPR(l[j-1]) -
```

```
w[j]*T0-BPR(l[j]));
 return f;
}

double mee::g1_flash_condensate()
{
double f;
vs=x[1];
for(int i=1; i<=6; i++)
l[i]=x[i+1];
for(i= 1 ;i<=5;i++)
w[i]=x[i+7];
a=x[13];
for (i=1; i<=5; i++)
m[i]= x[i+13];

f = vs*(hs(w[0])-hs(w[1]))/lan0-m[1]*lanS(w[1])/lan0;
return f;
}

double mee::g2_flash_condensate(int j)
{
double f;
vs=x[1];
for(int i=1; i<=6; i++)
l[i]=x[i+1];
for(i= 1 ;i<=5;i++)
w[i]=x[i+7];
a=x[13];
for (i=1; i<=5; i++)
m[i]= x[i+13];

f = (vs+l[j]-l11)*(hs(w[j-1])-hs(w[j]))/lan0-m[j]*lanS(w[j])/lan0;
return f;
}

double mee::g_flash_product()
{
double f;
vs=x[1];
for(int i=1; i<=6; i++)
l[i]=x[i+1];
for(i= 1 ;i<=5;i++)
w[i]=x[i+7];
a=x[13];
for (i=1; i<=5; i++)
m[i]= x[i+13];

f = l[1]*(h(l[1],w[1])-H(w[2]))/lan0-l11/lan0*(h(l11,w[2])-H(w[2]));
return f;
}
```

```
void mee::func()
{
 for(int i= 1; i<=6; i++)
          UU[i]=U[i];

 f[1]=g1_mass_enthalpy();
 f[2]=g_flash_product();
 f[3]=g2_mass_enthalpy();
 f[4]=g3_mass_enthalpy();

 for(i=5; i<=7; i++)
         f[i]=g4_mass_enthalpy(i);

 f[8]=g1_heat_transfer();
 f[9]=g2_heat_transfer();
 f[10]=g3_heat_transfer();

 for(i=11; i<=13; i++)
         f[i]= g4_heat_transfer(i-7);

 f[14]=g1_flash_condensate();

 for(i=15;i<=18;i++)
         f[i]=g2_flash_condensate(i-13);
}

void mee::lu()
{
 double b[20], l[20][20], u[20][20];
 int n=18;
        for (int i=1; i<=n; i++)
                for(int j=1; j<=n; j++)
                        A[i][j]=J[i][j];

        func();
        for (i=1; i<=n; i++)
                B[i]=-f[i];
        for(i=1; i<=n; i++)
          for(j=1; j<=n; j++)
                {
                 l[i][j]=0.0;
                 u[i][j]=0.0;
                }
        for(i=1; i<=n;i++)
                l[i][1]=A[i][1];
        for(j=2; j<=n; j++)
                u[1][j]=A[1][j]/l[1][1];

        u[1][1]=1.0;
        for(j=2; j<=n; j++)
```

```
        {
         for (i=j; i<=n; i++)
         {
                 double sum =0.0;
                         for(int k=1; k<=j-1; k++)
                          sum+=l[i][k]*u[k][j];
                 l[i][j]=A[i][j]-sum;
                }
                u[i][j]=1.0;

         for(i=j+1; i<=n; i++)
         {
                 double sum = 0.0;
                  for (int k=1; k<=i-1; k++)
                          sum+=l[j][k]*u[k][i];
                 u[j][i]=(A[j][i]-sum)/l[j][j];
         }
        }
        b[1]=B[1]/l[1][1];

  for (i=2; i<=n; i++)
         {
                 float sum=0.0;
                  for (int k=1; k<=i-1; k++)
                   sum+=l[i][k]*b[k];
                 b[i]=(B[i]-sum)/l[i][i];
         }
  p[n]=b[n];
  for(j=n-1; j>=1; j--)
  {
  float sum=0.0;
  for (int k=j+1; k<=n; k++)
  sum+=u[j][k]*p[k];
  p[j]=b[j]-sum;
  }
}

void mee::Jacob()
{
 double h=0.00001;
 double fold[20];
 func();
 for(int i=1; i<=18; i++)
 fold[i]=f[i];
 for (i = 1; i<=18;i++)
 for(int j=1; j<=18; j++)
 {
 x[j]+=h;
 func();
 J[i][j]=(f[i]-fold[i])/h;
 x[j]-=h;
```

```
  }
}

void mee::new_J()
 {
 int n=N;
 double sum, Jold[20][20], Pold[20][20], xold[20], fold[20], dx[20], df[20];
 double C[20], de, W[20][20], Ne[20][20], JX[20], FJX[20];
 for (int i=1; i<=n; i++)
 {
 xold[i]=x[i];
 fold[i]=f[i];
 }
 Jacob();
 func();
 lu();
 for (i=1; i<=n;i++)
 x[i]+=p[i];
 func();
 for (i=1;i<=n;i++)
 {
 dx[i]=x[i]-xold[i];
 df[i]=f[i]-fold[i];
 }
 for(i=1; i<=n; i++)
 for (int j=1; j<=n; j++)
 {
 Pold[i][j]=PP[i][j];
 Jold[i][j]=J[i][j];
 }
 for(i=1; i<=n; i++)
 {
 sum=0.0;
 for(int j=1; j<=n; j++)
 sum+=Jold[i][j]*dx[j];
 JX[i]=sum;
 FJX[i]=df[i]-JX[i];
 }
 for (i=1; i<=n; i++)
 for(j=1;j<=n;j++)
 W[i][j]=FJX[i]*dx[j];
 for (i=1; i<=n; i++)
 for (int k=1; k<=n; k++)
 {
 sum=0.0;
 for(j=1; j<=n; j++)
 sum+=W[i][j]*Pold[j][i];
 Ne[i][k]=sum;
 }
 for (i=1; i<=n;i++)
 {
```

```
  sum=0.0;
  for(j=1; j<=n; j++)
  sum+=dx[j]*Pold[j][i];
  C[i]=sum;
  }
  sum=0.0;
  for (i=1; i<=n; i++)
  sum+=C[i]*dx[i];
  de=sum;
  for(i=1; i<=n; i++)
  for(j=1;j<=n;j++)
  J[i][j]=Jold[i][j]+Ne[i][j]/de;
  lu();
  for(i=1;i<=n; i++)
  x[i]+=p[i];
  func();
}

void mee::new_P(int kk)
{
 int const n =18;
 double sum, xold[n], fold[n], dx[n], df[n];
 double C[n],de, de1, alpha, Pold[n][n], W[n][n], Ne[n][n];
 for(int i=1; i<=n; i++)
 {
 xold[i]=x[i];
 fold[i]=f[i];
 }

 Jacob();
 func();
 lu();
 for(i=1;i<=n;i++)
 x[i]+=p[i];
 func();
        Jacob();
 for (i=1; i<=n; i++)
 {
 dx[i]=x[i]-xold[i];
 df[i]=f[i]-fold[i];
 }
 for(i=1;i<=n;i++)
 for(int j=1;j<=n;j++)
 Pold[i][j]=PP[i][j];
 for(i=1;i<=n;i++)
 {
 sum=0.0;
 for(int j=1;j<=n;j++)
 sum+=dx[j]*Pold[j][i];
 C[i]=sum;
 }
```

```
  sum=0.0;
  for(i=1; i<=n; i++)
  sum+=C[i]*dx[i];
  de1=sum;
  de=de1 + pow(alpha,kk+3);
  for(i=1; i<=n; i++)
  {
  sum=0.0;
  for (j=1;j<=n;j++)
  sum+=Pold[i][j]*dx[j];
  C[i]=sum;
  }
  for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
  W[i][j]=C[i]*dx[j];
  for(i=1;i<=n;i++)
  for(int k=1;k<=n;k++)
  {
  sum=0.0;
  for(j=1;j<=n;j++)
  sum+=W[i][j]*Pold[j][i];
  Ne[i][k]=sum;
  }
  for(i=1;i<=n;i++)
  for(j=1;j<=n;j++)
  PP[i][j]=Pold[i][j]-Ne[i][j]/de;
}

void mee::compute_U()
{
 const double p1=0.981,  p2=1.0,  p3=0.2334, p4=-.1006e-4, p5=0.1362e-9;
 const double p6=1.0, p7=-0.07858, p8=0.002735, p9=-0.1092e-4, p10=-1.02;
 const double p11= 0.0476, p12=-0.364e-3, p13=0.946e-6, p14=-0.0515, p15=0.0505;
 const double p16=0.01, p17=21800,  p18=0.4,  p19=19500, p20= 2000;
 const double p21=-95,  p22=15,  p23=0.12,  p24=-0.001,  p25=3.0;
 const double p26=-0.025,  p27=-0.12,  p28=4300,  p29=100,  p30=-14;
 const double p31=0.04;
 double di= 0.048,d0=0.05, Le=8.5;
 int Nt=300;
 double A1, q, fd, bound1,  bound2, c_v, Tmax, Z;
 double Tsub, C, c_temp, v1, v0;
 double Tsup;
 for (int i=1; i<=6; i++)
 X[i]=F*Xf/L[i];
 for(int j=1; j<=6; j++)
 {
        A1=AREA/(3.14*d0*Le)*3.14*Nt*di*di*0.25;
        double F=L[j];
        fd=F/A1;
        q=lanB(l[j],w[j])*(L[j+1]-L[j])/AREA;
        v1 = 1000*(v(T[j],X[j])+v(T[j+1],X[j+1]))/2;
```

```
if(v1<15)
c_v = p6+p7*v1+p8*pow(v1,2)+p9*pow(v1,3);
else
c_v = 1.576-0.095*log10(v1);

c_temp = p10+p11*t(T[j],X[j])+p12*pow(t(T[j],X[j]),2)+p13*pow(t(T[j],X[j]),3);
//overrlow
Tmax = p22+(p23+p24*t(T[j],X[j]))*fd+(p25+p26*t(T[j],X[j]))*log10(v1)+
p27*t(T[j],X[j]);
// overflow
if(v1<1)
{
v0=1;
bound1 = p19+p20*log10(v0)+p21*t(T[j],X[j]);
bound2 = p28+p29*log10(v0)+p30*t(T[j],X[j]);
}
else
{
bound1 = p19+p20*log10(v1)+p21*t(T[j],X[j]);
bound2 = p28+p29*log10(v1)+p30*t(T[j],X[j]);
}
Tsub = Tmax-T[j+1];
if(Tsub<0)    Tsub=0.0;
Z=1.0+p16*tanh((q-bound1)/p17)*pow(Tsub,p18);
double cf = Z+(p14+p15*Z)*fd;
C=c_v*c_temp*cf;
if(Tsub<0)
{
Tsup=T[j+1]-Tmax;          //overflow
double dq=Tsup*Cp(X[j])*fd*(3.14*di*di*0.25)*0.61/1.321;
q+=dq;
}
if(q>bound2)
{
if(q>20000)  q=20000;
U[j]=(p1*pow(C,p2)*(p3*q+p4*q*q+p5*pow(q,3)));
}
else
{
double vel=fd/den(T[j],X[j]);
double Re=den(T[j],X[j])*vel*di/(0.001*v1);
double Pr=0.001*v1*Cp(X[j])/k(T[j],X[j]);
double h=0.023*pow(Re,0.8)*pow(Pr,0.4)*k(T[j],X[j])/di;
double q2=bound2;
double u=p1*pow(C, p2)*(p3*q2+p4*q2*q2+p5*pow(q2,3));
if(Tsub<0.0) Tsub=0.0;
double c_sub=1.0+p31*Tsub*pow((bound2-q)/bound2,2);
U[j]=((h+(u-h)*q/bound2)*c_sub);
}
}
}
```

```
void mee::report_U()
{
 clrscr();
 cout<<"\n\n\n";
 for(int i=1; i<=6; i++)
 cout<<"\n old U["<<(i)<<"] = "<<UU[i]<<"   new U["<<(i)<<"] = "<<U[i];
 getch();
}

void mee::save_res()
 {
 //Output On The Screen

 cout<<endl;
 cout<<"\t Report of the Simulation \n\n";
 cout<<" Feed condition \n\n";
 cout<<" feed conc = "<<Xf<<"\n"
    <<" feed rate (kg/s) = "<<F<<"\n"
    <<" feed temp (deg C)= "<<Tf<<"\n\n";
 cout<<" last effect temp (deg C)= "<<T[6]<<"\n"
    <<" product conc = "<<X11<<"\n\n"
    <<" Steam temp (deg C)= "<<T0<<"\n";
 cout<<" Steam flowrate(kg/s): Vs = "<<Vs<<"\n\n";
 cout<<" Vapor flashed from the product flash tank(kg/s): fv = "<<L[1]- L11<<"\n\n";
 cout<<" Liquid flowrate from each effect (kg/s):\n\n";
 cout<<" L1 = "<<L[1]<<"\n"
    <<" L2 = "<<L[2]<<"\n"
    <<" L3 = "<<L[3]<<"\n"
    <<" L4 = "<<L[4]<<"\n"
    <<" L5 = "<<L[5]<<"\n"
    <<" L6 = "<<L[6]<<"\n\n";
 cout<<" Steam flashed from each flash tank (kg/s):\n\n";
 cout<<" M1 = "<<M[1]<<"\n"
    <<" M2 = "<<M[2]<<"\n"
    <<" M3 = "<<M[3]<<"\n"
    <<" M4 = "<<M[4]<<"\n"
    <<" M5 = "<<M[5]<<"\n\n";
 cout<<"\nPress Any Key For the next page of the output:";
 getch();
 cout<<"\nTemperature of each effects (deg C):\n\n";
 cout<<" T1 = "<<T[1]<<"\n"
    <<" T2 = "<<T[2]<<"\n"
    <<" T3 = "<<T[3]<<"\n"
    <<" T4 = "<<T[4]<<"\n"
    <<" T5 = "<<T[5]<<"\n"
    <<" T6 = "<<T[6]<<"\n\n\n";
 cout<<" Concentration in each effect: \n\n";
 cout<<" X1 = "<<X11<<"\n"<<" X2 = "<<Xf/l[2]<<"\n"
    <<" X3 = "<<Xf/l[3]<<"\n"<<" X4 = "<<Xf/l[4]<<"\n"
    <<" X5 = "<<Xf/l[5]<<"\n"<<" X6 = "<<Xf/l[6]<<"\n\n\n";
```

```
cout<<" The area of each effect: ( A = "<<AREA<<" )\n\n";
cout<<"Steam Economy = "<<(F-L11)/Vs<<"\n\n";
cout<<"The estimated overall heat transfer coeff.(w/sq.m.k): \n\n";
cout<<" U1="<<U[1]<<"\n";
cout<<" U2="<<U[2]<<"\n";
cout<<" U3="<<U[3]<<"\n";
cout<<" U4="<<U[4]<<"\n";
cout<<" U5="<<U[5]<<"\n";
cout<<" U6="<<U[6]<<"\n\n";
cout<<"\nEnd Of the OutPut.....Thanx.";
getch();

// saving the output file
char file[20],a;
cout<<endl;
cout<<"enter simulation output file name";
cin>>file;
ofstream out(file);
out<<endl;
out<<"\t Report of the Simulation \n\n";
out<<" Feed condition \n\n";
out<<" feed conc = "<<Xf<<"\n"
   <<" feed rate (kg/s) = "<<F<<"\n"
   <<" feed temp (deg C) = "<<Tf<<"\n\n";
out<<" last effect temp (deg C) = "<<T[6]<<"\n"
   <<" product conc = "<<X11<<"\n\n"
   <<" Steam temp (deg C) = "<<T0<<"\n";
out<<" Steam flowrate(kg/s): Vs = "<<Vs<<"\n\n";
out<<" Vapor flashed from the product flash tank(kg/s): fv = "<<L[1]- L11<<"\n\n";
out<<" Liquid flowrate from each effect (kg/s):\n\n";
out<<" L1 = "<<L[1]<<"\n"
   <<" L2 = "<<L[2]<<"\n"
   <<" L3 = "<<L[3]<<"\n"
   <<" L4 = "<<L[4]<<"\n"
   <<" L5 = "<<L[5]<<"\n"
   <<" L6 = "<<L[6]<<"\n\n";
out<<" Steam flashed from each flash tank (kg/s):\n\n";
out<<" M1 = "<<M[1]<<"\n"
   <<" M2 = "<<M[2]<<"\n"
   <<" M3 = "<<M[3]<<"\n"
   <<" M4 = "<<M[4]<<"\n"
   <<" M5 = "<<M[5]<<"\n\n";
out<<" Temperature of each effects (deg C):\n\n";
out<<" T1 = "<<T[1]<<"\n"
   <<" T2 = "<<T[2]<<"\n"
   <<" T3 = "<<T[3]<<"\n"
   <<" T4 = "<<T[4]<<"\n"
   <<" T5 = "<<T[5]<<"\n"
   <<" T6 = "<<T[6]<<"\n\n\n";
out<<" Concentration in each effect: \n\n";
out<<" X1 = "<<X11<<"\n"<<" X2 = "<<Xf/l[2]<<"\n"
```

```
    <<" X3 = "<<Xf/l[3]<<"\n"<<" X4 = "<<Xf/l[4]<<"\n"
    <<" X5 = "<<Xf/l[5]<<"\n"<<" X6 = "<<Xf/l[6]<<"\n\n\n";
  out<<" The area of each effect: ( A = "<<AREA<<" )\n\n";
  out<<"Steam Economy = "<<(F-L11)/Vs<<"\n\n";
  out<<"The estimated overall heat transfer coeff.(w/sq.m.k): \n\n";
  out<<" U1="<<U[1]<<"\n";
  out<<" U2="<<U[2]<<"\n";
  out<<" U3="<<U[3]<<"\n";
  out<<" U4="<<U[4]<<"\n";
  out<<" U5="<<U[5]<<"\n";
  out<<" U6="<<U[6]<<"\n\n";

  }


 void mee::heat_trans_area()
 {
  Area[1]=Vs*lanS(1)/(U[1]*(T0-T[1]-BPR(l[1])));
  Area[2]=(L[2]-L[1]+M[1])*lanS(w[1])/(U[2]*(T[1]+BPR(l[1])-T[2]-BPR(l[2])));
  Area[3]=(L[3]-L[2]+M[2]+L[1]-L11)*lanS(w[2])/(U[3]*(T[2]+BPR(l[2])-T[3]-BPR(l[3])));
  Area[4]=(L[4]-L[3]+M[3])*lanS(w[3])/(U[4]*(T[3]+BPR(l[3])-T[4]-BPR(l[4])));
  Area[5]=(L[5]-L[4]+M[4])*lanS(w[4])/(U[5]*(T[4]+BPR(l[4]-T[5]-BPR(l[5]))));
  Area[6]=(L[6]-L[5]+M[5])*lanS(w[5])/(U[6]*(T[5]+BPR(l[5])-T[6]-BPR(l[6])));
  AREA2=(Area[1]+Area[2]+Area[3]+Area[4]+Area[5]+Area[6])/6;
 }

double mee::BPR(double ll)
{
 double ff;
 ff=41.4*pow((Xf/ll+0.1),2.0);
 return ff;
}

double mee::lanB(double ll,double ww)
{
 double ff;
 ff=lanS(ww) + 0.5*BPR(ll);
 return ff;
}

double mee::lanS(double ww)
{
 double ff;
 ff=2499000-2208.2*T0*ww-2.2*T0*T0*ww*ww;
 return ff;
}

double mee::h(double ll, double ww)
{
 double ff;
 ff=4187*(1-0.54*Xf/ll)*(T0*ww+BPR(ll));
```

```
 return ff;
 }

 double mee::hs(double ww)
 {
  double ff;
  ff=1000*(4.2098*T0*ww-1.8038);
  return ff;
 }

 double mee::H(double ww)
 {
  double ff;
  ff=2497500+1993.5*T0*ww-2.2*T0*T0*ww*ww;
  return ff;
 }

 long double mee::v(double TT, double XX)
 {
  double a,b,c,d;
  a=0.4717-0.02472*TT+0.7059*pow(10,-5)*TT*TT;
  b=0.06973-0.5452*pow(10,-3)*TT+0.1656*pow(10,-5)*TT*TT;
  c=0.002046+0.3183*pow(10,-4)*TT-0.9761*pow(10,-7)*TT*TT;
  d=0.5793e-04-0.6129*pow(10,-6)*TT+0.1837*pow(10,-8)*TT*TT;
  return (0.001*exp(a+b*XX+c*XX*XX+d*pow(XX,3))); // in kg/m.s
 }

 double mee::Cp(double XX)
 {
  return (1.00-0.54*XX)*4187.0; //in J/kg.k !
 }

 double mee::k(double TT, double XX)
 {
  return (0.504-0.282*XX+1.35*pow(10,-3)*TT)* 1.163; // in w/m.k
 }

 double mee::den(double TT, double XX)
 {
  return 1007+600*XX-0.495*TT; // in kg/cum
 }

 double mee::t(double TT, double XX)
 {
  return TT+41.4*pow((XX+0.1),2); // in deg C
 }

 void main()
 {
  mee ev;
  int counter;
```

```
 char a,b;
 double f;
 clrscr();

 cout<<"\n\t SIMULATION OF MULTIPLE EFFECT EVAPORATOR\n\n\n";
 ev.get_input_data();
 ev.initial_U();
 mat:   ev.scale_down();
 counter=0;
 do
 {
 counter++;
 ev.new_P(counter);
 ev.new_J();
 ev.lu();
 ev.error();
 cout<< "\n    error() = "<< ev.error();
 cout<< "\n\n   do you repeat ?(y/n)" ;   cin>>a;
 }
 while (a=='y');
 ev.scale_up();
 ev.compute_U();
 ev.report_U();
 cout<<"\n\n do you want to repeat the iteration? (y/n)";    cin>>b;
 if (b=='y')   goto mat;
 cout<<"\n\n do you want to save ? (y/n)";    cin>>a;
 if(a=='y')
 {
  ev.heat_trans_area();
  ev.save_res();
 }
 else
 {
 getch();
 exit(0);
 }
}
```

INPUT						OUTPUT	
F(kg/s)	X_f	X_p	$T_s(^0C)$	$T_f(^0C)$	$T_6(^0C)$	$A(m^2)$	S_e
15	0.15	0.45	135	70	54	197.191237	5.342353
15	0.15	0.45	144	70	54	172.697557	5.21897
18.144	0.15	0.45	105	70	54	477.515606	5.728544
18.144	0.15	0.45	121	70	54	298.187346	5.507801
18.144	0.15	0.45	134	70	54	238.474137	5.342353
18.144	0.15	0.45	144	70	54	208.894965	5.21897
26.02	0.15	0.45	105	70	54	677.610795	5.728743
26.02	0.15	0.45	113	70	54	521.800383	5.615939
26.02	0.15	0.45	121	70	54	427.625371	5.507801
26.02	0.15	0.45	134	70	54	341.99168	5.342353
26.02	0.15	0.45	113	70	54	679.616027	5.615939
34.02	0.15	0.45	121	70	54	559.101273	5.507801
34.02	0.15	0.45	134	70	54	447.139006	5.342353
34.02	0.15	0.45	144	70	54	391.678059	5.21897
34.02	0.15	0.45	105	70	54	1132.823582	5.728743
34.02	0.15	0.45	113	70	54	868.997566	5.615939
43.5	0.15	0.45	121	70	54	714.900217	5.507801
43.5	0.15	0.45	134	70	54	571.738588	5.342353
43.5	0.15	0.45	144	70	54	500.822915	5.21897
43.5	0.1	0.45	105	70	54	534.21794	5.830504
43.5	0.1	0.45	113	70	54	412.725335	5.741843
18.144	0.1	0.45	121	70	54	341.216412	5.655605
18.144	0.1	0.45	134	70	54	274.024827	5.52218
18.144	0.1	0.45	144	70	54	240.263119	5.421455
18.144	0.12	0.45	105	70	54	516.385121	5.795271
18.144	0.12	0.45	113	70	54	390.51226	5.697932
18.144	0.12	0.45	121	70	54	322.402828	5.603841
18.144	0.12	0.45	134	70	54	259.779432	5.458212
18.144	0.12	0.45	144	70	54	227.695143	5.348896
18.144	0.18	0.45	105	70	54	444.000999	5.642467
18.144	0.18	0.45	134	70	54	217.022026	5.200716
18.144	0.18	0.45	144	70	54	190.141929	5.061399
18.144	0.2	0.45	105	70	54	416.719827	5.572521
18.144	0.2	0.45	113	70	54	310.439583	5.428817
18.144	0.2	0.45	134	70	54	202.842338	5.08924
18.144	0.2	0.45	144	70	54	177.660918	4.938976
18.144	0.15	0.45	105	70	54	550.953569	5.394733
18.144	0.15	0.45	113	70	54	401.133213	5.295035
18.144	0.15	0.45	134	70	54	322.654851	5.197037
18.144	0.15	0.45	144	45	54	253.783356	5.49458

18.144	0.15	0.45	105	45	54	221.825531	4.936705
18.144	0.15	0.45	121	45	54	509.268951	5.525014
18.144	0.15	0.45	134	55	54	311.270522	5.318421
18.144	0.15	0.45	144	55	54	247.20031	5.1632
18.144	0.15	0.45	105	80	54	460.874963	5.871243
18.144	0.15	0.45	113	80	54	352.223535	5.753423
18.144	0.15	0.45	121	80	54	290.97633	5.640254
18.144	0.15	0.45	134	80	54	232.593255	5.468845
18.144	0.15	0.45	113	95	54	339.052891	5.971253
18.144	0.15	0.45	121	95	54	280.104847	5.851409
18.144	0.15	0.45	134	95	54	223.829751	5.670044