REFERENCES

ภาษาไทย

ปราโมทย์ ไชยเวศ. ปิโตรเลียมเทคโนโลยี. 2533.

ปฐ[illegible]ศ์ ปัญญาวิวานนท์. "การสร้างแบบจำลองและการซิมูเลตกระบวนการรีฟอร์มิง". วิทยานิพน์ปริญญามหาบัณฑิต สาขาวิชาวิศวกรรมเคมี. บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย. 2533.

[illegible]ิทธิ์สกุล . 2535. ความรู้เกี่ยวกับอุตสาหกรรมปิโตรเคมีและการพัฒนาอุตสาหกรรมปิโตรเคมีในประเทศไทย. กรุงเทพมหานคร. บริษัท เดียร์บุ๊ค จำกัด.

[illegible]. Applied Numerical Methods for Engineers. New [illegible]rk: John Wiley&Sons, Inc, 1994.

Bi[illegible] .T., Damiano, J.J., and Blau, G.E. "Nonlinear Parameter Estimation : a Case Study Comparison". AIChe Journal. 32(No.1, 1986): 29-45.

Bickle, G.M., Beltramini, J.N., and Duong D.Do. "Role of Sulfur in Catalytic Reforming of Hydrocarbons on Platinum-Rhenium/Alumina". Ind.Eng.Chem.Res. 29(1990): 1801-1807.

Brid, R.B., Stewart, W.E., and Lightfoot, E.N. Transpont Phenomena, New York: Wiley, 1970.

Constantinides Alkis. Applied Numerical Methods with Personal Computers. Singapore: McGraw-Hill, 1987.

Churin, E.J., Aloe, P.E., and Figoli, N.S. "The Role of Re and S in the Pt-Re-S/Alumina Catalyst". J.Catal.

99(1986): 39-52.

Das, S., Srivastava, R.D. and Saraf, D.N. "Studies on Bi-metallic Catalyst Behavior During Naphtha Reforming". J.Chem.Tech.Biotechnol. 43 (1988): 95-105.

Drew, T.B., Cokelet, G.R., Hoopes, J.W., JR. Advances in Chemical Engineering. New York:Academic Press, 1970.

Edgar, T.F., and Himmelblau, D.M. Optimization of Chemical Processes. New York: McGraw-Hill Book Company, 1989.

Eley, D.D., Pines, H., Weisz, P.B. Advances in Catalyst and Related Subjects Vol.13. New York: Academic Press, 1962.

————Advances in Catalysis Vol.23, New York: Academic press, 1973.

Fogler, H.S. "Elements of Chemical Reaction Engineering". Prentice-Hill, Inc., 1986.

Froment, G.F. "Model Discrimination and Parameter Estimation in Heterogeneous Catalysis". AIChE Journal. 21 (No.6, 1975): 1041-1057.

Froment, G.F.and Bischoff, K.B., Chemical Reactor Analysis and Design. John Wiley&Sons, 1958.

Haensel, Vladimir. "The Platforming Processes: Personal Recollections". Paper presented at the 183d meeting of the ACS, Las Vegas, 28 March-2 April 1983

Heinemann, H. Ind.End.Chem. 43(1951): 2098.

Himmelblau, D.M. and Bischoff, K.B. Process Analysis and Simulation. Willey, New York, 1947.

———— Process Analysis by Statistical Methods. New York: John Wiley&Sons, Inc., 1970.

Jothimurugesan, K., Bhatha, S., and Srivastava, R. D. "Kinetics of Dehydrogenation of methylcyclohexane over a Platinum-Rhenium-Alumina Catalyst in the Presence of Added Hydrogen". Ind.Eng.Chem.Fundam. 24(No.4, 1985): 433-438.

Kirk, R.E., and Othmer, D.F. Encyclopedia of Chemical Technology. Vol. 2. New York : The Interscience Encyclopedia, Inc.

Little, D.M. Catalytic Reforming. Okishoma:PennWell Books, 1985.

Marin, G.B. and Froment, G.F. Chem.Eng.Science. 37 (No.5, 1982):759-773.

McKetta, J.J. and Cumningham, W.A. Encyclopedia Chemical Processing and Design Vol.4. New York and Basel: Mercel Dekker, Inc.

Mills, G.A., Heinemann, H., Milliken, T.H., and Oblad, A.G. Ind.Eng.Chem. 1953: 136.

Parera, J.M., Beltramini, C.A., Querini, E.E., Ramage, M.P., Graziani, K.R., and Krambeck, F.J. "Development of MOBIL 's Kinetic Reforming Model". Chemical Engineering Science. 35(1980): 41-48.

Reid, C.R., Prausnitz, J.M., Poling, B.E. The Properties of Gases&Liquid. Fourth Edition. McGraw-Hill, 1987.

Satterfield, C.N. Heterogeneous catalysis in practics. New York: McGraw-Hill Book Company, 1980.

Selman, D.M., Voorhies, A.Jr. "Methylcyclopentane Reforming with Platinum-Rhenium Catalyst: Mechanism Studies". Ind.Eng.Chem.Prod.Res.Dev. 14(No.2, 1975): 118-123.

Shoichiro Nakamura. Applied Numerical Methods in C. Prentice-Hall International, Inc., 1993.

Shum, V.K., Butt, J.B., and Sachtler, M.H. "The Effects of Rhenium and Sulfur on the Activity Maintenance and Selectivity of Platinum/Alumina Hydrocarbon Conversion Catalyst". J.Catal. 96(1985): 371-380.

Steiner, H. Discussions of Faraday Society. 8(1950): 264-270.

Van Trimpont, P.A., Marin, G.B. and Froment, G.F. "Kinetics of Methylcyclohexane Dehydrogenation on Sulfided Commercial Platinum/Alumina and Platinum-Rhenium /Alumina Catalyst". Ind.Eng.Chem.Fundam. 25(1986): 544-553.

————"Kinetics of the reforming of C_7 hydrocarbons on a Commercial Pt-Re/Alumina Catalyst". Applied Catalysts. 24(1986): 53-68.

Wiseman, P. Petrochemical. Great Britain: Eills Horwood Limited, 1986.

Yang, K.H. and Hougen, O.A. Chem.Eng.Prog. 46(No.1, 1953): 134-137.

Yaws, C.L.and Chiang, P.Y. Hydrocarbon Processing. 1988: 81-84.

APPENDIX A

PROPERTY DATA BANK

The symbols and equations are shown below. The enthalpy and Gibbs energy of formation at 298.2 K are for the ideal-gas state. The reference states chosen for the elements are as follows :

MW	=	Molecular weight, g/mol
T_c	=	Critical temperature, K
P_c	=	Critical pressure, bar
V_c	=	Critical volume, cm^3/mole
Z_c	=	Critical compressibility factor, P_cV_c/RT_c
Omega	=	Pitzer's acentric factor
Dipm	=	Dipole moment, debyes

A, B, C, D = Constant to calculate the isobaric heat capacity of the ideal gas with Cp in J/(mol.K) and T in Kelvins

$$Cp = A + BT + CT^2 + DT^3$$

ΔH_f	=	heat of formation, kcal/g-mol
ΔG_f	=	Gibbs free energy, kJ/g-mol

Table A-1 Physical properties constant(Reid et al.(1987))

COMPONENT	MW	Tc K	Pc bar	Vc cm/mole	Zc	Omega	Dipm debyes	ΔH_f kcal/mol
HYDROGEN	2.016	33.0	12.9	64.3	0.303	-0.216	0	0
METHANE	16.043	190.4	46.0	99.2	0.288	0.011	0	-17.88
ETHANE	30.000	305.4	48.8	148.3	0.285	0.099	0	-20.23
PROPANE	44.094	369.8	42.5	203.0	0.281	0.153	0	-24.82
BUTANE	58.124	425.2	38.0	255.0	0.274	0.199	0	-30.15
n-PANTANE	72.151	469.7	33.7	304.0	0.263	0.251	0	-35.00
n-HEXANE	86.178	507.5	30.1	379.0	0.264	0.299	0	-39.96
CYCLOHEXANE	84.162	553.5	40.7	308.0	0.273	0.212	0.3	-29.43
METHYLCYCLOPENTANE	84.162	532.7	37.8	319.0	0.272	0.231	0	-25.50
2-METHYLPENTANE	86.178	497.5	30.1	367.0	0.267	0.278	0	-41.66
3-METHYLPENTANE	86.178	504.5	31.2	367.0	0.273	0.272	0	-41.02
2,2-DIMETHYLBUTANE	86.178	488.8	30.8	359.0	0.272	0.232	0	-44.35
2,3-DIMETHYLBUTANE	86.178	500.0	31.3	358.0	0.269	0.247	0	-42.49
BENZENE	78.114	562.2	48.9	259.0	0.271	0.212	0	19.82
TOLUENE	92.141	591.8	41.0	316.0	0.263	0.263	0.4	11.95
METHYLCYCLOHEXANE	98.189	572.2	34.7	368.0	0.268	0.236	0	-36.99
ETHYLCYCLOHEXANE	98.189	569.5	34.0	375.0	0.269	0.271	0	-30.37
n-HEPTANE	100.205	540.3	27.4	432.0	0.263	0.349	0	-44.91
2-METHYLHEXANE	100.205	530.4	27.3	421.0	0.261	0.329	0	-46.63
3-METHYLHEXANE	100.205	535.3	28.1	404.0	0.255	0.323	0	-45.98

Table A-2 Heat capacity, $Cp = A + BT + CT^2 + DT^3$

(Reid et al.(1987), J/g-mole)

COMPONENT	A	B	C	D
HYDROGEN	2.714E+01	9.274E-03	-1.381E-05	7.645E-09
METHANE	1.925E+01	5.213E-02	1.197E-05	-1.132E-08
ETHANE	5.409E+00	1.781E-01	-6.938E-05	8.713E-09
PROPANE	-4.224E+00	3.063E-01	-1.586E-04	3.210E-08
BUTANE	9.487E+00	3.313E-01	-1.108E-04	-2.822E-09
n-PANTANE	-3.626E+00	4.873E-01	-2.580E-04	5.300E-08
n-HEXANE	4.413E+00	5.820E-01	-3.119E-04	6.490E-08
CYCLOHEXANE	-5.454E+01	6.113E-01	-2.523E-04	1.321E-08
METHYLCYCLOPENTANE	-5.011E+01	6.381E-01	-3.642E-04	8.041E-08
2-METHYLPENTANE	-1.057E+01	6.180E-01	-3.573E-04	8.058E-08
3-METHYLPENTANE	-2.386E+00	5.690E-01	-2.870E-04	5.033E-08
2,2-DIMETHYLBUTANE	-1.663E+01	6.293E-01	-3.481E-04	6.580E-08
2,3-DIMETHYLBUTANE	-1.461E+01	6.150E-01	-3.376E-04	6.820E-08
BENZENE	-3.392E+01	4.739E-01	-3.017E-04	7.130E-08
TOLUENE	-2.435E+01	5.125E-01	-2.765E-04	4.911E-08
METHYLCYCLOHEXANE	-6.192E+01	7.842E-01	-4.438E-04	9.366E-08
ETHYLCYCLOHEXANE	-5.531E+01	7.511E-01	-4.396E-04	1.004E-07
n-HEPTANE	-5.146E+00	6.762E-01	-3.651E-04	7.658E-08
2-METHYLHEXANE	-3.939E+01	8.642E-01	-6.289E-04	1.310E-07
3-METHYLHEXANE	-7.046E+00	6.837E-01	-3.734E-04	7.834E-08

Table A-3 Gibbs free energy of formation of gas (Yaws and Chiang, (1988)), $\Delta G_f = A + BT + CT^2$, kJ/g-mole), by T in kelvins.

COMPONENT	A	B	C	ΔG_f at 298 K
n-HEXANE	-170.447	5.5417E-01	5.0303E-05	-0.84
2-METHYLPENTANE	-177.675	5.6303E-01	4.8313E-05	-5.60
3-METHYLPENTANE	-174.861	5.6271E-01	5.0351E-05	-2.70
2,2-DIMETHYLBUTANE	-189.225	5.8649E-01	4.7623E-05	-10.22
2,3-DIMETHYLBUTANE	-181.310	5.7783E-01	4.9722E-05	-4.70
METHYLCYCLOPENTANE	-110.437	4.7401E-01	4.9123E-05	35.18
BENZENE	81.512	1.5282E-01	2.6522E-05	129.41
n-HEPTANE	-191.520	6.5052E-01	5.6444E-05	7.35
ETHYLCYCLOPENTANE	-131.223	5.7136E-01	5.4772E-05	43.91
METHYLCYCLOHEXANE	-160.038	6.1255E-01	4.6303E-05	26.61
TOLUENE	47.813	2.3831E-01	3.1916E-05	121.66
2-METHYLHEXANE	-198.645	6.5837E-01	5.6475E-05	2.56
3-METHYLHEXANE	-196.032	6.5427E-01	5.6454E-05	3.95
2,2-DIMETHYLPENTANE	-209.894	6.8563E-01	5.6352E-05	-0.57
2,3-DIMETHYLPENTANE	-203.028	6.6456E-01	5.6286E-05	0.01
2,4-DIMETHYLPENTANE	-205.762	6.8189E-01	5.6332E-05	2.44
3,3-DIMETHYLPENTANE	-205.762	6.7887E-01	5.6327E-05	1.98
2,2,3-TRIMETHYLBUTANE	-209.101	6.9811E-01	5.2621E-04	3.61

APPENDIX B

GROUPS IN KINETIC EQUATIONS

Table B-1 Groups in kinetic equation for reactions on solid catalyst (Yang and Hougen (1950)).

Driving-Force Groups

Reaction	$A \rightleftharpoons R$	$A \rightleftharpoons R + S$	$A + B \rightleftharpoons R$	$A + B \rightleftharpoons R + S$
Adsorption of A controlling	$p_A - \frac{p_R}{K}$	$p_A - \frac{p_R p_S}{K}$	$p_A - \frac{p_R}{K p_B}$	$p_A - \frac{p_R p_S}{K p_B}$
Adsorption of B controlling	0	0	$p_B - \frac{p_R}{K p_A}$	$p_B - \frac{p_R p_S}{K p_A}$
Desorption of R controlling	$p_A - \frac{p_R}{K}$	$\frac{p_A}{p_S} - \frac{p_R}{K}$	$p_A p_B - \frac{p_R}{K}$	$\frac{p_A p_B}{p_S} - \frac{p_R}{K}$
Surface reaction controlling	$p_A - \frac{p_R}{K}$	$p_A - \frac{p_R p_S}{K}$	$p_A p_B - \frac{p_R}{K}$	$p_A p_B - \frac{p_R p_S}{K}$
Impact of A controlling (A not adsorbed)	0	0	$p_A p_B - \frac{p_R}{K}$	$p_A p_B - \frac{p_R p_S}{K}$
Homogeneous reaction controlling	$p_A - \frac{p_R}{K}$	$p_A - \frac{p_R p_S}{K}$	$p_A p_B - \frac{p_R}{K}$	$p_A p_B - \frac{p_R p_S}{K}$

Replacements in the General Adsorption Groups
$(1 + K_A p_A + K_B p_B + K_R p_R + K_S p_S + K_I p_I)^a$

Reaction	$A \rightleftharpoons R$	$A \rightleftharpoons R + S$	$A + B \rightleftharpoons R$	$A + B \rightleftharpoons R + S$
Where adsorption of A is rate controlling, replace $K_A p_A$ by	$\frac{K_A p_R}{K}$	$\frac{K_A p_R p_S}{K}$	$\frac{K_A p_R}{K p_B}$	$\frac{K_A p_R p_S}{K p_B}$
Where adsorption of B is rate controlling, replace $K_B p_B$ by	0	0	$\frac{K_B p_R}{K p_A}$	$\frac{K_B p_R p_S}{K p_A}$
Where desorption of R is rate controlling, replace $K_R p_R$ by	$K K_R p_A$	$K K_R \frac{p_A}{p_S}$	$K K_R p_S p_B$	$K K_R \frac{p_A p_B}{p_S}$
Where adsorption of A is rate controlling with dissociation of A, replace $K_A p_A$ by	$\sqrt{\frac{K_A p_R}{K}}$	$\sqrt{\frac{K_A p_R p_S}{K}}$	$\sqrt{\frac{K_A p_R}{K p_B}}$	$\sqrt{\frac{K_A p_R p_S}{K p_B}}$
Where equilibrium adsorption of A takes place with dissociation of A, replace $K_A p_A$ by and similarly for other components adsorbed with dissociation	$\sqrt{K_A p_A}$	$\sqrt{K_A p_A}$	$\sqrt{K_A p_A}$	$\sqrt{K_A p_A}$

Table B-1 (continued)

Where A is not adsorbed, replace $K_A p_A$ by	0	0	0	0
and similarly for other components that are not adsorbed				

Kinetic Groups

Adsorption of A controlling	k_A
Adsorption of B controlling	k_B
Desorption of R controlling	$k_R K$
Adsorption of A controlling with dissociation	k_A
Impact of A controlling	$k_A K_B$
Homogeneous reaction controlling	k

Surface Reaction Controlling

	$A \rightleftharpoons R$	$A \rightleftharpoons R + S$	$A + B \rightleftharpoons R$	$A + B \rightleftharpoons R + S$
Without dissociation	$k_s K_A$	$k_s K_A$	$k_s K_A K_B$	$k_s K_A K_B$
With dissociation of A	$k_s K_A$	$k_s K_A$	$k_s K_A K_B$	$k_s K_A K_B$
B not adsorbed	$k_s K_A$	$k_s K_A$	$k_s K_A$	$k_s K_A$
B not adsorbed, A dissociated	$k_s K_A$	$k_s K_A$	$k_s K_A$	$k_s K_A$

Exponents of Adsorption Groups

Adsorption of A controlling without dissociation	$n = 1$
Desorption of R controlling	$n = 1$
Adsorption of A controlling with dissociation	$n = 2$
Impact of A without dissociation $A + B \rightleftharpoons R$	$n = 1$
Impact of A without dissociation $A + B \rightleftharpoons R + S$	$n = 2$
Homogeneous reaction	$n = 0$

Surface Reaction Controlling

	$A \rightleftharpoons R$	$A \rightleftharpoons R + S$	$A + B \rightleftharpoons R$	$A + B \rightleftharpoons R + S$
No dissociation of A	1	2	2	2
Dissociation of A	2	2	3	3
Dissociation of A (B not adsorbed)	2	2	2	2
No dissociation of A (B not adsorbed)	1	2	1	2

APPENDIX C

EQUILIBRIUM CONSTANT

The equilibrium constant can be calculated from

$$\Delta G_{T,P} = - RT \ln K \qquad (C-1)$$

For any given reaction

$$\Delta G = \sum n_p \Delta G_f - \sum n_r \Delta G_r \qquad (C-2)$$

where

ΔG_f is the standard free energy of formation of component i (Table A-3).

n_p is stoichiometric coefficients of moles for product.

n_r is stoichiometric coefficients of moles for reactant.

APPENDIX D

VISCOSITY OF GAS MIXTURE

The viscosity of gas mixture in reforming system is predicted by method of Wilke (Reid et al. (1987)).

$$n_m = \frac{\sum y_i n_i}{\sum y_i \phi_{ij}} \qquad (D\text{-}1)$$

where

$$\phi_{ij} = \frac{[\,1 + (n_i/n_j)^{1/2} (M_j/M_i)^{1/4}\,]^2}{[8(1+M_i/M_j]^{1/2}} \qquad (D\text{-}2)$$

ϕ_{ji} is found by interchanging subscripts or by

$$\phi_{ji} = \frac{n_j M_i \phi_{ij}}{n_i M_j} \qquad (D\text{-}3)$$

where n_m is the viscosity of the mixture.

n_i is pure component viscosity of i in the mixture.

y_i is the mole fraction of i in the mixture.

M_i is the molecular weight of i in the mixture.

The pure component viscosity, n_i is predicted by method of Chung et al. (Reid et al.(1987)).

$$n = \frac{40.785\ F_c\ (M/T)^{1/2}}{V_c^{2/3}\Omega_v} \qquad (D\text{-}4)$$

where n is the viscosity, μP.

M is the molecular weight.

T is the temperature, K.

V_C is the critical volume, cm^3/mole.

Ω_v is viscosity collision.

$$\Omega_v = A(T^*)^{-B} + C[\exp(-DT^*)] + E[\exp(-FT^*)] \qquad (D\text{-}5)$$

where $T^* = 1.2593$, $A = 1.16145$, $B = 0.14874$, $C = 0.52487$, $D = 0.77320$, $E = 2.16178$, $F = 2.43787$

$$F_C = 1 - 0.2756.\omega + 0.05905\ \mu_r^4 + k \qquad (D\text{-}6)$$

ω is the acetric factor and k is a special correction for highly polar substances is shown in APPENDIX A.

$$\mu_r = \frac{1.313\ \mu}{(V_c\ T_c)^{1/2}} \qquad (D\text{-}7)$$

μ is dipole moment, debyes

μ_r is a dimensionless dipole moment

APPENDIX E

Table E-1 Input data used in the simulation

Case	Feed	Temperature °C	Pressure bar	H_2/Hydrocarbon
1	n-Hexane	500	1	11
2	n-Hexane	505	5	4
3	n-Hexane	505	15	4
4	n-Heptane	480	10	10
5	n-Heptane	490	10.5	20
6	n-Heptane	500	1	8
7	MCH	325	1	5
8	MCH	350	1	5
9	MCH	375	1	5
10	MCH	425	1	5
11	MCH+Tol (0.14:0.16)	325	1	5
12	MCH+Tol (0.14:0.16)	350	1	5
13	MCH+Tol (0.14:0.16)	375	1	5
14	MCH+Tol (0.14:0.16)	425	1	5
15	MCH+Tol (0.11:0.34)	350	1	5
16	MCH+Tol (0.08:0.52)	350	1	5
17	MCH	400	8.15	53
18	MCH	400	8.5	16
19	MCH	400	9	8

Table E-2 Input data used in the simulation of case 20

Composition of mixed C_6 to C_7 hydrocarbon feed	
Hydrocarbon type	Volume %
n-hexane	10.00
2MP	2.15
3MP	2.15
MCP	10.30
n-heptane	24.40
SBP_7	13.40
MBP_7	13.40
5N7	7.35
MCH	7.35
benzene	1.70
toluene	7.30
	100.00
Operating Condition	
pressure, bar	14.7
temperature, °C	505
$W/F°_{HC}$	20
Hydrogen/Hydrocarbon	4

Table E-3 Input data used in the simulation of case 21

Composition of mixed C_6 to C_7 hydrocarbon feed	
Hydrocarbon type	Volume %
n-hexane	30.75
MCP	11.20
n-heptane	30.75
MCH	11.20
benzene	3.10
toluene	13.00
	100.00
Operating Condition	
pressure, bar	1
temperature, °C	490
W/F°_{HC}	40
Hydrogen/Hydrocarbon	7

APPENDIX F

PARAMETER ESTIMATION

1. Introduction

Parameter estimation arises in fitting models containing several unknown parameters to experimental data through adjustment of these parameters. Model formulation is not a unique process; many different formulations may be used to fit the data and optimize model parameters. Of particular concern are formulations that are sufficiently accurate to represent physical or chemical phenomena.

Model formulation can be grouped into linear and nonlinear. The parameter estimation technique use to identify the parameter is therefore based on the types of the models.

2. Review of Numerical Methods for Nonlinear Model

In the studying of kinetic , rate coefficient (k) and the adsorption coefficients (K_A, K_R, K_S) are the parameter in the rate equation which is nonlinear model, the word nonlinear has meaning as applied to the independent variable of the model and also as applied to the

parameter. The nonlinear estimation problem appears as simply an *optimization problem*. The optimization techniques are following

1. **Direct Search Method** Direct method do not require the use of derivatives in determining the search direction. A direct search method simply selects a starting vector x^0, evaluates f(x) at x^0, and then selects another vector x^1, and evaluates f(x) at x^1. The both search direction and step length are chosen simultaneously After one or more stages, the value of $f(x^k)$ is compared with the best previous value of f(x) from among the previous stages, and the decision is made to continue or terminate the procedure. The disadvantage of direct search method is that they are slow in comparison with derivative or simplex methods, especially as the number of parameters becomes large.

2. **Simplex Method** The second method of minimization of a nonlinear objective function is by using a regular geometric patterns (a simplex) to select points at the vertices of the simplex at which to evaluate f(x). Each search direction points away from the point yielding the highest value of the objective function, point A in Figure F-1, through the centroid of the simplex, termed a *reflection*, is formed composed of the remaining old point and one new point, B. Thus, the direction of search changes, but the step size is fixed for a given size simplex.

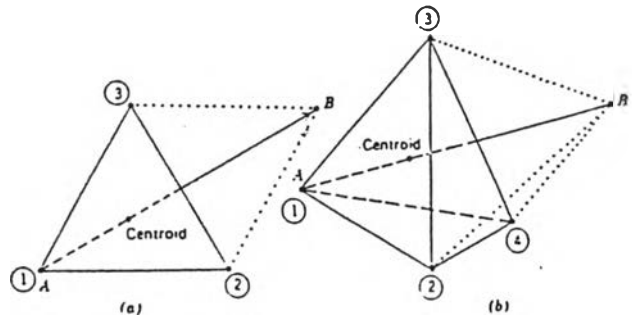

Figure F-1 Regular simplexes of two and three independent parameters.
(a) two variable simplex
(b) three variable simplex

Certain practical difficulties in the original procedure, namely that it did not provide for acceleration of the search, and encountered difficulty in carrying on the search in curving valleys or on curving ridges led to several improvements. In general it is recommended as better than the previously described direct search method because it takes less computer time even though the convergence to termination is slow.

3. **Gradient Method** This gradient(steepest-descent /ascent) method uses only the first derivatives of the f(x) in the calculations.

The gradient is the vector at a point x that gives the (local) direction of the greatest increase in f(x) and is orthogonal to the contour of f(x) at x. For maximization, the search direction is simply the gradient (when used the algorithm is called *steepest ascent*); for minimi-

zation, the search direction is the negative of the gradient(*steepest descent*).

$$\mathbf{s}^k = \nabla f(\mathbf{x})^k \qquad \text{(F-1)}$$

In steepest descent at the *k*th stage, the transition from point $\mathbf{x}^k$ to another point $\mathbf{x}^{k+1}$ can be viewed as given by the following expression:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta\mathbf{x}^k = \mathbf{x}^k + \lambda^k\mathbf{s}^k = \mathbf{x}^k - \lambda^k \nabla f(\mathbf{x}^k) \qquad \text{(F-2)}$$

where $\Delta\mathbf{x}^k$ = vector from $\mathbf{x}^k$ to $\mathbf{x}^{k+1}$

$\mathbf{s}^k$ = search direction, the direction of steepest descent

λ^k = scalar that determines the step length in direction $\mathbf{s}^k$

The negative of the gradient gives the direction for minimization but not the magnitude of the step to be taken, so that various steepest descent procedures are possible, depending upon the choice of λ^k.

4. **Newton's Method** Newton's method makes use of the second-order approximation of f(x) at $\mathbf{x}^k$, and thus employs second-order information about f(x), that is, information obtained from the second partial derivatives of f(x) with respect to the independent variables.

The minimum of f(x) in the direction of $\mathbf{x}^k$ is obtained by differentiating the approximation of f(x) with respect to each of the components of $\mathbf{x}$ and equating the

resulting expressions to zero to give

$$\nabla f(\mathbf{x}^k) = \nabla f(\mathbf{x}^k) + \mathbf{H}(\mathbf{x}^k)\Delta\mathbf{x}^k = 0 \qquad \text{(F-3)}$$

or

$$\mathbf{x}^{k+1} - \mathbf{x}^k = \Delta\mathbf{x}^k = -[\mathbf{H}(\mathbf{x}^k)]^{-1}\nabla f(\mathbf{x}^k) \qquad \text{(F-4)}$$

where $[\mathbf{H}(\mathbf{x}^k)]^{-1}$ is the inverse of the Hessian matrix $\mathbf{H}(\mathbf{x}^k)$.

Note that both the direction and step length are specified as a result of Eq.(E-3). If f(x) is actually quadratic, only one step is required to reach the minimum of f(x). However, for a general nonlinear objective function, the minimum of f(x) will not be reached in one step, so that Eq.(E-4) can be modified to

$$\mathbf{x}^{k+1} - \mathbf{x}^k = -\lambda^k [\mathbf{H}(\mathbf{x}^k)]^{-1}\nabla f(\mathbf{x}^k) \qquad \text{(F-5)}$$

where λ^k is the step length. The search direction **s** is given (for minimization) by

$$\mathbf{s}^k = -[\mathbf{H}(\mathbf{x}^k)]^{-1}\nabla f(\mathbf{x}^k) \qquad \text{(F-6)}$$

For the initial estimates of the parameters far from the final estimates, it is a characteristically slow method but converges rapidly near termination (in contrast to the gradient method which converges very slowly).

5. **Marquardt's Method** Marquardt, Levenberg, and others have suggested that the Hessian matrix of f(x) be

modified on each stage of the search as needed to ensure that the modified $\mathbf{H}(\mathbf{x})$, $\bar{\mathbf{H}}(\mathbf{x})$, is positive definite and well-conditioned. The procedure adds elements to the diagonal element of $\mathbf{H}(\mathbf{x})$

$$\bar{\mathbf{H}}(\mathbf{x}) = [\mathbf{H}(\mathbf{x}) + \beta \mathbf{I}] \tag{F-7}$$

where β is a positive constant large enough to make $\bar{\mathbf{H}}(\mathbf{x})$ positive definite when $\mathbf{H}(\mathbf{x})$ is not. Also it is possible to use

$$[\bar{\mathbf{H}}(\mathbf{x})]^{-1} = [\mathbf{H}^{-1}(\mathbf{x}) + \gamma \mathbf{I}] \tag{F-8}$$

Marquardt's method is definitely superior to either the Newton method or the Gradient method. Because either analytical or numerical derivatives at the minimum of f(x) are available. It is superior to the simplex method in that subsequent estimates of the precision of the parameters are easy to make. On the other hand, the simplex method has the advantage that the partial derivatives of f(x) need not be calculated at all, thus saving considerable computer time in estimation. For very complex models, the simplex method has proved the more effective in estimating the parameters in simulation studies.

3. Step of Parameter Estimation

1. Let Y_i is observation from experimental data and $\overline{Y_i}$ is expected response from the models.

For C_6 hydrocarbons

$$\overline{Y_1} = \frac{d\,X_{nHEX}}{d(W/F_{HC})} = -rate[1]-rate[2]-rate[6]$$

$$\overline{Y_2} = \frac{d\,X_{2MP}}{d(W/F_{HC})} = rate[1]-rate[3]-rate[4]-rate[8]$$

$$\overline{Y_3} = \frac{d\,X_{3MP}}{d(W/F_{HC})} = rate[2]+rate[3]-rate[7]$$

$$\overline{Y_4} = \frac{d\,X_{22DMB}}{d(W/F_{HC})} = rate[5]-rate[10]$$

$$\overline{Y_5} = \frac{d\,X_{23DMB}}{d(W/F_{HC})} = rate[4]-rate[5]$$

$$\overline{Y_6} = \frac{d\,X_{MCP}}{d(W/F_{HC})} = rate[6]-rate[7]$$

$$\overline{Y_7} = \frac{d\,X_{BZ}}{d(W/F_{HC})} = rate[7]$$

$$\overline{Y_8} = \frac{d\,X_{C5-}}{d(W/F_{HC})} = rate[8]+rate[9]+rate[10]$$

For C_7 hydrocarbons

$$\overline{Y_9} = \frac{d\,X_{nHEP}}{d(W/F_{HC})} = -rate[11]-rate[13]$$

$$\overline{Y_{10}} = \frac{d\,X_{Tol}}{d(W/F_{HC})} = rate[15]$$

$$\overline{Y_{11}} = \frac{d\,X_{MCH}}{d(W / F_{HC})} = rate[14]-rate[15]$$

$$\overline{Y_{12}} = \frac{d\,X_{5N7}}{d(W / F_{HC})} = rate[13]-rate[14]$$

$$\overline{Y_{13}} = \frac{d\,X_{iC7}}{d(W / F_{HC})} = rate[12]-rate[17]$$

$$\overline{Y_{14}} = \frac{d\,X_{C6}}{d(W / F_{HC})} = rate[16]+rate[17]$$

2. Let err is error between observation with expected response.

$$err_i^2 = (Y_i - \overline{Y_i})^2$$

3. Let E is total error

$$E_i^2 = \sum_{i=0}^{n} err_i^2$$

4. Optimization of E by using least square function in MATLAB program (Marquart' method-which is the best method as shown in the review of numerical method for nonlinear equation).

APPENDIX G

```
/*************************************************************************
* Program to simulate reforming processes for C6 to C7 hydrocarbons*
* Source file name is SIM.C                                              *
* File data bank for specific heat, Cp is cpmix.dat                      *
* File data bank for viscosity is vismix.dat                             *
*************************************************************************/

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

#include "sim.h"

FILE    *printer;
struct  cpm cpmix[MAX_COMPONENT];
struct  data_bank physical[MAX_COMPONENT];

double  Ke[REACTION];
static  char *feedname[MAXFEED] = {
                "Hydrogen", "C5 or C6", "n-Hexane", "MCP", "2MP",
                "3MP", "22DMB", 23DMB", "Bz", "Toluene", "ECP",
                "MCH", "n-Heptane", "SBP7", "MBP7"};

void main()                          /* main program */
{
double  heatreaction[REACTION];      /* for heat of reaction */
static  double  rate[REACTION];      /* for rate of reaction */
double  partialp[MAXFEED];           /* for partial pressure of C6 to C7*/

char    space[] ="                              ";
double  molefeed[MAXFEED+1] = {
                0,                  /*hydrogen*/
                0,                  /*C5 or C6*/
                0,0,0,0,0,0,0,      /*n-Hexane,MCP,2MP,3MP,22DMB,23DMB,Bz*/
                0,0,0,0,0,0         /*Tol,MCH,ECP,n-Heptane,SBP7,MBP7*/
  };

double  h2_c6 = 0;                  /*H2 in C6*/
double  c6_c6 = 0;                  /*C6 in C6*/
double  h2_c7 = 0;                  /*H2 in C7*/
double  c6_c7 = 0;                  /*C6 in C7*/

double  MOLEFEED[MAXFEED];
static  double  moleinitial[MAXFEED];
double  slope[MAXFEED];
double  totalmole = 0;
double  h2_hydrocarbon;
double  cpmixture, vismixture;

static  double  k[5][MAXFEED+1];    /*slope k[1]...k[4]*/
double  slopetemp = 0;
double  viscosity;

double  temperature;
double  temp1;
double  pressure;
double  feedhc = 0;
```

```
/************************************************************
*  Array of mole feed is                                    *
*    molefeed[0]   = Hydrogen                               *
*    molefeed[1]   = C5 or C6                               *
*    molefeed[2]   = n-Hexane                               *
*    molefeed[3]   = MCP                                    *
*    molefeed[4]   = 2MP                                    *
*    molefeed[5]   = 3MP                                    *
*    molefeed[6]   = 22DMB                                  *
*    molefeed[7]   = 23DMB                                  *
*    molefeed[8]   = Bz                                     *
*    molefeed[9]   = Toluene                                *
*    molefeed[10]  = ECP                                    *
*    molefeed[11]  = MCH                                    *
*    molefeed[12]  = n-Heptane                              *
*    molefeed[13]  = SBP7                                   *
*    molefeed[14]  = MBP7                                   *
************************************************************/

int     feed_count;
int     compo_count;
int     i = 0, j;
int     iii = 0, jjj = 0;
int     row = 1;                    /*row of the monitor*/
int     col = 1;
int     count = 0, n;
float   initial, final, h, printin;
int     nnp, nnc;

if((compo_count = read_datafile(NULL,NULL))<1)
  exit(1);
print("Number of component read from tables = %d\n",compo_count);
if((printer = setoutput())==NULL)
  exit(2);

/*recieve temperature, pressure etc.*/
clrscr();
row = 1; col = 1;
gotoxy(col,row);
cprintf("INPUT FEED :");
n = 0, i = 0;
row = 2;

while(i<MAXFEED) {
  char namebuf[20];
  double feedvalue;

  gotoxy(20,1); cprintf("count = %2d",i);
  gotoxy(3,row+1); cprintf(space);
  gotoxy(3,row+2); cprintf(space);
  gotoxy(3,row); cprint("\nFeed name(ctrl-Z to exit):");
  if((n = scanf("%s",namebuf)) == EOF) {
    break; }
    else {
      if((n = check_feedname(namebuf))>=0) {
        gotoxy(3,row+1); cprintf("\nFeed[%d] value : ",n);
        feedvalue = getvalue(20,row+1);
        scanf("%lf",&feedvalue);
        moleinitial[n] = molefeed[n] = feedvalue;
        gotoxy(55,row+1); cprintf("%-10.10s %lf",feedname[n],molefeed[n]);
        i++;
```

```
      } else {
        gotoxy(3,row+4); cprintf("Invalid feed name. Press any key...");
        getch();
        gotoxy(3,row+4); cprintf(space);
      }
    }
  }
feed_count = i;
if(i<1) {
  fprintf(stderr,"%s","No input feed. Program stop.\n");
  exit(4);                          /*no feed value inputed*/
} else {
  gotoxy(3,row);
  printf("Number of feed input = %d\n",feed_count);
  for(i=0; i<MAXFEED; i++) {
    if(feedname[i] == NULL) break;
    gotoxy(3,row+1+i);
    cprintf("%-10.10s %1f",feedname[i],molefeed[i]);
  }
  gotoxy(3,row+2+i); cprintf("\nPress any key to continue...");
  getch();
}

/*************************************************************
*  Input Data for Simulation                                 *
*  variable:       meaning                                   *
*  temperature     degree C                                  *
*  pressure        bar                                       *
*  h2_hydrocarbon  H2 to Hydrocarbon ratio                   *
*  initial         W/Hydrocarbon flow rate = 0               *
*  final           W/Hydrocarbon flow rate at W/             *
*  printin         Print interval                            *
*************************************************************/

clrscr();
row = 3; col = 1;
gotoxy(col,row);
cprintf("Feed Temperature (C) :");
temperature = (double)getvalue(col+30,row);

gotoxy(col,row+1);
cprintf("Pressure (bar):");
pressure = (double)getvalue(col+30,row+1);

gotoxy(col,row+1);
cprintf("H2 to Hydrocarbon ratio :");
h2_hydrocarbon = (double)getvalue(col+30,row+2);

gotoxy(col,row+3);
cprintf("Initial condition<0>:");
initial = (double)getvalue(col+30,row+3);

gotoxy(col,row+4);
cprintf("Final condition>0:");
final = (double)getvalue(col+30,row+4);

gotoxy(col,row+5);
cprintf("Step size:");
h = (double)getvalue(col+30,row+5);

gotoxy(col.row+6);
cprintf("Print interval:");
```

```
printin = (double)getvalue(col+30,row+6);
printf("\n %lf %lf %lf %lf %lf %lf %lf\n",
        temperature,pressure,h2_hydrocarbon,initial,final,h,printin);
getch();

printin = printin ==0?1:printin;
h       = h ==0?1:h;
nnp = ((final-initial)/printin)+1;
nnc = (printin/h)+1;
temperature = temperature+273.16;

/*Mole of feed into the reactor*/
for(i=0,feedhc=0.0; i<MAXFEED; i++) {
  feedhc = feedhc + molefeed[i];
}

/*Use hydrogen/hydrocarbon ratio to calculate h2 input*/
molefeed[0]    = h2_hydrocarbon*feedhc;
moleinitial[0] = molefeed[0];

/*************************************************************
*                       Working Program                      *
*************************************************************/
gotoxy(1,18);
highvideo();
textbackground(GREEN);
textcolor(BLINK);
cprintf("Working");
gotoxy(10,18);
normvideo();
textbackground(CYAN);
textcolor(BLUE);
cprintf("Finish at W/Hydrocarbon feed = %d",nnp);
gotoxy(1,21);
cprintf("Internal Loop =\n");

/*Calculate and print output*/
cpmixture  = cal_cpmixture(temperature-273.16,cpmix,MAXFEED,molefeed);
vismixture = cal_vismixture(molefeed,physical,temperature-273.16,MAXFEED);

/*print output data*/
fprintf(printer, "7.4f",0.0);
for(i=0; i<MAXFEED; i++) {
  fprintf(printer, "%7.4f",molefeed[i]);
}
fprintf(printer, "%7.4f", temperature-273.16);
fprintf(printer, "%7.4f", cpmixture);
fprintf(printer, "%7.4f", vismixture);
fprintf(printer, "\n");

/*************************************************************
*           Loop for Fourth-order Range-Kutta               *
*          Meaning of array k[1] of Range-Kutta             *
*    k1[0]   = H2                                           *
*    k1[1]   = C5 or C6                                     *
*    k1[2]   = n-Hexane                                     *
*    k1[3]   = MCP                                          *
*    k1[4]   = 2MP                                          *
*    k1[5]   = 3MP                                          *
*    k1[6]   = 22DMB                                        *
*    k1[7]   = 23DMB                                        *
*    k1[8]   = Bz                                           *
```

```
*    kl[8]    = Toluene                                          *
*    kl[10]   = ECP                                              *
*    kl[11]   = MCH                                              *
*    kl[12]   = n-Heptane                                        *
*    kl[13]   = SBP7                                             *
*    kl[14]   = MBP7                                             *
*    kl[15]   = temperature,K                                    *
************************************************************/
  for(iii=1; iii<=nnp; ++iii) {      /*print interval*/
    gotoxy(1,20);
    cprintf("W/Hydrocarbon feed =");
    gotoxy(23,20);
    cprintf("%d",iii);

    for(jjj=1; jjj<nnc; ++jjj) {     /*internal loop*/
      int  order;
      gotoxy(23,21);
      cprintf("%d",jjj);
      temp1 = temperature;

      for(order=1; order<=4; order++) {
        totalmole = 0;
        for(i=0; i<MAXFEED; ++i)
          totalmole = totalmole + molefeed[i];
        for(i=0; i<MAXFEED; ++i)
          partialp[i] = molefeed[i]*pressure/totalmole;

        ratec6c7(partialp,rate,(double)(temp1-273.16));
        heat(temp1-273.16,heatreaction);

        h2_c6 = (rate[5]+3*rate[6]-rate[7]-rate[8]-rate[9])*feedhc;
        c6_c6 = (rate[7]+rate[8]+rate[9])*2*feedhc;
        h2_c7 = (rate[12]3*rate[14]-rate[15]-rate[16])*feedhc;
        c6_c7 = (rate[15]+rate[16])*2*feedhc;

        k[order][0]  = h2_c6+h2_c7;
        k[order][1]  = c6_c6+c6+c7;
        k[order][2]  = (-rate[0]-rate[1]-rate[5])*feedhc;
        k[order][3]  = (rate[5]-rate[6])*feedhc;
        k[order][4]  = (rate[0]-rate[2]-rate[3]-rate[7])*feedhc;
        k[order][5]  = (rate[1]+rate[2]-rate[8])*feedhc;
        k[order][6]  = (rate[4]-rate[9])*feedhc;
        k[order][7]  = (rate[3]-rate[4])*feedhc;
        k[order][8]  = rate[6]*feedhc;
        k[order][9]  = rate[14]*feedhc;
        k[order][10] = (rate[12]-rate[13])*feedhc;
        k[order][11] = (rate[13]-rate[14])*feedhc;
        k[order][12] = (-rate[10]-rate[12])*feedhc;
        k[order][13] = (rate[10]-rate[11]-rate[15])*feedhc;
        k[order][14] = (rate[11]-rate[16])*feedhc;

/*Call cp mixture function*/
        cpmixture = cal_cpmixture(temp1-273.16,cpmix,MAXFEED,molefeed);
        for(i=0; slopetemp=0; i<REACTION; i++) {
          slopetemp = slopetemp-heatraction[i]*rate[i];
        }
        k[order][15] = slopetemp/cpmixture;
```

```
        switch(order) {
          case 1:
            for(n=0; n<MAXFEED; n++) {
              molefeed[n] = moleinitial[n]+h*k[order][n]/2;
            }
            temp1 = temperature+h*k[order][15]/2;
            break;
          case 2:
            for(n=0; n<MAXFEED; n++) {
              molefeed[n] = moleintial[n]+h*k[order][n]/2;
            }
            temp1 = temperature+h*k[order][15]/2;
            break;
          case 3:
            for(n=0; n<MAXFEED; n++) {
              molefeed[n] = moleintial[n]+h*k[order][n];
            }
            temp1 = temperature+h*k[order][15];
            break;
        }                              /*end switch*/
      }
      for(n=0; n<MAXFEED; n++) {
        molefeed[n]    = moleinitial[n]+h*slope[n];
        moleinitial[n] = molefeed[n];
      }
      temperature = temperature+h*slopetemp;
    }                                  /*End loop internal of RK*/
    gotoxy(23,20);
    cprintf("        ");
    gotoxy(23,21);
    cprintf("        ");

/*Call: heat capacity function and
  Call: viscosity of mixture*/

    cpmixture  = cal_cpmixture(temperature-273.16,cpmix,MAXFEED,molefeed);
    vismixture = cal_vismixture(MOLEFEED,physical,temperature-273.16,
                                MAXFEED);

/*Print text file to output*/

    fprintf(printer,"%7d",(iii*printin));
    for(i=0; i<MAXFEED; i++) {
      fprintf(printer,"\n%7.4f",molefeed[i]);
    }
    fprintf(printer,"%7.4f",temperature-273.16);
    fprintf(printer,"%7.4f",cpmixture);
    fprintf(printer,"%7.4f",vismixture);
    fprintf(printer,"%\n");
  }                                    /*End loop external of RK*/
  fclose(printer);
  sound(200);
  delay(100);
  nosound();
}                                      /*End main program*/
```

```
/*************************************************************
*      CPMIXTURE FUNCTION                                    *
*      degree = temperature(C)                               *
*      cpmix  = A,B,C,D                                      *
*      nc     = total number of component(int)               *
*      conc   = concentration of mixture                     *
*************************************************************/
double cal_cpmixture(double degree,struct cpm cpmixv[],int nc,double
                     conc[])
{
  int     i;
  double  cptemp;
  double  K;                          /*temperature in Kelvin*/
  double  total = 0;.............../*for tatal mole*/
  double  cpmixture = 0;
  K = degree+273.16;
  for(i=0; i<nc; i++) {
    total = total+conc[i];
  }
  for(i=0; i<nc; i++) {
    cptemp    = cpmixc[i].a+(cpmixc[i].b)*K+(cpmixc[i].c)*K*K
                +(cpmixc[i].d)*K*K*K;
    cpmixture = cpmixture+cptemp*conc[i]/total;
  }
  return cpmixture                    /*return value of the cp mixture*/
}

/*************************************************************
*      Function equilibrium(temperature,keq)                 *
*      Equilibrium constant is depend on temperature         *
*      Call by: temperature C, array of kequilibrium         *
*      function is not return value but use pointer          *
*************************************************************/
void  equilibrium(double t,double Ke[])
{
  float   R = 1.987;                  /*Gas constant 1.987 cal/mol.K*/
  double  T;
  T = t+273.16;                       /*change Temperature C to K*/

/*************************************************************************
*             Reaction    n-Hexane <==> 2MP                              *
*************************************************************************/
  Ke[0]  = exp(-(-7.228+(8.86E-03)*T-(1.99E-06)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*             Reaction    n-Hexane <==> 3MP                              *
*************************************************************************/
  Ke[1]  = exp(-(-4.414+(8.54E-03)*T-(4.80E-08)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*             Reaction    2MP <==> 3MP                                   *
*************************************************************************/
  Ke[2]  = exp(-(2.814-(3.20E-04)*T-(2.038E-06)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*             Reaction    2MP <==> 2,3DMB                                *
*************************************************************************/
  Ke[3]  = exp(-(-3.635+(1.48E-02)*T-(1.409E-06)*T*T)*1000/R/T/4.1841;
```

```
/*************************************************************************
*              Reaction    2,3DMB <==> 2,2DMB                            *
*************************************************************************/
  Ke[4]  = exp(-(-7.915+(8.66E-03)*T-(2.099E-06)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    n-Hexane <==> MCP + H2                        *
*************************************************************************/
  Ke[5]  = exp(-(60.01-(8.016E-02)*T-(1.18E-06)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    MCP <==> Bz + H2                              *
*************************************************************************/
  Ke[6]  = exp(-(191.949-(3.21E-01)*T-(2.26E-05)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    n-Heptane <==> SBP7                           *
*************************************************************************/
  Ke[7]  = exp(-(-4.497+(8.98E-03)*T+(1.60E-08)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    SBP7 <==> MBP7                                *
*************************************************************************/
  Ke[8]  = exp(-(-10.605+(2.20E-02)*T-(1.36E-07)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    n-Heptane <==> 5N7 + H2                       *
*************************************************************************/
  Ke[9]  = exp(-(60.297-(7.916E-02)*T-(1.672E-06)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    5N7 <==> MCH                                  *
*************************************************************************/
  Ke[10] = exp(-(-28.815+(4.119E-02)*T-(8.469E-06)*T*T)*1000/R/T/4.1841;

/*************************************************************************
*              Reaction    MCH <==> Toluene + 3H2                        *
*************************************************************************/
  Ke[11] = exp(-(207.851-(3.74E-03)*T-(1.439E-05)*T*T)*1000/R/T/4.1841;

}

/*************************************************************************
* Function viscosity_pure() to calculate viscosity of pure component    *
* viscosity_pure(physical properties, temperature Kelvin)               *
*                                                                       *
*    parameter use are                                                  *
*    1. Tc          *Critical temperature(K), Tc                        *
*    2. Vc          *Critical volume(cm^3/mole), Vc                     *
*    3. w           *Acentric factor, w                                 *
*    4. Diploem     *Dipole moment(debyes)                              *
*    5. MW          *Molecular weight(MW)                               *
*************************************************************************/

#define NC 20
double  viscosity_pure(double t, struct data_bank physic_constant)

{

  int     i;
  double  fc;                     /*fc = 1-0.2756w+0.059035ur^4*/
  double  tstar;                  /*dimension less temperature*/
```

```
                                    /*T* = 1.2593Tr*/
  double  omega;                    /*viscosity collision*/
  double  ur;                       /*ur = 131.3*dipolemoment/(VcTc)^0.5
  double  vis_pure;                 /*viscosity pure component*/
  double  T;

  T = t+273.16;                     /*change temperature C to K*/

  ur = 131.3*physic_constant.dipolem/(pow((double)physic_constant.tc
             *physic_constant.vc,0.5));

  fc = 1-0.2756*physic_constant.acentric+0.059035*pow(ur,4);

  tstar = 1.2593*(T)/physic_constant.tc;

  omega = 1.16145*pow(tstar,-0.14876)+0.52487*exp((double)-0.77320*tstar)
             +2.16178*exp((double)-2.43787*tstar);

  vis_pure = 40.785*fc*pow((double)physic_constant.mw*t,0.5)
             /pow((double)physic_constant.vc,(double)2/3)/omega;

  return vis_pure;                  /*return viscosity of pure component*/
}

/*********************************************************************
*  Function vismixture() is to calculate viscosity of gas mixture*
*  Call: Vismixture(molefeed,temperature,number of component)     *
*********************************************************************/

#define  MAXMIX 40

double  cal_vismixture(double c[],struct data_bank physical_vis[],
                                  double temp_mix,int count)
{
  double  nviscos[MAX_COMPONENT];  /*viscosity of pure component*/
  int     i,j;
  double  total_mole = 0;          /*total mole*/
  double  y[MAX_COMPONENT];        /*mole fraction*/
  double  phe[MAX_COMPONENT][MAX_COMPONENT];
  double  phe1, phe2;
  double  visc_mix = 0;            /*viscosity of gas mixture*/
  double  sumphe;                  /*summation of the mole_fraction
                                          multiply with phe*/

/*************************************************************
*    Search all viscosity of pure component                  *
*    and save in array of nviscos                            *
*************************************************************/

  for(i=0; i<count; i++) {
    nviscos[i] = viscosity_pure(temp_mix,physical_vis[i]);
  }

/*************************************************************
*    Search all mole fraction and save in array of y         *
*************************************************************/

  for(i=0; i<count; i++) {
    total_mole = total_mole + c[i];
  }
```

```
  for(i=0; i<count; i++) {
    y[i] = c[i]/total_mole;
  }

/*************************************************************
*                        Search all phe                      *
*************************************************************/

  for(i=0; i<count; i++) {          /*first loop*/
    for(j=0; j<count; j++) {        /*second loop*/
      if(i!=j) {
        if(j<i)
          phe[i][j] = nviscos[i]*physical_vis[j].mw*phe[j][i]
                      /mviscos[j]/physical_vis[i].mw;
        else {
          phe1 = pow((double)(1+(sqrt(nviscos[i]/nviscos[j]))
                 *pow((double)physical_vis[j].mw/physical_vis[i].mw,
                 0.25)),2);
          phe2 = sqrt((double)8*(1+(physical_vis[i].mw/
                 physical_vis[j].mw)));
          phe[i][j] = phe1/phe2;
        }
      }                             /*End if i=j*/
    }                               /*End second loop*/
  }                                 /*End first loop*/

/*************************************************************
*           Search for viscosity of gas mixture              *
*************************************************************/

  for(i=0; i<count; i++) {
    sumphe = y[i];
    for(j=0; j<count; j++) {
      if(j!=i)
        sumphe = sumhe + y[j]*phe[i][j];
    }
    visc_mix = visc_mix + y[i]*nviscos[i]/sumphe;
  }
  return visc_mix;                  /*return viscosity of gas mixture*/
}

/*************************************************************
*    Function heat(temperature,heat of reaction)             *
*    Because heat of reaction is depend on temperature       *
*    Call by: temperature C,array of heat of reaction        *
*    function is not return value but use pointer            *
*************************************************************/

void  heat(double  temp, double  heat_reaction[])
{
  int  i;
  temp = temp+273.16;

/*************************************************************************
*             Reaction    n-Hexane <==> 2MP                              *
*************************************************************************/

  heat_reaction[0]  = -1.70*1000 + ((-6.15*(temp-298)) + (3.60E-02)*
                      (pow(temp,2)-pow(298,2))/2 + (-4.54E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (1.568E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;
```

```
/*************************************************************************
*              Reaction    n-Hexane <==> 3MP                             *
*************************************************************************/

  heat_reaction[1]  = -1.06*1000 + ((2.027*(temp-298)) + (-1.30E-02)*
                      (pow(temp,2)-pow(298,2))/2 + (2.49E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (-1.457E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    2MP <==> 3MP                                  *
*************************************************************************/

  heat_reaction[2]  = 0.64*1000 + ((8.184*(temp-298)) + (-4.90E-02)*
                      (pow(temp,2)-pow(298,2))/2 + (7.03E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (-3.025E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    2MP <==> 2,3DMB                               *
*************************************************************************/

  heat_reaction[3]  = -0.83*1000 + ((-4.04*(temp-298)) + (-3.03E-03)*
                      (pow(temp,2)-pow(298,2))/2 + (1.97E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (-1.236E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    2,3DMB <==> 2,2DMB                            *
*************************************************************************/

  heat_reaction[4]  = -0.86*1000 + ((-2.02*(temp-298)) + (1.430E-02)*
                      (pow(temp,2)-pow(298,2))/2 + (-1.05E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (3.005E-10)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    n-Hexane <==> MCP + H2                        *
*************************************************************************/

  heat_reaction[5]  = 14.46*1000+(((-1.8557E+01)*(temp-298))+(6.54E-02)
                      *(pow(temp,2)-pow(298,2))/2 + (-6.61E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (2.316E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    MCP <==> Bz + H2                              *
*************************************************************************/

  heat_reaction[6]  = 45.32*1000+(((9.76E+01)*(temp-298))+(-1.36E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (2.107E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (1.38E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    2MP + H2 <==> 2C5-                            *
*************************************************************************/

  heat_reaction[7]  = -7.02*1000 + ((-9.9647*(temp-298)) + (-1.08E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (1.472E-04)*
                      (pow(temp,3)-pow(298,3))/3 + (-5.80E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;
```

```
/*****************************************************************
*              Reaction    3MP + H2 <==> 2C5-                     *
*****************************************************************/

  heat_reaction[8]  = -7.66*1000+(((-1.81E+01)*(temp-298))+(-5.94E-02)
                      *(pow(temp,2)-pow(298,2))/2 + (7.692E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (-2.78E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*****************************************************************
*              Reaction    2,2DMB + H2 <==> 2C5-                  *
*****************************************************************/

  heat_reaction[9]  = -4.327*1000 + ((-3.905*(temp-298)) + (-1.19E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (1.380E-04)*
                      (pow(temp,3)-pow(298,3))/3 + (-4.59E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*****************************************************************
*              Reaction    n-Heptane <==> SBP7                    *
*****************************************************************/

  heat_reaction[10] = -1.399*1000+((-18.072E+01)*(temp-298))+(9.77E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (-1.361E-04)*
                      (pow(temp,3)-pow(298,3))/3 + (2.076E-07)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*****************************************************************
*              Reaction    SBP7 <==> MBP7                         *
*****************************************************************/

  heat_reaction[11] = -2.181*1000 + ((4.382*(temp-298)) + (-3.42E-02)
                      *(pow(temp,2)-pow(298,2))/2 + (6.150E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (-2.920E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*****************************************************************
*              Reaction    n-Heptane <==> 5N7 + H2                *
*****************************************************************/

  heat_reaction[12] = 14.57*1000 + ((-23.024*(temp-298)) + (8.417E-02)
                      *(pow(temp,2)-pow(298,2))/2 + (-8.831E-05)*
                      (pow(temp,3)-pow(298,3))/3 + (3.147E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*****************************************************************
*              Reaction    5N7 <==> MCH                           *
*****************************************************************/

  heat_reaction[13] = -6.62*1000 + ((-6.61*(temp-298)) + (3.31E-02)*
                      (pow(temp,2)-pow(298,2))/2 + (-4.20E-06)*
                      (pow(temp,3)-pow(298,3))/3 + (-6.74E-09)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*****************************************************************
*              Reaction    MCH <==> Toluene + 3H2                 *
*****************************************************************/

  heat_reaction[14] = 48.948*1000 + ((118.99*(temp-298)) + (-2.439E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (1.259E-04)*
                      (pow(temp,3)-pow(298,3))/3 + (-2.162E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;
```

```
/*************************************************************************
*              Reaction    SBP7 + H2 <==> 2C6-                           *
*************************************************************************/

  heat_reaction[15] = -4.957*1000 + ((6.596*(temp-298)) + (-2.412E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (2.811E-04)*
                      (pow(temp,3)-pow(298,3))/3 + (-1.068E-07)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

/*************************************************************************
*              Reaction    MBP7 + H2 <==> 2C6-                           *
*************************************************************************/

  heat_reaction[16] = -2.776*1000+((2.214.99*(temp-298))+(-2.0697E-01)
                      *(pow(temp,2)-pow(298,2))/2 + (2.196E-04)*
                      (pow(temp,3)-pow(298,3))/3 + (-7.758E-08)*
                      (pow(temp,4)-pow(298,4))/4)/4.1841;

)

/*************************************************************************
* rateC6C7()                                                             *
* Function to calculate rate of reaction of C6 to C7 hydrocarbon*
* Because heat of reaction is depend on temperature                      *
* Call by : ratec6c7(partial pressure, rate, temperature                 *
*************************************************************************/

/*******************************************
*       Reaction of C6 hydrocarbon         *
*******************************************/

/*******************************************
*       n-Hexane <==> 2MP                  *
*******************************************/
        const double A1 =   .   E+  ;           /*kmole/kg cat.hr*/
        const double E1 =   .   E+05;           /*J/mole*/

/*******************************************
*       n-Hexane <==> 3MP                  *
*******************************************/
        const double A2 =   .   E+  ;
        const double E2 =   .   E+05;

/*******************************************
*       2MP <==> 3MP                       *
*******************************************/
        const double A3 =   .   E+  ;
        const double E3 =   .   E+05;

/*******************************************
*       2MP <==> 2,3DMB                    *
*******************************************/
        const double A4 =   .   E+  ;
        const double E4 =   .   E+05;

/*******************************************
*       2,3DMB <==> 2,2DMB                 *
*******************************************/
        const double A5 =   .   E+  ;
        const double E5 =   .   E+05;
```

```
/*********************************************
*       n-Hexane <==> MCP + H2                *
*********************************************/
        const double A6 =  .   E+  ;
        const double E6 =  .   E+05;

/*********************************************
*       MCP <==> Bz + H2                      *
*********************************************/
        const double A7 =  .   E+  ;
        const double E7 =  .   E+05;

/*********************************************
*       2MP + H2 <==> 2C5-                    *
*********************************************/
        const double A8 =  .   E+  ;
        const double E8 =  .   E+05;

/*********************************************
*       3MP + H2 <==> 2C5-                    *
*********************************************/
        const double A9 =  .   E+  ;
        const double E9 =  .   E+05;

/*********************************************
*       2,2DMB + H2 <==> 2C5-                 *
*********************************************/
        const double A10 =  .   E+  ;
        const double E10 =  .   E+05;

/*********************************************
*       Reaction of C7 hydrocarbon            *
*********************************************/

/*********************************************
*       n-Heptane <==> SBP7                   *
*********************************************/
        const double A11 =  .   E+  ;          /*kmole/kg cat.h*/
        const double E11 =  .   E+05;          /*J/mole*/

/*********************************************
*       SBP7 <==> MBP7                        *
*********************************************/
        const double A12 =  .   E+  ;
        const double E12 =  .   E+05;

/*********************************************
*       n-Heptane <==> 5N7 + H2               *
*********************************************/
        const double A13 =  .   E+  ;
        const double E13 =  .   E+05;

/*********************************************
*       5N7 <==> MCH                          *
*********************************************/
        const double A14 =  .   E+  ;
        const double E14 =  .   E+05;
```

```
/*****************************************
*       MCH <==> Toluene + 3H2           *
*****************************************/
        const double A15 =  .   E+  ;
        const double E15 =  .   E+05;

/*****************************************
*       SBP7 + H2 <==> 2C6-              *
*****************************************/
        const double A16 =  .   E+  ;
        const double E16 =  .   E+05;

/*****************************************
*       MBP7 + H2 <==> 2C6-              *
*****************************************/
        const double A17 =  .   E+  ;
        const double E17 =  .   E+05;

void  ratec6c7(double  partial[], double  rate[], double  temp)
{
  int     i;
  double  phe;
  double  acid_ad, metal_ad;
  double  R = 1.987*4.1841          /*R = gas constant*/
                                    /*  = 1.987 cal/mole*/
                                    /*1 cal = 4.1841 J*/

/*common adsorption term for C6 hydrocarbon*/
  const double  Khex  =  .   ;
  const double  Kmcp  =  .   E+  ;

/*common adsorption term for C7 hydrocarbon*/
  const double  Kc6-  =  . ;
  const double  Kp7   =  . ;
  const double  Ktola =  . ;
  const double  Knhep =  . ;
  const double  Kmch  =  . ;
  const double  Ktolm =  . ;

/*for C6 hydrocarbon*/
  phe      = pow((1 + (Khex*(partial[2]+partial[4]+partial[5]+partial[6]
             +partial[7])/partial[0]) + (Kmcp*partial[3]/partial[0])),2);

/*for C7 hydrocarbon*/
  acid_ad  = (partial[0]+Kc6-*partial[1]+Kp7*(partial[12]*partial[13]
             +partial[14])+Ktola*partial[9]*partial[0])/partial[0];
  metal_ad = 1+Knhep*partial[12]+Kmch*partial[11]+Ktolm*partial[9];

/*Call function equilibrium() for Ke[]*/
  equilibrium(temp,Ke);

/*Isomerization*/
  temp = temp + 273.16;

/*****************************************************************
*               Rate equation for C6 hydrocarbons                *
*****************************************************************/

  rate[0]  = A1*exp(-E1/(R*temp))*(partial[2]-(partial[4]/Ke[0]))/
             (partial[0]*phe);
  rate[1]  = A2*exp(-E2/(R*temp))*(partial[2]-(partial[5]/Ke[1]))/
             (partial[0]*phe);
```

```
  rate[2]  = A3*exp(-E3/(R*temp))*(partial[4]-(partial[5]/Ke[2]))/
             (partial[0]*phe);
  rate[3]  = A4*exp(-E4/(R*temp))*(partial[4]-(partial[7]/Ke[3]))/
             (partial[0]*phe);
  rate[4]  = A5*exp(-E5/(R*temp))*(partial[7]-(partial[6]/Ke[4]))/
             (partial[0]*phe);
  rate[5]  = A6*exp(-E6/(R*temp))*(partial[2]-(partial[3]*partial[0]
             /Ke[5]))/(partial[0]*phe);
  rate[6]  = A7*exp(-E7/(R*temp))*(partial[3]-(partial[8]*
             pow(partial[0],3))/Ke[6])/(partial[0]*phe);
  rate[7]  = A8*exp(-E8/(R*temp))*(partial[4]/phe;
  rate[8]  = A9*exp(-E9/(R*temp))*(partial[5]/phe;
  rate[9]  = A10*exp(-E10/(R*temp))*(partial[6]/phe;

/*************************************************************************
*               Rate equation for C7 hydrocarbons                        *
*************************************************************************/

  rate[10] = A11*exp(-E11/(R*temp))*(partial[12]-(partial[13]/Ke[7]))/
             (partial[0]*acid_ad);
  rate[11] = A12*exp(-E12/(R*temp))*(partial[13]-(partial[14]/Ke[8]))/
             (partial[0]*acid_ad);
  rate[12] = A13*exp(-E13/(R*temp))*(partial[12]-(partial[10]*partial[0]
             /Ke[9]))/(partial[0]*acid_ad);
  rate[13] = A14*exp(-E14/(R*temp))*(partial[10]-(partial[11]/Ke[10]))/
             (partial[0]*acid_ad);
  rate[14] = A15*exp(-E15/(R*temp))*(partial[11]-(partial[9]
             *pow(partial[0],3))/Ke[11])/metal_ad;
  rate[15] = A16*exp(-E16/(R*temp))*partial[13]/(partial[0]*acid_ad);
  rate[16] = A17*exp(-E17/(R*temp))*partial[14]/(partial[0]*acid_ad);

}                                    /*end function ratec6c7()*/

/*Function getvalue()*/
double  getvalue(x,y)
int     x,y;
{
  char    s[30];
  int     c;
  int     i = 0;
  int     check;
  int     get_ch;
  double  num;
  gotoxy(x,y);
  check = 1;
  while((c=getch()) != '\r'  check == 1) {
    if(c == 0x1B) {
      gotoxy(3,y+1);
      cprintf("Exit program<Y/N> :");
      if((get_ch == 'y'  get_ch == 'Y') {
        gotoxy(3,y+1);
        cprintf("          ");
        exit(1);
      }
      else if(get_ch == 0) {
        getch();
        gotoxy(3,y+1);
        cprintf("          ");
        gotoxy(x+1,y);
      }
      else {
        gotoxy(3,y+1);
```

```
        cprintf("          ");
        gotoxy(x+1,y);
      }
    }
    else if(c == 0x8) {
      i = 0;
      check = 1;
      gotoxy(x,y);
        cprintf("          ");
        gotoxy(x,y);
    }
    else if(c == 0){
      getch();
      i = 0;
      check = 1;
      gotoxy(x,y);
        cprintf("          ");
        gotoxy(x,y);
    }
    else if(c == '\r') {
      check = 1;
      i = 0;
    }
    else {
      if(c>'0' && c<='9'  c=='.') {
        check = 2;
        cprintf("%c",c);
        s[i] = c;
        i = i+1;
      }                                /*remove character*/
      else {
        i = 0;
        check = 1;
        sound(200);
        delay(100);
        nosound();
        gotoxy(x,y);
        cprintf("          ");
        gotoxy(x,y);
      }
    }
  }                                    /*end while*/

  s[i] = '\0';
  if(s[0] == NULL)
    return 12;
  num = u_atoi(s);
  return(num);
}

double  u_atoi(s)
char    s[];
{
  int     i;
  int     j;
  double  n, n1 = 0;
  n = 0;
  for(i=0; s[i]>='0'&& s[i]<='9'&& s[i]!='.'; i=i+1)
    n = 10*n + s[i] - '0';
  if(s[i] == '.') {
    for(i=i+1,j=1; s[i]>='0' && s[i]<='9'; i=i+1,j=j+1)
      n1 = n1 + (s[i]-'0')/pow(10,j);
```

```
        n = n + n1;
    }
    return n;
}

int check_feedname(char *fname)
{
    int  i;
    for(i=0; i<MAX_COMPONENT, i++) {
      if(stricmp(fname,feedname[i])==0)
        return i;
    }
    if(i>=MAX_COMPONENT)
      return -1;
}
```

VITA

Miss Aussanee Wapakpetch was born on October 5, 1967 in Bangkok, Thailand. She finished her primary school from Somthawin Rachadamri and secondary school from Bordindaja. She earned her B.Sc.(Chemistry) from Srinakharinwirot University, Bangkok. After graduated, she joined The Shell Company of Thailand in Marketing Department for a half year. She then pursues her post-graduate study at the Department of Chemical Engineering, Faculty of Engineering, Chulalongkorn University.