The Institute of Paper Chemistry

Appleton, Wisconsin

Doctor's Dissertation

A Dynamic Model of Kraft-Anthraquinone Pulping

Mark Alan Burazin

June, 1986
(Reprinted 1990)

A DYNAMIC MODEL OF KRAFT-ANTHRAQUINONE PULPING

A thesis submitted by

Mark Alan Burazin

B.S. 1979, Carroll College

M.S. 1981, Lawrence University

in partial fulfillment of the requirements
of The Institute of Paper Chemistry
for the degree of Doctor of Philosophy
from Lawrence University,
Appleton, Wisconsin

June, 1986

TABLE OF CONTENTS

SUMMARY

Since the introduction of the kraft pulping process the ability to predict the behavior of kraft pulping has steadily improved. With the advent of sufficient computing resources, dynamic models have been developed to examine the combined effects of reaction and diffusion, in some cases predicting lignin and yield profiles within the chip. The predictive capabilities of these models are limited by the available kinetic data and by restrictive assumptions which limit the useful simulation space.

The shortcomings of prior dynamic kraft pulping models prompted the development of a dynamic, distributed-parameter model of a wood chip for kraft, kraft-anthraquinone, soda-anthraquinone, and soda pulping conditions. The model uses improved kinetic equations and fewer, less restrictive assumptions. The reaction rates of lignin, cellulose, glucomannan, xylan, extractives, acetyl groups, and pulp viscosity are modeled using differential equations valid during the initial, bulk, and residual delignification phases. Nonlinear heat and mass transfer are modeled in three space dimensions.

A competitive, sequential experimental design strategy was used to select the reaction equation for each species from several mechanistically based candidates and to simultaneously determine the kinetic parameters of the best candidates.

The reaction and diffusion equations are solved using the method of lines with arbitrary-order, finite difference discretization. The finite difference formulas are generated automatically, incorporating boundary conditions when advantageous. The finite difference grid is automatically generated, optionally concentrating grid points near the chip surface to maximize accuracy. Auxiliary programs have been developed to generate plots of one-dimensional and two-dimensional chip sections along any line or plane of grid points, respectively.

The model has been tested and has been found to accurately predict the results of experiments from the reaction kinetics study. The numerical solution method accurately predicts average chip properties, such as lignin and cellulose contents, even when optimized for speed at the expense of accuracy.

The model is less accurate in predicting industrial cooks, especially lignin at long times and low liquor to wood ratio, and xylan at low effective alkali and low liquor:wood. Continuing the reaction kinetics study with liquor:wood as an added variable would quickly improve the accuracy of industrial cook simulations.

GLOSSARY

A	area
a,b,c...	variables
Ac	acetyl groups
a.d.	air dry
AE	activation energy
AQ	anthraquinone
Bi	Biot number = $K L_h/D$
C	carbohydrates
C	joint design criterion
°C	degrees Celsius
cp	centipoise
D	diffusivity
D	model discrimination criterion
d	ordinary differential operator
∂	partial differential operator
E	extractives
E	parameter estimation criterion
e	exponential operator
ECCSA	effective capillary cross-sectional area
F	Faraday
°F	degrees Fahrenheit
f	generic function
G	G factor (related to cellulose cleavage extent)
H	H-factor (related to relative delignification extent)
h	grid spacing

H_t	fractional hemicellulose content at end of initial phase
H^+	hydrogen ion
K	mass transfer coefficient
k	reaction rate constant
°K	degrees Kelvin
L	lignin
l	longitudinal
L_h	characteristic length (chip surface to center)
ln	natural logarithm
log	base 10 logarithm
l_t	fractional lignin content at end of initial phase
l:w	liquor:wood
l_+	cationic limiting conductance
l_-	anionic limiting conductance
$\underline{M}$	molar concentration (g-mole/dm^3)
MW	molecular weight
μl	liquor viscosity
μ_p	pulp viscosity
μ_s	solvent viscosity
μw	water viscosity
NaOH	sodium hydroxide
NaSH	sodium hydrosulfide
Na_2S	sodium sulfide
n_+	cation valence
n_-	anion valence
o.d.	oven dry
ODE	ordinary differential Eq.

ODW	oven dry wood
PDE	partial differential Eq.
pH	-log [H+]
Π	probability
π	circle circumference:circle diameter
Π_b	probability of best (most likely) model
R	gas constant
r	radial
RH	relative humidity
S	% dissolved solids/100
T	temperature
$T^\dagger$	1/T - 1/433K
t	time or tangential
u	generic concentration (mass or molar)
V	volume
Y	%yield/100
$\approx$	approximately equal to
$\vert$	evaluated at
$\equiv$	identical to
∞	infinity
$\int$	integral
[=]	in units of
[ ]	mass or molar concentration
$\Vert$	parallel
$\propto$	proportional to
:	ratio

INTRODUCTION

The kraft (sulfate) pulping process was introduced by C. F. Dahl in 1879 when he substituted sodium sulfate for sodium carbonate as a makeup chemical in the soda pulping of straw.[1] Today the kraft process is the most popular process for the manufacture of wood pulp, accounting for 78% of United States pulp production in 1984.[2]

"The main advantages of sulfate pulping, listed below, give a first characterization of the process and the resulting pulps:

- low demands on wood species and wood quality, including all types of softwoods and hardwoods, even in combination, and toleration of high amounts of extractives as well as considerable portions of decayed wood and bark residues
- short cooking times
- well established processing of the spent liquor, including the recovery of the pulping chemicals, generation of process heat, and the production of valuable by-products such as tall oil and turpentine from pine species
- excellent pulp strength properties."[3]

KRAFT PULPING MODELS

Kraft pulping is a fascinating process which is extremely challenging to model. The reactants in wood are present as a heterogeneous mixture of polymers. The pulping reagents, sodium hydroxide (NaOH) and sodium hydrosulfide (NaSH), must diffuse into the wood to react. Sodium hydroxide greatly influences diffusion rates in wood by swelling the wood structure. This swelling is highly nonlinear in NaOH concentration [NaOH]. As pulping proceeds, the polymers are degraded, opening up the wood structure and significantly increasing diffusion rates. Under typical industrial conditions reaction rates are often comparable to diffusion rates; approximations corresponding to reaction limited or diffusion limited conditions do not apply.

Diffusion in Wood

The relationship between diffusivity in pulping liquor and diffusivity in liquor saturated wood has been approximated experimentally as the effective capillary cross-sectional area (ECCSA). ECCSA is defined as the ratio of conductivity through liquor saturated wood to conductivity through liquor. ECCSA has been determined as a function of pH (= log [NaOH] + 14) at 100% yield for aspen[4] and spruce,[5] and as a function of yield at pH 13.2 for pine.[5] The effect of pH on the ECCSA of spruce is shown in Fig. 1. The effect of yield on pine ECCSA is shown in Fig. 2.

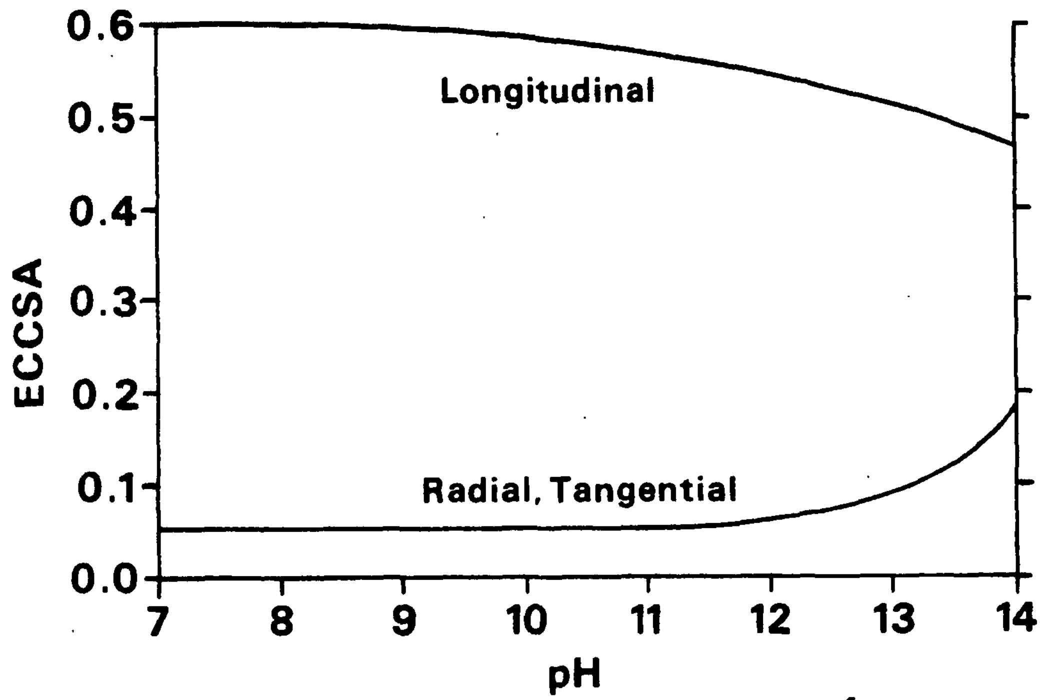


Fig. 1. ECCSA vs. pH for 100% yield spruce.[5]

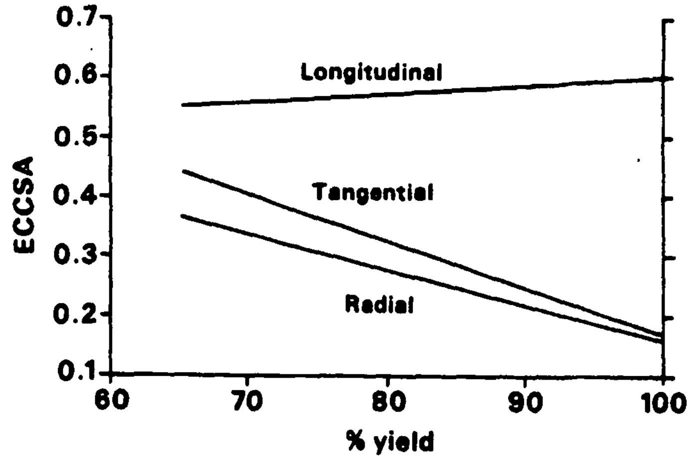


Fig. 2. ECCSA vs. % yield for pine at pH 13.2.[5]

Wood Composition

The major components of wood are lignin, carbohydrates, and extractives. Lignin is a highly branched three-dimensional heterogeneous polymer of three major precursors joined together by several types of bonds. Figure 3 shows Glasser and Glasser's softwood lignin model.[16] Lignin undergoes degradation reactions which produce dissolved lignin and condensation (cross-linking) reactions which produce residual lignin. Residual lignin is much less reactive than native lignin but otherwise is very difficult to distinguish from native lignin.

Carbohydrates are present as cellulose, a high molecular weight linear polymer of glucose, and hemicelluloses, low molecular weight polymers of several sugars. Structures of important softwood carbohydrates are shown in Fig. 4 to 6. The hemicelluloses are composed of low molecular weight linear backbones with short branches attached to the backbones. Carbohydrates undergo three important simultaneous reactions, peeling, stopping, and cleavage. The peeling reaction depolymerizes a carbohydrate polymer chain one sugar unit at a time from the reducing (hemiacetal) end of the chain. The stopping reaction oxidizes the

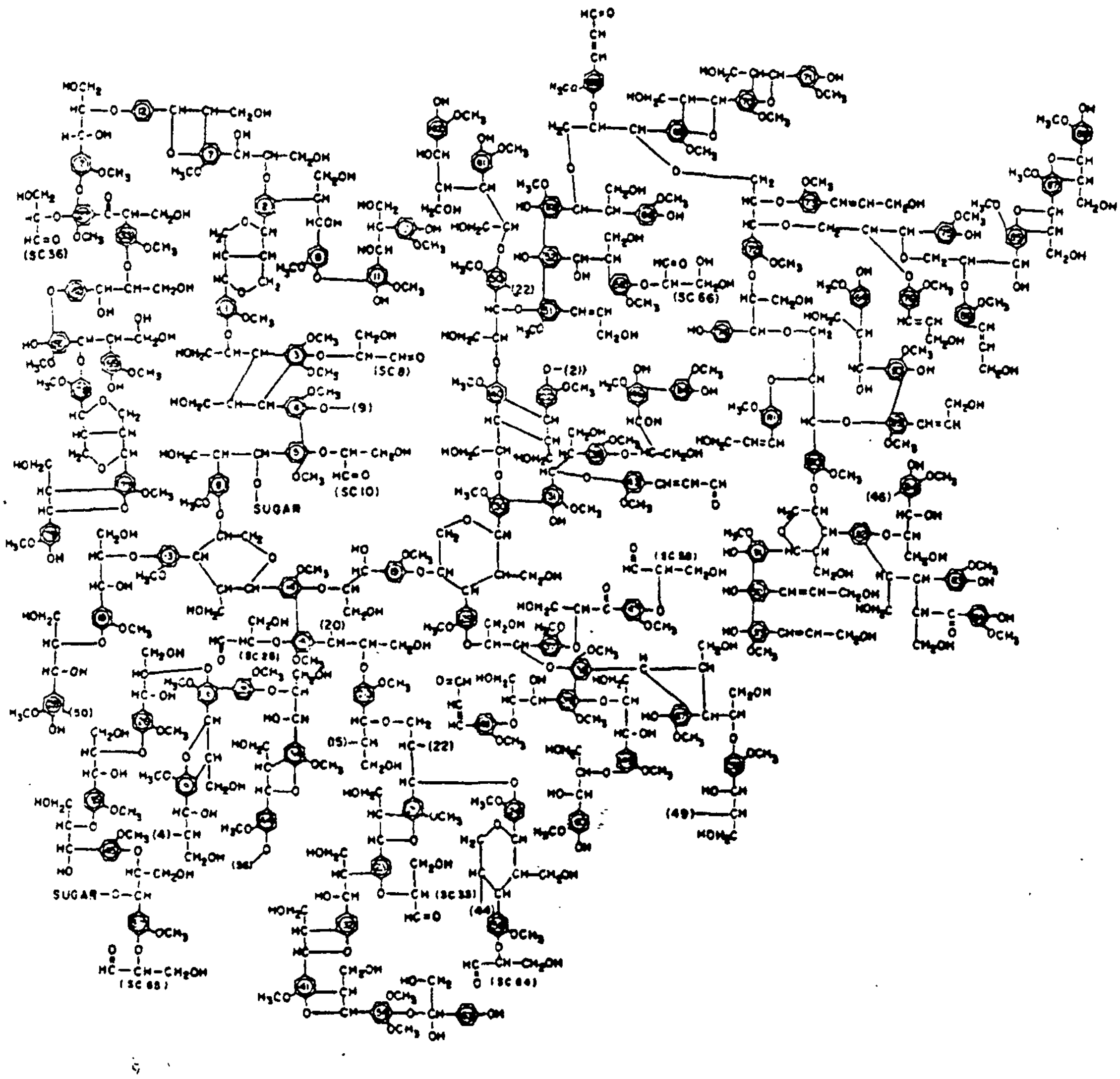


Fig. 3. Softwood lignin model designed by computerized evaluation (by courtesy of W. G. Glasser).[6]

reducing end to a metasaccharinic acid end which does not peel. The cleavage reaction breaks a carbohydrate chain in two, creating a new reducing end which subsequently may peel.

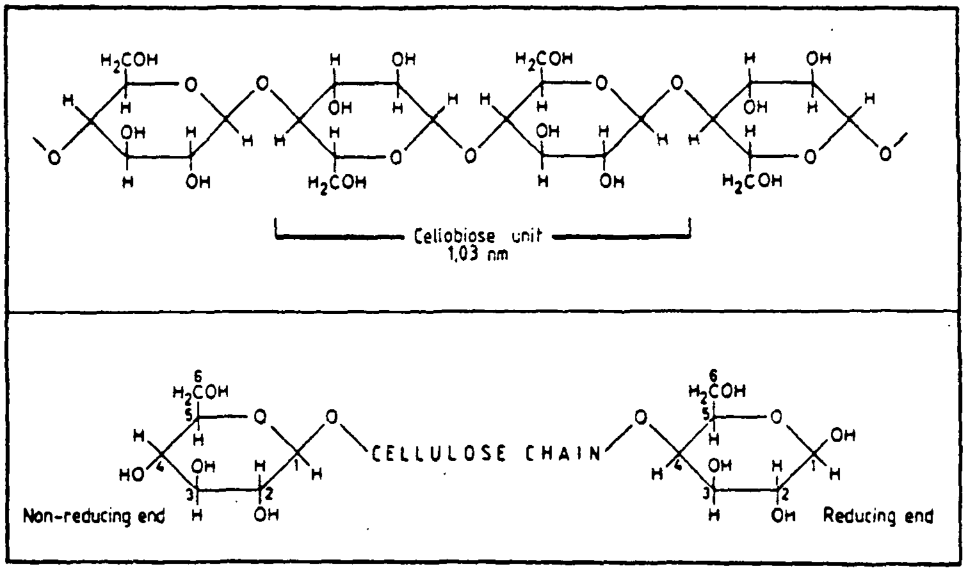


Fig. 4. Formula of cellulose. a) Central part of the molecular chain. b) Reducing and nonreducing end group of the molecule.[6]

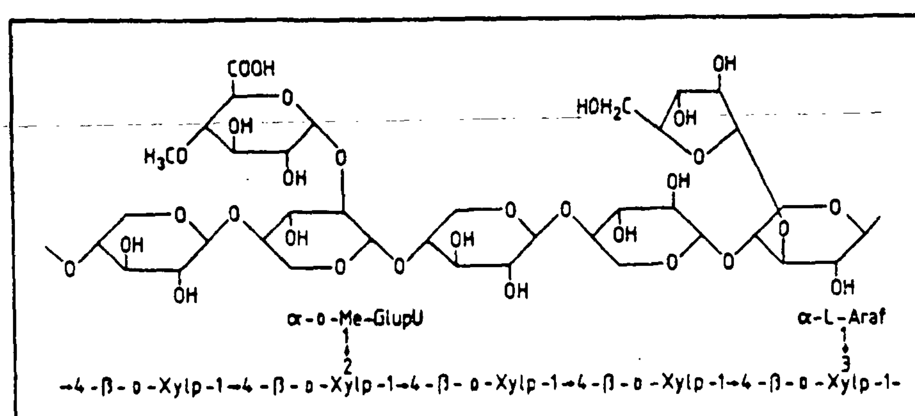


Fig. 5. Partial chemical structure of arabino-4-O-methylglucuronoxylan from softwood.

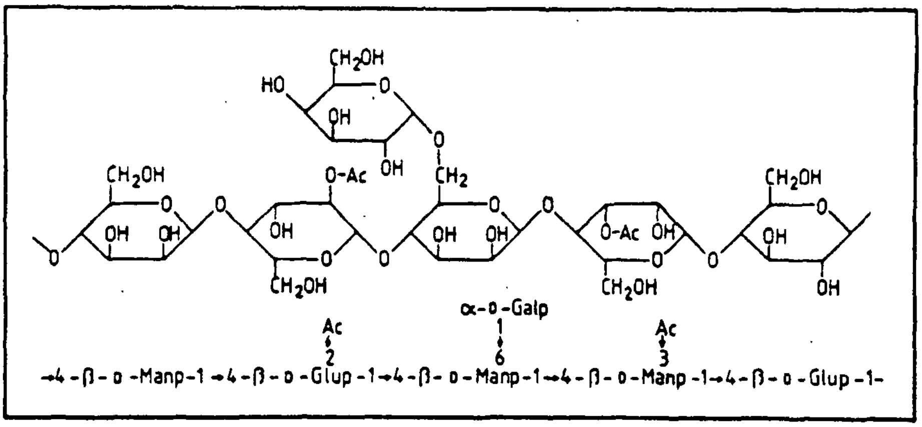


Fig. 6. Partial chemical structure of O-acetyl-galactoglucomannan from softwood.[6]

Extractives are a mixture of many species, most of which are soluble in pulping liquor. Virtually all extractives, as well as significant portions of carbohydrates and lignin, dissolve rapidly at the beginning of a cook, consuming much NaOH in the process. This period of rapid dissolution is known as the initial delignification phase or simply the initial phase. Most of the native lignin (≈ 75%) is removed during the bulk phase, which makes up the largest part of a typical kraft cook. The residual phase is the portion of the cook where residual lignin, probably formed during the cook, is slowly degraded. Figure 7 shows how the carbohydrate yield decreases as delignification proceeds. The breaks in the figure represent the transition points between the initial, bulk, and residual phases.

Steady State Models

One of the first pulping models was a graphical technique developed by Vroom.[8] He related the relative reaction rate of kraft delignification to the Arrhenius equation:

$$k = e^{(a - b/T)} \tag{1}$$

where k is the rate constant for the pulping reaction, relative to the rate constant at 100°C, T is temperature in °K, and a and b are constants. Vroom used data from a pulping study by Larocque and Maass[9] to determine b, then determined a from the condition that k = 1 at 100°C. The integral of the resulting rate expression is known as the H-factor:

$$H = \int e^{(43.20 - 16113/T)} dt \qquad (2)$$

where t is time in hours. The H-factor combines the effects of time and temperature into a single variable that can be directly related to delignification extent (one hour at 100°C ⟶ H = 1). Vroom found that cooks with widely different time-temperature histories fell on the same curve when plots of lignin vs. H-factor or yield vs. H-factor were made. Plots of lignin, yield, or kappa number (a relative indicator of lignin in pulp)[10] vs. H-factor can be used for control purposes provided wood species, chemical charge, and liquor to wood ratio (l:w) are held constant.

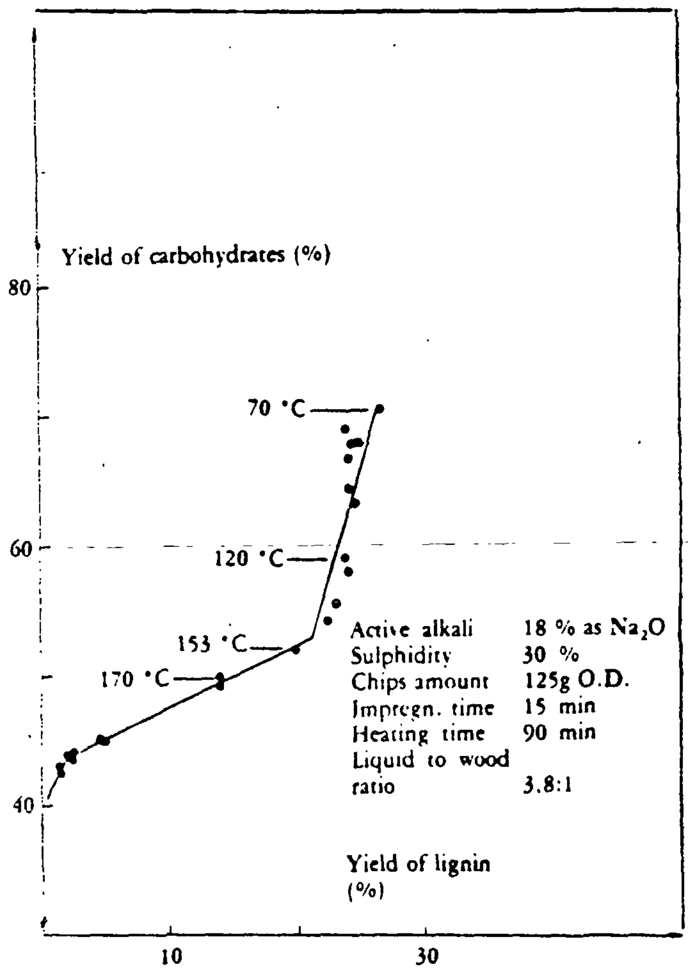


Figure 7. The dissolution of carbohydrates and lignin during pine sulfate cooking.[7]

Since the introduction of the H-factor several steady state models of the kraft process have been derived. These are prediction equations of pulp properties as a function of pulping conditions.[11-17] Some of the pulp properties modeled were total yield, screened yield, kappa number, and pulp viscosity (μ_p), which is a relative indicator of cellulose degree of polymerization and can be related to pulp strength.[18] Some of the independent variables considered were H-factor, effective alkali, sulfidity, AQ charge, and G factor. The G factor,[16]

$$G = \int e^{(57.71 - 21527/T)}dt \tag{3}$$

can be related to the extent of cellulose cleavage. The G factor may be used for pulp viscosity control much the same way the H-factor is used for kappa number control.

Greater predictive power may be obtained by using differential kinetic expressions[19-22] [$\partial u/\partial t = f$ (conditions) where u is the concentration of a generic species] rather than integral kinetic expressions [$u(t) = u(0)\, f$ (conditions)] since the differential expressions make it easier to account for conditions which vary over time. Differential kinetic expressions make possible dynamic computer models of kraft pulping.

Dynamic Models

A dynamic kraft pulping model consists of a set of differential equations representing reaction kinetics, mass transfer, and sometimes heat transfer. Each equation defines the rate of change of a component of interest as a function of pulp and liquor composition. These equations are numerically integrated over time to give a history of pulp properties over time. Dynamic models have the potential to be quite flexible.

One of the earliest dynamic kraft pulping models was developed by Johnsson.[23] He modeled the reaction of lignin, acetyl groups, and NaOH during the bulk phase, omitting carbohydrate and NaSH reactions, the initial phase, the residual phase, and heat of reaction. Acetyl groups were assumed to react before the bulk phase, consuming NaOH. The amount of NaOH consumed by the acetyl groups was proportional to the initial concentration of acetyl groups.

Ignoring heat of reaction is serious. Kraft pulping is exothermic,[24] which raises the temperature in the chip interior relative to the chip surface. A temperature rise of as little as 1°C will cause a significant increase in reaction rates.

Bulk phase delignification was assumed to be described by the relationship

$$\partial L/\partial t = -e^{(a - b/T)}[NaOH]\, L \qquad (4)$$

The effect of [NaSH] on delignification rate was not accounted for. Sulfidity was fixed at ≈ 40%. Carbohydrate reactions were implicitly modeled by assuming

$$\text{yield} = f\,(\text{kappa number only}) \qquad (5)$$

This assumes the relationship between lignin and carbohydrates is constant, i.e., all runs of this model would lie on the same lignin vs. carbohydrate curve. This ignores the effects of process conditions on yield selectivity.[13,25] The change in yield selectivity between the initial, bulk, and residual phases was handled by starting the cook in the bulk phase and ending it before the residual phase began. The rate of change of [NaOH] was proportional to $\partial L/\partial t$. Modeling $\partial[NaOH]/\partial t = f\,(\partial L/\partial t \text{ only})$ does not involve any further assumptions since $\partial \text{Carbohydrates}/\partial t$ is implicitly included through (5).

Johnsson's model accounted for the diffusion of NaOH and dissolved solids in one space dimension. One-dimensional diffusion underpredicts overall reaction rates by ignoring the increase in chemical concentration due to diffusion in the other two dimensions. Thermal diffusion, NaSH diffusion, and dissolved lignin diffusion were omitted, which does not introduce any new assumptions. The diffusivities of NaOH and dissolved solids were equal and depended only on temperature, ignoring the effects of [NaOH] and yield.

Pankonin[26] developed a model which accounted for the reaction of lignin, cellulose, glucomannan, xylan, NaOH, and NaSH during the bulk phase, omitting extractives and acetyl group reactions, the initial phase, the residual phase, and heat of reaction. Reaction rates for each species were fitted separately to

$$\partial u_i/\partial t = -(k_{iOH}\ [NaOH] + k_{iS}\ [Na_2S])\ \mu_i,\ k_{ij} = e^{(a_{ij} - b_{ij}/T)} \qquad (6)$$

This approach, although better than Johnsson's, is inadequate since the kinetic term for $[Na_2S]$ really represents the sum of two effects: the effect of [NaSH] and the effect of increased [NaOH] due to hydrolysis of Na_2S to NaSH and NaOH. This hydrolysis is virtually complete both at room temperature and at 170°C.[27-29] The rates of change of [NaOH] and $[Na_2S]$ were calculated as linear functions of wood species reaction rates. The model accounted for diffusion of NaOH, sodium sulfide (Na_2S), dissolved lignin, and dissolved carbohydrates in three dimensions, omitting thermal diffusion. Diffusion rates of all species were assumed to be equal and to depend on temperature, chip axis, and yield, omitting the effect of [NaOH].

Christensen[30] developed a model which accounted for the reaction rates of lignin, cellulose, glucomannan, xylan, extractives, NaOH, and NaSH, as well as heat of reaction, omitting acetyl group reactions. Lignin was split into high reactivity and low reactivity fractions, which accounted for the initial phase and bulk phase, respectively. The residual phase was not accounted for. The same basic equation was used for all species.

For lignin and xylan

$$\partial u_i/\partial t = -[k_{iOH}\,[NaOH] + k_{iSH}\,[NaOH]^{0.5}\,[NaSH]^{0.5}]\,u_i,$$
$$k_{ij} = a_{ij}\,e^{(b_{ij}/RT)} \qquad (7)$$

for cellulose and glucomannan

$$\partial u_i/\partial t = -[k_{iOH}\,[NaOH] + k_{iSH}\,[NaOH]^{0.5}\,[NaSH]^{0.5}]\,(u_i - u_{i\infty}),$$
$$k_{ij} = a_{ij}\,e^{(b_{ij}/RT)} \qquad (8)$$

The use of an unreactive fraction (u_∞) term is a convenient empirical simplification, but is not strictly correct. Some of the cooks that were run as part of the reaction kinetics study (discussed below) produced yields below 0.1%. The rates of change of [NaOH] and [Na_2S] were calculated as linear functions of wood component reaction rates. Heat of reaction was assumed proportional to $\partial Yield/\partial t$. Calculating $\partial T/\partial t$ as f ($\partial [NaOH]/\partial t$) has been suggested to be more accurate.[24]

The model accounted for heat and mass transfer across the chip surface. Diffusion within the chip was ignored. This assumes that chemical concentrations are equal throughout the chip, which would result in completely uniform pulp. This limits the applicability of Christensen's model to chips thin enough to produce no shives when pulped and introduces a systematic error for all chips. Mass transfer of NaOH, NaSH, and dissolved solids were accounted for, omitting dissolved lignin mass transfer. Mass transfer rates of all species were assumed to be identical and to depend only on temperature.

Tyler[31] developed a model which accounts for lignin and NaOH reactions in the bulk phase, omitting carbohydrate, extractives, and acetyl group reactions, the initial phase, the residual phase, and heat of reaction. The bulk phase reaction equation was

$$\partial L/\partial t = -a\ e^{(43.2\ -\ 16113/T)}\ [NaOH]^{0.75}\ [NaSH]^{0.25}\ L \qquad (9)$$

This equation has no utility for soda cooks ($\partial L/\partial t \longrightarrow 0$ as $[NaSH] \longrightarrow 0$). Yield was a linear function of lignin remaining. This ignores the effects of process conditions on yield selectivity. The rate of change of [NaOH] was proportional to $\partial L/\partial t$. Diffusion of NaOH was considered in one dimension, underestimating overall reaction rates. Diffusivity was assumed constant, omitting the effects of temperature, yield, and [NaOH].

The most recent model was developed by Gustafson.[32] He modeled lignin, carbohydrate, acetyl group, and NaOH reactions during the initial, bulk, and residual phases, omitting extractives reactions, NaSH reactions, and heat of reaction. The initial phase lignin equation was

$$\partial L/\partial t = -a \sqrt{T}\ e^{(b/T)}\ L \qquad (10)$$

The bulk phase lignin equation was

$$\partial L/\partial t = -[k_{OH}\ [NaOH] + k_S\ [NaOH]^{0.5}\ [S]^{0.4}]\ L,\ k_i = e^{(c_i\ -\ d_i/T)},$$

$$[S] = [Na_2S] + [NaSH] \qquad (11)$$

From the discussion above,[27-29] [S] should be replaced with [NaSH]. The transition from the initial phase equation to the bulk phase equation was made at 22% lignin on oven dry wood (ODW). The residual phase lignin equation was

$$\partial L/\partial t = -e^{(e - f/T)} [NaOH]^{0.7} L \qquad (12)$$

The transition point from bulk phase equation to residual phase equation was an input to the model. The initial phase carbohydrate equation was

$$\partial C/\partial t = g\ [NaOH]^{0.11}\ \partial L/\partial t \qquad (14)$$

where C is the concentration of carbohydrates. The rate of change of carbohydrate in the bulk and residual phases were proportional to $\partial L/\partial t$, ignoring the effect of process conditions on yield selectivity. The rate of change of [NaOH] was calculated from the rates of change of acetyl, lignin, and carbohydrates. The model accounts for NaOH diffusion in one dimension, underpredicting overall reaction rates, and omits thermal diffusion, NaSH diffusion, dissolved lignin diffusion, and dissolved solids diffusion. The effects of temperature, yield, and [NaOH] on diffusivity were accounted for.

Table 1 summarizes the dynamic pulping models discussed above.

It seems clear that there is room for improvement. A greatly improved model could be developed simply by combining the best features of all the models discussed above. Unfortunately, such a model would be lacking in several areas.

None of the model authors appear to have critically examined the pulping kinetics equations used in their models. For the most part, they seem to have used equations from the literature, or have used data sets from the literature and did some curve fitting to obtain the equations. Most pulping studies in the literature have concentrated on pulping conditions close to typical industrial practice. These studies define the behavior of the process, but typically do not provide enough information to discriminate between sets of reaction equations. For a set of reaction equations to be applicable over a wide range

Table 1. Comparison of dynamic kraft pulping models.

Author	Johnsson	Pankonin	Christensen	Tyler	Gustafson
Reference No.	23	26	30	31	32
Reaction of					
Native lignin	x	x	x	x	x
Residual lignin					x
Dissolved lignin					
Carbohydrates					x
Cellulose		x	x		
Glucomannan		x	x		
Xylan		x	x		
Extractives				x	
Acetyl	x				x
NaOH	x	x	x	x	x
NaSH		x	x		
AQ					
Viscosity					
Delignification Phases Modeled					
Initial			x		x
Bulk	x	x	x	x	x
Residual					x
Heat of Reaction			x		
Reaction Rate = Function of					
Temperature	x	x	x	x	x
[NaOH]	x	x	x	x	x
[NaSH]		x	x	x	x
[AQ]					
Digester Types Modeled					
Batch	x		x	x	x
Continuous	x	x	x		x
Diffusion of					
Temperature			x		
NaOH	x	x	x	x	x
NaSH		x	x		
AQ					
Dissolved lignin		x			
Dissolved solids	x	x	x		
Number of Space Dimensions for Diffusions					
	One	Three	Zero[a]	One	One
Diffusion Rate = Function of					
Temperature	x	x	x		x
Species					
Dissolved Solids					
Chip axis		x			
Yield		x			x
pH					x

[a]Chip surface mass transfer only.

of conditions the equations should be based on plausible reaction mechanisms, and the data used to fit the equation parameters should include experiments specifically designed to place the equations in jeopardy. There can be no confidence in extrapolating reaction equations fitted to data which can be fitted equally wel several different sets of reaction equations.

None of the models account for lignin condensation reactions. Experimental work by Kleinert[33] showed that the amount of lignin present at the bulk phase to residual phase transition decreases significantly as temperature increases, and implied that most if not all residual lignin is formed during the cook. He also found that the transition takes place at lower values for kraft pulping than for soda pulping. Gustafson does allow the lignin content at the transition point to be a model input, but this does not facilitate a priori estimation of new pulping conditions. A single set of equations applicable during all three phases seems preferable.

None of the models account for the effect of species or dissolved solids on diffusion rate. Direct measurements of lignin fragment diffusivity made by Benko[34] show that the diffusion of NaOH is approximately 12 times faster than lignin fragment diffusion. Dissolved solids concentration increases steadily during the cook, increasing liquor viscosity, which in turn decreases diffusivity. Dissolved solids are expected to be higher in the chip center than at the chip surface, resulting in lower diffusivities at the chip center relative to the chip surface.

Pulp viscosity and AQ are not considered by the models but would be useful additions. Pulp viscosity is a reasonable estimate of relative pulp strength. As such it makes a good constraint on explorations of experimental space.

Anthraquinone is an effective catalyst with limited applicability due to its high cost. Including AQ would facilitate efforts to find optimal conditions for its use.

I believe there is a need for a more fundamental model of the kraft pulping process which is not limited to simulating the status quo, but rather is able to extrapolate beyond current industry practice and explore wide ranges of operating conditions with confidence. In order to improve the predictive power of a dynamic model of kraft pulping, restrictive assumptions need to be removed. Kinetic data sufficient to account for all reactants independently is needed in order to optimize things such as yield selectivity and viscosity selectivity. Condensation reactions must be accounted for in order to better understand the relationship between fiber liberation point and rejects level. The kinetic equations used should be valid over as wide a range of operating conditions as possible and should be valid for both soda and kraft pulping. Equations also valid for soda-AQ and kraft-AQ pulping would be useful (AQ addition will become extremely popular if and when the cost comes down). More generally, the kinetic equations should be easily expandable to accommodate future pulping additives. Diffusion of all species should be accounted for in three space dimensions to get a better assessment of pulp variability within the chip.

THESIS OBJECTIVES

Since the introduction of the kraft pulping process, knowledge pertaining to the effects of diffusion limitation of pulping rates on pulp properties has been restricted mostly to large scale effects - increased rejects, reduced yield, reduced viscosity, etc. Little is known about the small scale effects of diffusion limitation - changes in pulp property profiles within the chip. Dynamic models of kraft pulping have been developed to look at the combined effects of reaction and diffusion but are hampered by a lack of reaction rate data and by assumptions built into the models which limit their usefulness.

It was this lack of a flexible model backed by adequate kinetic data which prompted the present work. The objective of this thesis was to develop a dynamic model which describes the chemical reactions, heat transfer, and mass transfer occurring in a wood chip during kraft-AQ, kraft, soda-AQ, and soda pulping. The most important desired features were: (1) accurate species averages anytime during a cook, (2) accurate three-dimensional species profiles anytime during a cook, (3) mechanistically plausible reaction rate equations for all species, and (4) reliable results when the chip model is extrapolated beyond the data which generated the model.

RESULTS AND DISCUSSION

APPROACH

A dynamic model of the kraft pulping process was developed in three stages. The first two stages were conducted in parallel. Preliminary numerical analysis work was done to identify a satisfactory numerical solution method. Several numerical solution methods were tested. Simultaneously, a reaction kinetics study was conducted to determine adequate reaction models valid during the initial, bulk, and residual delignification phases. A competitive sequential experimental design technique was used to aid in the selection of a model for each species. Iterations of experiment, data analysis, and prediction were used to identify the most appropriate of several proposed candidate models for each reacting species and to determine sufficiently precise parameters for those models.

The final stage consisted of combining literature data on diffusion and stoichiometry with the reaction kinetics equations into the numerical solution "harness." The numerical solution method chosen collapses the three dimensional nonlinear partial differential equations (PDE's) into nonlinear ordinary differential equations (ODE's) using the method of lines with finite difference discretization. The resulting system of ODE's was solved with a stiff ODE solver.

The utility of the model was tested with the aid of a limited number of data sets from the literature.

ASSUMPTIONS

The process of improving on earlier computer models often involves relaxing assumptions that were needed in the earlier models. The assumptions here are necessitated either by continued ignorance of certain aspects of the kraft process or by insufficient computational resources. The assumptions and the rationale for the assumptions are given below.

1. The chip is completely filled with liquor or water at the start of a cook. This assumption is necessary since normal diffusion equations are not valid under two-phase (gas and liquid) conditions. This assumption is valid only for preimpregnated or presteamed chips.
2. The chip can be approximated as a cuboid oriented as shown in Fig. 8.

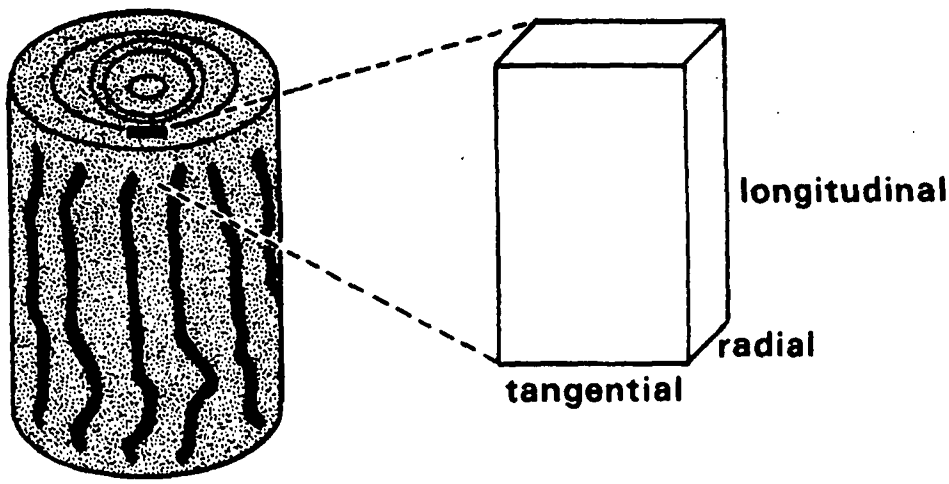


Fig. 8. Chip orientation with respect to original log.

The length direction of the cuboid is parallel to the wood fibers, which is normal for industrial chips. The cuboid width and thickness directions were arbitrarily assigned to the wood tangential and radial axes, respectively. Approximating a chip as a cuboid reduces computation time by ignoring chip surface irregularities. This is an approximation because the ends of industrial chips are not squared off and because of fissures formed during the chipping process. The chip model would probably significantly underpredict the chemical concentrations at the center of a heavily fissured chip.

3. The chip is a homogeneous anisotropic gel under pulping conditions. This assumption drastically reduces computation time by ignoring cellular structure, growth rings, etc. The only distinguishing feature between the chip axes is the anisotropic behavior of diffusivity. As a direct result of this assumption, the profiles generated by the model must be considered to be the average of many identically sized chips.
4. The chip has three planes of symmetry, one through the center of each chip axis. This assumption reduces computation time by a factor of eight.
5. Mass transfer at the chip surface is much greater than diffusion within the chip. This allows Bi (the ratio of mass transfer between bulk liquor and chip surface to diffusion between chip surface and chip interior) to be set large enough that its exact value has no influence on the results. Gustafson[32] showed numerically that varying Bi from 10 to 10,000 had no effect on overall delignification rate. He estimated Bi = 100 for commercial digesters and larger for laboratory digesters.
6. Wood is composed of lignin, cellulose, glucomannan, xylan, extractives, and acetyl groups.

DIFFERENTIAL EQUATIONS

The governing equation of the chip model is

$$\partial u/\partial t = \partial(D_x\, \partial u/\partial x)/\partial x + \partial(D_y\, \partial u/\partial y)/\partial y + \partial(D_z\, \partial u/\partial z)/\partial z + R(u) \qquad (14)$$

where u is the concentration of diffusing species, D is diffusivity, R is reaction rate, and x, y, and z are mutually orthogonal axes within the chip. The axes are arbitrarily aligned with the longitudinal, radial, and tangential planes in the wood. The subscripts on D acknowledge the anisotropy of diffusion in wood. Expanding (14) gives

$$\partial u/\partial t = D_x\, \partial u^2/\partial x^2 + \partial D_x/\partial x\, \partial u/\partial x + D_y\, \partial u^2/\partial y^2 + \partial D_y/\partial y\, \partial u/\partial y + D_z\, \partial u^2/\partial z^2 + \partial D_z/\partial z\, \partial u/\partial z + R(u) \qquad (15)$$

The boundary conditions at the chip surface are

$$\partial u/\partial x = k(u_{surface} - u_{bulk})/D_x,$$
$$\partial u/\partial y = k(u_{surface} - u_{bulk})/Dy, \text{ and}$$
$$\partial u/\partial z = k(u_{surface} - u_{bulk})/Dz \qquad (16)$$

where k is the mass transfer coefficient of forced convection between bulk liquor and chip surface. Equation (16) sets mass transfer between bulk liquor and chip surface to be equal to diffusion from the chip surface into the chip. The boundary conditions at the chip center planes are

$$\partial u/\partial x = \partial u/\partial y = \partial u/\partial z = 0 \qquad (17)$$

Equation (17) is a direct consequence of the assumption of symmetry across the chi center planes. The bulk liquor mass balance equation for a batch digester is

$$\partial u_{bulk}/\partial t = k(u_{surface} - u_{bulk})\, A_{chip}/V_{bulk} \qquad (18)$$

where A_{chip} is chip surface area and V_{bulk} is bulk liquor volume. Other digester types are simulated by modifying (18) appropriately.

PARAMETERS AND THEIR SOURCES

Bulk Liquor Diffusion Coefficients

The diffusion rates of all species are calculated from the Stokes-Einstein equation[34] for the diffusion of a sphere in a continuous solvent:

$$D = k_B\ T/(6\ \pi\ \mu_s\ R) \tag{19}$$

where k_B is Boltzmann's constant, μ_s is solvent viscosity, and R is the radius of the diffusing species. For any single species $k_B/(6\ \pi\ R)$ is constant, which implies the proportionality

$$D \propto T\ /\ \mu_s \tag{20}$$

This leads to a prediction equation for bulk liquor diffusivity:

$$D_l = D_{298}\ (\mu_{w(298)}/298)\ (T_l/\mu_l) \tag{21}$$

where D_l is the diffusivity of a species in liquor, D_{298} is the diffusivity in water at 298°K, $\mu_{w(298)}$ is the viscosity of water at 298°K, T_l is the liquor temperature, and μ_l is the liquor viscosity.

Values for D_{298} were obtained from a variety of sources. The diffusivities of NaOH[35] (2.12×10^{-9} m^2/s) and glucose[36] (0.673×10^{-9} m^2/s) (used to approximate the diffusivity of dissolved solids) were taken directly from diffusivity tables. The diffusivity of NaSH (1.51×10^{-9} m^2/s) was calculated from specific ionic conductances[36] using the Nernst-Haskell equation[37] for the diffusion of a single salt at infinite dilution:

$$D = (R\ T/F^2)\ (1/n_+ + 1/n_-)/(1/l_+ + 1/l_-) \tag{22}$$

where R is the gas constant, F is the Faraday, n_+ is the cation valence, n_- is the anion valence, l_+ is the cationic limiting conductance, and l_- is the anionic limiting conductance. The diffusivity of AQ (0.650×10^{-9} m^2/s) was estimated from the Wilke-Chang equation,[37] an empirical modification of the Stokes-Einstein equation:

$$D_{ab} = 7.4 \times 10^{-12} [(\phi MW_b)^{0.5} T]/(\mu_b V_a^{0.6}) \; [=] \; m^2/s \tag{23}$$

where D_{ab} is the diffusivity of a in b, ϕ is a solvent association parameter, MW is molecular weight, and V is molar volume at the boiling point. A value of 2.26 is recommended for ϕ_{water}.[38] Molar volume was calculated using a table of additive volumes.[37] Diffusivity of dissolved lignin (DL) (0.184×10^{-9} m^2/s) was back calculated from the work of Benko,[34] who determined the molecular weight of DL by carrying out diffusion experiments in glass diffusion cells calibrated to potassium chloride (KCl). The equation used was

$$MW_{DL} = (D_{KCl}/D_{DL})^2 \; MW_{KCl} \tag{24}$$

where MW_{DL} was reported by Benko. Solving for D_{DL} gives

$$D_{DL} = (MW_{KCl}/MW_{DL})^{0.5} \; D_{KCl} \tag{25}$$

Liquor viscosity was regressed against water viscosity. Water viscosity data[39] from 273 to 645°K were fitted to[40]

$$100/\mu_w = 2.260 \{(T - 285.5) + [(T - 285.5)^2 + 9854]^{0.5}\} - 142.2,$$
$$\mu_w \; [=] \; cp, \; T \; [=] \; °K \tag{26}$$

The value of $\mu_{w(298)}$ for (21) was taken directly from the water viscosity data.[39] Black liquor viscosity data[41-43] between 5 and 50% solids were fitted to[43]

$$\ln(\mu_l) = (10.63\ S^3 + 1.302\ S^2 + 1)\ \ln(\mu_w) + 27.31\ S^3 + 4.108\ S \qquad (27)$$

where S = % solids/100.

Diffusion Rates in Wood

Diffusion in wood was modeled as the product of bulk liquor diffusion and ECCSA. Two sets of data were used: ECCSA *vs.* pH at 100% yield for spruce[5] (Fig. 1), and ECCSA *vs.* yield at pH 13.2 for pine[5] (Fig. 2). Gustafson[32] fit the spruce radial and tangential data to

$$ECCSA_{r\&t} = 0.05 + 0.1299\ [NaOH]^{0.55} \qquad (28)$$

where [NaOH] is assumed to be $10^{(pH - 14)}$*M*. I used the same type of equation to fit the spruce longitudinal data:

$$ECCSA_l = 0.608 - 0.139\ [NaOH]^{0.175} \qquad (29)$$

Pankonin[26] fit the pine data to

$$ECCSA_l = 0.1446\ Y + 0.4565 \qquad (30)$$

$$ECCSA_t = -0.7850\ Y + 0.9550, \text{ and} \qquad (31)$$

$$ECCSA_r = -0.6056\ Y + 0.7620, \qquad (32)$$

where Y = % yield/100. The original pine curves extend from 100% yield down to 65% yield. I empirically extrapolated the lines down to zero yield using quadratic splines which agreed with ECCSA(65% yield), $\partial ECCSA/\partial yield\,|_{65\%\ yield}$, and ECCSA(0% yield) (= 1 by definition). This resulted in

$$ECCSA_l = 1.286\ Y^2 - 1.528\ Y + 1 \qquad (33)$$

$$ECCSA_t = 0.1065\ Y^2 - 0.9235\ Y + 1, \text{ and} \qquad (34)$$

$$ECCSA_r = 0.5633\ Y^2 - 1.338\ Y + 1 \qquad (35)$$

Equations (30) to (32) are for yields between 100 and 65%; Eq. (33) to (35) are for yields below 65%. The regions in the yield - [NaOH] plane where ECCSA is known are summarized in Fig. 9.

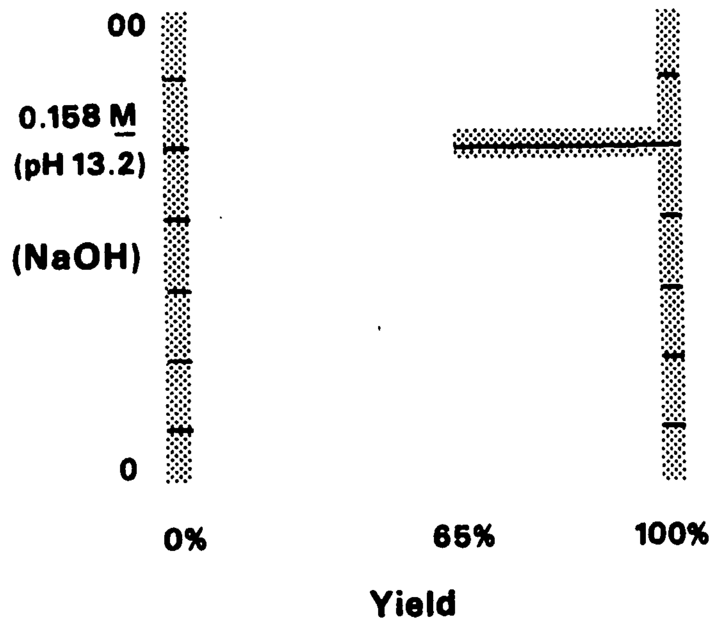


Fig. 9. Regions where ECCSA is known.

Combined formulas for ECCSA (longitudinal, radial, and tangential) as a function of [NaOH] and yield were obtained by interpolation. The constraints used for each interpolating equation were that ECCSA(0% yield) = 1 for all [NaOH], and that sections of the interpolant surface parallel to the yield or [NaOH] axis should have the same general "shape" as the spruce and pine equations respectively. This gives

$$\mathrm{ECCSA}([\mathrm{NaOH}], Y) = 1 - c\,[1 - \mathrm{ECCSA}([\mathrm{NaOH}])]\,[1 - \mathrm{ECCSA}(Y)] \qquad (37)$$

where c = 2.030 for $\mathrm{ECCSA}_{\mathrm{longitudinal}}$, 1.108 for $\mathrm{ECCSA}_{\mathrm{radial\ \&\ tangential}}$. The factor c forces ECCSA([NaOH], Y) equal to ECCSA(Y) (derived from the pine data) at pH 13.2 and 100% yield. The combined equation was forced through the pine data rather than the spruce data, since the reaction kinetics study used loblolly pine.

Thermal Diffusion

Unlike chemical diffusion, thermal diffusion in wood is isotropic. This is because the thermal diffusivity of wood is nearly identical to the thermal diffusivity of water.[44] This allows the use of tabulated data for water thermal diffusivity (D_T). Data from room temperature up to the critical point were fitted to

$$D_T = -1.7080 \times 10^{-12}\, T^2 + 1.4498 \times 10^{-9}\, T - 1.3489 \times 10^{-7} \qquad (38)$$

where D_T [=] m^2/s.

Chemical Consumption

The rates of consumption (except for the acetyl groups term) of NaOH and NaSH are taken from Christensen:[30]

$$R_{NaOH} = 0.15\, \partial L/\partial t + 0.4\, \partial C/\partial t + 0.1\, \partial E/\partial t + 0.6775\, \partial Ac/\partial t \text{ and} \qquad (39)$$

$$R_{NaSH} = 0.056\, \partial L/\partial t \qquad (40)$$

where L is lignin, C is carbohydrate, E is extractives, Ac is acetyl groups, and R_{NaOH} [=] R_{NaSH} [=] $\partial L/\partial t$ [=] $\partial C/\partial t$ [=] $\partial E/\partial t$ [=] $\partial Ac/\partial t$ [=] $g/dm^3/hr$. The acetyl term was calculated assuming one mole of NaOH was neutralized for each mole of acetyl.[32] The consumption of AQ was estimated from bulk liquor [AQ] vs. time data[45,46] as

$$R_{AQ} = -[AQ]\sqrt{T}\, e^{(20.48 - 10,690/T)} \qquad (41)$$

where R_{AQ} [=] mM/hr.

Heat of Reaction

The heat of reaction (= 13.345 kcal/g-mole) was calculated from the heats of formation[36] of aqueous H^+, aqueous OH^-, and liquid H_2O. This is equivalent to assuming the heat is liberated as the result of a strong acid-base neutralization. The product {heat capacity x density} of water (= 1 $kcal/dm^3/°K$) was used to approximate {heat capacity x density} of the wood-liquor mixture, giving an expression for heat generation:

$$R_T = -13.345\ [NaOH]/\partial t \tag{42}$$

where R_T [=] °K/hr and $\partial[NaOH]/\partial t$ [=]M/hr.

REACTION KINETICS STUDY

A reaction kinetics study was undertaken to determine the most appropriate kinetic equations for lignin, cellulose, glucomannan, xylan, and pulp viscosity, as well as to determine precise parameters for these equations.

Sequential Experimental Design

Experimental conditions for the reaction kinetics study were determined using the joint design criterion (**C**), developed by Hill, Hunter, and Wichern,[47] to drive a sequential experimental design. The sequential experimental design began with a small initial set of experiments, then iterated through a cycle of experiment, data analysis, and prediction. After each set of experiments, all the data were analyzed and used to predict conditions which maximized **C**. The optimization of **C** was conducted once each for lignin, cellulose, glucomannan, and xylan. The four **C** optimal points in experimental space were used for the next set of experiments. The cycle of experiment, analysis, and prediction was continued until an arbitrary stopping criterion was satisfied.

Joint design criterion **C** postulates the existence of a finite number of possible mechanistic models, one of which is assumed to be the correct representation of the system being studied. The parameters for each model are determined by fitting the models to the available data. Experimental conditions which maximize **C** maximize model discrimination[48] (**D**) and parameter estimation[49] (**E**) criteria simultaneously.

Model discrimination criterion **D** increases our knowledge as to which model is correct. **D** is essentially the sum of squares of the differences in model predictions between all unique pairs of models [weighted by functions of relative probability (Π) and variance (s^2)]. **D** is optimal for experimental conditions where the responses of the candidate models disagree the most. Running **D** optimal experiments maximizes the likelihood that only one model will be able to fit both the data generated by the run and the prior data.

Parameter estimation criterion **E** increases the precision of the model parameters. **E** is the determinant of the transpose product of an array composed of the sensitivities of model predictions to small changes in model parameters. **E** is optimal for conditions where models are the most sensitive, i.e., a small change in model parameters causes a big change in model prediction. Running **E** optimal experiments minimizes the confidence region about the model parameters.

Simultaneous optimization of model discrimination and parameter estimation is accomplished by using the relative probability of the most likely model (Π_b) to weight model discrimination against parameter estimation. Relative probabilities are calculated from the variances of the models from the experimental data.

The joint design criterion is **C** = d **D** + e **E**, where d and e are weights (d + e = 1). Both **D** and **E** are weighted by relative probability, so that unlikely models have little influence on **D**, **E**, or **C**. The weights d and e are calculated from Π_b. If all models are equally likely to be correct (Π_b =1/#models) then d = 1, e = 0 (pure model discrimination). If the best model fits the data perfectly and the others not at all (Π_b = 1) then d = 0, e = 1 (pure parameter estimation). As the experimental method proceeds, Π_b increases, and **C** shifts from emphasizing **D** to emphasizing **E**.

The models were fitted using NONLINWOOD, a nonlinear regression program.[50] I wrote a program named MDPE (Model Discrimination/Parameter Estimation) to optimize **C**. A sample data deck, sample output, and the source code listing for MDPE are presented in Appendix VI.

The reaction kinetics study began with 12 runs - a five variable fractional factorial in eight runs plus four center points. The variables were time, temper ture, [NaOH], [NaSH], and [AQ]. The responses were lignin, cellulose, glucomanna and xylan. Thereafter, I ran NONLINWOOD once per model to fit the candidate mode to the data and ran MDPE once per response to optimize **C** for each response. The four conditions obtained were used for the next set of runs. Ten iterations of the sequential design were performed, resulting in a total of 52 usable runs. The data and simple statistics of the data are presented in Appendix I. The models used in the reaction kinetics study are summarized in Appendix II. The experimental space spanned by the study is shown in Fig. 10 to 13.

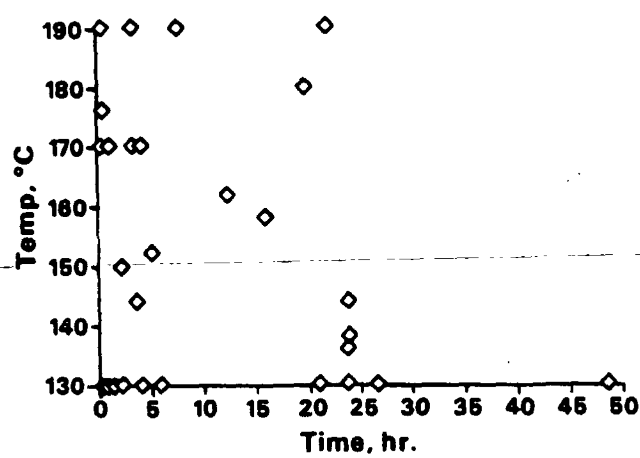


Fig. 10. Experimental conditions: temperature *vs.* time.

Many candidate models were considered for each response. For example, 62 separate lignin models were considered at one time or another, although no more than 10 were entertained at any one time. As the reaction kinetic study progressed, inferior models were discarded and new models were postulated.

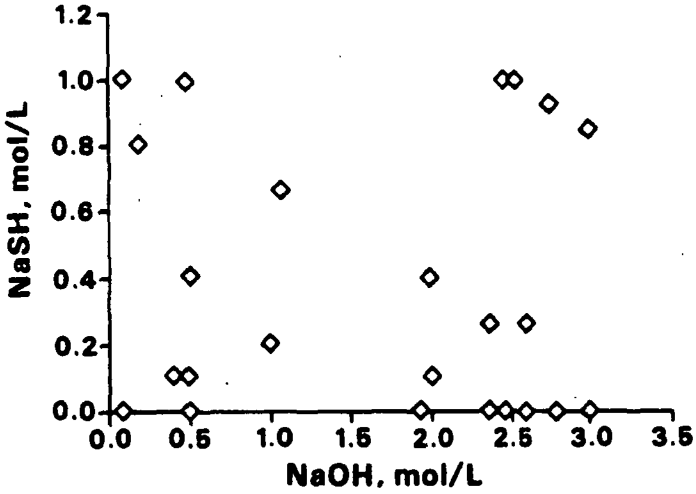


Fig. 11. Experimental conditions: NaSH vs. NaOH.

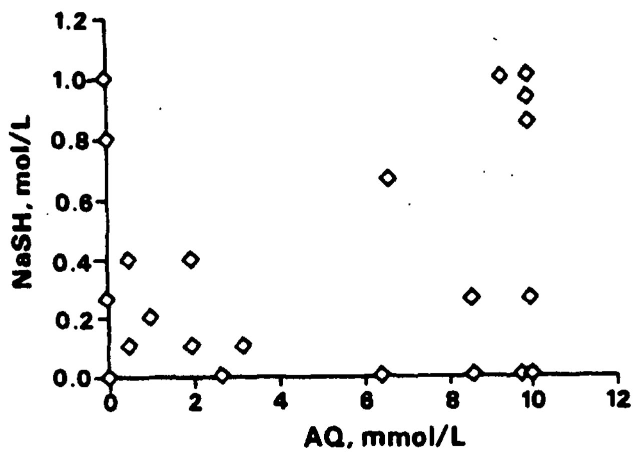


Fig. 12. Experimental conditions: NaSH vs. AQ.

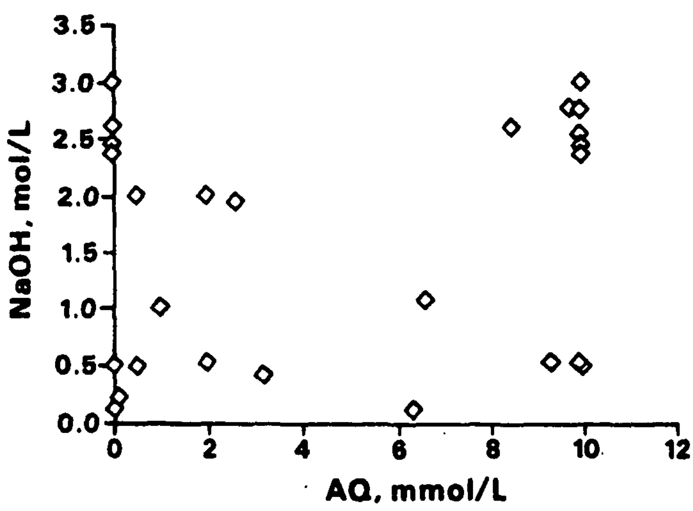


Fig. 13. Experimental conditions: NaOH vs. AQ.

Each model represents a mechanistically plausible reaction scheme. For instance, carbohydrate models accounting for just the peeling reaction were considered as well as models accounting for simultaneous peeling, stopping, and cleavage. The reaction variables looked at include the order of species dissolution with respect to [species], [NaOH], [NaSH], and [AQ]. Almost all of the models used parallel reaction pathways due to the experimental observation that alkaline pulping proceeds in the absence of NaSH and/or AQ. Parallel reaction pathways make it simple to handle new pulping catalysts; extra pathways are added to account for the additional possible reactions.

Bulk and Residual Phase Kinetics

The final product of the reaction kinetics study was a set of equations, valid during the bulk and residual phases, which describes the rate of change of lignin, cellulose, glucomannan, xylan, and pulp viscosity. The equations below represent the best candidate model for each species and have been tested over a wide experimental space using experimental conditions specifically designed to weed out inferior candidates. Error estimates of the parameters of the best model are given in Appendix II along with descriptions of the other models investigated. The best lignin model consisted of the reaction network shown in Fig. 14.

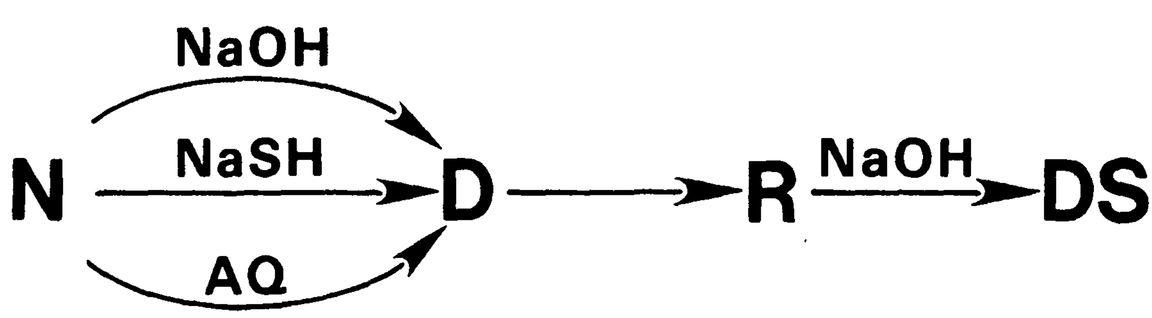


Fig. 14. Reaction network for best lignin model.

The network can be summarized as

$$\text{lignin} = N + R,$$

$$\partial \text{lignin}/\partial t = \partial N/\partial t + \partial R/\partial t,$$

$$\partial N/\partial t = -k_{bulk}\,N,$$

$$\partial R/\partial t = k_{condense}\,D - k_{residual}\,R, \text{ and}$$

$$\partial D/\partial t = k_{bulk}\,N - k_{condense}\,D, \qquad (43)$$

where N is native lignin, R is residual lignin, D is dissolved lignin, DS is dissolved solids, and t [=] hr. The bulk delignification phase rate constant k_{bulk} was calculated as the sum of three parallel reactions, dependent on NaOH, NaSH, and AQ, respectively:

$$k_{bulk} = [NaOH]\sqrt{T}\,e^{(-4.155 - 19{,}610\,T^{\dagger})} + \sqrt{[NaOH]}\sqrt{[NaSH]}\sqrt{T}\,e^{(-3.248 - 10{,}820\,T^{\dagger})} + \sqrt{[NaOH]}\sqrt{[AQ]}\sqrt{T}\,e^{(-3.838 - 15{,}900\,T^{\dagger})}, \qquad (44)$$

where T [=] °K, $T^{\dagger} = 1/T - 1/433$, [NaOH] [=] M, [NaSH] [=] M, and [AQ] [=] mM. Using parallel reactions allows the model to accurately predict the bulk phase rate when [NaSH] and/or [AQ] = 0. The condensation reaction rate depended only on [D]; it was assumed that the unreacted lignin and/or carbohydrates would always have enough sites available for condensation so that site availability would not limit condensation rate. The condensation reaction activation energy was fixed to a value expected for diffusion controlled reactions after attempts to fit the activation energy resulted in values approaching zero:

$$k_{condense} = \sqrt{T}\,e^{(-5.973 - 2{,}500\,T^{\dagger})} \qquad (45)$$

Dissolution of residual lignin was assumed to form unreactive dissolved lignin which was lumped together with dissolved solids (DS). The rate expression for residual phase delignification is

$$k_{residual} = [NaOH] \sqrt{T}\, e^{(-6.002 - 9,991\, T^{\dagger})} \quad (46)$$

The three carbohydrate fractions (cellulose, glucomannan, and xylan) were best modeled by the reaction network shown in Fig. 15.

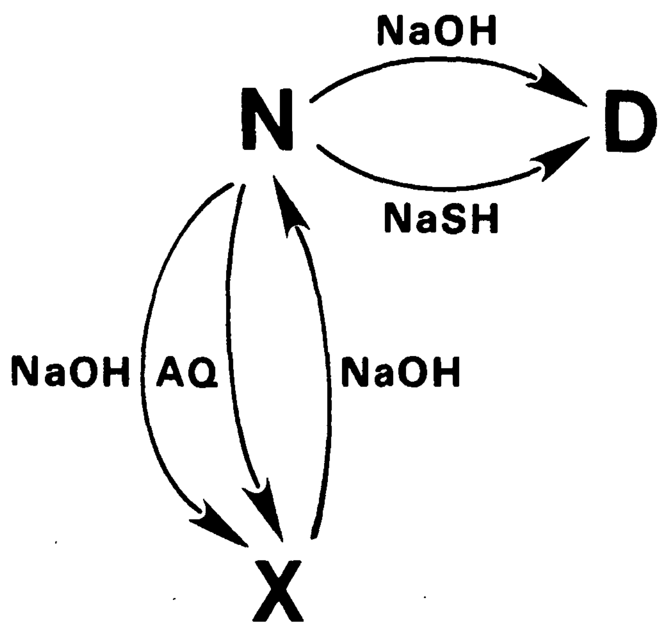


Fig. 15. Reaction network for best carbohydrate models.

The carbohydrate reaction network can be summarized as

$$C = N + X,$$

$$\partial C/\partial t = \partial N/\partial t + \partial X/\partial t,$$

$$\partial N/\partial t = k_{cleave}\, X - (k_{peel} + k_{stop})\, N, \text{ and}$$

$$\partial X/\partial t = k_{stop}\, N - k_{cleave}\, X \quad (47)$$

where C is cellulose, glucomannan, or xylan, N is the native fraction, D is the dissolved fraction, and X is the oxidized fraction. Peeling converts N to D, stopping converts N to X, and cleavage converts X back to N. The only distinction between N and X is that X does not peel. Peeling was modeled as two parallel reactions driven by NaOH and NaSH:

$$k_{peel\ cel} = [NaOH]\sqrt{T}\,e^{(-5.044\ -\ 6,697\ T^{\dagger})} + \sqrt{NaSH]}\sqrt{T}\,e^{(-6.683\ -\ 8,013\ T^{\dagger})} \quad (48)$$

$$k_{peel\ gm} = [NaOH]\sqrt{T}\,e^{(-0.8574\ -\ 6,682\ T^{\dagger})} + \sqrt{NaSH]}\sqrt{T}\,e^{(-4.695\ -\ 12,070\ T^{\dagger})}, \text{ and} \quad (49)$$

$$k_{peel\ xyl} = [NaOH]\sqrt{T}\,e^{(-2.303\ -\ 15,600\ T^{\dagger})} + \sqrt{NaSH]}\sqrt{T}\,e^{(-6.316\ -\ 2,500\ T^{\dagger})} \quad (50)$$

The activation energy for xylan peeling due to NaSH was fixed to a value expected for diffusion controlled reactions after attempts to fit the activation energy resulted in values approaching zero.

The stopping reaction was modeled as two parallel reactions driven by NaOH and AQ.

$$k_{stop\ cel} = [NaOH]\sqrt{T}\,e^{(-2.322\ -\ 9,611\ T^{\dagger})} + [NaOH]\sqrt{[AQ]}\sqrt{T}\,e^{(-4.620\ -\ 16,720\ T^{\dagger})} \quad (51)$$

$$k_{stop\ gm} = [NaOH]\sqrt{T}\,e^{(-1.510\ -\ 3,126\ T^{\dagger})} + [NaOH]\sqrt{[AQ]}\sqrt{T}\,e^{(-2.359\ -\ 10,770\ T^{\dagger})}, \text{ and} \quad (52)$$

$$k_{stop\ xyl} = [NaOH]\sqrt{T}\,e^{(-2.739\ -\ 8,860\ T^{\dagger})} + [NaOH]\sqrt{[AQ]}\sqrt{T}\,e^{(-6.339\ -\ 22,300\ T^{\dagger})} \quad (53)$$

The cleavage reaction was modeled as a single reaction in NaOH:

$$k_{cleave\ cel} = [NaOH]\sqrt{T}\,e^{(-3.319\ -\ 22,840\ T^{\dagger})}, \quad (54)$$

$$k_{cleave\ gm} = [NaOH]\sqrt{T}\,e^{(-4.766\ -\ 12,640\ T^{\dagger})}, \text{ and} \quad (55)$$

$$k_{cleave\ xyl} = [NaOH]\sqrt{T}\,e^{(-3.746\ -\ 12,110\ T^{\dagger})} \quad (56)$$

Pulp viscosity[18] was best modeled as a single pathway reaction driven by NaOH:

$$\partial \mu_p/\partial t = -(\mu_p - 1.268)^2 \, [NaOH] \sqrt{T} \, e(-6.398 - 19{,}110 \, T^{\dagger}), \quad (57)$$

where μ_p [=] cp. Equation (57) is consistent with the experimental observation of Kubes <u>et al</u>.[16] that plots of $1/\mu_p$ <u>vs</u>. time are linear. The constant 1.268 represents pulp viscosity at infinite time (approximated as the cuene viscosity[1] of a 0.5% glucose solution). The fitted value of μ_p (t = 0) was 57.67 cp.

<u>Initial Phase Kinetics</u>

During the initial phase some of the lignin and hemicellulose rapidly reacts. These reactions are fast compared to the heatup period used in the reaction kinetics study (≈ 20 minutes). Because of the relatively slow heatup period, initial phase kinetics were not explicitly studied. Initial phase kinetic data[21,22] of lignin and hemicellulose dissolution were used to augment the bulk and residual phase kinetic equations. The initial phase rates were estimated at a single temperature from data reported in the literature and combined with the reported activation energies to give

$$\begin{aligned} \partial L/\partial t &= \partial N/\partial t + \partial R/\partial t + k_a + k_b \quad \text{for } N > 0.87, \\ \partial L/\partial t &= \partial N/\partial t + \partial R/\partial t + k_b \quad \text{for } 0.87 > N > 0.76, \\ \partial L/\partial t &= \partial N/\partial t + \partial R/\partial t \quad \text{for } 0.76 > N, \end{aligned} \quad (58)$$

$$k_a = d\,(N - 0.87)\, e^{(17.33 - 6000/T)}, \quad (59)$$

$$k_b = d\, N\, e^{(22.12 - 8800/T)}, \quad (60)$$

$$\begin{aligned} \partial H/\partial t &= \partial N/\partial t + \partial X/\partial t + k_c \quad \text{for } N + X > H_t, \\ \partial H/\partial t &= \partial N/\partial t + \partial X/\partial t \quad \text{for } H_t > N + X, \end{aligned} \quad (61)$$

$$k_c = d\,(N + X) \sqrt{T}\, e^{(9.251 - 4738/T)}, \text{ and} \quad (62)$$

$$d = -(1 - \{1/(1 + 100\,[NaOH])\}), \quad (63)$$

where L is lignin and H is hemicellulose. The transition between initial and bulk phases (H_t) was fitted along with the other parameters in the glucomannan and xylan models above. The fitted transition values are 0.752 for glucomannan and 0.868 for xylan. The fitted transition value for cellulose was 1.042, indicating that not all the cellulose is accessible before the wood comes in contact with alkali. The initial phase reaction rate for lignin is insensitive[21] to [NaOH] over a wide range of concentration. The initial phase reaction rate for carbohydrates is independent[22] of [NaOH] for pH > 12. Equation (66) was derived from a dissociation equilibrium equation. The term d was empirically included to insure that the reaction rates rapidly approached zero below pH 12 and were fairly insensitive to [NaOH] above pH 12.

Implications for Kraft-AQ Pulping

The fact that the models above were selected over many rejected models (Appendix II) implies several things about kraft-AQ pulping. Delignification is first order with respect to [lignin]. Cellulose, glucomannan, and xylan dissolution reactions are first order in [cellulose], [glucomannan], and [xylan], respectively. The change in pulp viscosity with time is second order with respect to ($\mu_p - \mu_\infty$) where $\mu_\infty = 1.268$ cp (the 0.5% cuene viscosity of glucose) worked much better than $\mu_p = 0$. Models with inaccessible fraction parameters are inappropriate; the reaction kinetics study showed these terms must equal zero for lignin, cellulose, glucomannan, and xylan.

Lignin networks consisting of bulk delignification, lignin condensation, and residual delignification reactions work much better than bulk delignification only schemes. Direct condensation of native lignin appears to be insignificant compared to condensation of dissolved lignin. Dissolved lignin appears to be able to condense to lignin or carbohydrate; models with a condensation rate dependence of {[lignin] x [dissolved lignin]} were inferior to models with a condensation rate dependence of [dissolved lignin]. Reaction schemes which include explicit conversion between native lignin and a reactive intermediate showed that the conversion rates were much faster than bulk delignification. Reaction schemes without the conversion reactions gave essentially the same results.

Carbohydrate networks consisting of simultaneous peeling, stopping, and cleavage reactions worked much better than peeling only networks. Carbohydrate peeling is accelerated by NaSH for all three carbohydrate fractions, the reaction rate being proportional to $\sqrt{[NaSH]}$. The carbohydrate stopping reaction is accelerated by AQ for all three fractions; the rate is proportional to [NaOH] $\sqrt{[AQ]}$.

NUMERICAL SOLUTION METHODS

The equations to be solved, (15) and (18), are nonlinear parabolic partial differential equations in three space dimensions. Seventeen PDE's are used to model the chip interior: native and oxidized cellulose, glucomannan, and xylan; extractives and acetyl groups; native, residual, and dissolved lignin; NaOH, NaSH, AQ, dissolved solids, temperature, and viscosity. Eight ODE's are used to model the bulk liquor: dissolved lignin, NaOH, NaSH, AQ, dissolved solids, temperature, G-factor, and H-factor. PDE's are considered nonlinear if their coefficients, in this case D_x, D_y, D_z, and R, are not constants. Nonlinear PDE's cannot be solved analytically but they can be solved numerically. An analytic solution is a closed form equation from which the properties of interest may be calculated exactly, everywhere in solution space. A numerical solution approximates the properties at discrete points (grid points) in solution space. Interpolation is used to estimate properties between grid points. The accuracy of a numerical solution is a polynomial function of the distance between the grid points (grid spacing). For problems with known analytical solutions, the numerical solution rapidly approaches the analytical solution as the grid spacing decreases. For nonlinear problems, the numerical solution quickly converges as grid spacing decreases; the asymptotic solution is taken to be the correct solution.

The method of lines[51] is used to solve (15) and (18). A three-dimensional grid is laid out with the grid points at the intersections. In order to attain maximum accuracy, the grid points are bunched up at the chip surface, since that is where concentration gradients are steepest. The partial derivatives at a particular grid point are calculated as a linear combination of the function values and/or the derivative values at the grid point and its neighbors along the axes, as shown in Fig. 16.

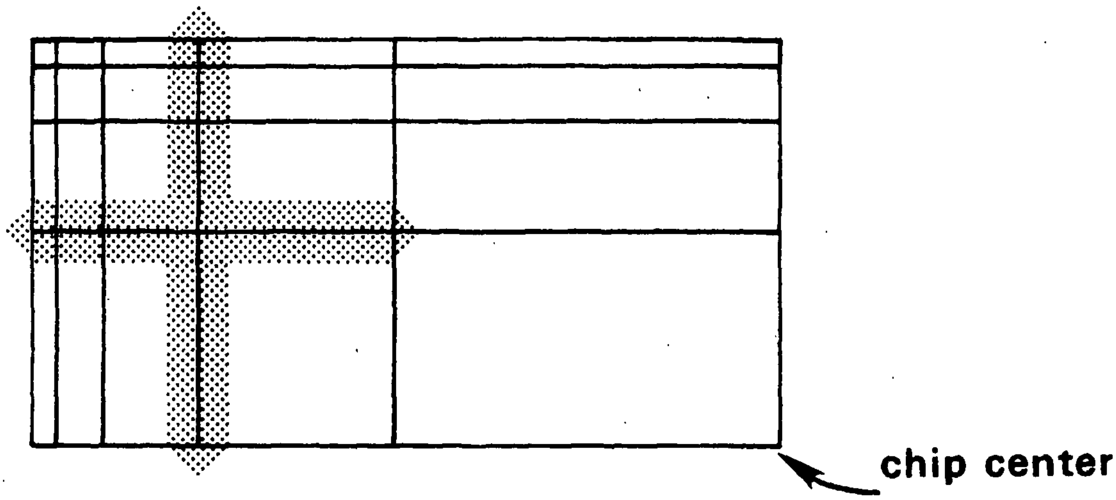


Fig. 16. Numerical solution grid.

The chip surface is the top and left sides of the rectangle; the chip center is the bottom right corner. The shaded cross lies over the points used to determine $\partial^2u/\partial x^2$, $\partial^2u/\partial y^2$, $\partial u/\partial x$, $\partial u/\partial y$, $\partial D_x/\partial x$, and $\partial D_y/\partial y$, evaluated at the intersection of the cross. The points used for $\partial^2u/\partial z^2$, $\partial u/\partial z$, and $\partial D_z/\partial z$ lie normal to the plane of the figure. Figure 16 assumes that $O(h^4)$ ("fourth order") accuracy is desired; the number of points required in general is the sum of the order of accuracy required plus the order of the derivative estimated. Even though only five points in each direction are shown in shadow, six values are available in the x (length) direction and seven values in the y (width) direction due to the derivative boundary conditions available at the chip surface and chip center planes.

The determination of the grid point locations and the finite difference formulas by the model is completely automatic. The only data that need to be supplied are the chip dimensions and the number of grid points in each direction. Optionally, the user may also specify the desired order of accuracy and the grid spacing between the chip surface and the first interior point (h) (the smallest grid spacing in the system and identical in all three directions). The grid spacings are automatically scaled in a geometric progression increasing toward the center. Boundary conditions are automatically incorporated into the formulas when they are the closest remaining points available. When advantageous, symmetry conditions about the chip center planes are exploited to increase the number of points available for the finite difference formulas. An example of exploiting the symmetry conditions is given in Appendix IV.

Nine finite difference formulas are generated for each grid point ($\partial u/\partial x$, $\partial u/\partial y$, $\partial u/\partial z$, $\partial u^2/\partial x^2$, $\partial u^2/\partial y^2$, $\partial u^2/\partial z^2$, $\partial D_x/\partial x$, $\partial D_y/\partial y$, and $\partial D_z/\partial z$). The finite difference formulas are calculated using the method of undetermined coefficients.[52] The method forces the finite difference formula to successfully predict the derivative of each of the functions x^j, $j = 0 \longrightarrow N-1$, where N is the number of points in the formula. This creates a set of linear equations which is solved by Gaussian elimination. An example of the method of undetermined coefficients is given in Appendix V.

The method of lines reduces the set of nonlinear partial differential equations to a stiff set of nonlinear ordinary differential equations, which is solved using DGEAR from the IMSL library.[52] For example a 3 x 3 x 3 grid replaces the system of 17 PDE's and 8 ODE's with 467 (17 x 3 x 3 x 3 + 8) ODE's. A sample data deck, sample output, and the source code listing for the chip model are given in Appendix VII.

SIMULATION STUDY

In order to assess the predictive power of the chip model, a brief simulation study was conducted. The model was used to simulate selected runs from the reaction kinetics study above (series CU), kraft pulping studies by Aurell and Hartler[53] (series CV) and Akhtaruzzaman[54] (series CW), and a low lignin kraft-AQ study by McDonough and Van Drunen[25,55] (series CT).

For all the runs a 3 x 3 x 3 point grid was used with even grid spacing and fourth order finite difference formulas. This gave acceptable accuracy and relatively fast run times, ranging from two to seven hours of processor time on a Burroughs 6930 mainframe computer. The run times increase with decreasing chip size.

Series CU used two widely different conditions from the reaction kinetics study. The conditions for the two simulations are given in Table 2.

Representative results are given in Fig. 17 and Fig. 18. In these figures, the solid line is the chip model output, the solid diamonds are the reaction equation predictions, and the open diamonds are the experimental results. In general, the simulation results are in excellent agreement with the predictions of the kinetic models, indicating that the chip model is internally consistent.

Table 2. Conditions for series CU.

	Common Parameters
Wood	Southern pine (*Pinus elliottii*, *P. palustris*, *P. rigida*, or *P. taeda*)
Lignin	30.1%
Cellulose	40.2% (38.6 x 1.042)
Glucomannan	16.7%
Xylan	8.7%
Extractives	3.3%
Acetyl	1.3%
Liquor:wood	40
Chip size	4 x 4 x 4 mm
Chip density	430 kg m^{-3}
	Unique Parameters
CU/2 simulates reaction kinetics study runs E-3 and F-1	
[NaOH]	1.002M
[NaSH]	0.198M
[AQ]	1.010 mM
Temp.	2 min at 27°C, exponential rise to 190°C (reaches 189°C at t = 21.5 min), t_{end} = 38.4 min
CU/4 simulates reaction kinetics study runs N-2 and N-4	
[NaOH]	2.999M
[NaSH]	0
[AQ]	0
Temp.	2 min at 27°C, exponential rise to 144°C (reaches 143°C at t = 17.5 min), t_{end} = 24 hr

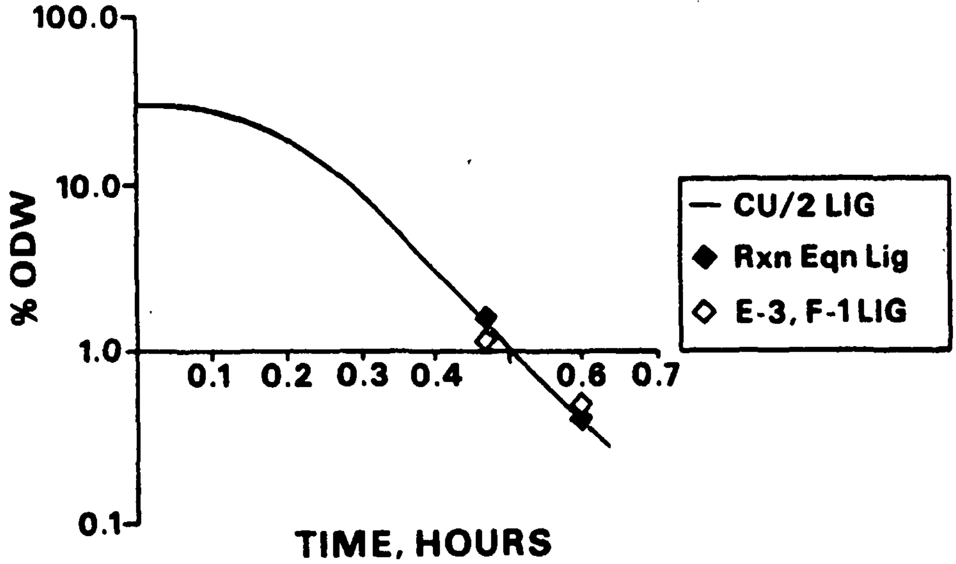


Fig. 17. Reaction kinetics study CU/2 lignin vs. time.

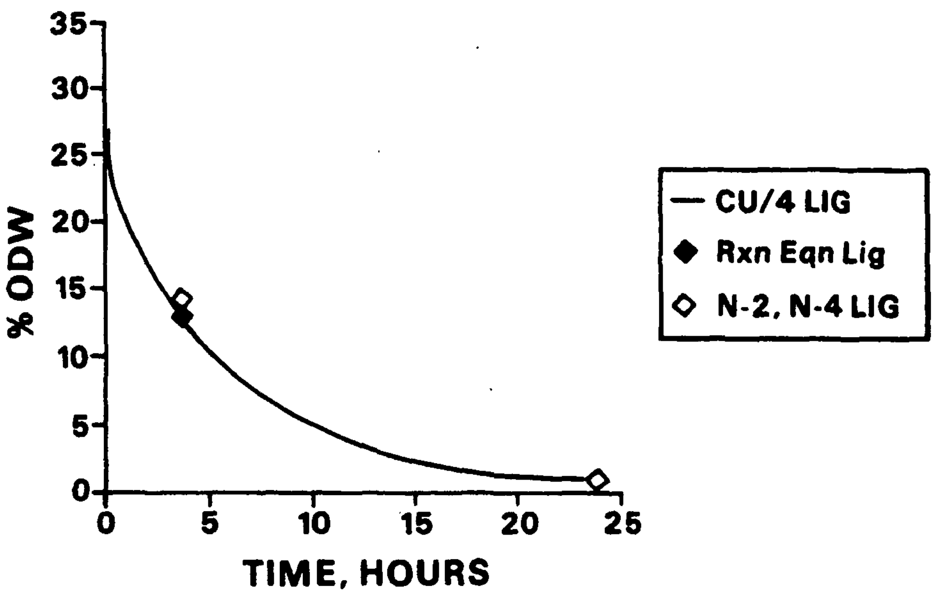


Fig. 18. Reaction kinetics study CU/4 lignin vs. time.

Series CV simulated two runs from Aurell and Hartler.[53] The study was one of the first to report carbohydrate data along with the usual lignin and yield data. Estimating chip size was a challenge. Industrial wood chips were screened with round hole screens. The fraction used passed through a 29 mm screen and was held on a 16 mm screen. I used the average of the two screen hole diameters (22.5 mm) as the chip length and width, and used 20% of the chip length (4.5 mm) as the chip thickness. The conditions for series CV are presented in Table 3.

Table 3. Conditions for series CV.

	Common Parameters
Wood	*Pinus sylvestris* (Scotch pine)
Lignin	27.3%
Cellulose	38.8% (37.3 x 1.042)
Glucomannan	19.3%
Xylan	9.8%
Extractives	4.0%
Acetyl	1.3%
Liquor:wood	4
Sulfidity	25%
AQ charge	0
Chip size	22.5 x 22.5 x 4.5 mm
Chip density[56]	358 kg m^{-3}
Temperature	2 hr from 70°C to 170°C, 2 hr at 170°C
	Unique Parameters
CV/1 simulates	Aurell and Hartler,[53] Table 1, chip sample B
EA	12.21%
CV/2 simulates	Aurell and Hartler,[53] Table 3, chip sample B
EA	19.38%

The results are presented in Fig. 19 to 23. In these figures, the solid line are the chip model predictions, the solid diamonds are the experimental results at low EA, and the open diamonds are the experimental results at high EA.

The chip model is somewhat slow in delignification under kraft conditions at low l:w, as shown in Fig. 19. This is in contrast with the kraft-AQ runs discussed below (series CT), in which delignification is too fast, especially at high AQ concentrations.

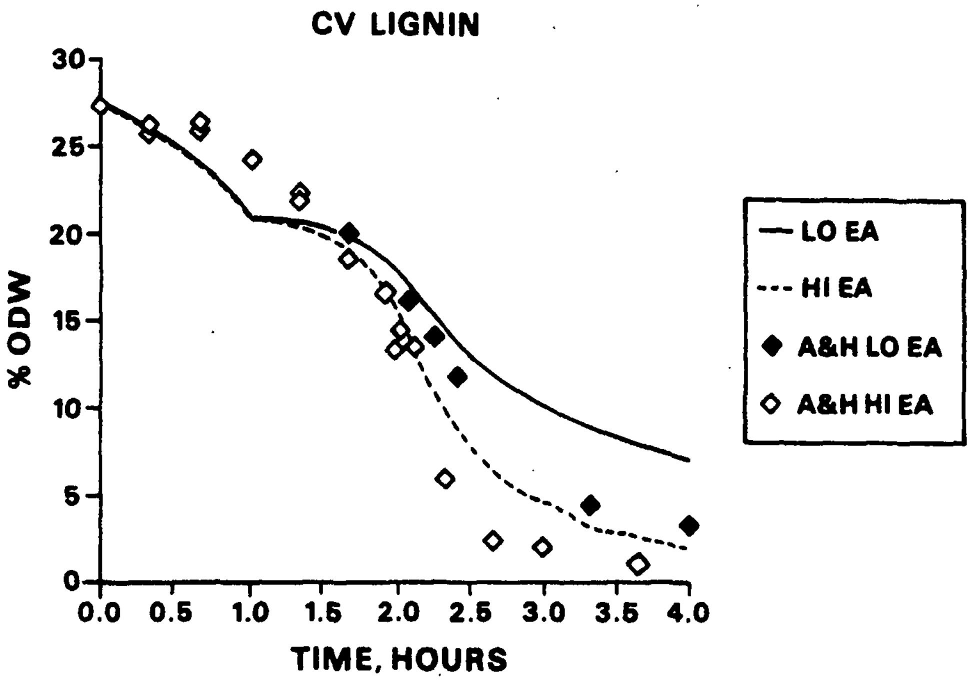


Fig. 19. Aurell and Hartler[53] series CV lignin *vs*. time.

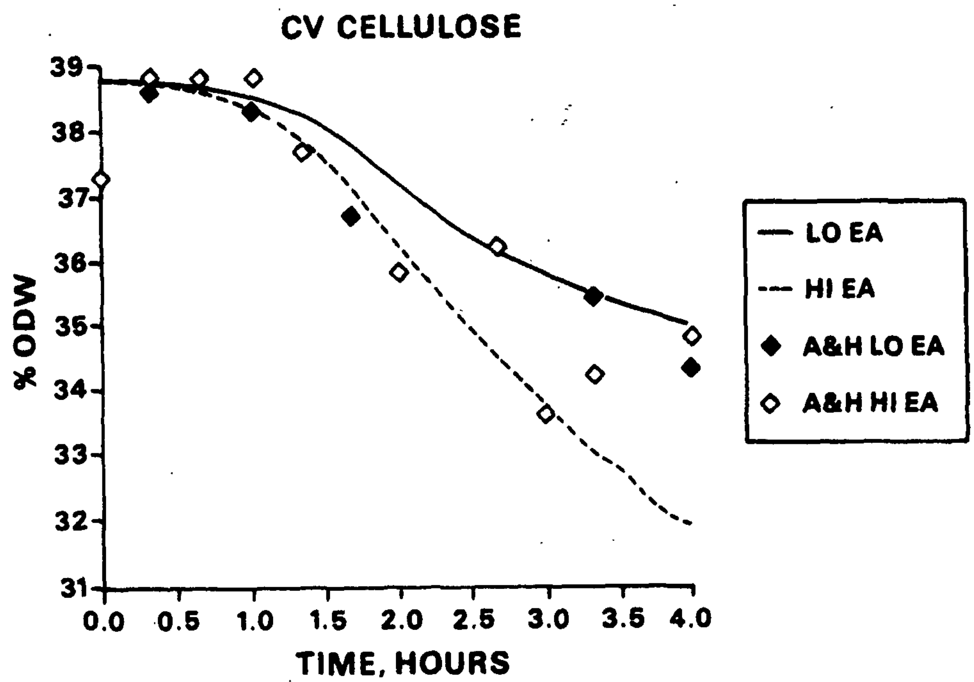


Fig. 20. Aurell and Hartler[53] series CV cellulose *vs*. time.

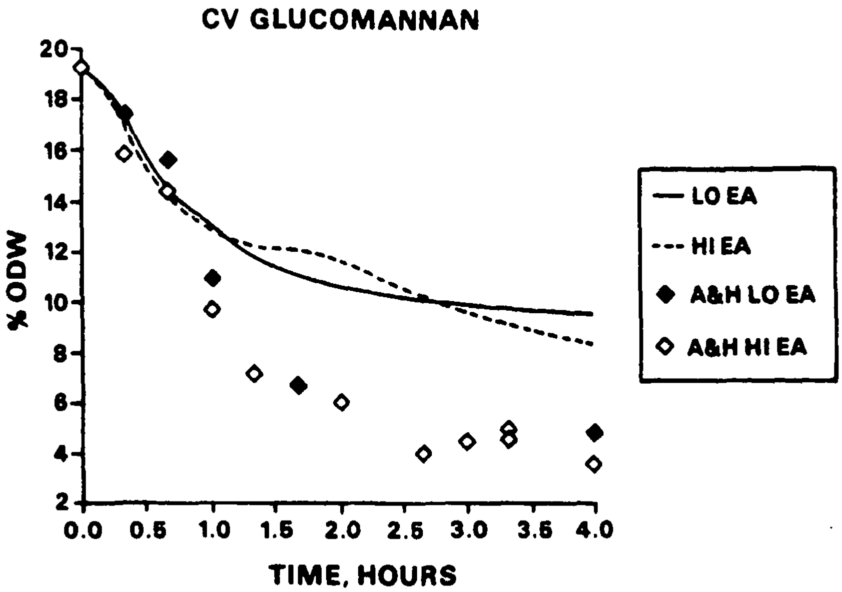


Fig. 21. Aurell and Hartler[53] series CV glucomannan vs. time.

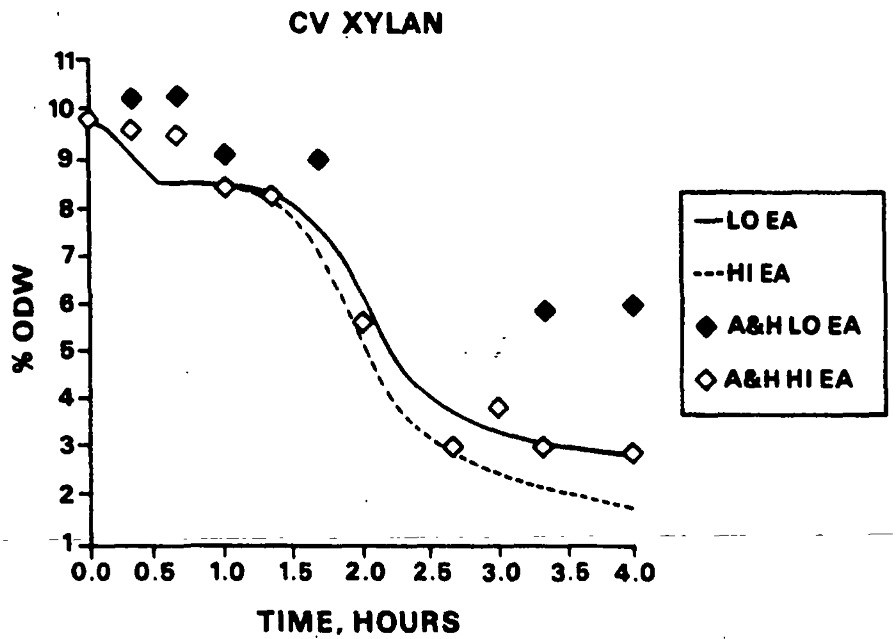


Fig. 22. Aurell and Hartler[53] series CV xylan vs. time.

The cellulose predictions are well within experimental error, as shown in Fig. 20. Particularly encouraging is that the cellulose fractional yield at time zero, 1.042, which was fitted using the reaction kinetics study (southern pine) data, works quite well with Scotch pine. This implies that a small portion of the cellulose is inaccessible to carbohydrate analysis in the original wood but is accessible after a short exposure to alkaline pulping liquor.

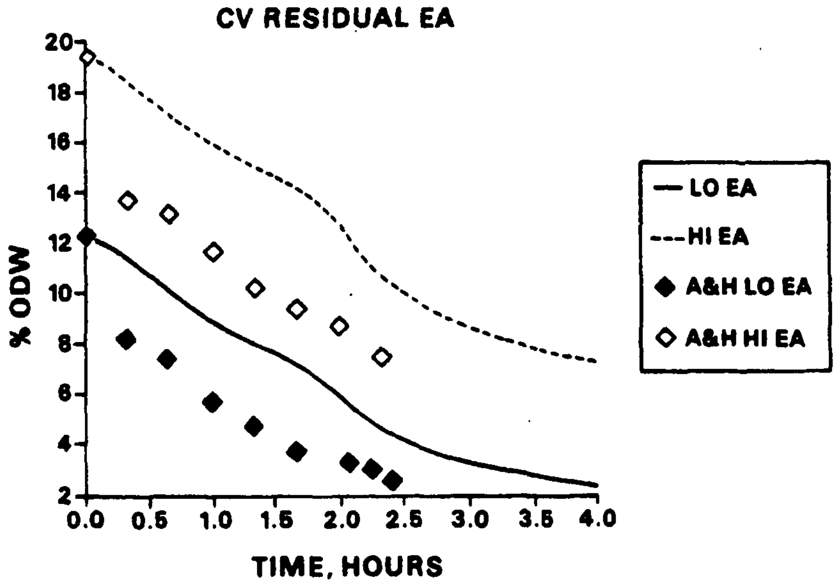


Fig. 23. Aurell and Hartler[53] series CV residual EA vs. time.

The glucomannan results (Fig. 21) indicate that the chip model correctly predicts that the glucomannan reaction rate is relatively insensitive to [NaOH], but overpredicts the glucomannan concentration in the bulk phase.

The xylan results (Fig. 22) indicate that the chip model predicts the xylan reaction rate fairly well at high EA but overpredicts the reaction rate at low EA. This may be due to xylan precipitation at high [xylan] and low [NaOH], conditions which did not exist in the high l:w cooks used in the reaction kinetics study.

The residual EA results (Fig. 23) are surprising. The chip model predicts the rate of change of EA fairly well after the first twenty minutes, but does not account for the initial drop in concentration seen experimentally. Apparently, there is a neutralization reaction in the initial phase which does not involve lignin, carbohydrates, extractives, or acetyl groups. This initial drop is larger for the high EA case (≈ 6% EA drop) than the low EA case (≈ 4% EA drop).

Series CW simulates the work of Akhtaruzzaman,[54] who did an extensive study of the effect of chip size on pulp properties. The experimental conditions for series CW are given in Table 4. It was difficult to make meaningful comparisons between the simulation results and Akhtaruzzaman's experimental results since most of the reported pulp properties were based on screened pulp. Estimating screened pulp properties from a three dimensional grid is a formidable programmin challenge, and unfortunately time did not permit me to implement it.

The only accessible result was total yield, as shown in Fig. 24. In this figure the solid symbols are predictions from regression equations of the experimental data, and the open symbols are the chip model results. The figure indicates that the chip model correctly predicts the trends of chip size effect on total yield but underestimates the magnitude of the effects. This may be due to poorly impregnated chips, as suggested by Gustafson.[32] The rest of the results, given in Appendix III, show the same trend.

Series CT simulated an extensive low lignin pulping study by McDonough and Van Drunen.[25] This study was chosen to test the model well into the residual phase over a large experimental space. Chip size was easy to estimate since I was able to obtain a sample of the chips used in the study. The dimensions used represent the average of 20 randomly selected chips. The simulation conditions are summarized in Table 5.

Results for lignin are shown in Fig. 25 to 29. Additional results are given Appendix III. In these figures the closed symbols represent the predictions of

Saffran's[55] regression equations of McDonough and Van Drunen's data, and the open symbols represent the output of the chip model.

Table 4. Conditions for series CW.

Common Parameters	
Series CV simulates	Akhtaruzzaman[54]
Wood	_Pinus sylvestris_ (Scotch pine)
Lignin	27.3%
Cellulose	38.8% (37.3 x 1.042)
Glucomannan	19.3%
Xylan	9.8%
Extractives	4.0%
Acetyl	1.3%
Liquor:wood	4
EA	17.05%
Sulfidity	30%
AQ charge	0
Chip density[56]	358 kg m^{-3}
Temperature	1 hr from 27°C to 170°C, 4.25 hr at 170°C
Unique Parameters	
CW/5 chip size	33 x 33 x 12 mm
CW/6 chip size	33 x 33 x 3 mm
CW/8 chip size	16 x 16 x 12 mm
CW/9 chip size	16 x 16 x 3 mm

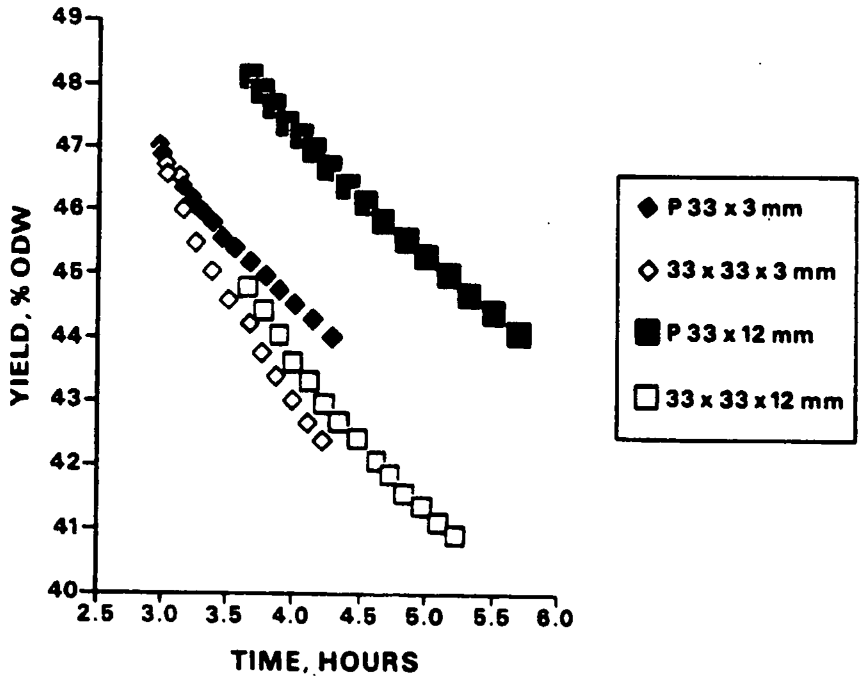


Fig. 24. Akhtaruzzaman[54] series CW yield vs. time.

Two points can be made from these results: bulk delignification tends to be too fast, and conditions where dissolved lignin concentration is high (very fast bulk delignification or low l:w) tends to give lignin condensation rates which are too high. These two points probably are related. The reaction kinetics study used a high (40:1) l:w, resulting in low dissolved lignin concentrations, hence low lignin condensation rates and low lignin contents in the residual phase. The data were fitted using a logarithmic transformation to emphasize the residual phase data. Any systematic error in the residual phase data is magnified by the transformation and the 10x increase in dissolved lignin concentration (40:1 l:w ——> 4:1 l:w).

As an additional demonstration of the predictive capabilities of the chip model, plots were made of the spatial variation of pulp properties in one and two dimensions. The orientations for these plots are shown in Fig. 30 and 31.

Table 5. Conditions for series CT.

	Common Parameters
Series CT simulates	McDonough and Van Drunen[25]
Wood	Southern pine (*Pinus elliottii*, *P. palustris*, *P. rigida*, or *P. taeda*)
Lignin	28.4%
Cellulose	40.1% (38.5 x 1.042)
Glucomannan	17.7%
Xylan	7.7%
Extractives	3.3%
Acetyl	1.3%
Liquor:wood	4
EA	18%
Sulfidity	25%
AQ charge	0.1%
Chip size	16.5 x 17.7 x 4.3 mm
Chip density	430 kg m^{-3}
Temperature	30 min at 80°C, 90 min to 173°C, 2.75 hr at 173°C
	Unique Parameters
CT/2 temp.	30 min at 80°C, 90 min to 185°C, 1.25 hr at 185°C
CT/3 temp.	30 min at 80°C, 90 min to 161°C, 7.25 hr at 161°C
CT/4 EA	21.5%
CT/5 EA	15.1%
CT/6 sulfidity	44.1%
CT/7 sulfidity	14.2%
CT/8 AQ charge	0.322%
CT/9 AQ charge	0.031%
CT/10 liquor:wood	5.83
CT/11 liquor:wood	2.74

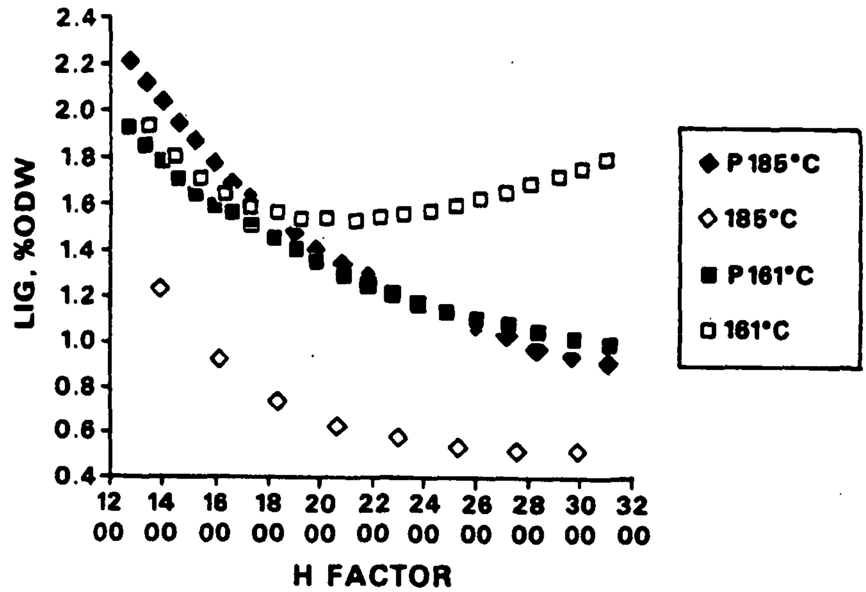


Fig. 25. McDonough and Van Drunen[25] series CT lignin vs. H-factor: effect of temp.

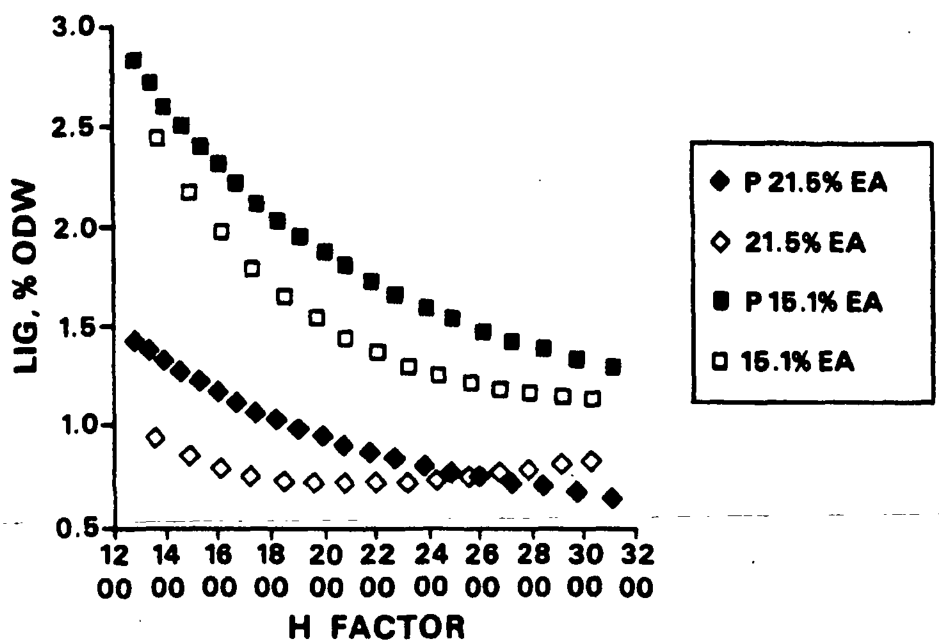


Fig. 26. McDonough and Van Drunen[25] series CT lignin vs. H-factor: effect of EA.

The 1D profiles are shown in Fig. 32 to 37. Figures 32 (3 mm thick) and 33 (12 mm thick) show the change in [lignin] with time. The top profiles are at time = 0, the bottom profiles are at time = 5 hr, and the rest are spaced at 1 hr intervals [(0, 5, 1)]. The [lignin] profiles generated for the 12 mm chip are probably more uniform than would be seen in an industrial cook since the chip model assumes complete penetration of the chip with cooking liquor, which is difficult to achieve in practice for such a thick chip.

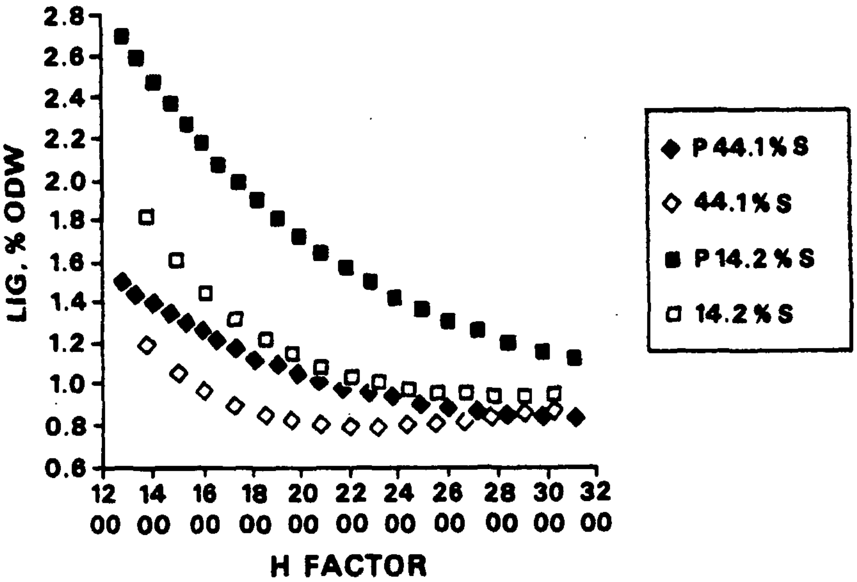


Fig. 27. McDonough and Van Drunen[25] series CT lignin vs. H-factor: effect of sulfidity.

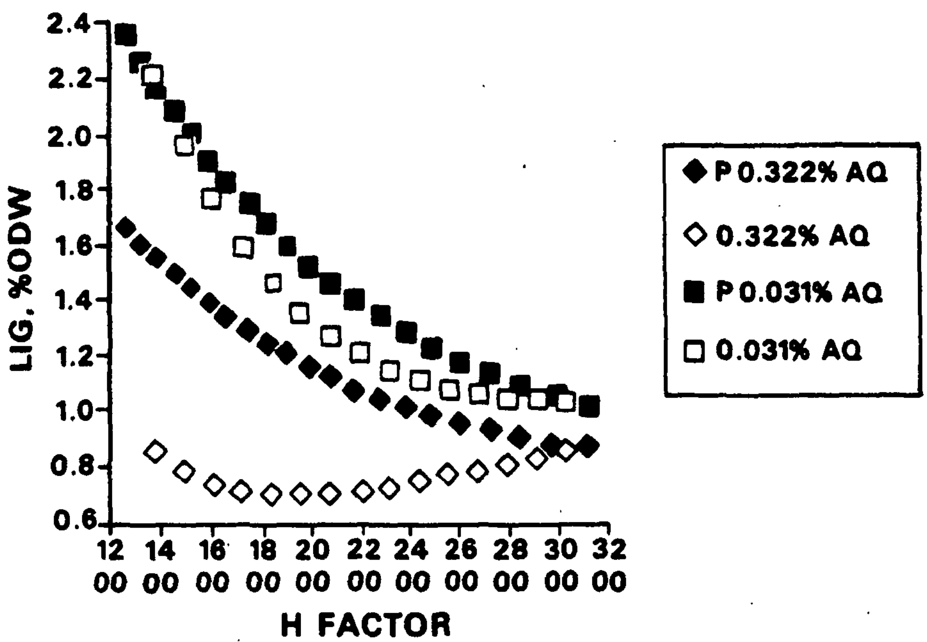


Fig. 28. McDonough and Van Drunen[25] series CT lignin vs. H-factor: effect of AQ.

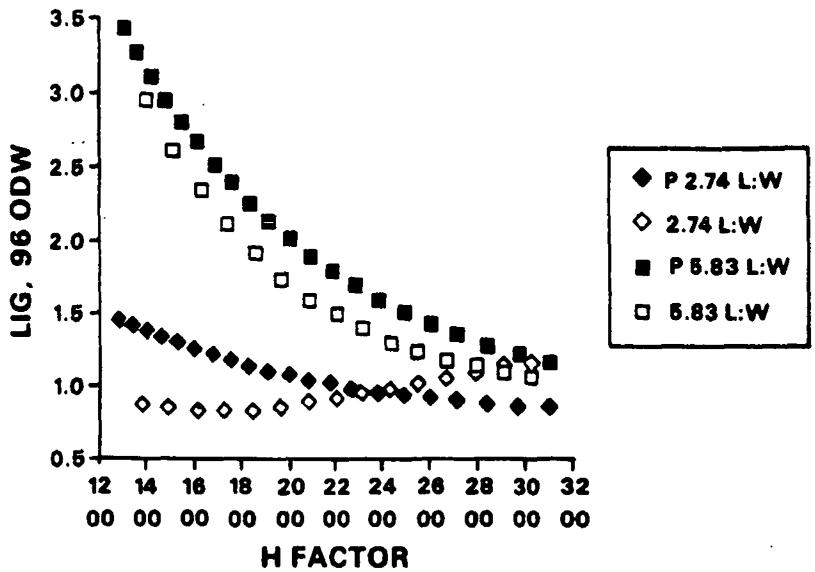


Fig. 29. McDonough and Van Drunen[25] series CT lignin *vs.* H-factor: effect of l:w.

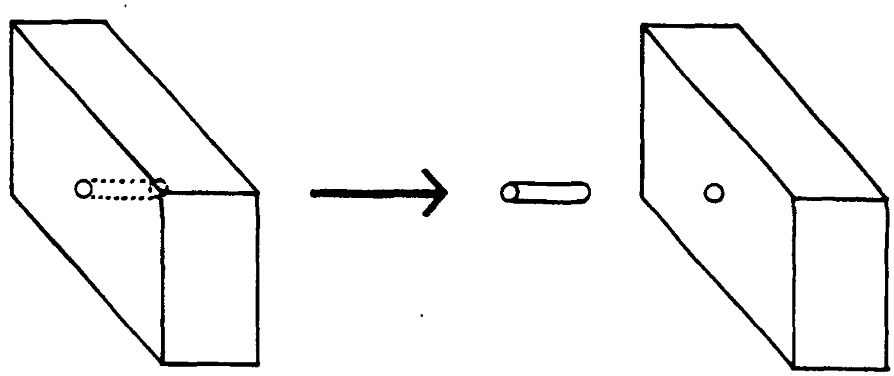


Fig. 30. Orientation of 1D plot.

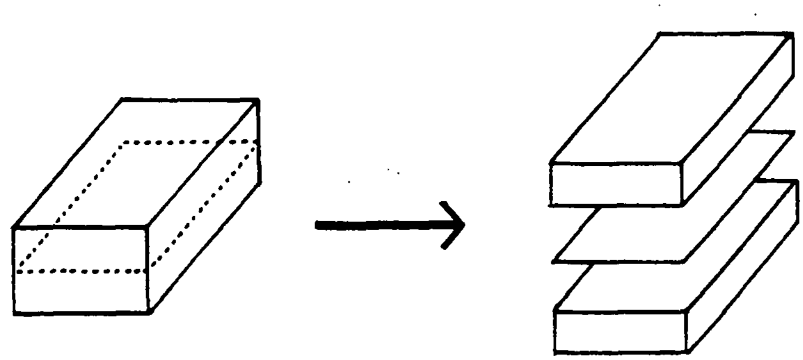


Fig. 31. Orientation of 2D plot.

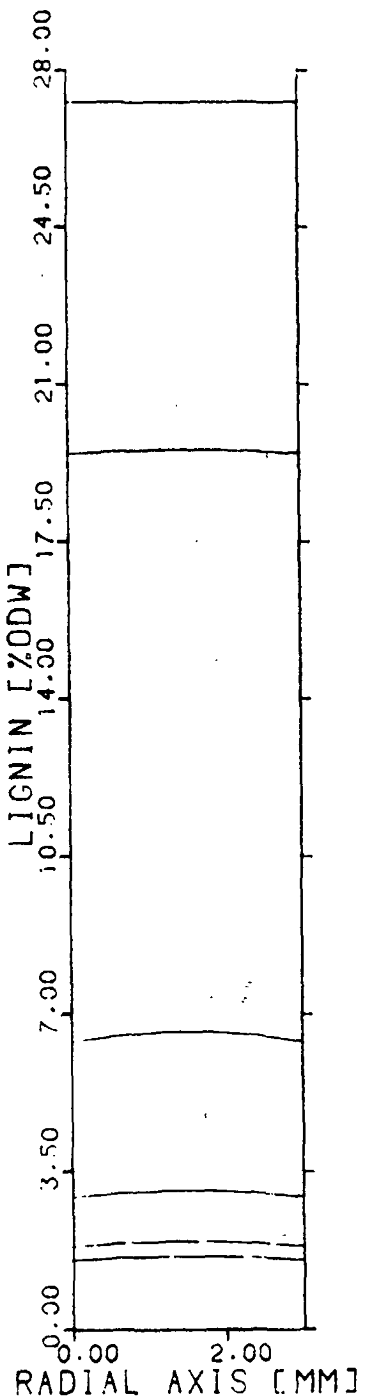


Figure 32. Lignin across the chip thickness. Run CW/6 time [hr] (0,5,1).

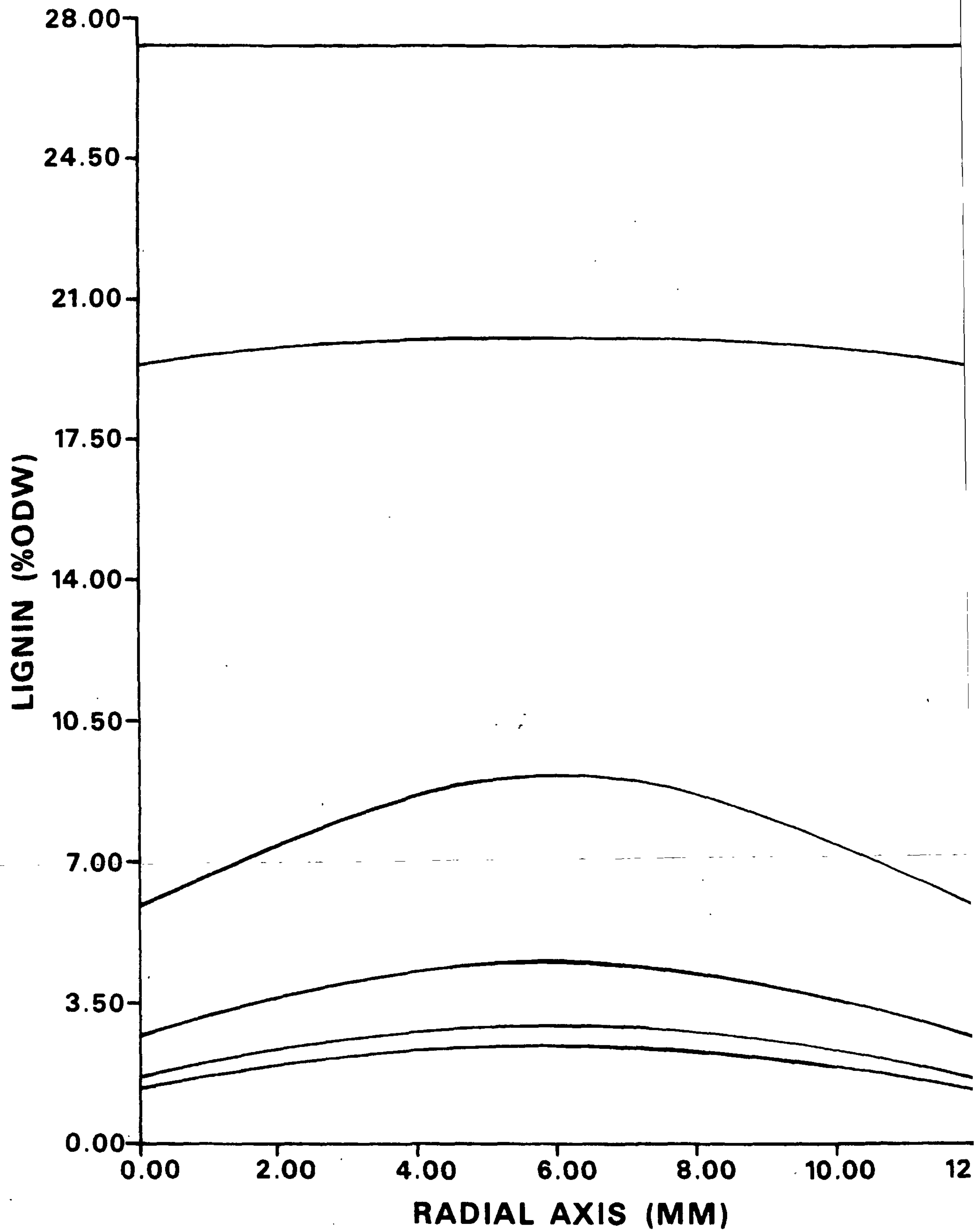


Figure 33. Lignin across chip thickness run CW/5 time [hr] (0,5,1).

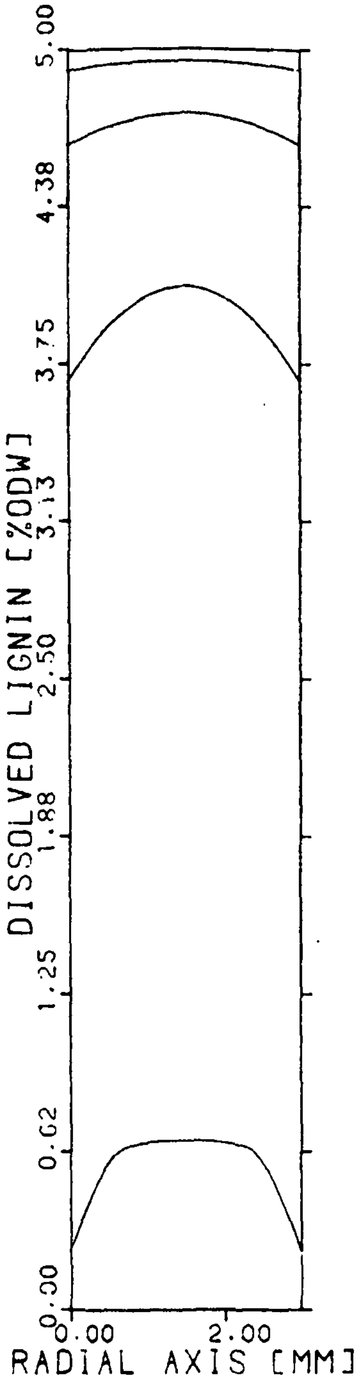


Figure 34. Dissolved lignin across chip thickness run CW/6 time [hr] (1,5,1).

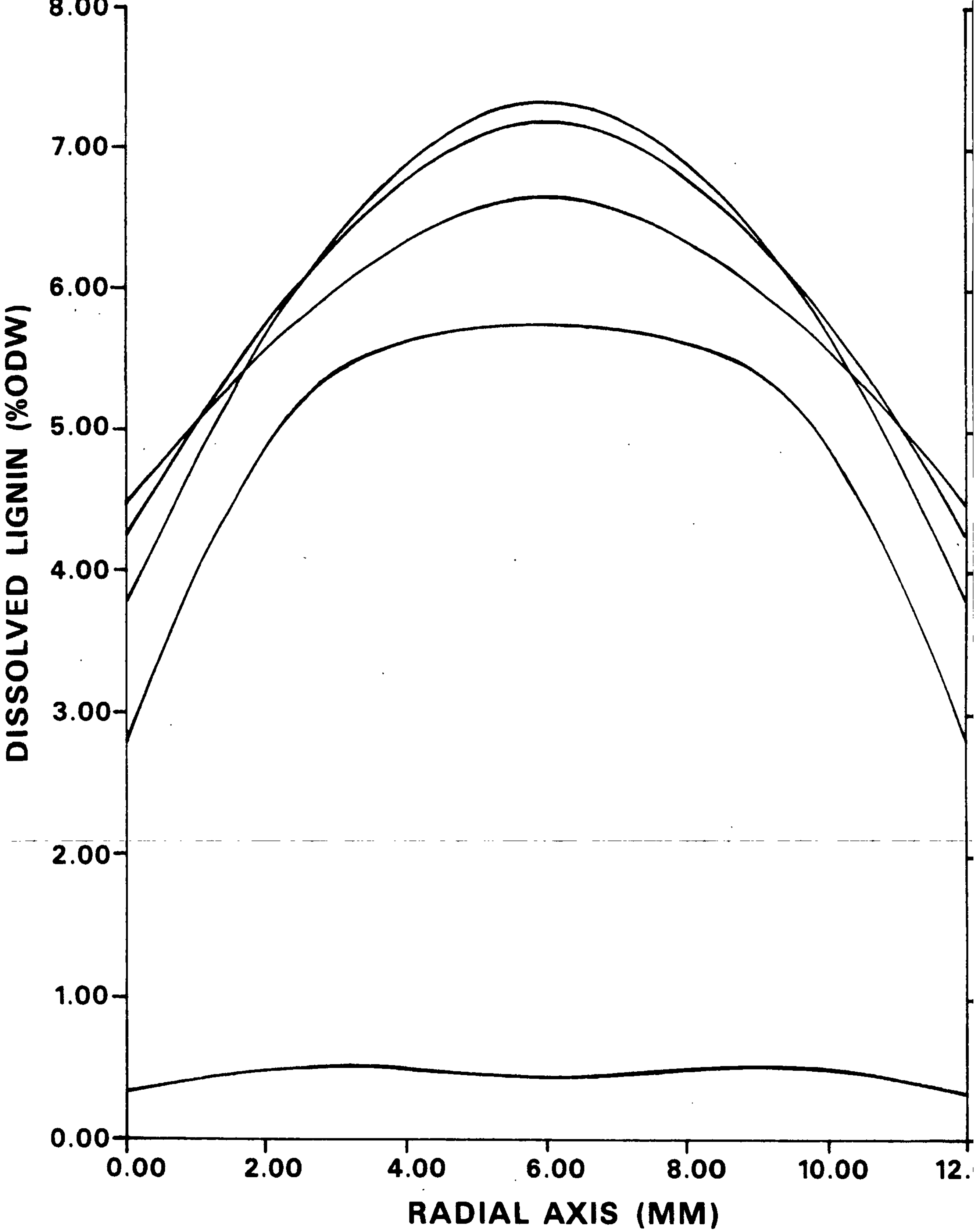


Figure 35. Dissolved lignin across chip thickness run CW/5 time [hr] (1,5,1).

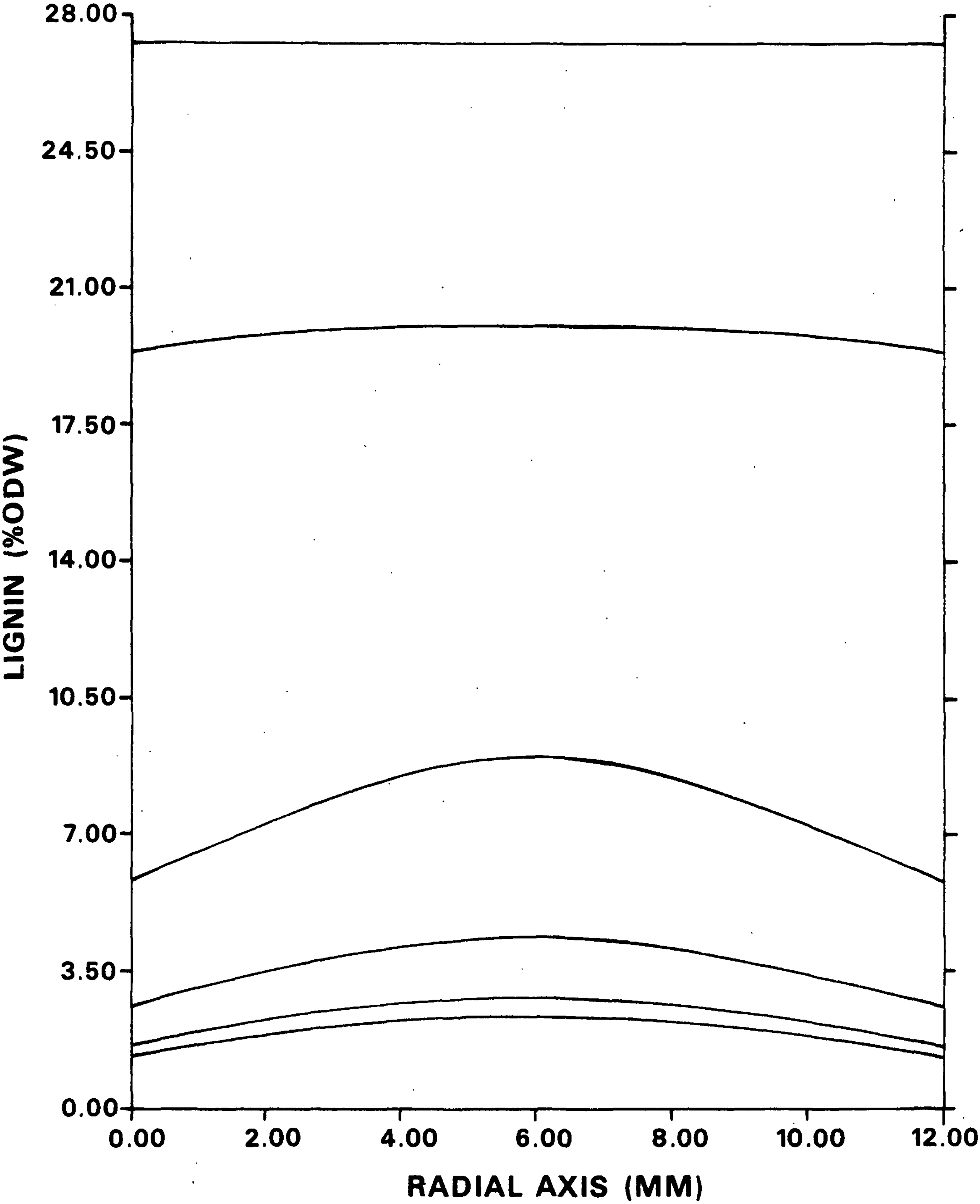


Figure 36. Lignin across chip thickness run CY/5 time [hr] (0,5,1).

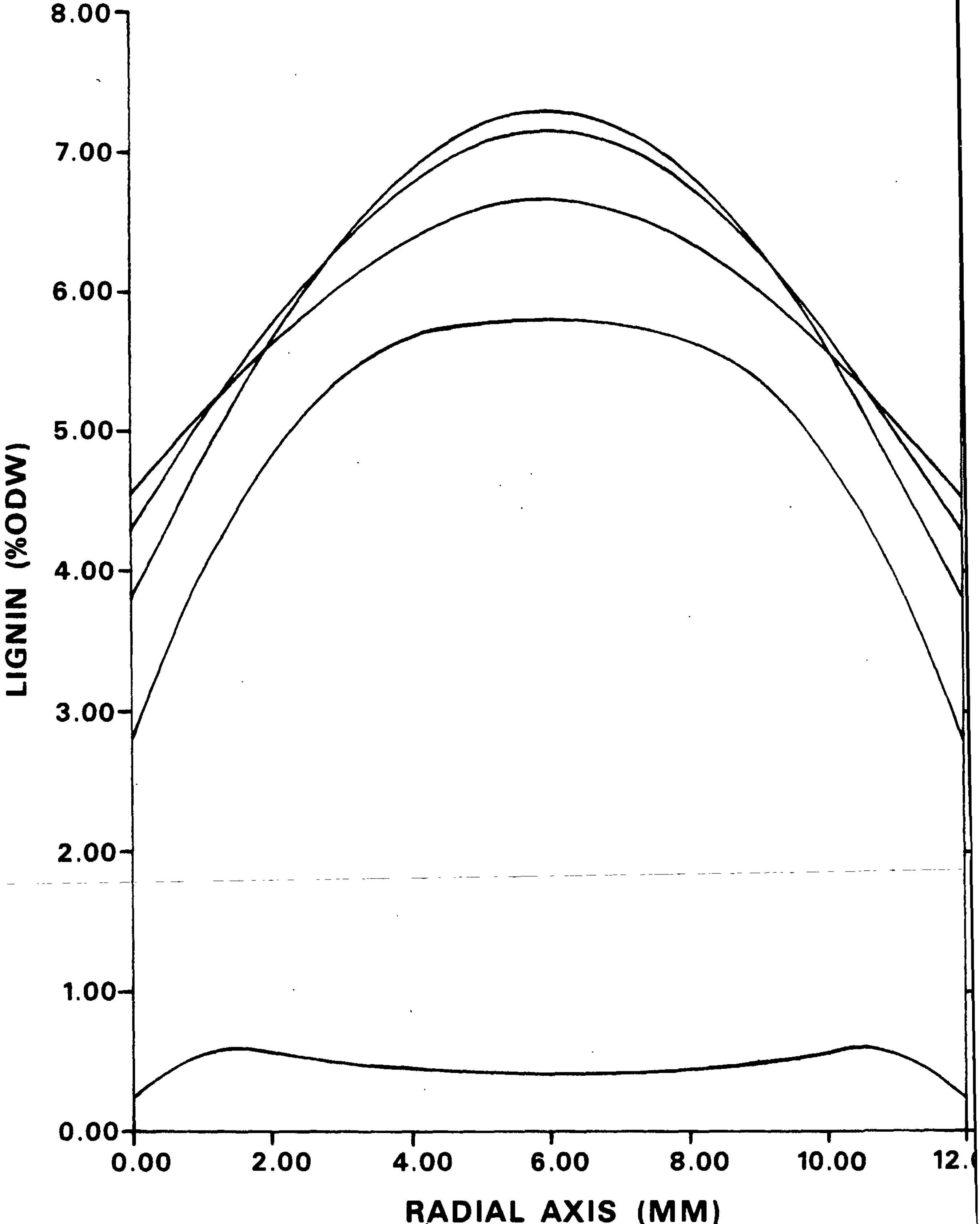


Figure 37. Dissolved lignin across chip thickness run CY/5 time [hr] (1,5,1).

The [dissolved lignin] profiles, Fig. 34 (3 mm thick) and 35 (12 mm thick), show quite a bit of variation. The surface concentration of dissolved lignin increases with time in these profiles, which are plotted at (1, 5, 1). For the 12 mm thick chip, the center [dissolved lignin] peaks at t = 3 hr, while the surface [dissolved lignin] steadily increases. Note the bimodal peak at t = 1 hr. This is at the point were the bulk liquor has just reached cook temperature, the overall production rate of dissolved lignin is fastest, and the temperature gradient within the chip is sharpest. The reaction rate is fastest at the chip surface and steadily decreases toward the chip center, so that [dissolved lignin] increases away from the chip center. Mass transfer at the chip surface efficiently moves the dissolved lignin into the bulk liquor, where [dissolved lignin] is lowest, which causes [dissolved lignin] to increase away from the chip surface. The combination of these two effects leads to the observed bimodal profile.

In order to confirm the accuracy of the model with an evenly spaced 3 x 3 x 3 (3^3) grid, an additional run, CY/5 was done. Run CY/5 is identical to CW/5 except that the 3^3 grid was replaced with an evenly spaced 5 x 5 x 5 (5^3) grid. This substantially increased execution time as expected (2 hr ——> 35 hr). The [lignin] profiles for the 3^3 grid (Fig. 33) and the 5^3 grid (Fig. 36) are in excellent agreement. The [dissolved lignin] profiles for the 3^3 grid (Fig. 35) and the 5^3 grid (Fig. 37) are in excellent agreement except for the time = 1 hr profiles, where the bimodal peaks are sharper and are moved toward the chip surface.

Figures 38 and 39 are 2D [lignin] contour plots at time = 3.375 hr of the 3^3 and 5^3 grids, respectively. The contour lines agree well from 2.1% ODW to about 3.1% ODW, then begin to diverge, corresponding to a drastic decrease in [lignin] gradient in both plots, which greatly magnifies differences between the two runs. I believe that the "wiggles" in Fig. 38 are an artifact of the cubic spline interpolation algorithm used to generate the 2D plots. The 3D projection plots qualitatively show that the [lignin] distributions resulting from the two grids quite similar.

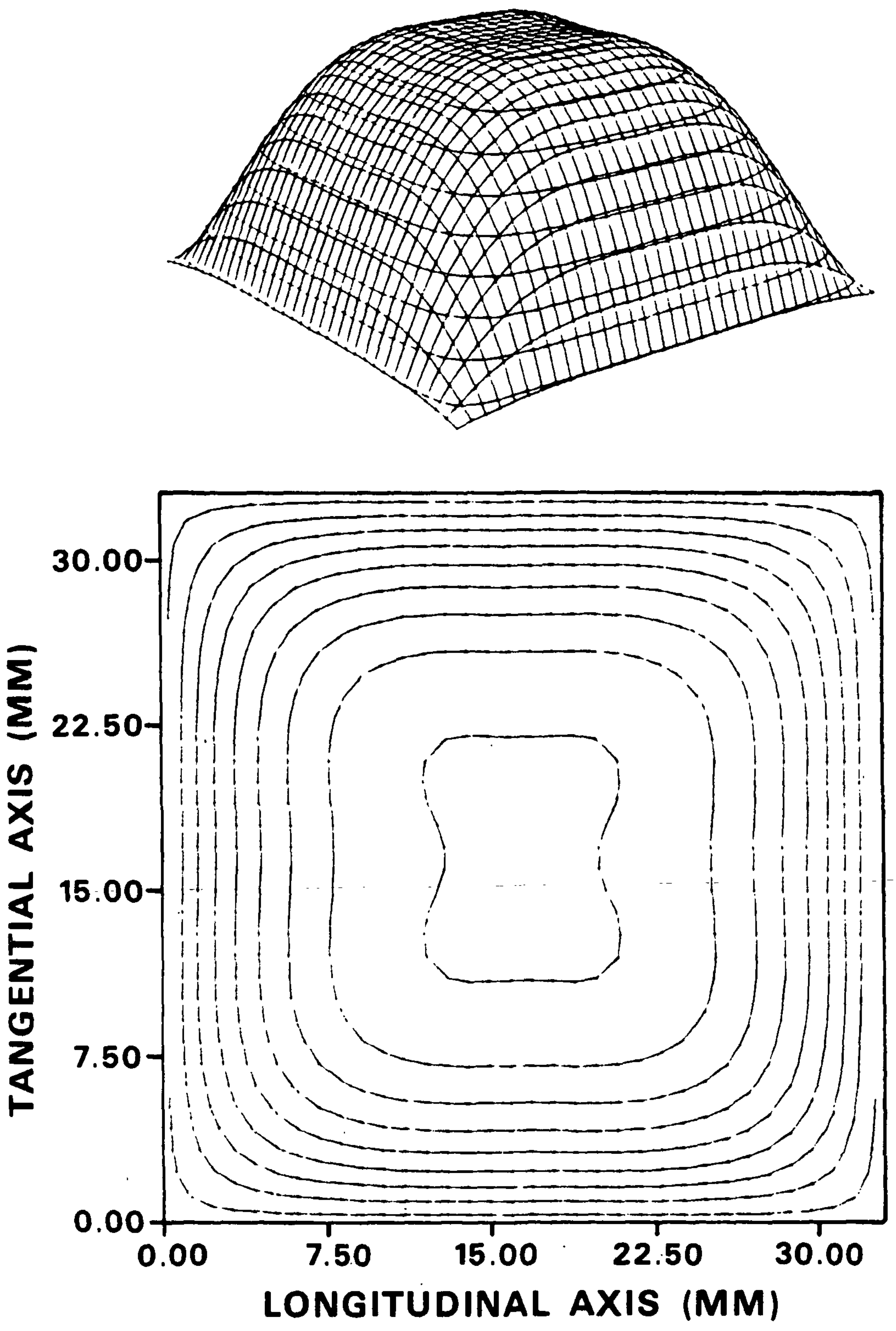


Figure 38. Lignin [%ODW] (2.1, 3.7, 0.2) run CW/5 time = 3.375 hours.

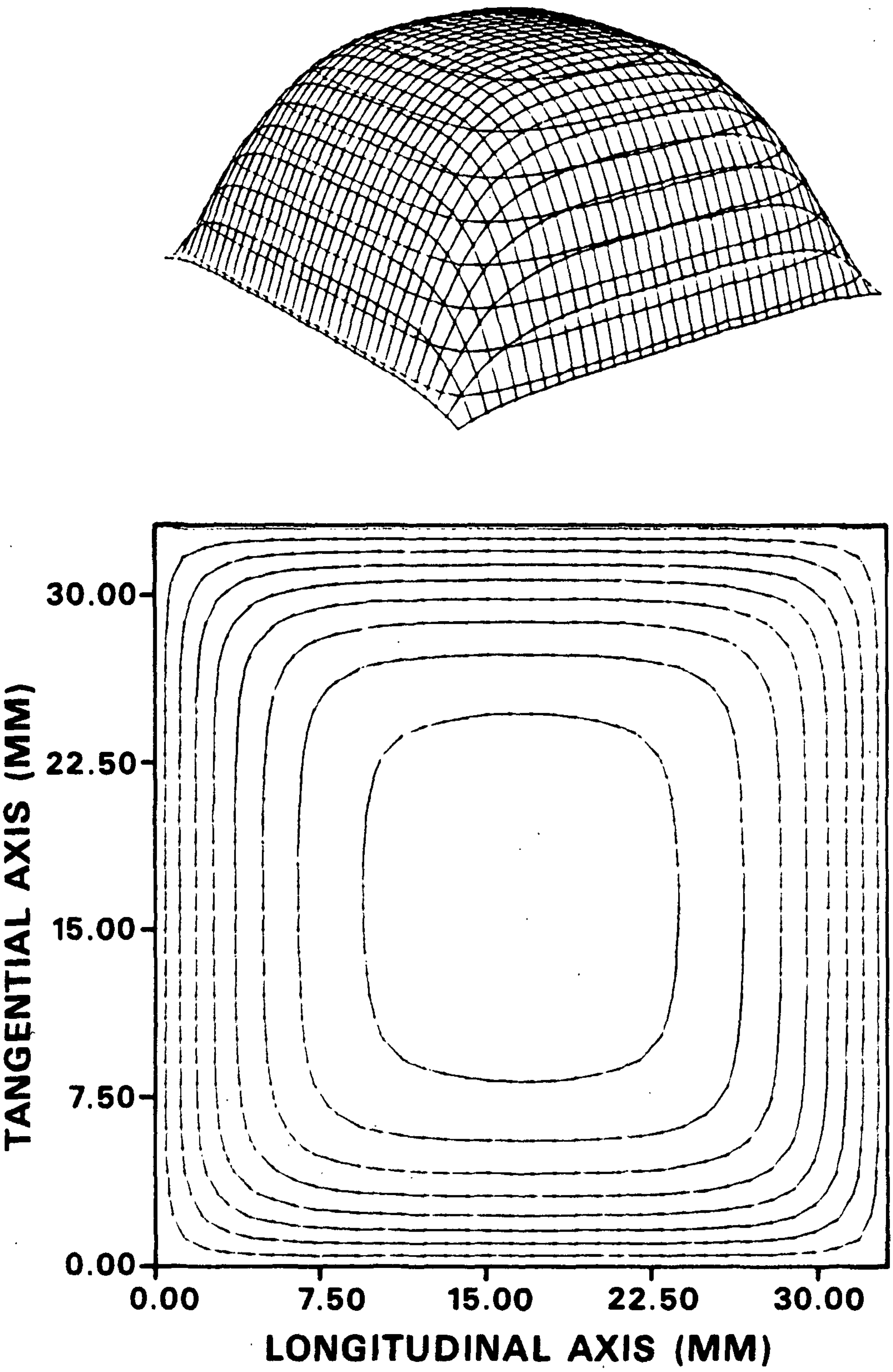


Figure 39. Lignin [%ODW] (2.1, 3.5, 0.2) run CY/5 time = 3.375 hours.

CONCLUSIONS

The chip model presented here makes a significant step forward in predictive capability, which will prove useful in future explorations and optimizations of kraft pulping. The major improvements of the chip model over prior dynamic models include

- it accounts for nonlinear heat and mass transfer of NaOH, NaSH, AQ, dissolved lignin, and dissolved solids in three space dimensions,
- it accounts for separate reactions of lignin, cellulose, glucomannan, xylan, and pulp viscosity,
- it accounts for lignin condensation and residual delignification,
- it accounts for reactions involving AQ,
- it may be used with virtually any type of digester,
- it may be easily used with other chemical pulping processes if reaction and diffusion data are available, and
- each kinetic equation used was chosen from several possible candidates using experiments specifically designed to discriminate between the candidates.

This last point is very important, for it is all too easy to propose a particular reaction mechanism, develop an equation based on that mechanism, then generate plausible data that "prove" the mechanism.

Knowing dissolved lignin concentrations and condensation rates in the chip center will greatly facilitate investigations of the relationship between lignin content and screened rejects. The addition of AQ reactions to the model will help to assess the economics of pulping with this catalyst. Separate accounting of the various species will allow yield selectivity and viscosity selectivity to be optimized.

The numerical solution method maximizes the accuracy attainable for a given number of grid points. The points are concentrated where they are needed and the boundary conditions are fully exploited to enhance accuracy.

The simulation study was a qualified success. The numerical solution method gave adequate numerical accuracy, even when speed was maximized at the expense of accuracy. The chip model accurately predicts the conditions used to generate the reaction equations. The chip model is less accurate when predicting industrial cooks, especially lignin in the residual phase and xylan at low [NaOH] and low l:w.

FUTURE WORK

The reaction kinetics study should be continued with l:w as an added experimental variable. This will require a more accurate representation of [NaOH] during the cook. Low l:w data added to the existing high l:w data should rapidly result in equations which give excellent agreement at both high and low l:w.

The applicability of the model could be enhanced by extending it to simulate chip mixtures, as was done by Gustafson[32] for one-dimensional chips. Computational speed and memory requirements should be directly proportional to the number of chips simulated.

An alternate avenue of exploration is to simulate chip surface irregularities and wood variation within the chip. This will require many grid points to follow the surface features and to handle internal variations such as growth rings and knots.

EXPERIMENTAL

WOOD SHAVINGS

The wood supply for the reaction kinetics study had to be processed into a form which met two related criteria: diffusion resistance had to be as small as possible (at least one small dimension), and the wood had to pack loosely (minimal clumping). Wood shavings fulfilled these requirements well. First, they could be made very thin, and second, they were curly, naturally giving a low packing density (unlike chips or wood meal). The shavings used in the kinetics study were prepared using a power jointer. Two 61-cm long, 30-cm diameter bolts of southern pine (Pinus elliottii, Pinus palustris, Pinus rigida, or Pinus taeda) were debarked and cut in half lengthwise. One half of each bolt was run through the jointer. The shavings were air dried, then screened using a 1 mesh screen [(1 wire per inch (2.54 cm)] and a 2 mesh screen. The fraction which passed through the 1 mesh screen and was retained on the 2 mesh screen was used in this study. The resulting shavings were approximately 15 mm long and 40 mm wide, with an average thickness of 0.34 mm.

PULPING RUNS

All pulping runs used one of eight stainless steel, 450 mL capacity bombs. The bombs had an internal thermocouple arrangement which allowed liquor temperature to be monitored. The bombs were heated in a multiunit digester. The multiunit digester consisted of the bombs, an oil bath, an oil reservoir, a heater, and a temperature controller. The bombs were mounted in a rack inside the bath which rotated the bombs end over end. The movement of the gas bubble inside each bomb mixed the liquor well and enhanced mass transfer between liquor and shavings. Bath temperature was controlled using a Honeywell programmable digital controller, which was used to adjust the output of an electric heat exchanger in the recirculation line of the multiunit digester. Oil circulation in the bath was sufficient to eliminate temperature variations across the bath.

Before each set of cooks, the bombs to be used were randomly selected from the eight bombs available. The position (left to right) of each bomb in the oil bath was also chosen randomly. The bombs were rinsed with hot tap water, deionized water, and acetone, then allowed to air dry. Each bomb was filled with approximately 11 g a.d. wood shavings, weighed to 0.1 mg. Powdered AQ was weighed to 0.1 mg and poured on top of the shavings. Aliquots of NaSH solution (1.89M) and NaOH solution (5.38M) were measured to 1 mL using graduated cylinders and poured on top the shavings, together with enough deionized water to give 400 mL liquor. The deionized water was added first, NaSH solution second, and NaOH solution last. After the bombs were filled, the shavings were vacuum impregnated. The bombs were stoppered and vacuum was applied for 5 minutes, relieved, and reapplied for 5 minutes. The bombs were then sealed and mounted in the oil bath.

Rapid heat-up of the bombs was achieved by preheating the oil in the reservoir above the target temperature. At the start of a cook, oil was pumped into the bath, and bomb temperature rose quickly. Heat up was made even faster by ramping oil bath temperature down to the target temperature. The ramp was timed to end when bomb temperature was near equilibrium. A plot of a typical heat up curve is shown in Fig. 40.

At the end of a cook, the oil was stripped from the bombs with low pressure steam for five minutes. This quickly dropped bomb temperature to 100°C. The bombs were cooled to room temperature within a minute using cold tap water spray. Each bomb was opened, the liquor was vacuum filtered through a coarse fritted glass filter, and a liquor sample was stored in a polyethylene bottle. The pulp was washed with deionized water in a coarse fritted glass filter, broken up in a Waring blender, washed again, and leached overnight in 900 mL deionized water. The leached pulp was extracted in Soxhlet extractors using extra-coarse fritted glass extraction thimbles (two thimbles per cook), first with 150 mL ethanol for 12 cycles per thimble, then with 150 mL ethanol/benzene (1:2, volume/volume) for 24 cycles per thimble.

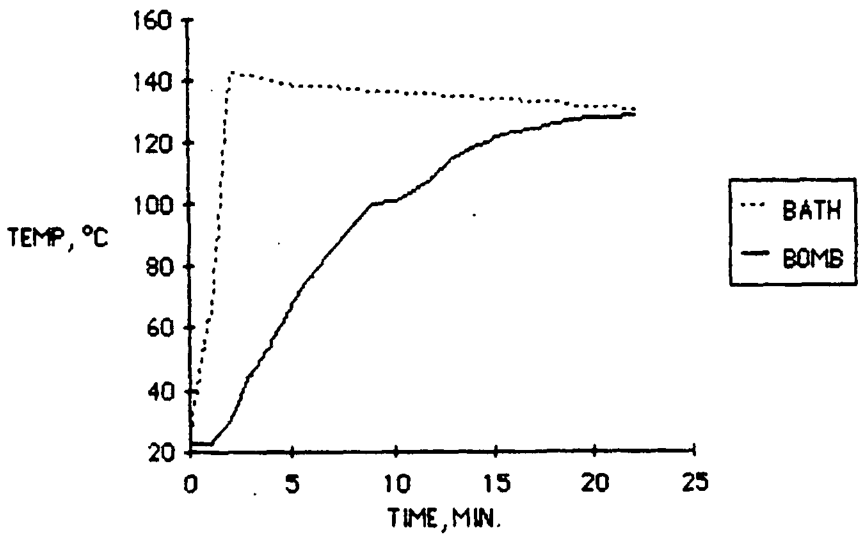


Fig. 40. Typical heat up curve for the multiunit digester.

The extracted pulp was spread out into culture dishes (90 mm diameter, 15 mm deep, three per cook), set in a fume hood overnight to allow the solvent to evaporate, and conditioned in a constant humidity room [50% RH, 73°F (23°C)] for a day. After conditioning the pulp was weighed to 0.1 mg. A small sample of pulp (at least 250 mg) was weighed to 0.1 mg and used for moisture determination (105°C for two hours). Approximately 330 mg of air dry pulp was weighed to 0.1 mg and analyzed for lignin[57] and sugars.[58] When pulp was available and the calculated kappa number of the pulp was less than 100 (kappa ≈ $\{([lignin]/[yield])/0.0015\}$[10]), viscosity determinations were done. A sample of pulp (2 g a.d. or all that was left, whichever was less) was bleached with 10 mL of sodium chlorite solution (200 g per liter) and 30 ml of acetate buffer (0.5 M, pH 3.1) for 24 hours at 30°C in sealed Kapak/Scotchpak plastic bags. The sample was washed in a glass Soxhlet extraction thimble, bleached and washed again, then reduced overnight with 10 mL of sodium borohydride solution (20 g per liter) and 30 mL of bicarbonate buffer (0.5<u>M</u>, pH 9.5). The reduced sample was washed and analyzed for cuene viscosity.[18]

ACKNOWLEDGMENTS

I would like to thank my thesis advisor, Dr. Tom McDonough, and the rest of my thesis advisory committee, Drs. Harry Cullinan, Bob Halcomb, and Earl Malcolm, for their enthusiasm and their counsel. The numerical analysis advice of Dr. Pete Parker is gratefully acknowledged.

I would like to thank The Institute of Paper Chemistry and its member companies for providing me with the opportunity and resources to pursue this thesis. The analytical assistance of Messrs. Art Webb and Harry Grady is greatly appreciated. Special thanks are due to Mr. Webb, whose flexibility in scheduling made the sequential experimental design strategy possible. I am grateful to the staff and students of the Institute for their expert repartee.

Finally, I would like to thank my family for their love and support, especially my father Larry, who inspired me, and my wife Allison, who made my goal her own and saw it through with remarkable serenity and patience.

LITERATURE CITED

1. Joint Textbook Committee of the Paper Industry. The pulping of wood, 2nd Edition, Volume 1. New York, NY, McGraw-Hill, 1969.

2. American Paper Institute Industry Fact Sheet. New York, NY, Feb. 27, 1985.

3. Fengel, D.; Wegener, G. Wood-chemistry, ultrastructure, reactions. New York, NY, de Gruyter, 1984.

4. Stone, J. E. The effective capillary cross-sectional area of wood as a function of pH. Tappi 40(7):539-41(1957).

5. Hartler, N. Penetration and diffusion in sulfate cooking. Paperi Puu 44(7):365-74(1962).

6. Glasser, W. G.; Glasser, H. R. The evaluation of lignin's chemical structure by experimental and computer simulation techniques. Paperi Puu 63(2):71-83(1981).

7. Rekunen, S.; Jutila, E.; Lahteenmaki, E.; Lonnberg, B.; Virkola, N. Examination of reaction kinetics in kraft cooking. Paperi Puu 62(2):80-90 (1980).

8. Vroom, K. E. The "H"-factor: a means of expressing cooking times and temperatures as a single variable. Pulp Paper Mag. Can. Convention Issue 1957:228-33.

9. Larocque, G. L.; Maass, O. The mechanism of the alkaline delignification of wood. Can. J. Res. 19B(1):1-16(1941).

10. TAPPI Standard Test Method T 236 os-76. Kappa number of pulp.

11. Kerr, A. J. The kinetics of kraft pulping - progress in the development of a mathematical model. Appita 24(3):180-8(1970).

12. Hatton, J. V. Development of yield prediction equations in kraft pulping. Tappi 56(7):97-100(1973).

13. Blain, T. J. Low-sulfidity pulping with anthraquinone. Tappi 62(6): 53-5(1979).

14. Fleming, B. I.; Bolker, H. I.; Kubes, G. J.; MacLeod, J. M.; Werthemann, D. P. Sulfide as a reducing agent in kraft delignification. Tappi 63(11): 73-7(1980).

15. Abbot, J. Kinetics of alkaline delignification of black spruce wood. Doctoral Dissertation. Montreal, Quebec, Canada, McGill University, 1982.

16. Kubes, G. J.; Fleming, B. I.; MacLeod, J. M.; Bolker, H. I. Viscosities of unbleached alkaline pulps. II. The G factor. J. Wood Chem. Technol. 3(3):313-33(1983).

17. Fleming, B. I.; Kubes, G. J. The viscosities of unbleached alkaline pulps IV. The effect of alkali. J. Wood Chem. Technol. 5(2):217-27(1985).

18. TAPPI Standard Test Method T 230 om-82. Viscosity of pulp (capillary viscometer method).

19. Daleski, E. J. The effect of elevated temperatures on the alkaline pulping processes. Tappi 48(6):325-30(1965).

20. LéMon, S.; Teder, A. Kinetics of the delignification in kraft pulping. I Bulk delignification of pine. Svensk Papperstid. 76(11):407-14(1973).

21. Olm, L.; Tistad, G. Kinetics of the initial stage of kraft pulping. Svensk Papperstid. 82(15):458-64(1979).

22. Kondo, R.; Sarkanen, K. V. Kinetics of lignin and hemicellulose dissolution during the initial stage of alkaline pulping. Holzforschung 38:31-6 (1984).

23. Johnsson, L. Acta Polytechnica Scandinavica. Mathematics and Computing Machinery Series No. 22. Stockholm, Sweden, 1971.

24. Miller, Paul T. A proposal to study the heat of reaction for kraft pulping. A-200 Problem 414A. Appleton, WI, The Institute of Paper Chemistry, 1983.

25. McDonough, T. J.; Van Drunen, V. J. Pulping to low residual lignin contents in the kraft-anthraquinone and kraft processes. Tappi 63(11): 83-7(1980).

26. Pankonin, B. M. A dynamic model of the kraft pulping process. Master's Thesis. Appleton, WI, The Institute of Paper Chemistry, 1979.

27. Teder, A.; Tormund, D. The equilibrium between sulfide and sulfide ions in kraft pulping. Svensk Papperstid. 76(16):607-9(1973).

28. Blythe, D. A. An investigation of sodium sulfide in cellulosic chain cleavage during kraft pulping. Doctoral Dissertation. Appleton, WI, The Institute of Paper Chemistry, 1984.

29. Smith, D. A.; Dimmel, D. R. High temperature proton exchange reactions by hydroxide in water. J. Wood Chem. Technol. 4(1):75-90(1984).

30. Christensen, T. A mathematical model of the kraft pulping process. Doctoral Dissertation. West Lafayette, IN, Purdue University, 1982.

31. Tyler, D. B. Predicting rejects from kraft cooking of overthick chips: a model incorporating caustic diffusion with delignification. Master's Thesis. Moscow, ID, University of Idaho, 1981.

32. Gustafson, R. R. A theoretical model of the kraft pulping process. Doctoral Dissertation. Seattle, WA, University of Washington, 1982.

33. Kleinert, T. N. Mechanisms of alkaline delignification. I. The overall reaction pattern. Tappi 49(2):53-7(1966).

34. Benko, J. The measurement of molecular weight of lignosulfonic acids and related materials by diffusion. I. Tappi 44(11):766-71(1961); II. Tappi 44(11):771-5(1961); III. Tappi 44(12):849-54(1961); IV. Tappi 47(8):508-14(1964).

35. Cussler, E. L. Diffusion-mass transfer in fluid systems. New York, NY, Cambridge University Press, 1984.

36. CRC Handbook of Chemistry and Physics, 65th Edition. Boca Raton, FL, Chemical Rubber Company Press, 1984.

37. Reid, R. C.; Prausnitz, J. M.; Sherwood, T. K. The properties of gases and liquids, 3rd Edition. New York, NY, McGraw-Hill, 1977.

38. Hayduk, W.; Laudie, H. Prediction of diffusion coefficients for non-electrolytes in dilute aqueous solutions. AIChE J. 20(3):611(1974).

39. Beaton, C. F. Viscosity of water and steam. Eng. Sci. Data Item 78040. B.P. Chem. Int. Ltd., London, England, 1978; CA 90:110165y.

40. Perry, J. H. Chemical Engineer's handbook, 4th Edition. New York, NY, McGraw-Hill, 1963.

41. Kobe, K. A.; McCormack, E. J. Viscosity of pulping waste liquors. Ind. Eng. Chem. 41(12):2847-8(1949).

42. Hedlund, A. I. Indunstad Svartluts Viskositet vid Hga Temperaturer. Svensk Papperstid. 54(12):408-11(1951).

43. TAPPI Monograph Series Number 32. Chemical Recovery in Alkaline Pulping Processes, 1968.

44. Lienhard, J. H. A heat transfer textbook. Englewood Cliffs, NJ, Prentice-Hall, 1981.

45. Fleming, B. I.; Kubes, G. J.; MacLeod, J. M.; Bolker, H. I. Polarographic analysis of soda-anthraquinone pulping liquor. Tappi 62(7):55-8(1979).

46. Fullerton, T. J., Forest Research Institute, Rotorua, New Zealand, Personal Communication, 1985.

47. Hill, W. J.; Hunter, W. G.; Wichern, D. W. A joint design criterion for the dual problem of model discrimination and parameter estimation. Technometrics 10(1):145-60(1968).

48. Box, G. E. P.; Hill, W. J. Discrimination among mechanistic models. Technometrics 9(1):57-71(1967).

49. Atkinson, A. C.; Hunter, W. G. The design of experiments for parameter estimation. Technometrics 10(2):271-89(1968).

50. Daniel, C.; Wood, F. S. Fitting equations to data, 2nd Edition. New York NY, Wiley, 1981.

51. Gerald, C. F. Applied numerical analysis. Reading, MA, Addison-Wesley, 1980.

52. Rice, J. R. Numerical methods, software, and analysis. IMSL Reference Edition. New York, NY, McGraw-Hill, 1983.

53. Aurell, R.; Hartler, N. Kraft pulping of pine. Part 1. The changes in the composition of the wood residue during the cooking process. Svensk Papperstid. 68(3):59-68(1965).

54. Akhtaruzzaman, A. F. M. Influence of chip dimensions in kraft pulping; predictive mathematical models. Paperi Puu 62(10):607-14(1980).

55. Saffran, K. E. The development of a simulation-optimization program for the pulp and paper industry. Doctoral Dissertation. Appleton, WI, The Institute of Paper Chemistry, 1984.

56. Echols, R. M. Variation in tracheid length and wood density in geographic races of Scotch pine. Yale University School of Forestry Bulletin Number 64. New Haven, CT, 1958.

57. Effland, M. J. Modified procedure to determine acid-insoluble lignin in wood and pulp. Tappi 60(10):143-4(1977).

58. Borchardt, L. G.; Piper, C. V. A gas chromatographic method for carbohydrates as alditol-acetates. Tappi 53(2):257-60(1970).

APPENDIX I

REACTION KINETICS STUDY DATA

Code	Time, hr	Temp., °K	t to T, min	NaOH, mole/L	NaSH mole/L	AQ, mmole/L	Lignin %ODW	Cel %ODW	GM %ODW	Xylan %ODW
A-1	2.390	150	17	1.002	0.198	1.007	6.19	33.8	6.01	2.98
A-2	2.390	150	17	1.002	0.198	1.005	5.53	35.8	6.82	3.07
A-3	2.390	150	17	1.002	0.198	0.999	5.61	38.9	7.20	3.50
A-4	2.390	150	17	1.002	0.198	1.006	5.93	39.6	7.29	3.68
B-1	1.367	130	16	0.501	0.099	0.502	21.5	38.4	7.16	7.30
B-2	4.228	130	16	2.007	0.099	2.007	14.2	35.4	8.80	5.12
B-3	4.228	130	16	0.511	0.402	0.507	16.2	38.7	7.18	7.00
B-4	1.367	130	16	2.018	0.402	1.989	19.9	37.1	9.80	6.27
C-4	4.299	170	18	2.018	0.402	0.507	0.177	21.0	1.63	0.184
E-1	0.473	130	19	1.002	0.198	1.029	26.2	42.2	9.96	8.03
E-2	23.993	136	20	1.002	0.198	1.013	1.67	35.3	5.94	2.68
E-3	0.473	190	21	1.002	0.198	1.007	1.13	36.9	5.59	1.16
E-4	0.506	170	19	1.002	0.198	0.992	9.86	40.3	7.83	3.75
F-1	0.604	190	22	1.002	0.198	1.013	0.494	34.0	4.90	1.00
F-2	0.477	176	24	1.002	0.198	1.018	5.02	37.5	6.35	2.31
F-3	2.459	130	24	1.002	0.198	0.993	19.7	36.5	9.03	5.30
F-4	24.008	138	27	1.002	0.198	0.999	1.18	36.2	5.96	2.54
G-1	--	--	--	--	--	--	30.0	39.2	16.1	8.95
G-2	--	--	--	--	--	--	30.0	38.8	16.6	8.84
G-1	--	--	--	--	--	--	30.2	38.4	16.8	8.41
G-1	--	--	--	--	--	--	30.1	38.0	17.2	8.51
H-1	1	27	--	1	0.2	1	28.4	40.3	16.4	8.04
H-2	1	27	--	1	0.2	1	28.2	40.7	16.8	8.54
H-3	1	27	--	1	0.2	1	28.2	37.6	16.6	7.06
H-4	1	27	--	1	0.2	1	28.5	39.2	17.6	8.55
I-1	7.767	190	18	0.420	0.099	3.208	0.174	19.7	2.00	0.426
I-2	24.005	130	19	0.128	1.002	0	13.4	34.1	8.00	4.41
I-3	0.628	190	18	0.094	0	6.404	12.4	39.8	4.67	7.15
I-4	3.396	190	18	0.229	0.798	0.101	0.404	28.4	2.62	1.73
J-1	5.824	130	21	1.085	0.662	6.679	7.86	35.8	8.26	4.74
J-2	48.730	130	20	2.798	0	9.780	0.522	31.9	4.66	0.740
J-3	48.730	130	20	3.001	0.846	10.010	0.424	32.2	2.89	0.601
J-4	26.800	130	24	3.001	0.846	10.012	0.304	34.5	3.32	0.711

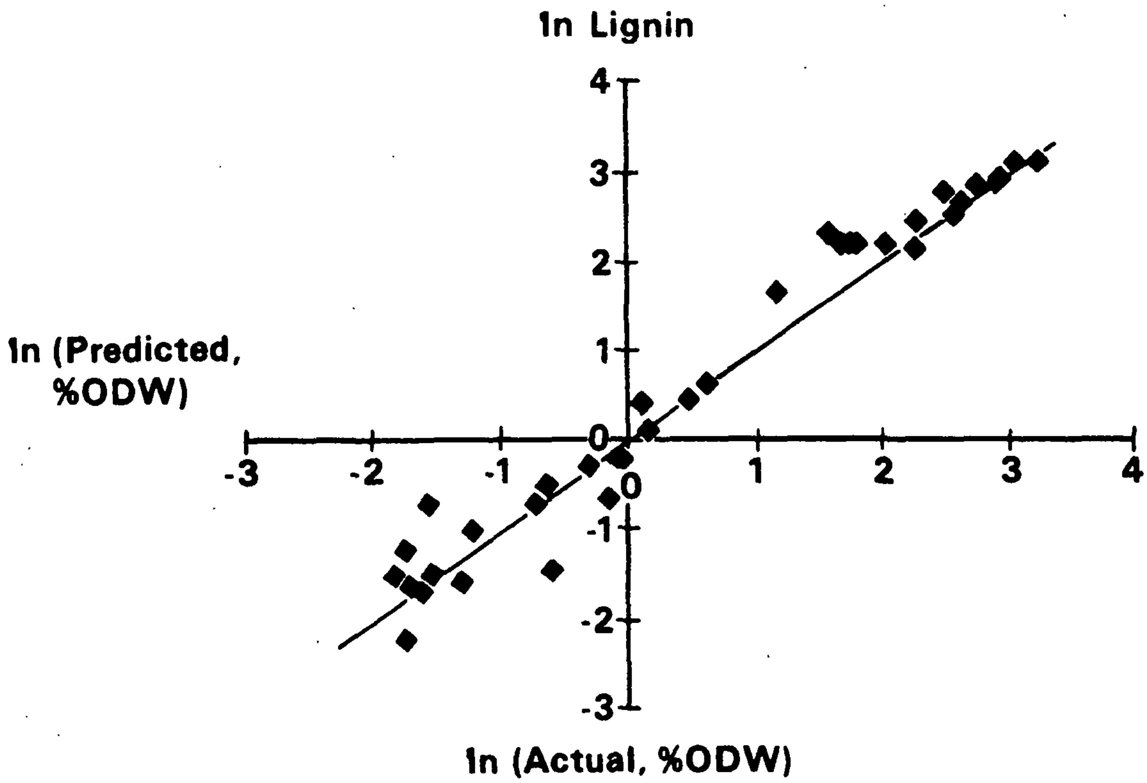
APPENDIX I (Continued)

Code	Time, hr	Temp., °K	t to T, min	NaOH, mole/L	NaSH, mole/L	AQ, mmole/L	Lignin, %ODW	Cel, %ODW	GM, %ODW	Xylan, %ODW
K-1	19.994	180	20	2.482	1.002	9.999	0	0	0	0
K-2	20.018	180	20	2.475	0	0	0	0	0	0
K-3	20.018	180	20	2.482	1.002	0	0	0	0	0
K-4	20.018	180	20	2.475	0	9.996	0	0	0	0
M-1	22.054	190	19	0.505	1.002	10.008	0.028	0.348	0.025	0.0
M-2	22.054	190	19	0.505	1.002	0	0.483	4.26	0.332	0.0
M-3	22.054	190	19	0.498	0	9.999	0.893	8.85	0.685	0.0
M-4	22.054	190	19	0.498	0	0	0.211	6.84	0.461	0.0
N-1	21.180	130	17	0.511	0	9.990	9.79	34.1	9.26	5.3
N-2	3.736	144	19	2.999	0	0	14.2	33.0	7.17	3.2
N-3	5.147	152	17	1.950	0.005	2.627	1.19	34.0	5.10	0.9
N-4	23.985	144	16	2.999	0	0	0.756	22.1	2.16	0.4
O-1	1.307	170	20	0.501	0.099	2.001	3.32	42.0	6.77	3.5
O-2	1.307	170	20	2.007	0.099	0.507	0.975	32.5	3.41	0.4
O-3	3.557	170	20	0.511	0.402	2.001	0.687	34.0	5.29	2.5
P-1	24.011	130	19	0.504	0.997	9.357	1.93	35.8	7.52	4.3
P-2	12.517	162	21	2.376	0.260	10.004	0.162	14.4	0.868	0.0
P-3	16.128	158	20	2.618	0.260	8.560	0.559	12.8	0.700	0.1
P-4	0.979	130	19	2.775	0.926	10.005	19.2	36.8	9.02	5.8
P-5	0.979	130	19	2.562	1.002	9.998	18.7	37.4	9.48	6.1
P-6	0.979	130	19	3.001	0.846	9.994	19.3	37.0	8.92	5.9
P-7	16.128	158	20	2.609	0	0	0.166	10.8	0.489	0.0
P-8	16.128	158	20	2.618	0.260	0	0.163	10.5	0.428	0.0
P-9	16.128	158	20	2.609	0	8.562	0.217	13.1	0.840	0.1
P-10	12.517	162	21	2.381	0	0	0.203	12.3	0.606	0.0
P-11	12.517	162	21	2.376	0.260	0	0.184	14.6	0.794	0.0
P-12	12.517	162	21	2.381	0	9.995	0.274	16.9	0.900	0.0

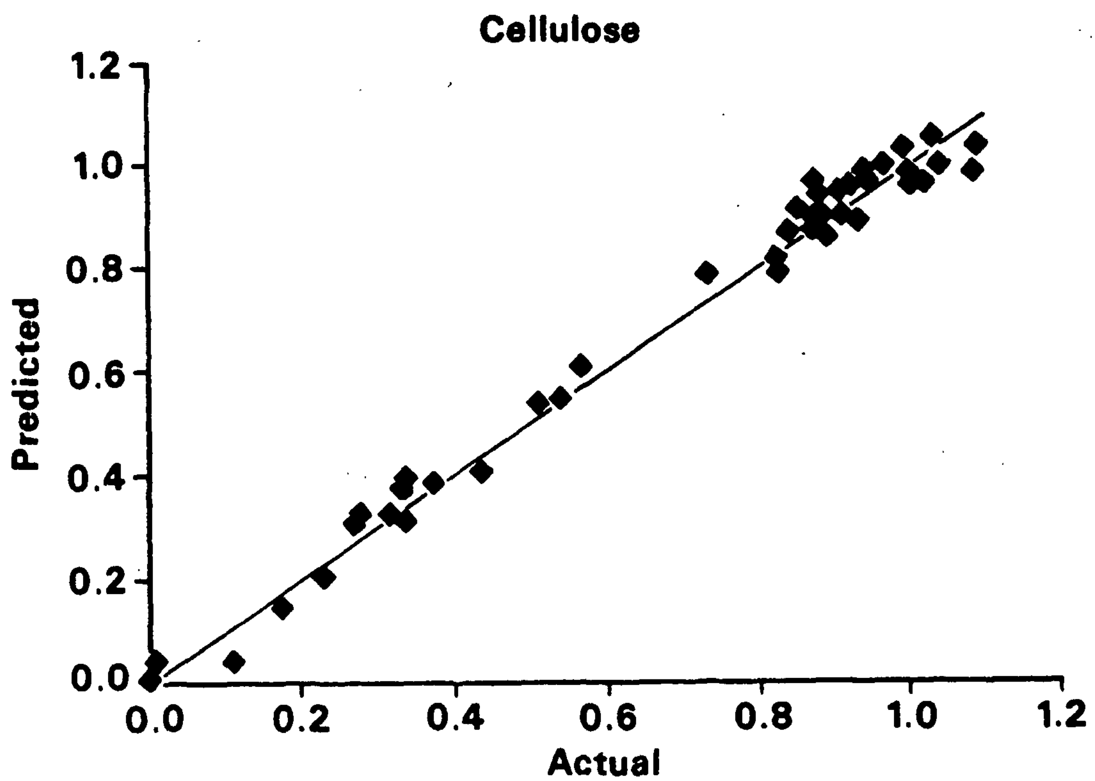
	Time	Temp.	NaOH	NaSH	AQ
Mean	11.806	156.3	1.578	0.336	3.796
Std. dev.	11.729	22.7	0.972	0.362	4.230

Correlation Matrix

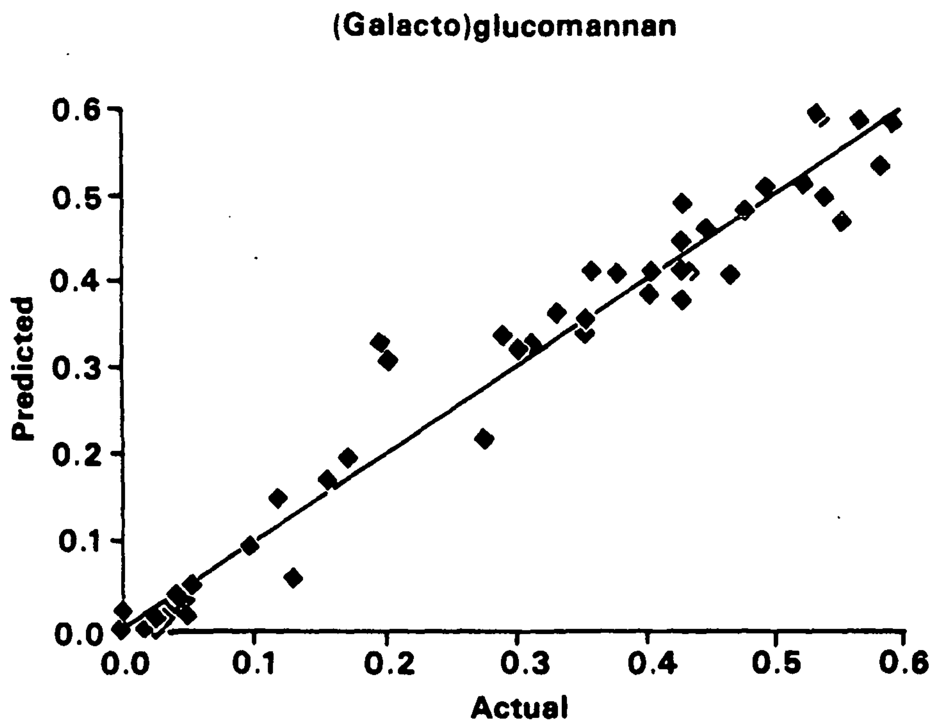
Time	1				
Temp.	-0.089	1			
NaOH	0.263	-0.244	1		
NaSH	0.155	-0.131	0.016	1	
AQ	0.358	-0.151	0.287	0.286	1



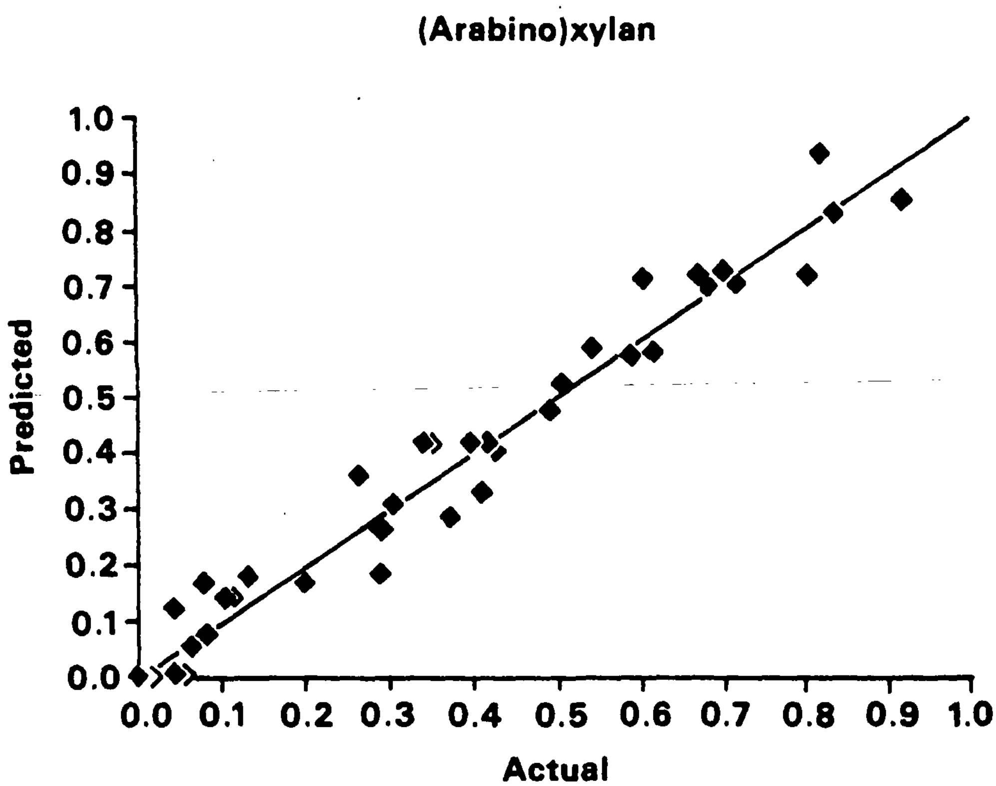
Reaction kinetics study predicted lignin vs. actual.



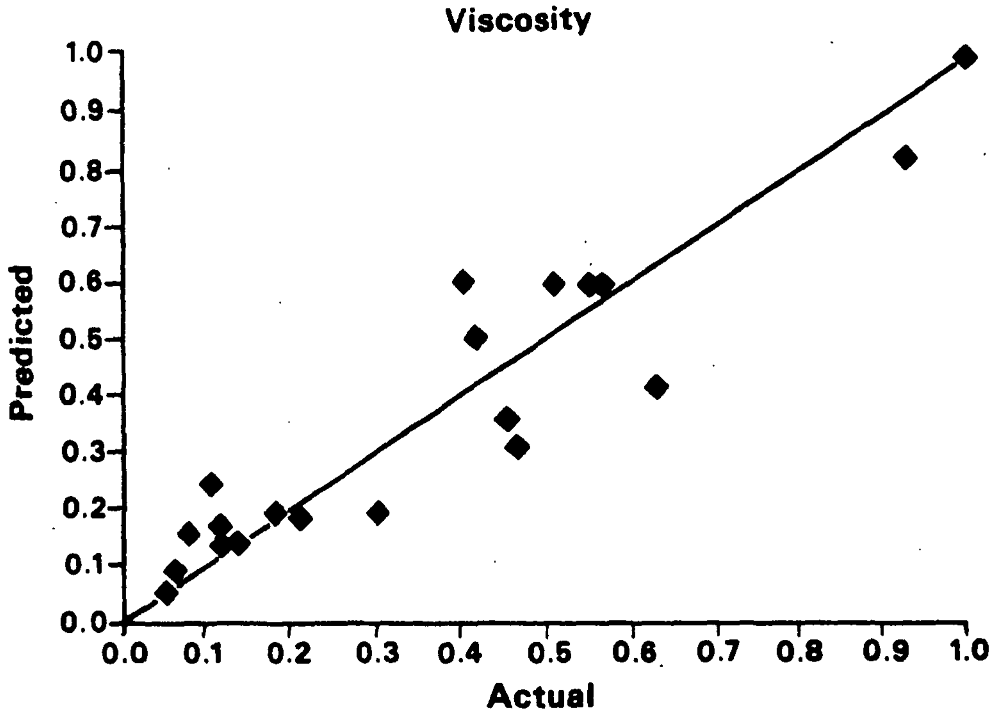
Reaction kinetics study predicted cellulose vs. actual.



Reaction kinetics study predicted glucomannan vs. actual.



Reaction kinetics study predicted xylan vs. actual.



Reaction kinetics study predicted viscosity vs. actual.

APPENDIX II

REACTION KINETICS STUDY MODELS

BEST MODELS

Lignin Model 621n 18 October 85

lignin = N + R,

∂lignin/∂t = $\partial N/\partial t$ + $\partial R/\partial t$ + k_a + k_b for N > 0.87,

∂lignin/∂t = $\partial N/\partial t$ + $\partial R/\partial t$ + k_b for 0.87 > N > 0.76,

∂lignin/∂t = $\partial N/\partial t$ + $\partial R/\partial t$ for 0.76 > N,

$k_a = d\ (N - 0.87)\ e^{(17.33 - 6000/T)}$,

$k_b = d\ N\ e^{(22.12 - 8800/T)}$,

$d = -(1-\{1/(1 + 100\ [NaOH])\})$,

$\partial N/\partial t = -k_{bulk}\ N$,

$\partial R/\partial t = k_{condense}\ D - k_{residual}\ R$, and

$\partial D/\partial t = k_{bulk}\ N - k_{condense}\ D$,

$$k_{bulk} = [NaOH]\ \sqrt{T}\ e^{(k_{NDOH} - AE_{NDOH}\ T^{\dagger})}$$
$$+ \sqrt{[NaOH]}\ \sqrt{[NaSH]}\ \sqrt{T}\ e^{(k_{NDSH} - AE_{NDSH}\ T^{\dagger})}$$
$$+ \sqrt{[NaOH]}\ \sqrt{[AQ]}\ \sqrt{T}\ e^{(k_{NDAQ} - AE_{NDAQ}\ T^{\dagger})},$$

$k_{condense} = \sqrt{T}\ e^{(k_{DR} - 2,500\ T^{\dagger})}$

$k_{residual} = [NaOH]\ \sqrt{T}\ e^{(k_{RD} - AE_{RD}\ T^{\dagger})}$,

where T [=] °K, $T^{\dagger} = 1/T - 1/433$, [NaOH] [=] M, [NaSH] [=] M, and [AQ] [=] mM.

APPENDIX II (Continued)

Name	Coefficient	S.E. Coefficient	t-Value	95% Confidence Limits	
k_{NDOH}	-4.15518	0.175	-23.7	-4.51	-3.80
AE_{NDOH}	19608.7	1830	10.7	1590	23300
k_{NDSH}	-3.24838	0.158	-20.6	-3.57	-2.93
AE_{NDSH}	10821.8	1130	9.6	8530	13100
k_{NDAQ}	-3.83789	0.236	-16.3	-4.31	-3.36
AE_{NDAQ}	15897.1	2170	7.3	11500	20300
k_{DR}	-5.97332	0.352	-17.0	-6.69	-5.26
k_{RD}	-6.00188	0.674	-8.9	-7.37	-4.64
AE_{RD}	9991.03	2010	5.0	5920	14100

Carbohydrates

$C = N + X,$

$\partial H/\partial t = \partial N/\partial t + \partial X/\partial t + k_c$ for $N + X > H_t,$

$\partial H/\partial t = \partial N/\partial t + \partial X/\partial t$ for $H_t > N + X,$

$k_c = d\ (N + X)\ \sqrt{T}\ e^{(9.251\ -\ 4738/T)}$, and

$d = -(1 - \{1/(1 + 100\ [NaOH])\}),$

$\partial N/\partial t = k_{cleave}\ X - (k_{peel} + k_{stop})\ N$, and

$\partial X/\partial t = k_{stop}\ N - k_{cleave}\ X,$

$k_{ND} = [NaOH]\ \sqrt{T}\ e^{(k_{NDOH}\ -\ AE_{NDOH}\ T^{\dagger})}$

$\quad + \sqrt{[NaSH]}\ \sqrt{T}\ e^{(k_{NDSH}\ -\ AE_{NDSH}\ T^{\dagger})},$

$k_{stop} = [NaOH]\ \sqrt{T}\ e^{(k_{NXOH}\ -\ AE_{NXOH}\ T^{\dagger})}$

$\quad + [NaOH]\ \sqrt{[AQ]}\ \sqrt{T}\ e^{(k_{NXAQ}\ -\ AE_{NXAQ}\ T^{\dagger})},$

$k_{cleave} = [NaOH]\ \sqrt{T}\ e^{(k_{XN}\ -\ AE_{XN}\ T^{\dagger})},$

Cellulose Model 30   5 October 85

Name	Coefficient	S.E. Coefficient	t-Value	95% Confidence Limit	
k_{NDOH}	-5.04455	0.0784	-64.4	-5.20	-4.89
AE_{NDOH}	6696.65	439	15.2	5810	7580
H_t	1.04253	0.0113	92.4	1.02	1.06
k_{NDSH}	-6.68305	0.233	-28.7	-7.15	-6.21
AE_{NDSH}	8013.12	512	15.7	6980	9050
k_{NXOH}	-2.32248	0.0885	-26.2	-2.50	-2.14
AE_{NXOH}	9611.41	378	25.4	8850	10400
k_{NXAQ}	-4.62038	0.0102	-451.7	-4.64	-4.60
AE_{NXAQ}	16718.2	402	41.6	15900	17500
k_{XN}	-3.31930	0.0908	-36.6	-3.50	-3.14
AE_{XN}	22844.2	1130	20.2	20600	25100

Glucomannan Model 30   13 October 85

Name	Coefficient	S.E. Coefficient	t-Value	95% Confidence Limits	
k_{NDOH}	-0.857458	0.0809	-10.6	-1.02	-0.694
AE_{NDOH}	6681.58	471	14.2	5730	7630
H_t	0.752114	0.0383	19.7	0.677	0.832
k_{NDSH}	-4.69499	0.343	-13.7	-5.39	-4.00
AE_{NDSH}	12067.8	1890	6.4	8250	15900
k_{NXOH}	-1.51020	0.113	-13.4	-1.74	-1.28
AE_{NXOH}	3125.76	320	9.8	2480	3770
k_{NXAQ}	-2.35942	0.185	-12.7	-2.73	-1.98
AE_{NXAQ}	10770.8	826	13.0	9100	12400
k_{XN}	-4.76589	0.112	-42.4	-4.99	-4.54
AE_{XN}	12636.8	395	32.0	11800	13400

Xylan Model 30 13 October 85

Name	Coefficient	S.E. Coefficient	t-Value	95% Confidence Limits	
k_{NDOH}	-2.30326	0.282	-8.2	-2.87	-1.73
AE_{NDOH}	15595.1	1670	9.3	12200	19000
H_t	0.867626	0.0395	21.9	0.787	0.947
k_{NDSH}	-6.31659	0.190	-33.3	-6.70	-5.93
AE_{NDSH}	2500	--	--	--	--
k_{NXOH}	-2.73932	0.505	-5.4	-3.76	-1.72
AE_{NXOH}	8860.34	2460	3.6	3890	13800
k_{NXAQ}	-6.33924	2.46	-2.6	-11.3	-1.36
AE_{NXAQ}	22304.0	10000	2.2	2070	42500
k_{XN}	-3.74624	0.318	-11.8	-4.39	-3.11
AE_{XN}	12112.0	2220	5.5	7630	16600

Viscosity

$\partial \mu_p / \partial t = -(\mu_p - 1.268)^2\ [\text{NaOH}]\ \sqrt{T}\ e^{(-6.398\ -\ 19{,}110\ T^{\dagger})}$,

where μ_p [=] cp.

Rejected Models

Lignin

Model No.	Description
3	'native' fraction (N) -> 'dissolved' fraction (D) (3 $\|\|$ pathways) $\propto$ [NaOH], N $\sqrt{[\text{NaOH}]}\ \sqrt{[\text{NaSH}]}$, N $\sqrt{[\text{NaOH}]}\ \sqrt{[\text{AQ}]}$; common activation energies (AE's) for all three paths.
4	model 3 with separate AE's.
5	model 4 with all paths $\propto$ N^2 instead of N.
6	N -> D (3 $\|\|$ pathways) $\propto$ $N^a\ [\text{NaOH}]^b$, $N^a\ [\text{NaOH}]^c\ [\text{NaSH}]^d$, Na $[\text{NaOH}]^e$ $[\text{NaOH}]^f$.

Model No.	Description
7	N -> D (3 $\|\|$ pathways) $\propto$ N [NaOH], N $\sqrt{[NaOH]}\sqrt{[NaSH]}$, $\sqrt{[NaOH]}\sqrt{[AQ]}$ N -> 'residual' fraction (R) $\propto$ N $\sqrt{[NaOH]}$; R -> D $\propto$ R [NaOH]; D -> R $\propto$ D.
11	N is in equilibrium with N^- as a f ([NaOH]); N^- -> 'reactive intermediate (Q) $\propto$ N^-; Q -> N^- $\propto$ Q; Q -> D (3 $\|\|$ pathways) $\propto$ Q [NaOH], Q [NaSH], Q [AQ]; Q -> R $\propto$ Q; R -> D $\propto$ R [NaOH]; D -> R D.
12	model 11 with Q -> D $\propto$ Q [NaOH], Q $\sqrt{[NaSH]}$, Q $\sqrt{[AQ]}$.
13	model 11 with D -> R $\propto$ (Q + R) D.
14	model 12 with D -> R $\propto$ (Q + R) D.
15	model 11 without Q -> R reaction.
16	model 12 without Q -> R reaction.
17	model 13 without Q -> R reaction.
18	model 14 without Q -> R reaction.
19	model 13 with Q -> D $\propto$ Q [NaOH], Q {[NaSH]/([NaSH] + a)}, Q {[AQ]/([AQ] + b)}.
20	model 15 with Q -> D $\propto$ Q [NaOH], Q {[NaSH]/([NaSH] + a)}, Q {[AQ]/([AQ] + b)}.
31	model 11 without N^- -> Q and Q -> N^- reactions (Q fixed = N^-).
32	model 31 with Q -> D $\propto$ Q [NaOH], Q $\sqrt{[NaSH]}$, Q $\sqrt{[AQ]}$.
33	model 31 with D -> R $\propto$ (Q + R) D.
34	model 32 with D -> R $\propto$ (Q + R) D.

Model No.	Description
35	model 31 without Q -> R pathway.
36	model 32 without Q -> R pathway.
37	model 33 without Q -> R pathway.
38	model 34 without Q -> R pathway.
42	model 32 with common AE for all three paths of Q -> D reaction; common AE for Q -> R and D -> R reactions.
44	model 34 with common AE for all three paths of Q -> D reaction; common AE for Q -> R and D -> R reactions.
45	model 35 with D -> R AE fixed to 2500.
46	model 36 with D -> R AE fixed to 2500.
47	model 37 with D -> R AE fixed to 2500.
48	model 38 with D -> R AE fixed to 2500.
50	model 46 with R -> unreactive 'dissolved solids' (DS) instead of R -> D.
51	model 36 with R -> DS instead of R -> D.
52	model 50 with R -> DS $\propto \sqrt{R}$.
53	model 46 with Q -> D $\propto Q\sqrt{[NaOH]}$, $Q\sqrt{[NaSH]}$, $Q\sqrt{[AQ]}$; initial phase accounted for; NaOH, NaSH, and AQ consumption accounted for.
54	model 53 with Q -> D $\propto Q\,[NaOH]$, $Q\sqrt{[NaOH]}\sqrt{[NaSH]}$, $Q\sqrt{[NaOH]}\sqrt{[AQ]}$.
55	N -> D (3 $\parallel$ pathways) $\propto N\,[NaOH]$, $N\sqrt{[NaSH]}$, $N\sqrt{[AQ]}$; D -> R $\propto$ D; R -> DS $\sqrt{R}\,[NaOH]$; initial phase accounted for; NaOH, NaSH, and AQ consumption accounted for; D -> R AE fixed to 2500.

Model No.	Description
56	model 54 with $R \rightarrow D \propto \sqrt{R}$; $Q \rightarrow D \propto Q$ [NaOH], $Q\sqrt{[NaSH]}$, $Q\sqrt{[AQ]}$.
57	model 56 with $Q \rightarrow D \propto Q\sqrt{[NaOH]}$, $Q\sqrt{[NaSH]}$, $Q\sqrt{[AQ]}$.
58	model 56 with $Q \rightarrow D \propto Q$ [NaOH], $Q\sqrt{[NaOH]}\sqrt{[NaSH]}$, $Q\sqrt{[NaOH]}\sqrt{[AQ]}$
59	model 58 with $D \rightarrow R \propto D^a$.
60	model 55 with $R \rightarrow DS \propto R$ [NaOH].
61	model 60 with 2 stage initial phase; fitted transition between stages.

Cellulose

Model No.	Description
101	$N \rightarrow D \propto N$ [NaOH].
102	$N \rightarrow D \propto N^2$ [NaOH].
103	$N \rightarrow D$ (2 $\|\|$ pathways) $\propto N$ [NaOH], $N\sqrt{[NaOH]}\sqrt{[NaSH]}$; $D \rightarrow N \propto N\sqrt{[NaOH]}\sqrt{[AQ]}$; all 3 pathways share common AE.
104	model 103 with separate AE's.
105	model 103 with all paths $\propto N^2$ instead of N.
106	$N \rightarrow D$ (2 $\|\|$ pathways) $\propto N$ [NaOH], $N\,[NaSH]^a$; $D \rightarrow N \propto N\,[NaOH]\,[AQ]^b$; transition between initial phase and bulk phase = f ([NaSH], [AQ]).
107	model 101 with transition between initial phase and bulk phase = f ([NaSH], [AQ]).
108	$N \rightarrow D \propto N$ [NaOH]; $N \rightarrow$ 'oxidized' fraction (X) $\propto N$ [NaOH]; $X \rightarrow N \propto X$ [NaOH].

109 model 107 with N -> D $\propto$ [NaOH]^{a};

transition between initial phase and bulk phase = f ([NaSH]^{b}, [AQ]^{c}).

110 N -> D $\propto$ N [NaOH]^{a};

N -> X $\propto$ N [NaOH]^{b};

X -> N $\propto$ N [NaOH]^{c};

transition between initial phase and bulk phase = f ([NaSH]^{d}, [AQ]^{e}).

111 model 108 with N -> X (2 || pathways) $\propto$ N [NaOH], N $\sqrt{[AQ]}$.

112 model 108 with N -> D (2 || pathways) $\propto$ N [NaOH], N $\sqrt{[NaSH]}$;

N -> X (2 || pathways) $\propto$ N [NaOH], N [NaOH] $\sqrt{[AQ]}$.

113 model 112 with N -> X (2 || pathways) $\propto$ N [NaOH], N [AQ].

114 model 108 with N -> D (2 || pathways) $\propto$ N [NaOH], N {[NaSH]/([NaSH] + a)};

N -> X (2 || pathways) $\propto$ N [NaOH], N {[AQ]/([AQ] + b)}.

115 N + 'cleaved' fraction (C) -> D (2 || pathways) $\propto$ (N + C) [NaOH], (N + C) [NaSH];

N + C -> X (2 || pathways) $\propto$ (N + C) [NaOH], (N + C) [NaOH] $\sqrt{[AQ]}$;

N + C -> C $\propto$ (N + C) [NaOH];

X -> N $\propto$ X [NaOH].

116 model 115 with N + C -> D $\propto$ (N + C) [NaOH], (N + C) $\sqrt{[NaSH]}$.

117 model 115 with N + C -> X $\propto$ (N + C) [NaOH], (N + C) $\sqrt{[AQ]}$.

118 model 117 with N + C -> D $\propto$ (N + C) [NaOH], (N + C) $\sqrt{[NaSH]}$.

119 model 112 with N -> D $\propto$ N [NaOH], N [NaSH].

120 model 119 with N -> X $\propto$ N [NaOH], N $\sqrt{[AQ]}$.

124 model 112 with N -> X pathways sharing a common AE.

125 model 124 with N -> X $\propto$ N [NaOH], N $\sqrt{[AQ]}$.

126 model 119 with N -> X pathways sharing a common AE.

127 model 120 with N -> X pathways sharing a common AE.

Glucomannan

Model No.	Description
201	model 101.
202	model 102.
.	.
.	.
.	.
220	model 120.
221	N -> D (2 $\|\|$ pathways) $\propto$ N [NaOH], N $\sqrt{[NaSH]}$.
222	N -> D (2 $\|\|$ pathways) $\propto$ N [NaOH], N [NaSH].
223	model 112 with N -> D $\propto$ N [NaOH].
224	model 124.
225	model 125.
226	model 126.
227	model 127.
228	model 112 with N -> X $\propto$ N [NaOH].
229	model 119 with N -> X $\propto$ N [NaOH].

Xylan

Model No.	Description
301	model 101
302	model 102
.	.
.	.
.	.
320	model 120

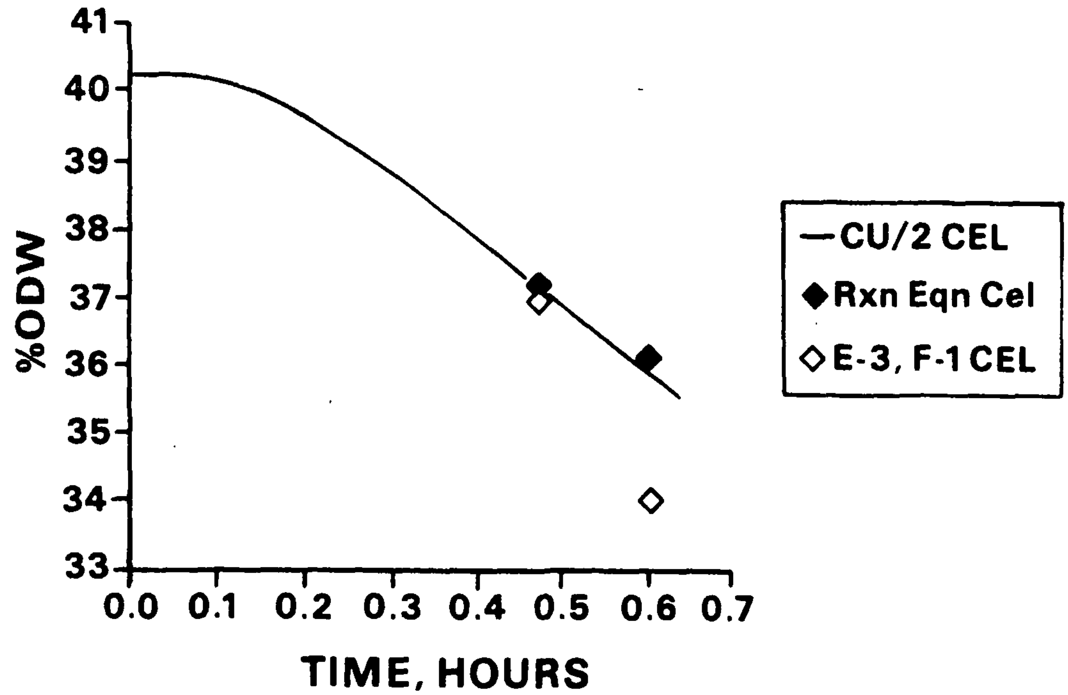
Model No.	Description
323	model 223
324	model 124
325	model 125
326	model 126
327	model 127

Viscosity

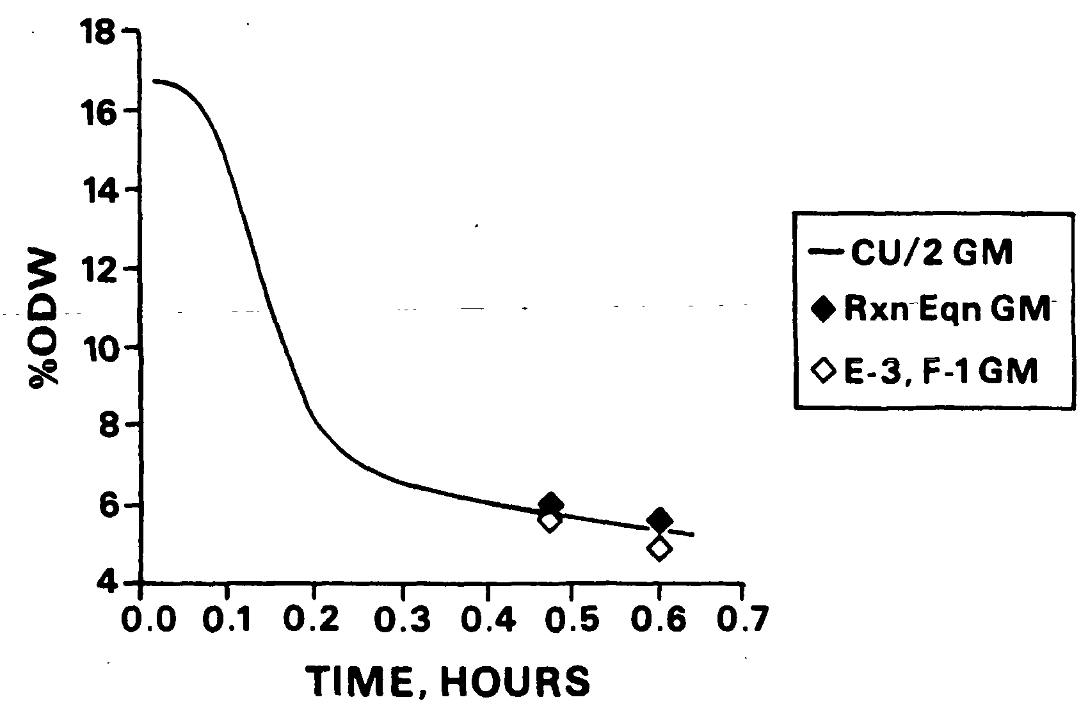
Model No.	Description
411	$\partial\mu_p/\partial t = -\mu_p$ [NaOH]
412	$\partial\mu_p/\partial t = -\mu_p^2$ [NaOH]
413	$\partial\mu_p/\partial t = -\mu_p^3$ [NaOH]
414	$\partial\mu_p/\partial t = -\mu_p^4$ [NaOH]
415	$\partial\mu_p/\partial t = -(\mu_p - 1.268)$ [NaOH]
416	$\partial\mu_p/t = -(\mu_p - 1.268)^2$ [NaOH]
417	$\partial\mu_p/t = -(\mu_p - 1.268)^3$ [NaOH]
418	$\partial\mu_p/t = -(\mu_p - 1.268)^4$ [NaOH]

APPENDIX III

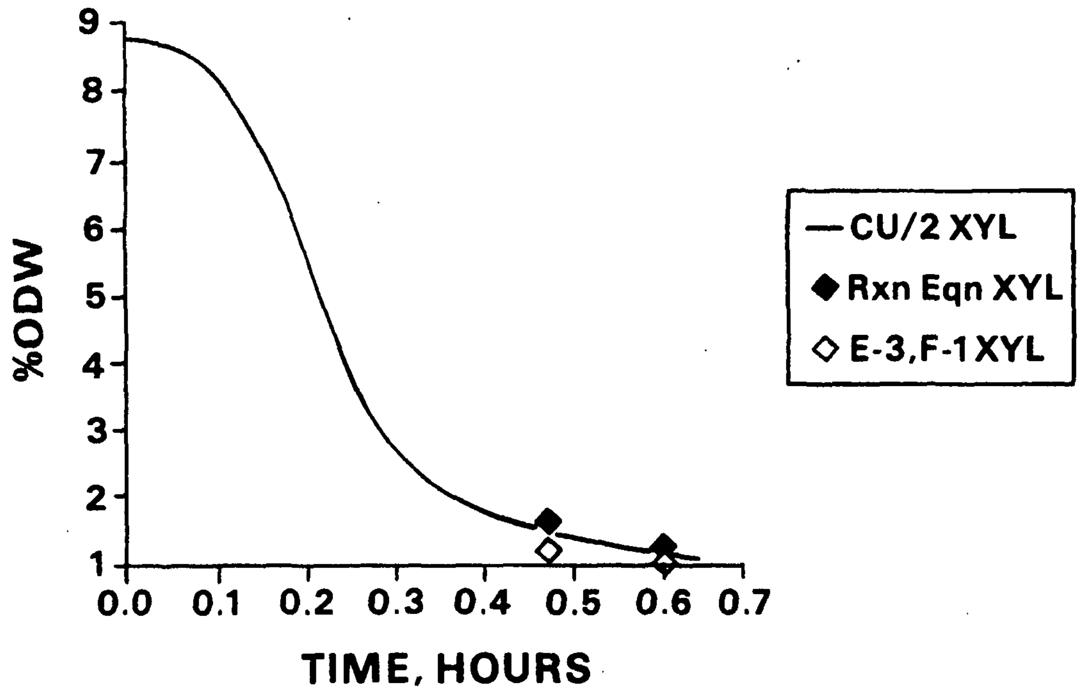
SIMULATION STUDY RESULTS



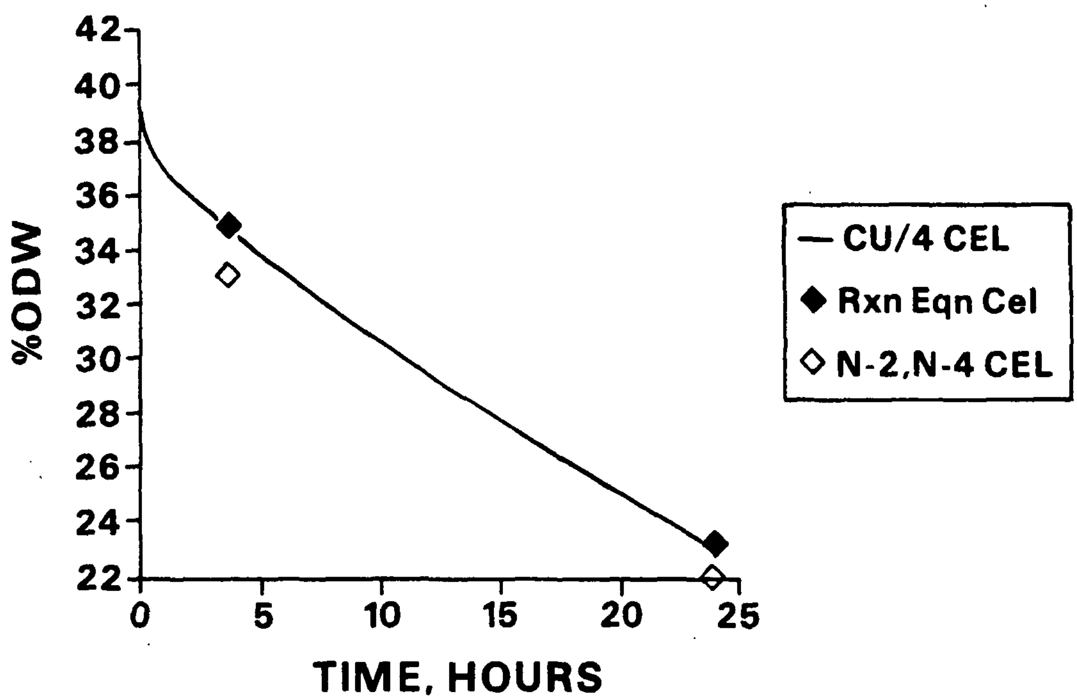
Reaction kinetics study series CU cellulose vs. time.



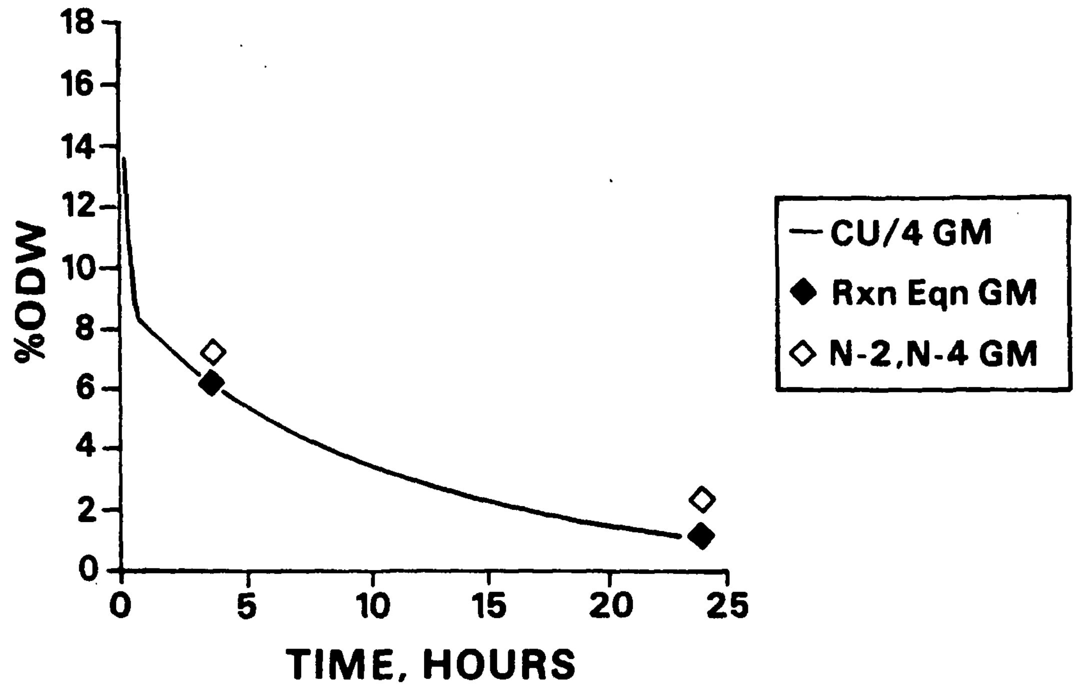
Reaction kinetics study series CU glucomannan vs. time.



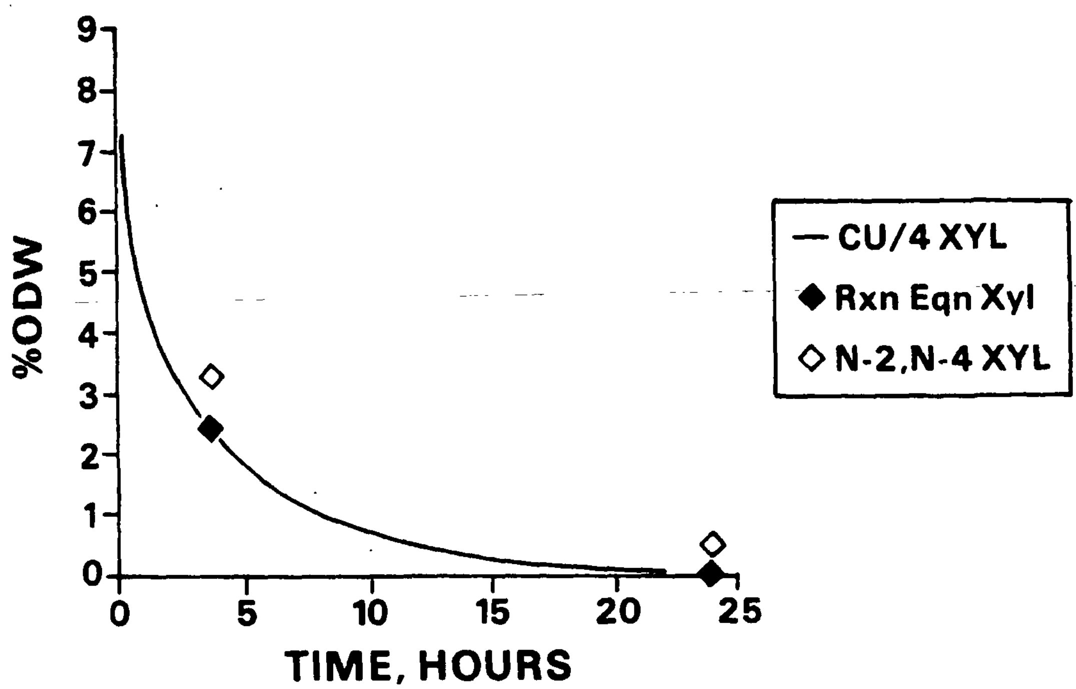
Reaction kinetics study series CU xylan vs. time.



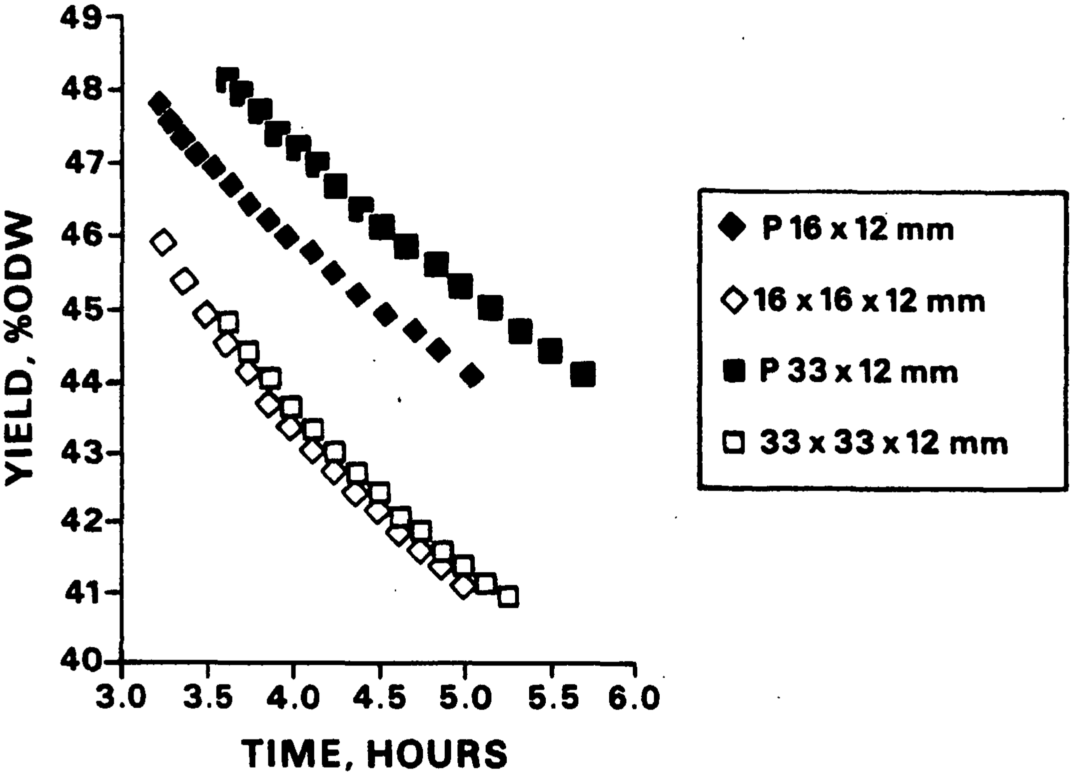
Reaction kinetics study series CU cellulose vs. time.



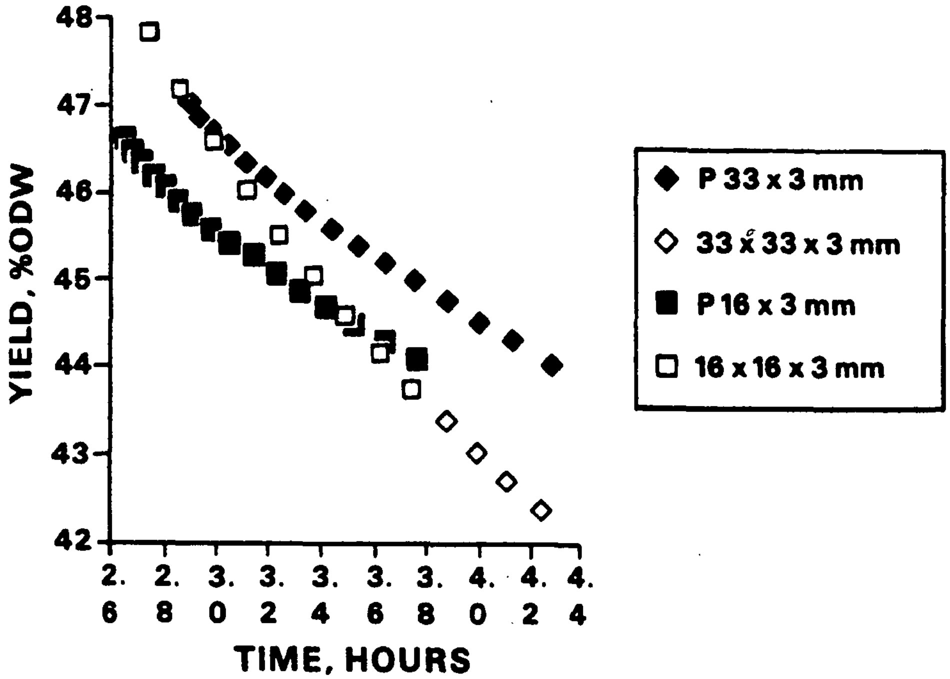
Reaction kinetics study series CU glucomannan vs. time.



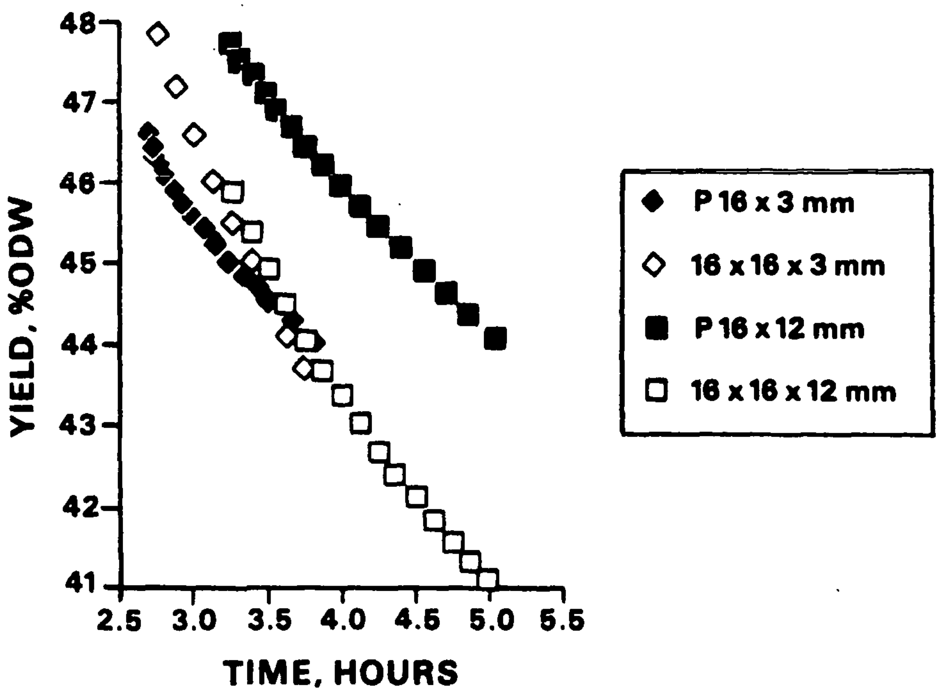
Reaction kinetics study series CU xylan vs. time.



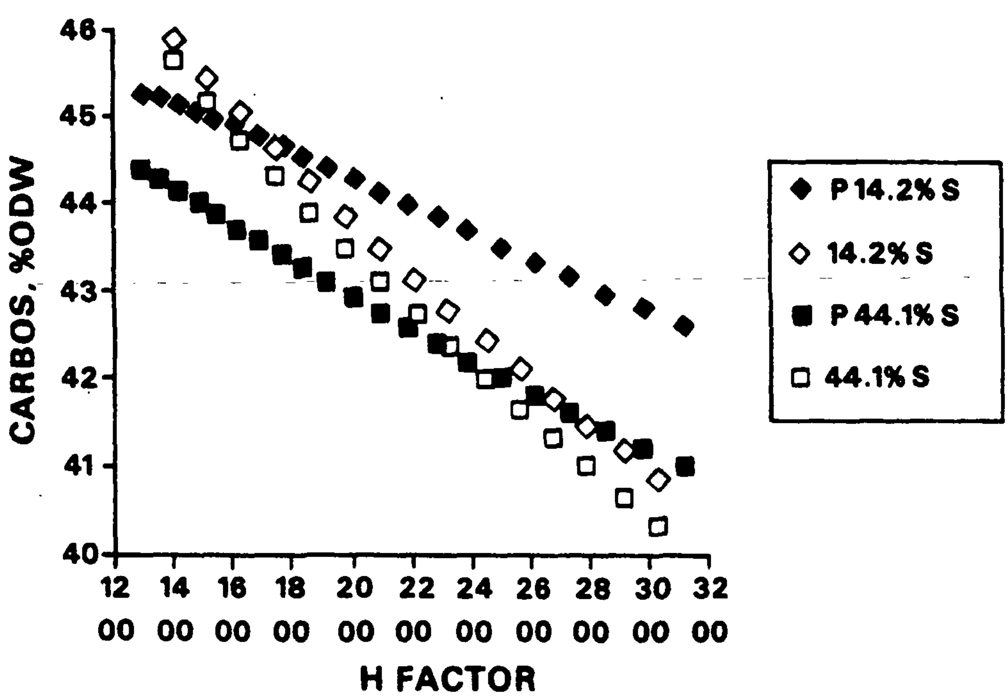
Akhtaruzzaman[54] series CW yield vs. time: effect of chip length at 12 mm thickness.



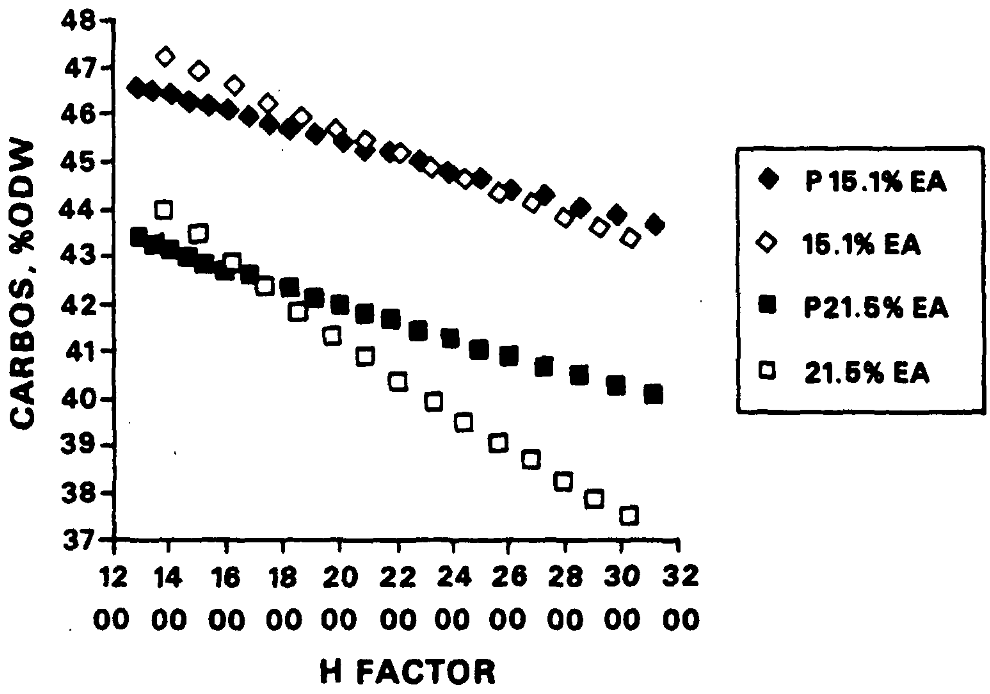
Akhtaruzzaman[54] series CW yield vs. time: effect of chip length at 3 mm thickness.



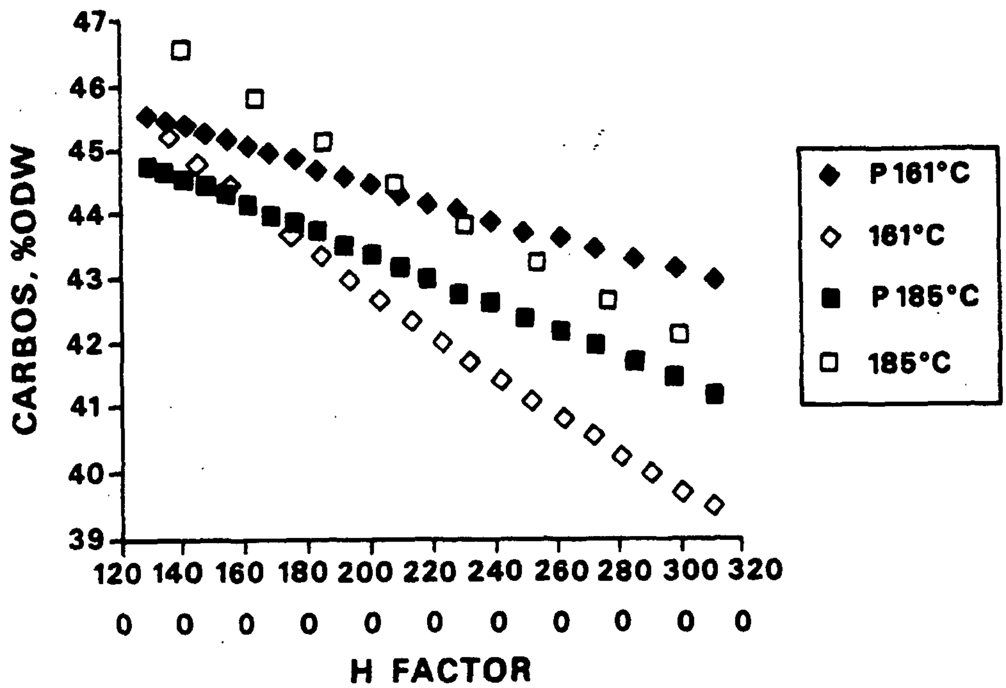
Akhtaruzzaman[54] series CW yield vs. time: effect of chip thickness at 16 mm length.



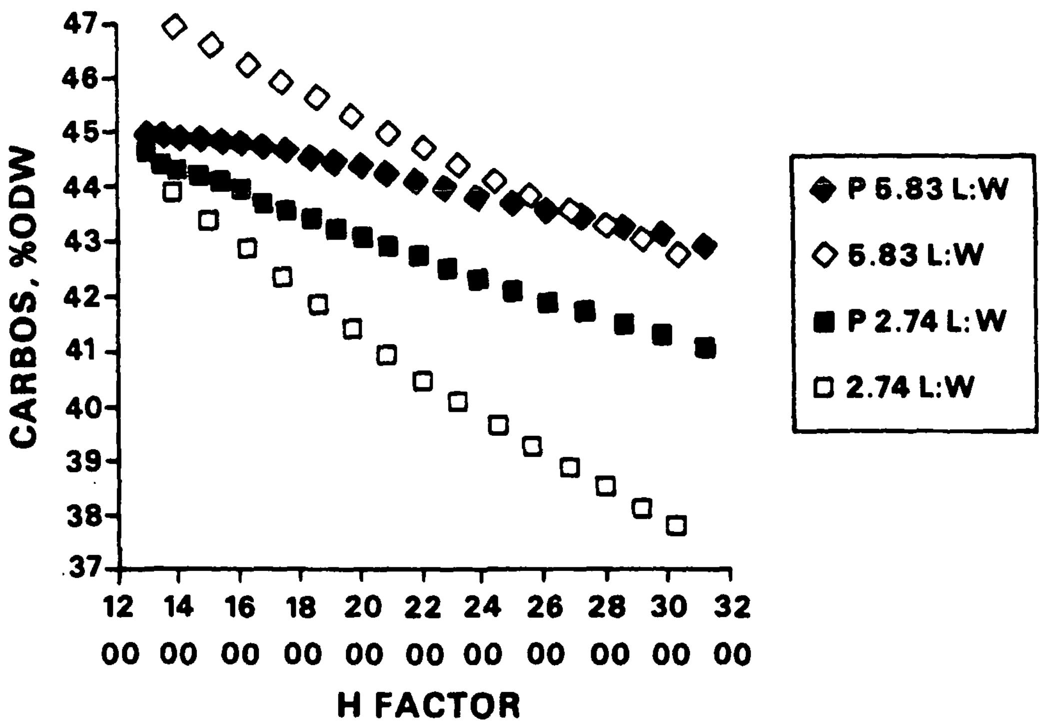
McDonough and Van Drunen[25] series CT carbohydrates vs. time: effect of sulfidity.



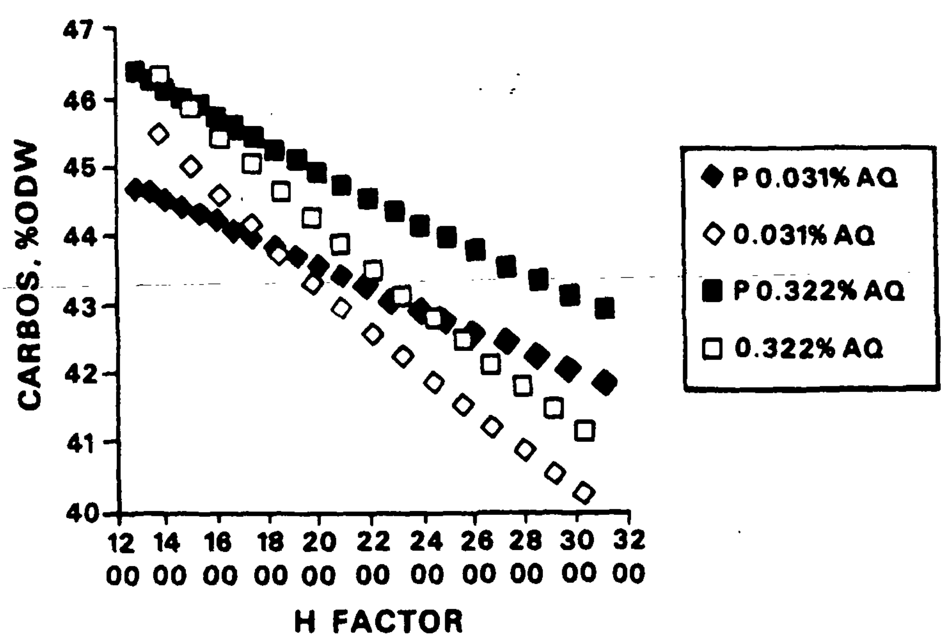
McDonough and Van Drunen[25] series CT carbohydrates vs. time: effect of effective alkali.



McDonough and Van Drunen[25] series CT carbohydrates vs. time: effect of temperature.



McDonough and Van Drunen[25] series CT carbohydrates vs. time: effect of liquor:wood.

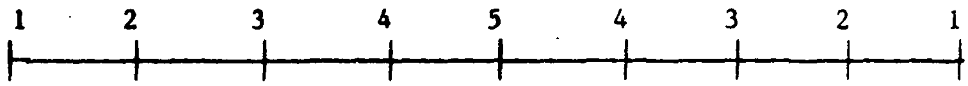


McDonough and Van Drunen[25] series CT carbohydrates vs. time: effect of AQ.

APPENDIX IV

CHIP SYMMETRY AND FINITE DIFFERENCE FORMULAS

The algorithm which generates the finite difference formulas takes full advantage of the symmetry conditions about the chip center-planes (see the assumptions section of results and discussion). The symmetry conditions provide the algorithm with a duplicate set of grid points whose positions and values are known, as shown below.



For simplicity's sake, the grid is shown in one dimension and with constant grid spacing. The boldface points are the actual grid points, the lightface points are the extra points generated by the symmetry condition. As an example, a popular fourth order finite difference formula is

$$\partial^2 u/\partial x^2 \Big|_{x0} \approx 1/12h^2 \, [-u(x_0-2h) + 16u(x_0-h) - 30u(x_0) + 16u(x_0+h) - u(x_0+2h)],$$

where h is grid spacing. If we let h = 1 we get

$$\partial^2 u/\partial x^2 \Big|_{x0} \approx 1/12 \, [-u(x_0-2) + 16u(x_0-1) - 30u(x_0) + 16u(x_0+1) - u(x_0+2)].$$

Evaluating the formula at x_0 = 3, 4, and 5 we get the following coefficients:

(Coefficients x 12)

1	2	3	4	5	4	3	2	1
-1	**16**	**-30***	**16**	**-1**				
	-1	**16**	**-30***	**16**	-1			
		-1	**16**	**-30***	16	-1		

The * signifies the point at which the derivative is evaluated. Taking advantage of symmetry involves recognizing that even though the lightface points are not really there, their function values are known to be exactly equ to the function values of their boldface counterparts. This allows the summation of coefficients of the boldface-lightface pairs giving

(Coefficients x 12)

1	2	3	4	5	4	3	2	1
-1	16	-30*	16	-1				
	-1	16	-31*	16				
		-2	32	-30*				

These formulas are as accurate as the first set of formulas would be if the lightface points actually were there. The second set of formulas allows the model to achieve greater accuracy than would otherwise be possible.

APPENDIX V

EXAMPLE OF THE METHOD OF UNDETERMINED COEFFICIENTS

Given: $\partial^2 f(x_0)/\partial x^2 \approx b_1 f(x_0) + b_2 f(x_1) + b_3 f(x_1)/\partial x + b_4 f(x_2)$

Find: b_1, b_2, b_3, and b_4 such that the above formula is exact for

$$f_0(x) = 1,\ f_1(x) = x,\ f_2(x) = x^2,\ \text{and } f_3(x) = x^3$$

Define $f'(x) = \partial f/\partial x$, $f''(x) = \partial^2 f/\partial x^2$

Let $x_0 = 0$ (the equations would hold for any coordinate system as long as the relative positions of x_0, x_1, and x_2 were maintained; however, by setting $x_0 = 0$ the resulting system of equations is considerably simplified)

$f_0''(x_0) = 0$	$f_0(x_0) = 1$	$f_0(x_1) = 1$	$f_0'(x_1) = 0$	$f_0(x_1) = 1$
$f_1''(x_0) = 0$	$f_1(x_0) = 0$	$f_1(x_1) = x_1$	$f_1'(x_1) = 1$	$f_1(x_2) = 2$
$f_2''(x_0) = 2$	$f_2(x_0) = 0$	$f_2(x_1) = x_1^2$	$f_2'(x_1) = 2x_1$	$f_2(x_2) = x_2^2$
$f_3''(x_0) = 0$	$f_3(x_0) = 0$	$f_3(x_1) = x_1^3$	$f_3'(x_1) = 3x_1^2$	$f_3(x_2) = x_2^3$

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & x_1 & 1 & x_2 \\ 0 & x_1^2 & 2x_1 & x_2^2 \\ 0 & x_1^3 & 3x_1^2 & x_2^3 \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \\ 0 \end{bmatrix}$$

Define $[B] = [b_1\ b_2\ b_3\ b_4]^T$, $[C] = [0\ 0\ 2\ 0]^T$, and $z = x/h$ [in the chip model, h is the shortest distance between two grid points (x's). Normalizing distance by h greatly increases the numerical accuracy of the formula coefficients]

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & hz_1 & 1 & hz_2 \\ 0 & h^2z_1{}^2 & 2hz_1 & h^2z_2{}^2 \\ 0 & h^3z_1{}^3 & 3h^2z_1{}^2 & h^3z_2{}^3 \end{bmatrix} [B] = [C]$$

Define $[D] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & h & 0 & 0 \\ 0 & 0 & h^2 & 0 \\ 0 & 0 & 0 & h^3 \end{bmatrix}$

Define $[E] = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & z_1 & 1 & z_2 \\ 0 & z_1{}^2 & 2z_1 & z_2{}^2 \\ 0 & z_1{}^3 & 3z_1{}^2 & z_2{}^3 \end{bmatrix}$

Define $[F] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/h & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

$$[D]\ [E]\ [F]\ [B] = [C]$$

$$\{[D]\ [E]\ [F]\}^{-1}\ \{[D]\ [E]\ [F]\}\ [B] = \{[D]\ [E]\ [F]\}^{-1}\ [C]$$

$$[B] = [F]^{-1}\ [E]^{-1}\ [D]^{-1}\ [C]$$

$$[B] = [F]^{-1}\ [E]^{-1} \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1/h & 0 & 0 \\ 0 & 0 & 1/h^2 & 0 \\ 0 & 0 & 0 & 1/h^3 \end{bmatrix} [C]$$

$$[B] = [F]^{-1} [E]^{-1} [0 \; 0 \; 2/h^2 \; 0]^T$$

$$[B] = 1/h^2 \, [F]^{-1} [E]^{-1} [0 \; 0 \; 2 \; 0]^T$$

$$[B] = 1/h^2 \, [F]^{-1} [E]^{-1} [C]$$

$$[B] = 1/h^2 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & h & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} [E]^{-1} [C]$$

$$[B] = \begin{bmatrix} 1/h^2 & 0 & 0 & 0 \\ 0 & 1/h^2 & 0 & 0 \\ 0 & 0 & 1/h & 0 \\ 0 & 0 & 0 & 1/h^2 \end{bmatrix} [E]^{-1} [C]$$

In the chip model, the x's are normalized by h, $[E]^{-1}$ [C] is calculated by Gaussian elimination with row and column pivoting, then [B] is calculated using the formula above. The calculation of the finite difference formula coefficients is done in subroutine FINITE in Appendix VII.

APPENDIX VI

MDPE (MODEL DISCRIMINATION/PARAMETER ESTIMATION)

```
#FILE (MARK)MDPE ON STUDENTS
$      SET $
$    RESET FREE
FILE  1(KIND = DISK, FILETYPE = 7, MYUSE = IN, TITLE = "MDPE/DATA")
FILE  2(KIND = DISK, FILETYPE = 7, MYUSE = IN, TITLE = "MDPE/TRIALS")
FILE  3(KIND = DISK, FILETYPE = 7, MYUSE = IN, TITLE = "MDPE/CHOOSE")
FILE  5(KIND = REMOTE, MYUSE = IO)
FILE  6(KIND = DISK, TITLE = "ERRORS", NEWFILE = TRUE, FILETYPE = 7,
    *  BLOCKSIZE = 420, MAXRECSIZE = 12, MYUSE = OUT, FLEXIBLE)
CFILE  7(KIND = PRINTER, TRAINID = EBCDIC96)
FILE  7(KIND = DISK, TITLE = "RESULTS", NEWFILE = TRUE, FILETYPE = 7,
    *  BLOCKSIZE = 420, MAXRECSIZE = 12, MYUSE = OUT, FLEXIBLE)
FILE 11(KIND = DISK, FILETYPE = 7, TITLE = "NONLINWOOD/PKW.")
$      LIMIT = 3
$      OPT = 0
$    RESET DBLTOSNGL
$      SET ERRLIST
$      SET LINEINFO
$      SET LIST
$    RESET LONG
$      SET OMITDEBUG
$      SET OWN
$      SET OWNARRAYS
$      SET TIME
$    RESET XREF

$      INCLUDE "IMSL/DBLE/USPKD"
$      INCLUDE "IMSL/DBLE/UGETIO"
$      INCLUDE "IMSL/DBLE/UERSET"
$      INCLUDE "IMSL/DBLE/UERTST"
$      INCLUDE "IMSL/DBLE/VIPRFF"
$      INCLUDE "IMSL/DBLE/LUDATN"
$      INCLUDE "IMSL/DBLE/LUELMN"
$      INCLUDE "IMSL/DBLE/VTPROF"
$      INCLUDE "IMSL/DBLE/VCVTSF"
$      INCLUDE "IMSL/DBLE/LINV3F"
$      INCLUDE "IMSL/DBLE/ZXMJN"
$      INCLUDE "UTIL/DBLE/SKIP"
$      INCLUDE "IMSL/DBLE/ZXMWE"
$      INCLUDE "UTIL/DBLE/SIMPLX"
$      INCLUDE "IMSL/DBLE/ZSRCH"
$      INCLUDE "IMSL/DBLE/ZXMIN"
$      INCLUDE "UTIL/DBLE/CONST"
$      INCLUDE "UTIL/DBLE/SKIPIN"
$      INCLUDE "UTIL/DBLE/SKIPOT"
$      INCLUDE "IMSL/DBLE/ZXMWD"
$      INCLUDE "UTIL/DBLE/ZXSIMP"
$      INCLUDE "UTIL/DBLE/ZXMING"
$      INCLUDE "MDPE/FW"
$      INCLUDE "UTIL/DBLE/FPENAL"
$      INCLUDE "UTIL/DBLE/TIMER"
$      INCLUDE "UTIL/DBLE/ROUND"
$      INCLUDE "UTIL/DBLE/EOFILE"
```

```
$       OPT = 0
$       SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$       SET OWNARRAYS % Always set

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE INITOP
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

$       INCLUDE 'MDPE/COMMON'

      DO 100 JW = 1, MW
         DO 100 JM = 1, MM
            DO 100 JV = 1, 5
               AOPT(JV, JM, JW) = .TRUE.
  100 CONTINUE

      DO 200 JW = 1, MW
         DO 200 JM = 1, MM
            DO 200 JV = 6, MV
               AOPT(JV, JM, JW) = .FALSE.
  200 CONTINUE

      DO 300 JV = 1, 5
         AOPT(JV,  1, 1) = .FALSE.
         AOPT(JV,  2, 1) = .FALSE.
         AOPT(JV,  8, 1) = .FALSE.
         AOPT(JV,  9, 1) = .FALSE.
         AOPT(JV, 10, 1) = .FALSE.
  300 CONTINUE

      DO 400 JW = 2, 4
         AOPT(4,  1, JW) = .FALSE.
         AOPT(5,  1, JW) = .FALSE.
         AOPT(4,  2, JW) = .FALSE.
         AOPT(5,  2, JW) = .FALSE.
         AOPT(4, 11, JW) = .FALSE.
  400 CONTINUE

      AOPT(5, 3, 3) = .FALSE.
      AOPT(5, 5, 3) = .FALSE.

      DO 500 JW = 5, MW
         DO 500 JM = 1, MM
            DO 500 JV = 1, MV
               AOPT(JV, JM, 5) = .FALSE.
  500 CONTINUE

      RETURN
      END

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION CNVERT(
     &  UNVERT, HIGH  , LOW   , NSIG  )
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT
     &  LOGICAL(A - Z)

      INTEGER
     &  NSIG

      DOUBLE PRECISION
     &  UNVERT, HIGH  , LOW
     &, ROUND                                           % 23 Jan 85 %%

C---------------------------------------------------------------------

      CNVERT = ROUND((HIGH - LOW) * UNVERT + LOW, NSIG)

      RETURN
      END      % CNVERT

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      BLOCK DATA
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT
     &  LOGICAL(A - Z)

      LOGICAL
     &  OLD1  , OLD2

      COMMON
     & /CE    / OLD1
     & /CFW   / OLD2

      DATA
     &  OLD1  /.FALSE./
     &, OLD2  /.FALSE./

      END      % BLOCK DATA

$     OPT = 0
$     SET OWN       % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE DFDBX
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----Partial derivatives of FW with respect to parameter estimates----
C     at constant JM, JR, & JW
C-----Creation date:  8 Aug 84----Last update: 14 Feb 85--------------

      IMPLICIT
     &  LOGICAL(A - Z)
```

```
      DOUBLE PRECISION
     &  DUMMY  , F      , HUSE
     &, FW                                                    % 23 Jan 85 %%

$     INCLUDE "MDPE/COMMON"

C-----------------------------------------------------------------------

      DO 100 JP = 1, NP(JM)
         BP(JP) = AP(JP, JM)
         BWK1(JP) = BP(JP)
  100 CONTINUE

      DO 200 JV = 1, NV
         BV(JV) = AV(JV, JR)
  200 CONTINUE

      F = FW(DUMMY)

      HUSE = DABS(BP(1)) * H
      HUSE = DMAX1(H, HUSE)
      BP(1) = BP(1) + HUSE
      AX(JR, 1) = (FW(DUMMY) - F) / HUSE

      DO 300 JP = 2, NP(JM)
         HUSE = DABS(BP(JP)) * H
         HUSE = DMAX1(H, HUSE)
         BP(JP-1) = BWK1(JP-1)
         BP(JP) = BP(JP) + HUSE
         AX(JR, JP) = (FW(DUMMY) - F) / HUSE
  300 CONTINUE

      END       % DFDBX

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FSS
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates BSS, variance of---------------------
C     model JM about FW
C-----Creation date: 09 Jan 84----Last update:  9 Oct 84--------------

      IMPLICIT
     &  LOGICAL (A - Z)

      DOUBLE PRECISION
     &  DUMMY

      DOUBLE PRECISION
     &  SUM    , Y      , YJ
```

```
     &, FW                                                     % 23 Jan 85 %%

$      INCLUDE "MDPE/COMMON"

 9000 FORMAT(/,' enter FSS',/)
 9010 FORMAT(/,' variances [0 & JM]',/)
 9020 FORMAT(/,' parameter vector BP [Parameters]',/)
 9030 FORMAT(/,' variable  vector BV [Parameters]',/)
 9099 FORMAT(/,' exit  FSS',/)

C----------------------------------------------------------------------

      IF(DEBUGG .GE. 1)
     &  WRITE(PRINTR, 9000)

      IF(DEBUGG .LT. 10) GO TO 1
        CALL SKIPOT(PRINTR, 1)
        WRITE(PRINTR, *//) DEBUGG, JM, JW, NM, NR, NV, NW, SW
        CALL SKIPOT(PRINTR, 1)
    1 CONTINUE

C-----BP(JP), the parameter vector for FW

      DO 10 JP = 1, NP(JM)
         BP(JP) = AP(JP, JM)
   10 CONTINUE

      IF(DEBUGG .LT. 100) GO TO 2
        WRITE(PRINTR, 9020)
        WRITE(PRINTR,    /) (BP(JP), JP = 1, NP(JM))
    2 CONTINUE

C-----BSSM(JM), variance of model JM about experimental data AY

      SUM = 0D0
      DO 100 JR = 1, NR

C--------BV(JV), variable list for FW

         DO 110 JV = 1, NV
            BV(JV) = AV(JV, JR)
  110    CONTINUE

C--------Calculate variance as sum of squares deviation
C--------of model JM from runs 1 through NR

         Y   = AY(JW, JR)
         YJ  = FW(DUMMY)
         SUM = (Y - YJ) * (Y - YJ) + SUM

         IF(DEBUGG .LT. 1000) GO TO 3
           WRITE(PRINTR, 9030)
           WRITE(PRINTR,    /) (BV(JV), JV = 1, NV)
    3    CONTINUE
```

```
         IF(DEBUGG .LT. 100) GO TO 4
           CALL SKIPOT(PRINTR, 1)
           WRITE(PRINTR,  *//) JR, JW, Y, YJ, SUM
           CALL SKIPOT(PRINTR, 1)
    4    CONTINUE

  100 CONTINUE

      BSSM(JM) = SUM / NRM1

      IF(DEBUGG .LT. 10) GO TO 5
        WRITE(PRINTR, 9010)
        WRITE(PRINTR,     /) BSSW(JW), BSSM(JM)
    5 CONTINUE

      IF(DEBUGG .GE. 1)
     &  WRITE(PRINTR, 9099)

      END      % FSS

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FPROB
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates current probabilities----------------
C     Pi,n,i=1,m from prior probabilities Pi,n-1
C     Pj,k => Pr(model j correct based on data from runs 1-k)
C-----Creation date: 16 Dec 83----Last update: 26 Oct 84--------------

      IMPLICIT
     &  LOGICAL (A - Z)

      INTEGER
     &  J      , JRM1

      DOUBLE PRECISION
     &  DWARF , FW     , SUM    , V      , VJ     , Y      , YJ
     &, DUMMY

$     INCLUDE "MDPE/COMMON"

 9000 FORMAT(' ')
 9010 FORMAT(/' FPROB debug summary',50('-')/)
 9020 FORMAT(/' BPR [Models]'/)
 9030 FORMAT(/' BSSM [Models]'/)
 9040 FORMAT(/' BV [Variables]'/)
 9050 FORMAT(/' APR [Models x run ',I3,']'/)
 9060 FORMAT(/' BP [Parameters]'/)
 9110 FORMAT(/5X, ' loop 20 summary ***'/)
```

```
C-----------------------------------------------------------------------

      DWARF  = DSQRT(SMALL)
      JRM1   = JR - 1
      SUM    = 0D0
      V      = BSSW(SW)
      Y      = AY(SW, JR)

      DO 10 JV = 1, NV
   10    BV(JV) = AV(JV, JR)

      DO 20 JM = 1, NM

         DO 30 JP = 1, NP(JM)
   30       BP(JP) = AP(JP, JM)

         VJ = BSSM(JM)
         YJ     = FW(
     &     DUMMY )

         BPR(JM) = 1D0 / DSQRT(2D0 * PI * (V + VJ))
     &           * DEXP((-5D-1 / (V + VJ)) * (Y - YJ) ** 2)

         IF(BPR(JM) .LT. DWARF) BPR(JM) = DWARF

         SUM = APR(JM, JRM1) * BPR(JM) + SUM

C--------Borrow BY from FD for temporary storage of predicted Y's

         BY(JM) = YJ

         IF(DEBUGG .LT. 1000) GO TO 20
           WRITE(PRINTR, 9110)
           WRITE(PRINTR, *//) JM, V, VJ, Y, YJ, SUM
           WRITE(PRINTR, *//) 'BPR(JM)', BPR(JM)
           WRITE(PRINTR, *//) 'APR(JM, JRM1)', APR(JM, JRM1)

   20 CONTINUE

      IF(SUM .LT. DWARF) SUM = DWARF

      DO 40 JM = 1, NM
         APR(JM, JR) = APR(JM, JRM1) * BPR(JM) / SUM
   40    IF(APR(JM, JR) .LT. DWARF) APR(JM, JR) = DWARF

      IF(DEBUGG .LT. 1) GO TO 99
        WRITE(PRINTR, 9010)
        WRITE(PRINTR, *//) JR, JRM1, NM, NR, NV, NW, PI, SUM, SW, V
     &, Y

        IF(DEBUGG .LT. 10) GO TO 99
          WRITE(PRINTR, 9020)
          WRITE(PRINTR, /) (BPR(J), J = 1, NM)
```

```
      WRITE(PRINTR, 9030)
      WRITE(PRINTR, /) (BSSM(J), J = 1, NM)
      WRITE(PRINTR, 9040)
      WRITE(PRINTR, /) (BV(JV), JV = 1, NV)
      WRITE(PRINTR, 9050) JR
      WRITE(PRINTR, /) (APR(J, JR), J = 1, NM)
      WRITE(PRINTR, 9060)
      WRITE(PRINTR, /) (BP(JP), JP = 1, NP(JM))

   99 RETURN

      END      % FPROB

$     OPT = 0
$     SET OWN       % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION FDRVTE(
     &  X     )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This function calculates FDRVTE, the predicted response,---------
C     given parameters BP(*) for model JM, response JW
C     and varible settings BV(*) <BP(JP) is replaced by X>
C-----Creation date: 28 Dec 83----Last update:  9 Oct 83--------------

      IMPLICIT
     &  LOGICAL (A - Z)

      DOUBLE PRECISION
     &  DUMMY , FW     , X

      INTEGER
     &  J

$     INCLUDE "MDPE/COMMON"

C-----------------------------------------------------------------------

      DO 100 J = 1, NP(JM)
         BP(J) = AP(J, JM)
  100 CONTINUE

      BP(JP) = X

      DO 200 J = 1, NV
         BV(J) = AV(J, JR)
  200 CONTINUE

      FDRVTE = FW(
     &    DUMMY )

      END      % FDRVTE
```

```
$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FD    (
     &  D      )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates D, the criterion---------------------
C     for model discrimination
C-----Creation date: 16 Dec 83----Last update:  9 Oct 84--------------

      IMPLICIT
     &  LOGICAL (A - Z)

      DOUBLE PRECISION
     &  D
     &, FW     , PJ     , PK     , V      , VJ     , VK     , YJ     , YK
     &, DUMMY  , SUM

      INTEGER
     &  J      , K

$     INCLUDE "MDPE/COMMON"

C---------------------------------------------------------------------

      SUM      = ODO
      V        = BSSW(SW)

      DO 100 JV = 1, NV
         BV(JV) = AV(JV, NRP1)
  100 CONTINUE

      DO 200 JM = 1, NM
         DO 210 JP = 1, NP(JM)
            BP(JP) = AP(JP, JM)
  210    CONTINUE

         BY(JM) = FW(DUMMY)

  200 CONTINUE

      DO 300 J = 1, NM - 1
         PJ = APR(J, NR)
         VJ = BSSM(J)
         YJ = BY(J)
         DO 300 K = J + 1, NM
            PK = APR(K, NR)
            VK = BSSM(K)
            YK = BY(K)
```

```
           SUM    =
     &       ((VJ - VK) ** 2 / ((V + VJ) * (V + VK)) +
     &         (YJ - YK) ** 2 * (1D0 / (V + VJ) + 1D0 / (V + VK)))
     &      * PJ * PK
     &      + SUM

  300 CONTINUE

      D       = SUM * 5D-1

      END      % FD

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FE    (
     &  E      )
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates E, the parameter----------------------
C     estimation criterion for model JM given AV and FW
C-----Creation date: 18 Dec 83----Last update: 26 Oct 84---------------

      IMPLICIT
     &  LOGICAL (A - Z)

      DOUBLE PRECISION
     &  D1     , D2     , E      , SUM    , X

      INTEGER
     &  IER

      LOGICAL
     &  OLD

      COMMON
     & /CE    / OLD

$     INCLUDE "MDPE/COMMON"

C-----AX(JR,JP) = dFW(AP, AV(JR))/dAP(JP)------------------------------

      IF(BOLD(JM)) GO TO 1
         DO 100 JR = 1, NR
            CALL DFDBX                                   % time>0 %%%%%
  100    CONTINUE
         DO 200 JP = 1, NP(JM)
            DO 200 JR = 1, NR
               AAX(JR, JP, JM) = AX(JR, JP)
  200    CONTINUE
         BOLD(JM) = .TRUE.
         OLD = .TRUE.
```

```
    1 CONTINUE

      IF(OLD) GO TO 2
         DO 300 JP = 1, NP(JM)
            DO 300 JR = 1, NR
               AX(JR, JP) = AAX(JR, JP, JM)
  300    CONTINUE
         OLD = .TRUE.
    2 CONTINUE

      JR = NRP1
      CALL DFDBX

C-----Transpose product AX'AX

      CALL VTPROF(
     &  AX     , NRP1   , NP(JM), MR     , BXTX   )

C-----Convert from symmetric storage to full storage

      CALL VCVTSF(
     &  BXTX   , NP(JM), AXTX   , MP     )

C-----Determinant of AX'AX [E]

      CALL LINV3F(
     &  AXTX   , BWK1   , 4      , NP(JM), MP     , D1     , D2     , BWK2
     &, IER    )

      E = D1 * 2D0 ** D2

      IF(DEBUGG .LT. 100 .OR. MOD(JFEVAL, 101) .NE. 0) GO TO 9
         IF(FCLOSE) CALL EOFILE(PRINTR)
         CALL SKIPOT(PRINTR, 1)
         WRITE(PRINTR, /) JFEVAL
         WRITE(PRINTR, /) 'AP', (AP(JP, JM), JP = 1, NP(JM))
         WRITE(PRINTR, /) 'AV', (AV(JV, JR), JV = 1, NV)
         WRITE(PRINTR, /) 'AX', (AX(JR, JP), JP = 1, NP(JM))
         CALL SKIPOT(PRINTR, 1)
         IF(FCLOSE) LOCK PRINTR
    9 CONTINUE

      END      % FE

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FC    (
     &  C      )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates C, the combined----------------------
```

```
C     criterion for simultaneous model discrimination
C     and parameter estimation
C-----Creation date: 16 Dec 83----Last update: 28 Oct 84---------------

      IMPLICIT
     &  LOGICAL (A - Z)

      DOUBLE PRECISION
     &  C
     &, DD     , DWARF , EE     , PRBEST, W1     , W2

      LOGICAL
     &  OLD

      COMMON
     & /CE     / OLD

$     INCLUDE "MDPE/COMMON"

C----------------------------------------------------------------------

      CALL FD(DD    )

      DD     = DD / DMAX

      EE     = 0D0
      PRBEST = 0D0

      DO 100 JM = 1, NM
         OLD = .FALSE.

         CALL FE(BE(JM))

         BEMAX(JM) = DMAX1(BEMAX(JM), SMALL)                         %%%%%

         EE     = APR(JM, NR) * BE(JM) / BEMAX(JM) + EE
         PRBEST = DMAX1(APR(JM, NR), PRBEST)
  100 CONTINUE

      W1     = (DFLOAT(NM) * (1D0 - PRBEST) / DFLOAT(NM-1)) ** LAMBDA
      W2     = 1D0 - W1

      C      = W1 * DD + W2 * EE

      END      % FC

$     OPT = 0
$     SET OWN       % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FCNVRT(N     , X     )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
C-----This subroutine converts variables from-------------------------
C     ZXMWD range [X] to FW range [AV]
C-----Creation date: 14 Jan 84----Last update: 12 Jan 85--------------

      IMPLICIT
     &  LOGICAL (A - Z)

      INTEGER
     &  N
     &, JOPT

      DOUBLE PRECISION
     &  X(N)

$     INCLUDE 'MDPE/COMMON'

C---------------------------------------------------------------------

C-----Transform variables being optimized from optimizer range to data
C     range.  Fill in rest with their default values.  Note that FOPT
C-----is always true whenever this routine is called.

      JOPT = 0
      DO 100 JV = 1, NV
         AV(JV, NRP1) = BVDEF(JV)
         IF(.NOT. BOPT(JV)) GO TO 100
            JOPT = JOPT + 1
            AV(JV, NRP1) = (BVHI(JV) - BVLO(JV)) * X(JOPT) + BVLO(JV)
  100 CONTINUE

      END       % FCNVRT

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FZC  (
     &  N     , X     , F     )
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine interfaces ZXMWD and FC-------------------------
C-----Creation date: 30 Dec 83----Last update: 14 Jan 84--------------

      IMPLICIT
     &  LOGICAL (A - Z)

      INTEGER
     &  N

      DOUBLE PRECISION
     &  F     , X(N)
     &, FPENAL
```

```
$     INCLUDE "MDPE/COMMON"
C-----convert variables from ZXMWD range [X] to FW range [AV]----------

      CALL FCNVRT(N, X)

C-----estimate joint criterion C

      CALL FC(F)

      F = -F
      IF(ZOPT .EQ. 3) F = F + FPENAL(NOPT, AV(1, NRP1), BVLO, BVHI)*1D2

      END      % FZC

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FZD   (
     & N     , X     , F     )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates interfaces ZXMWD and FD---------------
C-----Creation date: 14 Jan 84----Last update: 14 Jan 84---------------

      IMPLICIT
     &  LOGICAL (A - Z)

      INTEGER
     &  N

      DOUBLE PRECISION
     &  F     , X(N)
     &, FPENAL

$     INCLUDE "MDPE/COMMON"

C-----convert variables from ZXMWD range [X] to FW range [AV]----------

      CALL FCNVRT(N, X)

C-----estimate model discrimination criterion D

      CALL FD(F)

      F = -F
      IF(ZOPT .EQ. 3) F = F + FPENAL(NOPT, AV(1, NRP1), BVLO, BVHI)*1D2

      END      % FZD

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set
```

```
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FZE    (
     & N      , X      , F      )
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates interfaces ZXMWD and FE--------------
C-----Creation date: 14 Jan 84----Last update: 11 Aug 84--------------

      IMPLICIT
     & LOGICAL (A - Z)

      INTEGER
     & N

      DOUBLE PRECISION
     & F      , X(N)
     &, FPENAL

$     INCLUDE "MDPE/COMMON"

C-----convert variables from ZXMWD range [X] to FW range [AV]---------

      CALL FCNVRT(N, X)

C-----estimate parameter esimation criterion E

      CALL FE(F)

C     F = -F                  % Not after 11 August 84!            %%%%%%

      F = -DLOG(DMAX1(F, SMALL))%ln transform experiment 11 Aug 84%%%%%
      IF(ZOPT .EQ. 3) F = F + FPENAL(NOPT, AV(1, NRP1), BVLO, BVHI)*1D2

      END      % FZE

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FCMAX
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine optimizes C, the combined criterion-------------
C     for simultaneous model discrimination and parameter
C     estimation, using IMSL routine ZXMWD
C-----Creation date: 16 Dec 83----Last update: 12 Jan 85--------------

      IMPLICIT
     & LOGICAL (A - Z)

$     INCLUDE "MDPE/COMMON"
```

```
      DOUBLE PRECISION
     & DUMMY , F
     &, CNVERT, FW    , ROUND , FPENAL                    % 23 Jan 85 %%

      INTEGER
     & IER   , J

      REAL
     & FZC

      EXTERNAL
     & FZC

 9001 FORMAT(/,' MAXIMUM ''C'' VALUE [1]',/)

C-----------------------------------------------------------------------

      DO 100 JV = 1, NV
         BOPT(JV) = .FALSE.
  100 CONTINUE

      DO 200 JM = 1, NM
         DO 200 JV = 1, NV
            BOPT(JV) = BOPT(JV) .OR. AOPT(JV, PM(JM), SW)
  200 CONTINUE

      IF(SECOND) BOPT (2) = .FALSE.
      IF(SECOND) BVDEF(2) = BTRIAL(JT)

      NOPT = 0
      DO 300 JV = 1, NV
         IF(BOPT(JV)) NOPT = NOPT + 1
  300 CONTINUE

      NSRCH = 2 ** NOPT

      DO 400 JV = 1, NOPT
         BZA(JV) = 0D0
         BZB(JV) = 1D0
  400 CONTINUE

      IF(DEBUGG .LT. 0) GO TO 301
         IF(FCLOSE) CALL EOFILE(PRINTR)
         WRITE(PRINTR, *//) 'Just before Cmax', NOPT,
     &   NSRCH, NV
         WRITE(PRINTR, /) 'BOPT', (BOPT(JV), JV = 1, NV)
         IF(FCLOSE) LOCK PRINTR
  301 CONTINUE

      GO TO(1, 2, 3), ZOPT
        CALL MESSAG(PRINTR, 'FCMAX', 9, .TRUE.)

    1   CONTINUE
        CALL ZXMWD (
```

```
     &    FZC    , NOPT   , NSIG   , BZA    , BZB    , NSRCH , BZX    , F
     &  , BZWORK, IWORK  , IER    )
        GO TO 9

   2    CONTINUE
        CALL ZXSIMP(
     &    FZC    , NOPT   , NSIG   , NSRCH  , PRINTR, MP+1   , BZA    , BZB
     &  , BZX    , F      , ASIMP )
        GO TO 9

   3    CONTINUE
        CALL ZXMING(
     &    FZC    , NOPT   , NSIG   , NSRCH  , PRINTR, IOPT   , BZA    , BZB
     &  , BZX    , F      , BZWORK, IWORK )
        GO TO 9

   9 CONTINUE

C-----routine FZC makes sure F is negative so that when F is
C-----minimized (by ZXMWD), -F (which is positive) is maximized.

      CMAX    = -F

      DO 500 JV = 1, NOPT
         BVMAX(JV) = CNVERT(BZX(JV), BVHI(JV), BVLO(JV), NSIG)
  500 CONTINUE

C-----Borrow BY from FD for temporary storage of predicted Y's

      DO 600 JV = 1, NV
         BV(JV) = BVMAX(JV)
  600 CONTINUE

      DO 700 JM = 1, NM
         DO 610 JP = 1, NP(JM)
            BP(JP) = AP(JP, JM)
  610    CONTINUE
         BY(JM) = FW(DUMMY)
  700 CONTINUE

      IF(FCLOSE) CALL EOFILE(PRINTR)
      WRITE(PRINTR, 9001)
      WRITE(PRINTR, /    ) (BVMAX(JV), JV = 1, NV)
      WRITE(PRINTR, /    ) 'CMAX', ROUND(CMAX, NSIG)
      WRITE(PRINTR, /) 'Predicted Y''s',(ROUND(BY(J), NSIG),J = 1,NM)
      IF(ZOPT .EQ. 1) WRITE(PRINTR, /)
     &  'NSIG'    , ROUND(BZWORK(1), NSIG)
      IF(ZOPT .EQ. 3) WRITE(PRINTR, /)
     &  'NFE'     , ROUND(BZWORK(2), NSIG)
     &, 'NORM'    , ROUND(BZWORK(1), NSIG)
     &, 'NSIG'    , ROUND(BZWORK(3), NSIG)
     &, 'PENALTY', ROUND(FPENAL(NOPT, BVMAX, BVLO, BVHI) * 1D2, NSIG)
      IF(FCLOSE) LOCK PRINTR
```

```
      RETURN
      END       % FCMAX

$     OPT = 0
$     SET OWN         % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FDMAX
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine optimizes D, the model discrimination------------
C     criterion, using IMSL routine ZXMWD
C-----Creation date: 14 Jan 84----Last update: 12 Jan 85---------------

      IMPLICIT
     &  LOGICAL (A - Z)

$     INCLUDE "MDPE/COMMON"

      DOUBLE PRECISION
     &  DUMMY , F
     &, CNVERT, FW    , ROUND , FPENAL                         % 23 Jan 85 %%

      INTEGER
     &  IER   , J

      REAL
     &  FZD

      EXTERNAL
     &  FZD

 9001 FORMAT(/,' MAXIMUM ''D'' VALUE [1]',/)

C---------------------------------------------------------------------

      DO 100 JV = 1, NV
         BOPT(JV) = .FALSE.
  100 CONTINUE

      DO 200 JM = 1, NM
         DO 200 JV = 1, NV
            BOPT(JV) = BOPT(JV) .OR. AOPT(JV, PM(JM), SW)
  200 CONTINUE

      NOPT = 0
      DO 300 JV = 1, NV
         IF(BOPT(JV)) NOPT = NOPT + 1
  300 CONTINUE

      NSRCH = 2 ** NOPT

      DO 400 JV = 1, NOPT
```

```
         BZA(JV) = 0D0
         BZB(JV) = 1D0
  400 CONTINUE

      IF(DEBUGG .LT. 0) GO TO 301
         IF(FCLOSE) CALL EOFILE(PRINTR)
         WRITE(PRINTR, *//) 'Just before Dmax', NOPT,
     &   NSRCH, NV
         WRITE(PRINTR, /) 'BOPT', (BOPT(JV), JV = 1, NV)
         IF(FCLOSE) LOCK PRINTR
  301 CONTINUE

      GO TO(1, 2, 3), ZOPT
        CALL MESSAG(PRINTR, 'FCMAX', 9, .TRUE.)

    1   CONTINUE
        CALL ZXMWD (
     &    FZD    , NOPT   , NSIG   , BZA    , BZB    , NSRCH , BZX    , F
     &  , BZWORK, IWORK  , IER    )
        GO TO 9

    2   CONTINUE
        CALL ZXSIMP(
     &    FZD    , NOPT   , NSIG   , NSRCH , PRINTR, MP+1   , BZA    , BZB
     &  , BZX    , F      , ASIMP )
        GO TO 9

    3   CONTINUE
        CALL ZXMING(
     &    FZD    , NOPT   , NSIG   , NSRCH , PRINTR, IOPT   , BZA    , BZB
     &  , BZX    , F      , BZWORK, IWORK )
        GO TO 9

    9 CONTINUE

C-----routine FZD makes sure F is negative so that when F is
C-----minimized (by ZXMWD), -F (which is positive) is maximized.

      DMAX = -F

      DO 500 JV = 1, NOPT
         BVMAX(JV) = CNVERT(BZX(JV), BVHI(JV), BVLO(JV), NSIG)
  500 CONTINUE

C-----Borrow BY from FD for temporary storage of predicted Y's

      DO 600 JV = 1, NV
         BV(JV) = BVMAX(JV)
  600 CONTINUE

      DO 700 JM = 1, NM
         DO 610 JP = 1, NP(JM)
            BP(JP) = AP(JP, JM)
  610    CONTINUE
```

```
          BY(JM) = FW(DUMMY)
  700 CONTINUE

      IF(FCLOSE) CALL EOFILE(PRINTR)
      WRITE(PRINTR, 9001)
      WRITE(PRINTR, /   ) (BVMAX(JV), JV = 1, NV)
      WRITE(PRINTR, /   ) 'DMAX', ROUND(DMAX, NSIG)
      WRITE(PRINTR, /) 'Predicted Y''s',(ROUND(BY(J), NSIG),J = 1,NM)
      IF(ZOPT .EQ. 1) WRITE(PRINTR, /)
     &  'NSIG'    , ROUND(BZWORK(1), NSIG)
      IF(ZOPT .EQ. 3) WRITE(PRINTR, /)
     &  'NFE'     , ROUND(BZWORK(2), NSIG)
     &, 'NORM'    , ROUND(BZWORK(1), NSIG)
     &, 'NSIG'    , ROUND(BZWORK(3), NSIG)
     &, 'PENALTY', ROUND(FPENAL(NOPT, BVMAX, BVLO, BVHI) * 1D2, NSIG)
      IF(FCLOSE) LOCK PRINTR

      RETURN
      END       % FDMAX

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FEMAX
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine optimizes E, the parameter estimation------------
C     criterion, using IMSL routine ZXMWD
C-----Creation date: 14 Jan 84----Last update: 12 Jan 85--------------

      IMPLICIT
     &  LOGICAL (A - Z)

$     INCLUDE "MDPE/COMMON"

      DOUBLE PRECISION
     &  DUMMY , F
     &, CNVERT, FW    , ROUND , FPENAL                    % 23 Jan 85 %%

      INTEGER
     &  IER   , J

      REAL
     &  FZE

      EXTERNAL
     &  FZE

      LOGICAL
     &  OLD

      COMMON
```

```
     & /CE     / OLD

 9001 FORMAT(/,' MAXIMUM ''E'' VALUE FOR MODEL ',I2,' [1]',/)
C-----------------------------------------------------------------------

      OLD = .FALSE.

      DO 100 JV = 1, NV
         BOPT(JV) = AOPT(JV, PM(JM), SW)
  100 CONTINUE

      NOPT = 0
      DO 200 JV = 1, NV
         IF(BOPT(JV)) NOPT = NOPT + 1
  200 CONTINUE

      NSRCH = 2 ** NOPT
C     NSRCH = 3 ** NOPT                                %%%%% 11 FEB 85 %%%%

      DO 300 JV = 1, NOPT
         BZA(JV) = 0D0
         BZB(JV) = 1D0
  300 CONTINUE

      IF(DEBUGG .LT. 0) GO TO 301
         IF(FCLOSE) CALL EOFILE(PRINTR)
         WRITE(PRINTR, *//) 'Just before Emax', NOPT,
     &   NSRCH, NV
         WRITE(PRINTR, /) 'BOPT', (BOPT(JV), JV = 1, NV)
         IF(FCLOSE) LOCK PRINTR
  301 CONTINUE

      GO TO(1, 2, 3), ZOPT
        CALL MESSAG(PRINTR, 'FCMAX', 9, .TRUE.)

    1   CONTINUE
        CALL ZXMWD (
     &    FZE    , NOPT   , NSIG   , BZA    , BZB    , NSRCH , BZX    , F
     &  , BZWORK, IWORK  , IER    )
        GO TO 9

    2   CONTINUE
        CALL ZXSIMP(
     &    FZE    , NOPT   , NSIG   , NSRCH , PRINTR, MP+1   , BZA    , BZB
     &  , BZX    , F      , ASIMP )
        GO TO 9

    3   CONTINUE
        CALL ZXMING(
     &    FZE    , NOPT   , NSIG   , NSRCH , PRINTR, IOPT   , BZA    , BZB
     &  , BZX    , F      , BZWORK, IWORK )
        GO TO 9
```

```
    9 CONTINUE

C-----routine FZE makes sure F is negative so that when F is
C-----minimized (by ZXMWD), -F (which is positive) is maximized.

C     BEMAX(JM) = -F      % ln transform experiment 11 Aug 84    %%%%%%
      BEMAX(JM) = DEXP(-F) % 11 Aug 84                           %%%%%%

      DO 400 JV = 1, NOPT
         BVMAX(JV) = CNVERT(BZX(JV), BVHI(JV), BVLO(JV), NSIG)
  400 CONTINUE

C-----Borrow BY from FD for temporary storage of predicted Y's

      DO 500 JV = 1, NV
         BV(JV) = BVMAX(JV)
  500 CONTINUE

      DO 600 J = 1, NM
         DO 510 JP = 1, NP(J)
            BP(JP) = AP(JP, J)
  510    CONTINUE
         BY(J) = FW(DUMMY)
  600 CONTINUE

      IF(FCLOSE) CALL EOFILE(PRINTR)
      WRITE(PRINTR, 9001) PM(JM)
      WRITE(PRINTR, /   ) (BVMAX(JV), JV = 1, NV)
      WRITE(PRINTR, /   ) 'EMAX', ROUND(BEMAX(JM), NSIG)
      WRITE(PRINTR, /) 'Predicted Y''s',(ROUND(BY(J), NSIG),J = 1,NM)
      IF(ZOPT .EQ. 1) WRITE(PRINTR, /)
     &  'NSIG'    , ROUND(BZWORK(1), NSIG)
      IF(ZOPT .EQ. 3) WRITE(PRINTR, /)
     &  'NFE'     , ROUND(BZWORK(2), NSIG)
     &, 'NORM'    , ROUND(BZWORK(1), NSIG)
     &, 'NSIG'    , ROUND(BZWORK(3), NSIG)
     &, 'PENALTY', ROUND(FPENAL(NOPT, BVMAX, BVLO, BVHI) * 1D2, NSIG)
      IF(FCLOSE) LOCK PRINTR

      RETURN
      END      % FEMAX

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE INITAL
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine gets the necessary initial data------------------
C-----Creation date: 03 Jan 84----Last update: 17 Jan 85---------------
```

```
      IMPLICIT
     &  LOGICAL (A - Z)

      INTEGER
     &  J     , K

      DOUBLE PRECISION
     &  ROUND , YIELD

$     INCLUDE "MDPE/COMMON"

 9000 FORMAT(/,' enter INITAL',/)
 9001 FORMAT(/,' PARAMETERS [Models x Parameters]',/)
 9002 FORMAT(/,' VARIABLES [Runs x Variables]',/)
 9003 FORMAT(/,' EXPERIMENTAL DATA [Runs x Responses]',/)
 9004 FORMAT(/,' RESPONSE VARIANCE',/)
 9005 FORMAT(/,' DEBUG LEVEL',/)
 9006 FORMAT(/,' CALCULATE PROBABILITIES FROM SCRATCH?',/)
 9007 FORMAT(/,' # INITIAL SEARCH POINTS',/)
 9008 FORMAT(/,' # SIGNIFICANT DIGITS',/)
 9010 FORMAT(/,' UPPER BOUNDS ON VARIABLES [Variables]',/)
 9015 FORMAT(/,' DO FIXED TEMPERATURE CMAX TRIALS INSTEAD?',/)
 9020 FORMAT(/,' LOWER BOUNDS ON VARIABLES [Variables]',/)
 9025 FORMAT(/,' DEFAULT VALUES OF UNOPTIMIZED VARIABLES'
     &         ,' [Variables]',/)
 9030 FORMAT(/,' TRANSFORMED DATA ',50('-'),/)
 9035 FORMAT(/,' TEMPS FOR FIXED TEMP CMAX TRIALS [trials]',/)
 9090 FORMAT(/)
 9099 FORMAT(/,' exit INITAL',/)
 9110 FORMAT(1X,A6)
 9120 FORMAT(/,' READ IN PARAMETER ESTIMATION MAXIMA? [models]',/)
 9130 FORMAT(/,' PARAMETER ESTIMATION MAXIMA [models]',/)

C-----------------------------------------------------------------------

      DISKIN = 1
      ERRORS = 6
      PRINTR = 7
      REMOTE = 5

C-----Get machine dependent constants

      CALL CONST

C-----Initialize optimization flag array

      CALL INITOP

      OPEN  (DISKIN) % just being
      REWIND(DISKIN) % carefull...

      WRITE(PRINTR, 9000) % say hi

C-----actual # of models, parameters, runs, variables, responses, &
```

```
C     prior function values
C-----[NM, NP, NR, NV, & NW respectively]

      READ (DISKIN, /)    NF, NM, NR, NT, NV, NW
      WRITE(PRINTR, *//) NF, NM, NR, NT, NV, NW
      CALL SKIPOT(PRINTR, 1)
      READ (DISKIN, /  ) (NP(J), J = 1, NM)
      WRITE(PRINTR, /  ) ' NP ', (NP(J), J = 1, NM)
      CALL SKIPOT(PRINTR, 1)
      NRM1    = NR - 1
      NRP1    = NR + 1

C-----select response to optimize [SW]
C-----and optimization method [ZOPT]

      READ (DISKIN, /    ) SW, ZOPT, IOPT

      IF(SW .EQ. 1) WRITE(PRINTR, *// ) SW, ' (Lignin)'
      IF(SW .EQ. 2) WRITE(PRINTR, *// ) SW, ' (Celluose)'
      IF(SW .EQ. 3) WRITE(PRINTR, *// ) SW, ' (Galactoglucomannan)'
      IF(SW .EQ. 4) WRITE(PRINTR, *// ) SW, ' (Arabinoxylan)'
      IF(SW .EQ. 5) WRITE(PRINTR, *// ) SW, ' (Extractives)'
      CALL SKIPOT(PRINTR, 1)

      IF(ZOPT .EQ. 1) WRITE(PRINTR, *// ) ZOPT, ' (ZXMWD)'
      IF(ZOPT .EQ. 2) WRITE(PRINTR, *// ) ZOPT, ' (ZXSIMP)'
      IF(ZOPT .EQ. 3) WRITE(PRINTR, *// ) ZOPT, ' (ZXMING)'
      CALL SKIPOT(PRINTR, 1)

      IF(ZOPT .NE. 3) GO TO 30
        IF(IOPT .EQ. 0) WRITE(PRINTR, *//) IOPT, 'Identity Hessian'
        IF(IOPT .EQ. 1) WRITE(PRINTR, *//) IOPT, 'Read in Hessian'
        IF(IOPT .EQ. 2) WRITE(PRINTR, *//) IOPT, 'Diagonal Hessian'
        IF(IOPT .EQ. 3) WRITE(PRINTR, *//) IOPT, 'Full Hessian'
   30 CONTINUE

      IF(ZOPT .EQ. 1) GO TO 40
         WRITE(PRINTR, *//) ZOPT, 'is not currently working correctly'
     & , 'ZOPT = 1 (ZXMWD) will be used instead <17 Jan 85>'

         ZOPT = 1
   40 CONTINUE

C-----select integration method [METH] and iteration method [MITER]

      READ(DISKIN, /    ) METH, MITER

      CALL SKIPOT(PRINTR, 1)
      IF(METH .EQ. 1) WRITE(PRINTR, *// ) METH, ' (Adams(non-stiff))'
      IF(METH .EQ. 2) WRITE(PRINTR, *// ) METH, ' (Gear(stiff))'

      IF(MITER .EQ. 0) WRITE(PRINTR, *// ) MITER
     &                                    , ' (Functional iteration)'
      IF(MITER .EQ. 1) WRITE(PRINTR, *// ) MITER
```

```
     &                                , ' (Analytic, full Jacobian)'
      IF(MITER .EQ. 2) WRITE(PRINTR, *// ) MITER
     &                                , ' (Internal, full Jacobian)'
      IF(MITER .EQ. 3) WRITE(PRINTR, *// ) MITER
     &                                , ' (Internal, diagonal Jacobian)'
      CALL SKIPOT(PRINTR, 2)

C-----model pointer [PM], # of fractions reactant is split into [FS],
C-----and total number of fractions [FT]

      READ (DISKIN, /   )        (PM(JM), JM = 1, NM)
      WRITE(PRINTR, /   ) ' PM', (PM(JM), JM = 1, NM)
      CALL SKIPOT(PRINTR, 1)

      READ (DISKIN, /   )        (FS(JM), JM = 1, NM)
      WRITE(PRINTR, /   ) ' FS', (FS(JM), JM = 1, NM)
      CALL SKIPOT(PRINTR, 1)

      READ (DISKIN, /   )        (FT(JM), JM = 1, NM)
      WRITE(PRINTR, /   ) ' FT', (FT(JM), JM = 1, NM)
      CALL SKIPOT(PRINTR, 1)

C-----upper & lower limits of variables [variables]

      WRITE(PRINTR, 9010)

      READ (DISKIN, /) (BVHI(JV), JV = 1, NV)
      WRITE(PRINTR, /) (BVHI(JV), JV = 1, NV)

      WRITE(PRINTR, 9020)

      READ (DISKIN, /) (BVLO(JV), JV = 1, NV)
      WRITE(PRINTR, /) (BVLO(JV), JV = 1, NV)

C-----default values of unoptimized variables [variables]
C-----(also used to initialize optimum conditions display vector)

      WRITE(PRINTR, 9025)

      READ (DISKIN, /) (BVDEF(JV), JV = 1, NV)
      WRITE(PRINTR, /) (BVDEF(JV), JV = 1, NV)

      BACKSPACE DISKIN
      READ (DISKIN, /) (BVMAX(JV), JV = 1, NV)

C-----Do combined criteria optimization?

      CALL SKIPOT(PRINTR, 1)
      READ (DISKIN, /  ) DOC
      WRITE(PRINTR, *//) DOC

C-----Do model discrimination optimization?  Force if DOC true

      CALL SKIPOT(PRINTR, 1)
```

```
      READ (DISKIN, /  ) DOD
      DOD = DOC .OR. DOD
      WRITE(PRINTR, *//) DOD
      IF(DOC) WRITE(PRINTR, /) ' DOD forced true if DOC is true'

C-----Select which models have pre-determined parameter estimation
C     maxima using [BEOLD], and store in [BEMAX].  [BY], normally used
C-----by FD, is also use here for temporary storage

      WRITE(PRINTR, 9120)

      READ (DISKIN, /   ) (BEOLD(JM), JM = 1, NM)
      WRITE(PRINTR, /   ) (BEOLD(JM), JM = 1, NM)

      K = 0
      DO 1100 JM = 1, NM
         IF(BEOLD(JM)) K = K + 1
 1100 CONTINUE

      IF(K .EQ. 0) CALL SKIPIN(DISKIN, 1)
      IF(K .EQ. 0) GO TO 1300
        WRITE(PRINTR, 9130)

        READ (DISKIN, /   ) (BY(J), J = 1, K)
        WRITE(PRINTR, /   ) (BY(J), J = 1, K)

        J = 0
        DO 1200 JM = 1, NM
           IF(.NOT. BEOLD(JM)) GO TO 1200
             J = J + 1
             BEMAX(JM) = BY(J)
 1200   CONTINUE
 1300 CONTINUE

C-----parameters [parameters x models]

      WRITE(PRINTR, 9001)

      DO 100 JM = 1, NM
         DO 100 J = 1, NP(JM), 7
            K = MINO(J+6, NP(JM))
            READ (DISKIN, /)     (AP(JP, JM), JP = J, K)
            WRITE(PRINTR, /) JM, (AP(JP, JM), JP = J, K)
  100 CONTINUE

C-----variables [variables x runs]

      WRITE(PRINTR, 9002)

      DO 200 JR = 1, NR
         READ (DISKIN, 9110) LABEL(JR)
         READ (DISKIN, /)     (AV(JV, JR), JV = 1, NV)
         WRITE(PRINTR, 9110) LABEL(JR)
         WRITE(PRINTR, /) JR, (AV(JV, JR), JV = 1, NV)
```

```
  200 CONTINUE

C-----experimental data [responses x runs]

      WRITE(PRINTR, 9003)

      DO 250 JR = 1, MR
         DO 250 JW = 1, MW
            AY(JW, JR) = 999999999
  250 CONTINUE

      DO 300 JR = 1, NR
         READ (DISKIN, 9110) LABEL(JR)
         READ (DISKIN, /)     (AY(JW, JR), JW = 1, NW)
         WRITE(PRINTR, 9110) LABEL(JR)
         WRITE(PRINTR, /) JR, (AY(JW, JR), JW = 1, NW)
  300 CONTINUE

C-----power factor for W1 and W2

      CALL SKIPOT(PRINTR, 1)
      READ (DISKIN, /)   LAMBDA
      WRITE(PRINTR, *//) LAMBDA

C-----response variance [BSSW(SW)]

      WRITE(PRINTR, 9004)
      READ (DISKIN, /) (BSSW(JW), JW = 1, NW)
      WRITE(PRINTR, /) (BSSW(JW), JW = 1, NW)

C-----step size for DFDX [H]

      CALL SKIPOT(PRINTR, 1)
      READ (DISKIN, /) H
      WRITE(PRINTR, *//) H

C-----debug level

      READ (DISKIN, /)   DEBUGG
      WRITE(PRINTR, 9005)
      WRITE(PRINTR, *//) DEBUGG

C-----number of significant digits

      READ (DISKIN, /)   NSIG
      WRITE(PRINTR, 9008)
      WRITE(PRINTR, *//) NSIG

C-----is this first time through? T => calc probabilities from scratch

      READ (DISKIN, /)   FIRST
      WRITE(PRINTR, 9006)
      WRITE(PRINTR, *//) FIRST
```

```
C-----model likelyhoods [models x run 0]

      DO 400 JM = 1, NM
         APR(JM, 0) = 1D0 / DFLOAT(NM)
  400 CONTINUE

      IF(FIRST) GO TO 10

C-------model likelyhoods [models x runs]

        DO 500 JR = 1, NR - 1
           READ (DISKIN, /) (APR(JM, JR), JM = 1, NM)
  500   CONTINUE

        WRITE(PRINTR, 9020)
        DO 510 JR = 0, NR - 1
           WRITE(PRINTR, /) JR, (APR(JM, JR), JM = 1, NM)
  510   CONTINUE
   10 CONTINUE

C-----Run some fixed temp CMAX trials?

      READ (DISKIN, /   ) SECOND
      WRITE(PRINTR, 9015)
      WRITE(PRINTR, *// ) SECOND

      IF(.NOT. SECOND) GO TO 20
        READ (DISKIN, /  ) DMAX
        WRITE(PRINTR, *//) DMAX
        READ (DISKIN, /   )(BEMAX(JM), JM = 1, NM)
        WRITE(PRINTR, /  )' EMAX', (BEMAX(JM), JM = 1, NM)

        READ (DISKIN, /   )(BTRIAL(JT), JT = 1, NT)
        WRITE(PRINTR, 9035)
        WRITE(PRINTR, /   )(BTRIAL(JT), JT = 1, NT)
   20 CONTINUE

      WRITE(PRINTR, 9099) % say bye
      CLOSE(DISKIN)       % being carefull again...

C-----Now that the data is all read in, make necessary transformations.
C-----<Note: These must agree in function to those in MRPEST/TRANS!>

      DO 600 JR = 1, NR

C--------Keep Time in Hours

         AV(1, JR) = AV(1, JR)
         IF(JR .GT. 1) GO TO 1
           BVHI(1) = BVHI(1)
           BVLO(1)  = BVLO(1)
           BVDEF(1) = BVDEF(1)
           BVMAX(1) = BVMAX(1)
    1    CONTINUE
```

```
C--------Convert Temperature from degrees C to degrees K

         AV(2, JR) = AV(2, JR) + 273D0
         IF(JR .GT. 1) GO TO 2
           BVHI(2)   = BVHI(2)   + 273D0
           BVLO(2)   = BVLO(2)   + 273D0
           BVDEF(2) = BVDEF(2) + 273D0
           BVMAX(2) = BVMAX(2) + 273D0
           DO 610 JT = 1, NT
              BTRIAL(JT) = BTRIAL(JT) + 273D0
  610      CONTINUE
    2    CONTINUE

C--------OH & SH are in mol/1, AQ in mmol/1.  Leave them that way
C--------for now. (initial OH adjusted for est. consumption 9 Feb 85)

         YIELD = AY(1, JR)
         DO 900 K = 2, 4
            YIELD = AY(K, JR) + YIELD
  900    CONTINUE
         AV(3, JR) = AV(3, JR) + (YIELD * .297591D0 - 25.2407D0)/124.D0
         IF(JR .GT. 1) GO TO 3
           BVHI(3)   = BVHI(3)
           BVLO(3)   = BVLO(3)
           BVDEF(3) = BVDEF(3)
           BVMAX(3) = BVMAX(3)
    3    CONTINUE

         AV(4, JR) = AV(4, JR)
         IF(JR .GT. 1) GO TO 4
           BVHI(4)   = BVHI(4)
           BVLO(4)   = BVLO(4)
           BVDEF(4) = BVDEF(4)
           BVMAX(4) = BVMAX(4)
    4    CONTINUE

         AV(5, JR) = AV(5, JR)
         IF(JR .GT. 1) GO TO 5
           BVHI(5)   = BVHI(5)
           BVLO(5)   = BVLO(5)
           BVDEF(5) = BVDEF(5)
           BVMAX(5) = BVMAX(5)
    5    CONTINUE

C--------Convert t2Temp from minutes to hours [20 Jun 84]

         AV(6, JR) = AV(6, JR) / 6D1
         IF(JR .GT. 1) GO TO 6
           BVHI(6)   = BVHI(6)   / 6D1
           BVLO(6)   = BVLO(6)   / 6D1
           BVDEF(6) = BVDEF(6) / 6D1
           BVMAX(6) = BVMAX(6) / 6D1
    6    CONTINUE
```

```
C--------Convert dependent variables from %o.d. wood to
C--------fraction of initial wood concentration

           AY(1, JR) = AY(1, JR) / 27.938D0    % Lignin
           AY(2, JR) = AY(2, JR) / 35.952D0    % Celluose
           AY(3, JR) = AY(3, JR) / 15.657D0    % GGM
           AY(4, JR) = AY(4, JR) /  7.444D0    % AX
           AY(5, JR) = AY(5, JR) /  3.292D0    % Extractives

  600 CONTINUE

C-----transformed data & variables

      WRITE(PRINTR, 9030)

      WRITE(PRINTR, 9010)
      WRITE(PRINTR, /) (ROUND(BVHI(JV), NSIG), JV = 1, NV)

      WRITE(PRINTR, 9020)
      WRITE(PRINTR, /) (ROUND(BVLO(JV), NSIG), JV = 1, NV)

      WRITE(PRINTR, 9025)
      WRITE(PRINTR, /) (ROUND(BVDEF(JV), NSIG), JV = 1, NV)

      WRITE(PRINTR, 9002)
      DO 700 JR = 1, NR
         WRITE(PRINTR, 9110) LABEL(JR)
         WRITE(PRINTR,    /) (ROUND(AV(JV, JR), NSIG), JV = 1, NV)
  700 CONTINUE

      WRITE(PRINTR, 9003)
      DO 800 JR = 1, NR
         WRITE(PRINTR, 9110) LABEL(JR)
         WRITE(PRINTR,    /) (ROUND(AY(JW, JR), NSIG), JW = 1, NW)
  800 CONTINUE

      IF(.NOT. SECOND) GO TO 90
        CALL SKIPOT(PRINTR, 1)
        WRITE(PRINTR, *// ) DMAX
        CALL SKIPOT(PRINTR, 1)
        WRITE(PRINTR, /   ) 'EMAX', (BEMAX(JM), JM = 1, NM)
        WRITE(PRINTR, 9035)
        WRITE(PRINTR, /   ) (BTRIAL(JT), JT = 1, NT)
   90 CONTINUE

      END      % INITAL

$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C                 MAIN
```

```
  100 CONTINUE

      WRITE(PRINTR, /) (ROUND(BSSM(JM), NSIG), JM = 1, NM)

      IF(DEBUGG .LT. 0) GO TO 12
        CALL TIMER(PTHR, PTMIN, PTSEC, 2)
        CALL TIMER(IOHR, IOMIN, IOSEC, 3)
        WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
   12 CONTINUE

C-----Pr(model JR correct) for each model

      WRITE(PRINTR, 9030)
      JR = 0
      WRITE(PRINTR, /) JR, (ROUND(APR(JM, JR),NSIG), JM = 1, NM)

      IF(FIRST) GO TO 1
        JR   = NR
        FOPT = .FALSE.

        CALL FPROB
        WRITE(PRINTR, *//) JFEVAL

        DO 150 JR = 1, NR
           WRITE(PRINTR, /) JR, (ROUND(APR(JM, JR),NSIG), JM = 1, NM)
  150   CONTINUE
        GO TO 2
    1 CONTINUE

      FOPT = .FALSE.
      DO 200 JR = 1, NR
         CALL FPROB
         WRITE(PRINTR, *//) JFEVAL
         WRITE(PRINTR, /) JR, (ROUND(APR(JM, JR),NSIG), JM = 1, NM)
         WRITE(PRINTR, /) ROUND(AY(SW, JR),NSIG)
     &                   ,(ROUND(BY(JM),NSIG), JM = 1, NM)
  200 CONTINUE

    2 CONTINUE
      IF(DEBUGG .LT. 0) GO TO 13
        CALL TIMER(PTHR, PTMIN, PTSEC, 2)
        CALL TIMER(IOHR, IOMIN, IOSEC, 3)
        WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
   13 CONTINUE

      IF(DEBUGG .NE. -1) GO TO 19                               %%%%%%%
        CLOSE(PRINTR, DISP = CRUNCH)                            %%%%%%%
        STOP                                                    %%%%%%%
   19 CONTINUE                                                  %%%%%%%

      IF(SECOND) GO TO 20

C-------optimal E criterion value for each model [BEMAX(JM)]
```

```
      FCLOSE = .TRUE.
      FOPT   = .TRUE.

      IF(FCLOSE) LOCK PRINTR
      DO 300 JM = 1, NM
         IF(BEOLD(JM)) GO TO 300
           CALL FEMAX

           IF(FCLOSE) CALL EOFILE(PRINTR)
           WRITE(PRINTR, *//) JM, JFEVAL
           CALL TIMER(PTHR, PTMIN, PTSEC, 2)
           CALL TIMER(IOHR, IOMIN, IOSEC, 3)
           WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
           IF(FCLOSE) LOCK PRINTR
  300 CONTINUE

C-------optimal D criterion value [DMAX]

      FCLOSE = .TRUE.
      FOPT   = .TRUE.

      IF(FCLOSE) LOCK PRINTR
      IF(.NOT. DOD) GO TO 30
        CALL FDMAX

        IF(FCLOSE) CALL EOFILE(PRINTR)
        WRITE(PRINTR, *//) JFEVAL
        CALL TIMER(PTHR, PTMIN, PTSEC, 2)
        CALL TIMER(IOHR, IOMIN, IOSEC, 3)
        WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
        IF(FCLOSE) LOCK PRINTR
   30 CONTINUE

C-------optimal C criterion value [CMAX]

      FCLOSE = .TRUE.
      FOPT   = .TRUE.

      IF(FCLOSE) LOCK PRINTR
      IF(.NOT. DOC) GO TO 99
        CALL FCMAX

        IF(FCLOSE) CALL EOFILE(PRINTR)
        WRITE(PRINTR, *//) JFEVAL
        CALL TIMER(PTHR, PTMIN, PTSEC, 2)
        CALL TIMER(IOHR, IOMIN, IOSEC, 3)
        WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
        IF(FCLOSE) LOCK PRINTR
        GO TO 99
   20 CONTINUE

C-----Fixed temperature CMAX trials

      DO 400 JT = 1, NT
```

```
      FCLOSE = .TRUE.
      FOPT   = .TRUE.

      IF(FCLOSE) LOCK PRINTR
      CALL FCMAX

      IF(FCLOSE) CALL EOFILE(PRINTR)
      WRITE(PRINTR, *//) JT, JFEVAL

      CALL TIMER(PTHR, PTMIN, PTSEC, 2)
      CALL TIMER(IOHR, IOMIN, IOSEC, 3)
      WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
      IF(FCLOSE) LOCK PRINTR
400 CONTINUE

 99 CONTINUE
    CLOSE(ERRORS, DISP = CRUNCH)
    IF(FCLOSE) CALL EOFILE(PRINTR)
    IF(FCLOSE) CLOSE(PRINTR, DISP = CRUNCH)
    STOP
    END      % MAIN
```

```
#FILE (MARK)MDPE/FW ON STUDENTS
C-----Routines required by DGEAR

C$      INCLUDE 'IMSL/DBLE/USPKD'
C$      INCLUDE 'IMSL/DBLE/UGETIO'
C$      INCLUDE 'IMSL/DBLE/UERTST'
$      INCLUDE 'IMSL/DBLE/LUDATF'
$      INCLUDE 'IMSL/DBLE/LUELMF'
$      INCLUDE 'IMSL/DBLE/LEQT1B'
$      INCLUDE 'IMSL/DBLE/DGRCS'
$      INCLUDE 'IMSL/DBLE/DGRPS'
$      INCLUDE 'IMSL/DBLE/DGRST'
$      INCLUDE 'IMSL/DBLE/DGRIN'
$      INCLUDE 'MRPEST/FCN'
$      INCLUDE 'MRPEST/FCNJ'
$      INCLUDE 'IMSL/DBLE/DGEAR'
$      INCLUDE 'UTIL/DBLE/SPLINE'

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION FW    (
     &  DUMMY )
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This function calculates FW, the predicted response,------------
C     given parameters BP(*) for model JM, response JW
C     and varible settings BV(*)
C     Author: Mark A. Burazin
C-----Creation date: 16 Dec 83----Last update: 19 Jan 85--------------

      IMPLICIT
     &  LOGICAL(A-Z)

C-----Declaration of argument list

      DOUBLE PRECISION
     &  DUMMY

C-----Declaration of local variables

      LOGICAL
     &  FSAME , OLD    , OK

      INTEGER
     &  EDEP
     &, IOHR   , IOMIN  , PTHR   , PTMIN
     &, NC     , NOB    , NOBMAX, NODEP  , NOIND  , NVARX  , NVARY  , J
     &, ENOB   , K      , EN     , JFM1   , NFP1
     &, IER    , INDEX  , IWK    , N      %[DGEAR ]    %%%%%

      DOUBLE PRECISION
     &  GX     , GY     , TOL    , XEND   , SAME
     &, IOSEC  , PENALT, PTSEC  , XJ
     &, HSTEP  , WK                              % [DGEAR ]
```

```
     &, ONE    , P1     , TOLMAX, TOLMIN, ZERO   , TWOMIN
     &, YO     , YSH    , YAQ    , YSHPOW, YAQPOW, SH     , AQ
      REAL
     &  FCN    , FCNJ   % [DGEAR ]

      DIMENSION
     &  IWK    (3)
     &, GY     (3)
     &, SAME   (50)
     &, WK     (63)

      DATA
     &  ONE    /1D+0  /
     &, P1     /1D-1  /
     &, TOLMAX /1D-3  /
     &, TOLMIN /1D-8  /
     &, TWOMIN /3.33333333333333333333D-2     /
     &, ZERO   /0D+0  /

      EXTERNAL
     &  FCN    , FCNJ

      COMMON
     & /CFW    / SAME   , OLD

$     INCLUDE "MDPE/COMMON"
$     INCLUDE "MRPEST/COMMON"

 9010 FORMAT(/,' PT = ',I2,':',I2,':',F5.2,'  IO = ',I2,':',I2,':'
     &       ,F5.2,/)

C-----Reaction Kinetics Study------------------------------------------
C     This routine is a hybrid.  It is designed to perform the tasks
C     of FW for the Reaction Kinetics Study using IMSL routine DGEAR
C     and MRPEST/FCN.  In order to use FCN with only a single response,
C     CHOOSE is zeroed out except for the desired response.  The arrays
C     FCN expects are filled as necessary.  Only the parts FCN needs
C-----are filled.

C-----Match search capability.  Added 14 Jan 85
C     APAST contains variable and FW values for the last NF calls to
C     FW.  Search through APAST for a match with current variables.
C     If a match is found, reuse the old value of FW and return. Else
C     'shuffle' the old values to make room for the new values, toss
C-----the oldest, calculate the new FW, and store in APAST(NF).

      IF(NF .LE. 0) GO TO 4
         JF = 1
    1    CONTINUE
            JV = 0
    2       CONTINUE
               JV = JV + 1
               IF(JV .LE. NV) GO TO 3
                  FW = APAST(0, JF)
```

```
                 GO TO 11
    3          CONTINUE
            IF(APAST (JV, JF) .EQ. BV(JV) .AND.
     &         JMPAST(JF)     .EQ. JM          ) GO TO 2
            JF = JF + 1
         IF(JF .LE. NF) GO TO 1

         NFP1 = NF + 1
         DO 100 J = 1, NF - 1
            JF   = NFP1 - J
            JFM1 = JF - 1
            JMPAST(JF) = JMPAST(JFM1)
            DO 100 JV = 0, NV
               APAST(JV, JF) = APAST(JV, JFM1)
  100    CONTINUE
    4 CONTINUE


C-----POINT2 is an indirect pointer which allows MDPE to work with
C     models in arbitrary order.  POINT2(J, K) is the number of the
C     Jth model for response K.  The technique for reading the data
C-----just once requires the use of $SET OWN.

      IF(OLD) GO TO 5
         DO 200 K = 1, NW
            POINTR(K) = 1                             % input for [MRPEST]%%%
            POINTY(K) = 1                             % K for [MRPEST]  %%%%%
  200    CONTINUE
    5 CONTINUE

C-----Use IMSL routine DGEAR to calculate the predicted value
C     corresponding to observation ENOB, response ENOIND, and
C-----model JM

      EDEP   = SW                                                   %%%%%
      EOB    = JR                                                   %%%%%
      XEND   = BV(1)                                                %%%%%
      IF(AY(EDEP, EOB) .LT. 0 .OR. XEND .EQ. 0) GO TO 8
         ENC    = NP(JM)   % NC for MRPEST                         %%%%%%
         ENOIND = NV    % NOIND for MRPEST                          %%%%%

         DO 300 J = 1, ENODEP
            CHOOSE(J) = 0
  300    CONTINUE

         CHOOSE(EDEP) = PM(JM)          % 26 Nov 84                 %%%%%

         DO 400 J = 1, ENC                                           %%%%
            EB(J)  = BP(J) % B(J) for MRPEST                         %%%%
  400    CONTINUE

         DO 500 J = 1, ENOIND
            XJ = BV(J)                          % X(EOB, J) for MRPEST %%%%
            IF(.NOT. FOPT .OR. ZOPT .EQ. 1) GO TO 6                  %%%%
```

```
              XJ = DMAX1(BVLO(J), XJ)
              XJ = DMIN1(BVHI(J), XJ)
    6       CONTINUE
            EX(EOB, J) = XJ
  500     CONTINUE

          XEND   = EX(EOB, 1)                  % absolutely necessary %%%%
          INDEX  = 1                           % first call to DGEAR
          N      = FT(JM)                      % 26 Nov 84            %%%%
          GX     = TWOMIN                      % 19 jan 85            %%%%

C--------Y0 handling patch.  Added 18 Jan 85

          Y0     = EB(3) * P1                              % 30 Dec 84 %%%%
          YSH    = EB(ENC-1) * 1D-2
          YAQ    = EB(ENC) * 1D-2
          SH     = EX(EOB, 4)
          AQ     = EX(EOB, 5)
          IF(SW .LT. 2 .OR. SW .GT. 4) GO TO 511
            GO TO(501, 501, 502, 502), CHOOSE(EDEP) - 6
              GO TO 511

  501         CONTINUE
              Y0 = Y0 + YSH * DSQRT(SH) + YAQ * DSQRT(AQ)
              GO TO 511

  502         CONTINUE
              YSHPOW = EB(ENC-3)
              YAQPOW = EB(ENC-2)
              Y0     = Y0 + YSH * SH**YSHPOW + YAQ * AQ**YAQPOW
              GO TO 511
  511     CONTINUE
          GY(1) = Y0
          IF(N .EQ. 1) GO TO 7
            DO 600 J = 2, N
              GY(J) = ZERO
  600       CONTINUE
    7     CONTINUE
          TOL    = P1 ** NSIG
          TOL    = DMIN1(TOL, TOLMAX)
          TOL    = DMAX1(TOL, TOLMIN)
          HSTEP  = TOL                                                %%%%

          IF(XEND .LE. ZERO) GO TO 8
            CALL DGEAR (
     &        N     , FCN   , FCNJ  , GX    , HSTEP , GY    , XEND
     &      , TOL   , METH  , MITER , INDEX , IWK   , WK    , IER   )
    8 CONTINUE

        FW = AY(EDEP, EOB)
        IF(FW .LT. ZERO) GO TO 9
          FW = ZERO
          DO 700 J = 1, FS(JM)
            FW = GY(J) + FW
```

```
 700      CONTINUE
   9 CONTINUE

     IF(NF .LE. 0) GO TO 10
        APAST(0, 1) = FW
        JMPAST(1) = JM
        DO 800 JV = 1, NV
           APAST(JV, 1) = BV(JV)
 800      CONTINUE
  10 CONTINUE

     JFEVAL = JFEVAL + 1
  11 CONTINUE

     DO 900 J = 1, 49
        K = 51 - J
        SAME(K) = SAME(K-1)
 900 CONTINUE

     SAME(1) = FW
     FSAME = .TRUE.
     DO 1000 J = 2, 50
        FSAME = FSAME .AND. (SAME(J) .EQ. SAME(1))
1000 CONTINUE

     IF((MOD(JFEVAL, 101) .NE. 1 .OR. DEBUGG .LT. 100) .AND.    %%%%%%
    &   (.NOT. FSAME              .OR. .NOT. FOPT     )) GO TO 9999 %%%

        IF(FCLOSE) CALL EOFILE(PRINTR)
        CALL SKIPOT(PRINTR, 1)
        WRITE(PRINTR, /) 'FW status report #', JFEVAL / 101
        WRITE(PRINTR, *//) JFEVAL, JM, EDEP, ENC, ENOIND, EOB, FOPT
        WRITE(PRINTR, *//) NSIG , N, GX, HSTEP, GY, XEND, TOL
        WRITE(PRINTR, *//) METH, MITER, INDEX
        WRITE(PRINTR, *//) FW, IER
        WRITE(PRINTR, /) 'EB', (EB(J), J = 1, NP(JM))
        WRITE(PRINTR, /) 'CHOOSE', (CHOOSE(J), J = 1, NW)
        WRITE(PRINTR, /) 'BV', (BV(J), J = 1, NV)
        WRITE(PRINTR, /) 'EX(EOB, *)', (EX(EOB, J), J = 1, ENOIND)
        WRITE(PRINTR, /) 'AY(EDEP, EOB)', AY(EDEP, EOB)

        IF(MOD(JFEVAL, 101) .NE. 1) GO TO 12
           CALL SKIPOT(PRINTR, 1)
           CALL TIMER(PTHR, PTMIN, PTSEC, 2)
           CALL TIMER(IOHR, IOMIN, IOSEC, 3)
           WRITE(PRINTR, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
           CALL SKIPOT(PRINTR, 1)
  12    CONTINUE
        IF(FCLOSE) LOCK PRINTR

        IF(.NOT. FSAME) GO TO 14
           IF(FCLOSE) CALL EOFILE(PRINTR)
           WRITE(PRINTR, /  ) 'Infinite loop in MDPE/FW'
           WRITE(PRINTR, *//) SAME
```

```
           IF(NF .EQ. 0) GO TO 13
              DO 1100 JF = 1, NF
                 WRITE(PRINTR, /) 'JF', JF, 'JMPAST(JF)', JMPAST(JF),
     &           'APAST(*, JF)', (APAST(JV, JF), JV = 0, NV)
 1100         CONTINUE
   13      CONTINUE
           CLOSE(PRINTR, DISP = CRUNCH)
           STOP
   14   CONTINUE
 9999 CONTINUE
C     IF(MOD(JFEVAL, 83) .NE. 82) GO TO 15
C        IF(FCLOSE) CALL EOFILE(PRINTR)
C        CALL CHEKPT
C        WRITE(PRINTR, /) 'CHECKPOINT #', JFEVAL/83 + 1, 'TAKEN'
C        IF(FCLOSE) LOCK PRINTR
   15 CONTINUE

      RETURN
      END    % FW
```

```
#FILE (MARK)MDPE/COMMON ON STUDENTS
$   RESET LIST

      DOUBLE PRECISION
     & AAX    (70,25,15) % sensitivities (MR x MP x MM)
     &, AP    (25, 15)   % parameters (MP x MM)
     &, APAST ( 7, 99)   % prior fcn calls(MV x MF)
     &, APR   (15, 0:70) % probabilities (MM x 0:MR)
     &, ASIMP (26, 26)   % SIMPLX work array(MP+1 x MP+1)
     &, AV    ( 7, 70)   % variables (MV x MR)
     &, AX    (70, 25)   % sensitivities (MR x MP)
     &, AXTX  (25, 25)   % X'X (MP x MP)
     &, AY    ( 8, 70)   % experimental data (MW x MR)
     &, BX    (25,  1)   % sensitivities (MP x 1)
     &, BXT   ( 1, 25)   % BX' (1 x MP)
     &, BXT2  ( 1, 25)   % BX' [inverse(X'X)] (1 x MP)
     &, B11   ( 1,  1)   % BX' [inverse(X'X)] BX (1 x 1)
C
     &, BE    (15)       % E criterion (MM)
     &, BEMAX (15)       % max E criterion (MM)
     &, BP    (25)       % parameters (MP)
     &, BPR   (15)       % probabilities (MM)
     &, BSSM  (15)       % model variance (MM)
     &, BSSW  ( 8)       % response variance (MM)
     &, BTRIAL( 5)       % Fixed temperature CMAX temperatures (MT)
     &, BV    ( 7)       % variables (MV)
     &, BVDEF ( 7)       % default values for unoptimized vars. (MV)
     &, BVHI  ( 7)       % upper bounds for variables (MV)
     &, BVLO  ( 7)       % lower bounds for variables (MV)
     &, BVMAX ( 7)       % display vector for F(C, D, & E)MAX (MV)
     &, BWK1  (25)       % work vector for LINV3F (MP)
     &, BWK2  (50)       % work vector for LINV3F (2*MP)
     &, BXTX  (625)      % X'X (MP*MP)
     &, BY    (15)       % work vector for FD (MM)
     &, BZA   ( 7)       % lower bounds for ZXMWD (MV)
     &, BZB   ( 7)       % upper bounds for ZXMWD (MV)
     &, BZX   ( 7)       % variables for ZXMWD (MV)
     &, BZWORK(105)      % work for ZXMWD (MV*(MV + 1)/2 + 11*MV)
C
     &, BIG              % largest positive d.p.#
     &, CMAX             % max C criterion
     &, DMAX             % max D criterion
     &, EPS              % smallest d.p.# e > 0 such that 1-e < 1 < 1+e
     &, H                % initial step size for DRVTE
     &, LAMBDA           % power factor for C calculation
     &, PI               % circle circumference / circle diameter
     &, SMALL            % smallest positive d.p.#
C
      INTEGER
     & FS     (15)       % # fractions which make up reactant (MM)
     &, FT    (15)       % # fractions total (reactant + products) (MM)
     &, IWORK ( 7)       % work for ZXMWD (MV)
     &, JMPAST(99)       % past values of JM (MF)
```

```
     &, LABEL (70)         % labels (MR)
     &, NP    (15)         % actual # parameters (MM)
     &, PM    (15)         % pointer vector for models (MM)
C
     &, DEBUGG             % debug level
     &, DISKIN             % input data disk file
     &, ERRORS             % error message disk file
     &, IHIGH              % largest positive integer
     &, ILOW               % largest negative integer
     &, IOPT               % Hessian options selector for ZXMING
     &, JF                 % current prior function being searched
     &, JFEVAL             % # function evaluations so far
     &, JM                 % current model
     &, JP                 % current parameter
     &, JR                 % current run
     &, JT                 % current(fixed temp CMAX) trial temp
     &, JV                 % current variable
     &, JW                 % current response
     &, METH               % DGEAR integration method
     &, MF                 % max # prior fcns to be searched for a match
     &, MITER              % DGEAR iteration method
     &, MM                 % max # models
     &, MP                 % max # parameters
     &, MR                 % max # runs
     &, MT                 % max # trial temps
     &, MV                 % max # variables
     &, MW                 % max # responses
     &, NF                 % actual # prior fcns searched for a match
     &, NM                 % actual # models
     &, NOPT               % # variables used during optimization
     &, NR                 % actual # runs
     &, NRM1               % NR - 1
     &, NRP1               % NR + 1
     &, NSIG               % # significant figures to print
     &, NSRCH              % # starting points for optmization(ZXMWD)
     &, NT                 % actual # trial temps
     &, NV                 % actual # variables
     &, NW                 % actual # responses
     &, PRINTR             % printer file
     &, REMOTE             % CRT file
     &, SW                 % response to use
     &, ZOPT               % 1 => ZXMWD, 2 => ZXSIMP
C
      LOGICAL
     &  AOPT  (7, 15,  8)% True => Optimize (MV x MM x MW)
     &, BEOLD (15)         % True => use EMAX info from prior runs (MM)
     &, BOLD  (15)         % True => generate AAX(JM) from scratch (MM)
     &, BOPT  ( 7)         % True => Optimize using this variable (MV)
     &, DOC                % True => do combined criteria optimization
     &, DOD                % True => do model discrimination optimization
     &, FCLOSE             % True => close & reopen results file
     &, FIRST              % True => generate APR from scratch
     &, FOPT               % True => optimization in progress
     &, SECOND             % True => do(fixed temp CMAX) trials
```

```
C
      COMMON/CMDPE1/
     &  AAX   %(70,25,15) % sensitivities (MR x MP x MM)
C
      COMMON/CMDPE2/
     &  AP    %(25, 15)   % parameters (MP x MM)
     &, APAST%( 7, 99)    % prior fcn calls(MV x MF)
     &, APR   %(15, 0:70) % probabilities (MM x 0:MR)
     &, ASIMP%(26, 26)    % SIMPLX work array(MP+1 x MP+1)
     &, AV    %( 7, 70)   % variables (MV x MR)
C
      COMMON/CMDPE3/
     &  AX    %(70, 25)   % sensitivities (MR x MP)
     &, AXTX  %(25, 25)   % X'X (MP x MP)
     &, AY    %( 8, 70)   % experimental data (MW x MR)
     &, BX    %(25,  1)   % sensitivities (MP x 1)
     &, BXT   %( 1, 25)   % BX' (1 x MP)
     &, BXT2  %( 1, 25)   % BX' [inverse(X'X)] (1 x MP)
     &, B11   %( 1,  1)   % BX' [inverse(X'X)] BX (1 x 1)
C
     &, BE    %(15)       % E criterion (MM)
     &, BEMAX%(15)        % max E criterion (MM)
     &, BP    %(25)       % parameters (MP)
     &, BPR     %(15)     % probabilities (MM)
     &, BSSM    %(15)     % model variance (MM)
     &, BSSW    %( 8)     % response variance (MM)
     &, BTRIAL  %( 5)     % Fixed temperature CMAX temperatures (MT)
     &, BV      %( 7)     % variables (MV)
     &, BVDEF   %( 7)     % default values for unoptimized vars. (MV)
     &, BVHI    %( 7)     % upper bounds for variables (MV)
     &, BVLO    %( 7)     % lower bounds for variables (MV)
     &, BVMAX   %( 7)     % display vector for F(C, D, & E)MAX (MV)
     &, BWK1    %(25)     % work vector for LINV3F (MP)
     &, BWK2    %(50)     % work vector for LINV3F (2*MP)
     &, BXTX    %(625)    % X'X (MP*MP)
     &, BY      %(15)     % work vector for FD (MM)
     &, BZA     %( 7)     % lower bounds for ZXMWD (MV)
     &, BZB     %( 7)     % upper bounds for ZXMWD (MV)
     &, BZX     %( 7)     % variables for ZXMWD (MV)
     &, BZWORK %(105)     % work for ZXMWD (MV*(MV + 1)/2 + 11*MV)
C
      COMMON/CMDPE4/
     &  CMAX              % max C criterion
     &, DMAX              % max D criterion
     &, H                 % initial step size for DRVTE
     &, LAMBDA            % power factor for C calculation
C
     &, FS    %(15)       % # fractions which make up reactant (MM)
     &, FT    %(15)       % # fractions total (reactant + products) (MM)
     &, IWORK %( 7)       % work for ZXMWD (MV)
     &, JMPAST % (99)     % past values of JM (MF)
     &, LABEL %(70)       % labels (MR)
     &, NP      %(15)     % actual # parameters (MM)
     &, PM    %(15)       % pointer vector for models (MM)
```

```
C
      &, DEBUGG            % debug level
      &, DISKIN            % input data disk file
      &, ERRORS            % error message disk file
      &, IOPT              % Hessian options selector for ZXMING
      &, JF                % current prior function being searched
      &, JFEVAL            % # function evaluations so far
      &, JM                % current model
      &, JP                % current parameter
      &, JR                % current run
      &, JT                % current(fixed temp CMAX) trial temp
      &, JV                % current variable
      &, JW                % current response
      &, METH              % DGEAR integration method
      &, MF                % max # prior fcns to be searched for a match
      &, MITER             % DGEAR iteration method
      &, MM                % max # models
      &, MP                % max # parameters
      &, MR                % max # runs
      &, MT                % max # trial temps
      &, MV                % max # variables
      &, MW                % max # responses
      &, NF                % actual # prior fcns searched for a match
      &, NM                % actual # models
      &, NOPT              % # varables used during optimization
      &, NR                % actual # runs
      &, NRM1              % NR - 1
      &, NRP1              % NR + 1
      &, NSIG              % # significant figures to print
      &, NSRCH             % # starting points for optmization(ZXMWD)
      &, NT                % actual # trial temps
      &, NV                % actual # variables
      &, NW                % actual # responses
      &, PRINTR            % printer file
      &, REMOTE            % CRT file
      &, SW                % response to use
      &, ZOPT              % 1 => ZXMWD, 2 => ZXSIMP
C
      &, AOPT %(7, 15,  8)% True => Optimize (MV x MM x MW)
      &, BEOLD%(15)        % True => use EMAX info from prior runs (MM)
      &, BOLD %(15)        % True => generate AAX(JM) from scratch (MM)
      &, BOPT %( 7)        % True => Optimize using this variable (MV)
      &, DOC               % True => do combined criteria optimization
      &, DOD               % True => do model discrimination optimization
      &, FIRST             % True => generate APR from scratch
      &, FCLOSE            % True => close & reopen results file
      &, FOPT              % True => optimization in progress
      &, SECOND            % True => do(fixed temp CMAX) trials
C
       COMMON/CCONST/
      &  BIG               % largest positive d.p.#
      &, EPS               % smallest d.p.# e > 0 such that 1-e < 1 < 1+e
      &, PI                % circle circumference / circle diameter
      &, SMALL             % smallest positive d.p.#
```

```
C
     &, IHIGH              % largest positive integer
     &, ILOW               % largest negative integer

      MF = 99    % Max # prior function calls to be searched for a match
      MM = 15    % Max # models
      MP = 25    % Max # parameters per model
      MR = 70    % Max # runs(sets of observations)
      MT =  5    % Max # fixed temp CMAX trials
      MV =  7    % Max # variables per run(independent variables)
      MW =  8    % Max # responses per run(dependent variables)

$     POP LIST
```

```
#FILE (MARK)UTIL/DBLE/FPENAL ON STUDENTS
C----------------------------------------------------------------------

C   Util routine name   - FPENAL

C   Creation date       -  2 Nov 84
C   Latest revision     -  2 Nov 84

C   Purpose             - Penalty function for optimization methods.

C   Usage               - P = FPENAL(N, X, A, B)

C   Arguments     N     - The number of parameters. (INPUT)
C                 X     - Parameter vector of length N. (INPUT)
C                 A, B  - Boundary vectors of length N. (INPUT)
C                           A(j) <= X(j) <= B(j)
C                 FPENAL - Penalty for exceeding bounds. (OUTPUT)

C   Req.d UTIL routines - None

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION FPENAL(N     , X     , A     , B     )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT
     &  LOGICAL(A - Z)

      DOUBLE PRECISION
     &  X     , A     , B

      INTEGER
     &  N
     &, J

      DIMENSION
     &  X(N)   , A(N)   , B(N)

C----------------------------------------------------------------------

      FPENAL = 0D0
      DO 100 J = 1, N
         IF(X(J) .LT. A(J)) FPENAL =
     &     FPENAL + ((A(J) - X(J)) / (B(J) - A(J))) ** 2
         IF(X(J) .GT. B(J)) FPENAL =
     &     FPENAL + ((X(J) - B(J)) / (B(J) - A(J))) ** 2
  100 CONTINUE

      END    % FPENAL
```

```
#FILE (MARK)UTIL/DBLE/SIMPLX ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name   - SIMPLX

C   Creation date       -  9 Aug 84
C   Latest revision     - 27 Nov 84

C   Author              - Mark A. Burazin

C   Purpose             - Local minimum(with constraints) of a
C                           function of N variables.

C   Usage               - CALL SIMPLX(FCN, N, NSIG, MAXNIT, OUT, IR
C                           A, B, X, F, WORK)

C   Arguments     FCN    - A USER SUPPLIED SUBROUTINE WHICH CALCULATES
C                           F GIVEN X(1),X(2),...,X(N).
C                           FCN IS REFERENCED AS FOLLOWS,
C                             CALL FCN(N,X,F)
C                             WHERE X IS A VECTOR OF LENGTH N
C                             FCN MUST APPEAR IN AN EXTERNAL STATEMENT
C                               IN THE CALLING PROGRAM.
C                             FCN MUST NOT ALTER THE VALUES OF
C                               X(I),I=1,...,N,  OR N.
C                 N      - The number of unknown parameters. (INPUT)
C                           N <= IR-ONE
C                 NSIG   - CONVERGENCE CRITERION. (INPUT)  NSIG IS THE
C                           NUMBER OF DIGITS OF ACCURACY REQUIRED IN
C                           THE PARAMETER ESTIMATES.
C                 MAXNIT - Maximum # of iterations allowed. (INPUT)
C                 OUT    - Desired unit device for output. (INPUT)
C                 IR     - Row dimension of WORK matrix exactly as
C                           dimensioned in the calling program. (INPUT)
C                           IR >= N+1
C                 A,B    - Constraint vectors of length IR. (INPUT)
C                           A(I) <= X(I) <= B(I), I = 1, N
C                 X      - Parameter estimate vector of length IR.
C                           On input, X contains initial guesses.
C                           On output, X contains the final estimates.
C                 F      - Value of the function at the final parameter
C                           estimates. (INPUT)
C                 WORK   - IR by IR real work matrix.

C   Req.d UTIL routines - FBOUND, SKIP, SKIPOT

C   Reference           - Article in Byte...

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE SIMPLX(FCN    , N      , NSIG   , MAXNIT, OUT    , IR
     &                 , A      , B      , X      , F     , SIMP   )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT
     &  LOGICAL(A - Z)
```

```
      DOUBLE PRECISION
     &  A      , B      , X      , F      , SIMP

      INTEGER
     &  N      , NSIG   , MAXNIT, OUT    , IR

      DOUBLE PRECISION
     &  WORK2
     &, NEXT   , CENTER, MEAN   , ERROR , PP     , QQ
     &, H      , ROOT2 , RTNP1  , MAXERR, STEP   , ALFA   , BETA   , GAMMA
     &, ZERO   , ONE    , TWO

      INTEGER
     &  BEST   , NIT    , NMAX   , NP1    , P      , V      , WORST

      DIMENSION
     &  A(1)   , B(1)   , X(1)   , SIMP(IR, IR)
     &, NEXT(20)        , CENTER(20)      , ERROR(21)       , PP(20), QQ(20)
     &, STEP(20)        , BEST  (21)      , WORST(21)

      DATA
     &  H       /.125D0/
     &, NMAX    /  20D0/
     &, ALFA    /   1D0/
     &, BETA    /  .5D0/
     &, GAMMA   /   2D0/
     &, ZERO    /   0D0/
     &, ONE     /   1D0/
     &, TWO     /   2D0/

 9001 FORMAT(' ***  Fatal error in UTIL routine ZXSIMP'/
     &       ' ***  N (',I3,') > NMAX (',I3,')'/
     &       ' ***  Program Halted')

C-----Make certain N <= current vector dimensions(NMAX)-----------%%%%%
C-----NMAX = *** 20 *** as of 20 Nov 84---------------------------%%%%%

      IF(N .LE. NMAX) GO TO 1
        CALL SKIPOT(OUT, 3)
        WRITE(OUT, 9001) N, NMAX
        CLOSE(OUT, DISP = CRUNCH)
        STOP
    1 CONTINUE

C-----Initial housekeeping

      MAXERR = 10D0 ** (-NSIG)
      NIT    = -1
      NP1    = N + 1
      ROOT2  = DSQRT(TWO)
      RTNP1  = DSQRT(DFLOAT(NP1))
      DO 100 P = 1, N
         SIMP(P, 1) = X(P)
         STEP(P) = DMIN1(B(P) - X(P), X(P) - A(P)) * H
```

```
  100 CONTINUE

      DO 110 P = 1, NP1
         BEST (P) = 1
         WORST(P) = 1
  110 CONTINUE


C-----Create NP1 verticies

      CALL FCN(N, SIMP(1, 1), SIMP(NP1, 1))

      DO 200 P = 1, N
         PP(P) = STEP(P) * (RTNP1 + N - 1) / (N * ROOT2)
         QQ(P) = STEP(P) * (RTNP1 - 1) / (N * ROOT2)
  200 CONTINUE

      DO 220 V = 2, NP1
         DO 210 P = 1, N
            SIMP(P, V) = SIMP(P, 1) + QQ(P)
  210    CONTINUE
         SIMP(V-1, V) = SIMP(V-1, 1) + PP(V-1)
         CALL FCN(N, SIMP(1, V), SIMP(NP1, V))
  220 CONTINUE

C-----Start of iteration loop------------------------------------------

    2 CONTINUE
      DONE = .TRUE.
      NIT  = NIT + 1

C-----Rank verticies by parameter value(1-N) and response(NP1)

      DO 300 V = 1, NP1
         DO 300 P = 1, NP1
            IF(SIMP(P, V) .LT. SIMP(P, BEST (P))) BEST (P) = V
            IF(SIMP(P, V) .GT. SIMP(P, WORST(P))) WORST(P) = V
  300 CONTINUE

C-----Convergence check

      DO 400 P = 1, NP1
         IF(.NOT. DONE) GO TO 400
           ERROR(P) = (SIMP(P, WORST(P)) - SIMP(P, BEST (P)))
     &               / DMAX1(DABS(SIMP(P, WORST(P))), ONE)
           IF(ERROR(P) .GT. MAXERR) DONE = .FALSE.
  400 CONTINUE

C-----Exit loop here on convergence or completion of alloted loops-----

      IF(DONE .OR. NIT .GT. MAXNIT) GO TO 9

C-----Centroid of simplex(excluding worst)

      DO 500 P = 1, N
```

```
        CENTER(P) = ZERO
  500 CONTINUE

      DO 520 V = 1, NP1
         IF(V .EQ. WORST(NP1)) GO TO 520
         DO 510 P = 1, N
            CENTER(P) = CENTER(P) + SIMP(P, V)
  510    CONTINUE
  520 CONTINUE

C-----'Next' vertex is specular reflection of worst

      DO 600 P = 1, N
         CENTER(P) = CENTER(P) / N
         NEXT(P) = (ONE + ALFA) * CENTER(P)
     &           - ALFA * SIMP(P, WORST(NP1))
  600 CONTINUE

      CALL FCN(N, NEXT(1), NEXT(NP1))

C-----Is reflected point better than best so far?

      IF(NEXT(NP1) .GE. SIMP(NP1, BEST (NP1))) GO TO 3

C-------Reflected point best!  Accept it and expand in same direction

        DO 700 P = 1, NP1
           SIMP(P, WORST(NP1)) = NEXT(P)
  700   CONTINUE

        DO 710 P = 1, N
           NEXT(P) = GAMMA * SIMP(P, WORST(NP1))
     &             + (ONE - GAMMA) * CENTER(P)
  710   CONTINUE

        CALL FCN(N, NEXT(1), NEXT(NP1))

C-------If expanded point best accept it and return to start of loop
C-------else just return to start of loop

        IF(NEXT(NP1) .GE. SIMP(NP1, WORST(NP1))) GO TO 2
          DO 720 P = 1, NP1
             SIMP(P, WORST(NP1)) = NEXT(P)
  720     CONTINUE
          GO TO 2

    3 CONTINUE

C-----Reflected point not best.  Is it better than worst?

      IF(NEXT(NP1) .GT. SIMP(NP1, WORST(NP1))) GO TO 4

C-------Reflected point better than worst.  Accept it & return to
C-------start of loop
```

```
        DO 800 P = 1, NP1
           SIMP(P, WORST(NP1)) = NEXT(P)
  800   CONTINUE
        GO TO 2

    4 CONTINUE

C-----Reflected point worse than worst.  Reject it and contract toward
C-----center away from worst.

      DO 900 P = 1, N
         NEXT(P) = BETA * SIMP(P, WORST(NP1))
     &           + (ONE - BETA) * CENTER(P)
  900 CONTINUE

      CALL FCN(N, NEXT(1), NEXT(NP1))

C-----Is contracted point better than worst?

      IF(NEXT(NP1) .GT. SIMP(NP1, WORST(NP1))) GO TO 5

C-------Contracted point better than worst.  Accept it and return to
C-------start of loop.

        DO 1000 P = 1, NP1
           SIMP(P, WORST(NP1)) = NEXT(P)
 1000   CONTINUE
        GO TO 2

    5 CONTINUE

C-----Contracted point worse than worst.  Reject it, shrink all but the
C-----best vertex toward the best vertex, and return to loop start.

      DO 1020 V = 1, NP1
         DO 1010 P = 1, N
            SIMP(P, V) = SIMP(P, V) * BETA
     &              + SIMP(P, BEST (NP1)) * (ONE - BETA)
 1010    CONTINUE
         CALL FCN(N, SIMP(1, V), SIMP(NP1, V))
 1020 CONTINUE
      GO TO 2

    9 CONTINUE

C-----Final housekeeping---------------------------------------------

      DO 1100 P = 1, N
         X(P) = SIMP(P, BEST (NP1))
 1100 CONTINUE

      F = SIMP(NP1, BEST (NP1))

C     WRITE(OUT, /) 'NIT', NIT

      RETURN
      END   % SIMPLX
```

```
#FILE (MARK)UTIL/DBLE/ZXMING ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name   - ZXMING

C   Creation date       - 31 Oct 84
C   Latest revision     - 27 Nov 84

C   Author              - Mark A. Burazin

C   Purpose             - Global minimum(with constraints) of a
C                           function of N variables

C   Usage               - CALL ZXMING(FCN, N, NSIG, NSRCH, OUT, IOPT,
C                           A, B, X, F, WORK, M    )

C   Arguments     FCN   - A USER SUPPLIED SUBROUTINE WHICH CALCULATES
C                           F GIVEN X(1),X(2),...,X(N).
C                           FCN IS REFERENCED AS FOLLOWS,
C                             CALL FCN(N,X,F)
C                           WHERE X IS A VECTOR OF LENGTH N.
C                           FCN MUST APPEAR IN AN EXTERNAL STATEMENT
C                           IN THE CALLING PROGRAM.
C                           FCN MUST NOT ALTER THE VALUES OF
C                             X(I),I=1,...,N,  OR N.
C                 N     - The number of unknown parameters. (INPUT)
C                 NSIG  - CONVERGENCE CRITERION. (INPUT)  NSIG IS THE
C                           NUMBER OF DIGITS OF ACCURACY REQUIRED IN
C                           THE PARAMETER ESTIMATES.
C                 NSRCH - Number of starting points to be generated.
C                           (INPUT) Suggested value = MIN(2**N+5,100)
C                 OUT   - Desired unit device for output. (INPUT)
C                 IOPT  - Options selector. (INPUT)
C                           IOPT = 0 causes ZXMIN to initialize Hessian
C                             matrix H to Identity matrix.
C                           IOPT = 1 indicates H has been initialized
C                             by user to positive definite matrix.
C                           IOPT = 2 causes ZXMIN to compute diagonal
C                             values of Hessian and set H to diagonal
C                             matrix containing these values.
C                           IOPT = 3 cause ZXMIN to compute estimate
C                             of Hessian in H.
C                 A,B   - CONSTRAINT VECTORS OF LENGTH N. (INPUT)
C                           X(I) IS REQUIRED TO SATISFY:
C                                A(I) .LE. X(I) .LE. B(I)
C                 X     - Parameter estimate vector of length N.
C                           On input, X contains initial guesses.
C                           On output, X contains the final estimates.
C                 F     - VALUE OF THE FUNCTION AT THE FINAL
C                           PARAMETER ESTIMATES. (OUTPUT)
C                 WORK  - A vector of length N*(N+1)/2 + 5*N used as
C                          working space.  On output, WORK(I), contains
C                          for
```

```
C                              I = 1, the 2-norm of the gradient.
C                              I = 2, the number of calls to FCN.
C                              I = 3, an estimate of the significant
C                                digits in the final parameter estimates.
C                              I = 3*N+1 through 4*N, an estimate of the
C                                gradient dF/dX(I), I=1,...,N at the final
C                                parameter estimates.
C                              I = 4*N+1 through N*(N+1)/2 + 4*N, an
C                                estimate of the Hessian at the final
C                                parameter estimates. This is also where
C                                the Hessian can be initialized on input
C                                (IOPT = 1).
C                 M       - Integer work vector of length N.

C   Routines req'd        - UERTST, UGETIO, ZXMIN, ZXMJN

C   Reference             - IMSL documentation of ZXMIN & ZXMWD...

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE ZXMING(FCN    , N      , NSIG   , NSRCH  , OUT    , IOPT
     &                        , A      , B      , X      , F      , WORK   , M      )
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT
     &  LOGICAL(A - Z)

      DOUBLE PRECISION
     &  A      , B      , X      , F      , WORK

      INTEGER
     &  N      , NSIG   , NSRCH  , OUT    , IOPT   , M

      REAL
     &  FCN

      DOUBLE PRECISION
     &  BIG    , FBEST

      INTEGER
     &  IER    , IP     , IW     , NLONG  , NM1    , NP1    , NSHORT
     &, P      , PG     , PH     , PXBEST, PW     , S

      DIMENSION
     &  A(1), B(1), X(1), WORK(1), M(1)
     &, IW(9)

      EXTERNAL
     &  FCN

      DATA
     &  BIG    /1.94882838205028079124 4D+29603/
     &, NLONG /     200/
     &, NSHORT/       4/
```

```
C-----Initial housekeeping---------------------------------------------

      IP     = 0                % First call flag for ZSRCH
      FBEST  = BIG
      NM1    = N - 1
      NP1    = N + 1
      PW     = 1                % Pointers
      PG     = PW + 3*N         % to vectors
      PH     = PG + N           % used by
      PXBEST = PH + N*(NP1)/2   % ZSRCH

      DO 100 P = PXBEST, PXBEST+NM1
         WORK(P) = BIG
  100 CONTINUE

C-----Test NSRCH starting points with NSHORT*NP1 iterations of SIMPLX

      DO 210 S = 1, NSRCH

         CALL ZSRCH (A, B, N, NSRCH, IP, X, M, IW, IER)

         CALL ZXMIN (FCN, N, NSIG, NSHORT*NP1, IOPT, X, WORK(PH)
     &               , WORK(PG), F, WORK(PW), IER)

         IF(F .GE. FBEST) GO TO 210
           DO 200 P = 1, N
              WORK(P+PXBEST-1) = X(P)
  200      CONTINUE
           FBEST = F
  210 CONTINUE

C     CALL SKIPOT(OUT, 1)
C     WRITE(OUT, /) 'FBEST', FBEST, 'XBEST', (XBEST(P), P = 1, N)
C     CALL SKIPOT(OUT, 1)

C-----Optimize best point for NLONG*NP1 iterations of SIMPLX

      DO 300 P = 1, N
         X(P) = WORK(P+PXBEST-1)
  300 CONTINUE

      CALL ZXMIN (FCN, N, NSIG, NLONG*NP1, IOPT, X, WORK(PH), WORK(PG)
     &            , F, WORK(PW), IER)

      END   % ZXSIMP
```

```
#FILE (MARK)UTIL/DBLE/ZXSIMP ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name   - ZXSIMP

C   Creation date       - 10 Aug 84
C   Latest revision     - 27 Nov 84

C   Author              - Mark A. Burazin

C   Purpose             - Global minimum(with constraints) of a
C                           function of N variables

C   Usage               - CALL ZXSIMP(FCN, N, NSIG, NSRCH, OUT, IR,
C                           A, B, X, F, WORK)

C   Arguments    FCN    - A USER SUPPLIED SUBROUTINE WHICH CALCULATES
C                           F GIVEN X(1),X(2),...,X(N).
C                           FCN IS REFERENCED AS FOLLOWS,
C                             CALL FCN(N,X,F)
C                           WHERE X IS A VECTOR OF LENGTH N.
C                           FCN MUST APPEAR IN AN EXTERNAL STATEMENT
C                           IN THE CALLING PROGRAM.
C                           FCN MUST NOT ALTER THE VALUES OF
C                             X(I),I=1,...,N,  OR N.
C                N      - The number of unknown parameters. (INPUT)
C                           N must be <= IR-1.
C                NSIG   - CONVERGENCE CRITERION. (INPUT)  NSIG IS THE
C                           NUMBER OF DIGITS OF ACCURACY REQUIRED IN
C                           THE PARAMETER ESTIMATES.
C                NSRCH  - Number of starting points to be generated.
C                           (INPUT) Suggested value = MIN(2**N+5,100)
C                OUT    - Desired unit device for output. (INPUT)
C                IR     - Row dimension of work matrix exactly as
C                           declared in the calling program. (INPUT)
C                           IR must be >= N+1.
C                A,B    - CONSTRAINT VECTORS OF LENGTH IR. (INPUT)
C                           X(I) IS REQUIRED TO SATISFY:
C                                A(I) .LE. X(I) .LE. B(I)
C                X      - Parameter estimate vector of length IR.
C                           On input, X contains initial guesses.
C                           On output, X contains the final estimates.
C                F      - VALUE OF THE FUNCTION AT THE FINAL
C                           PARAMETER ESTIMATES. (OUTPUT)
C                WORK   - IR by IR real work matrix.

C   Routines req'd      - SIMPLX, SKIP, SKIPOT

C   Reference           - IMSL documentation of ZXMWD...

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE ZXSIMP(FCN    , N      , NSIG   , NSRCH  , OUT    , IR
     &                , A      , B      , X      , F      , WORK   )
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT
     &  LOGICAL(A - Z)

      DOUBLE PRECISION
     &  A       , B       , X       , F       , WORK

      INTEGER
     &  N       , NSIG    , NSRCH   , OUT     , IR

      REAL
     &  FCN

      DOUBLE PRECISION
     &  BIG     , FBEST   , XBEST

      INTEGER
     &  IER     , IP      , IW      , M       , NLONG   , NMAX    , NP1     , NSHORT
     &, P       , S

      DIMENSION
     &  A(1)    , B(1)    , X(1)    , WORK(IR, IR)
     &, XBEST(20)         , IW(20)
     &, M(20) % Dimension(M) must = NMAX from calling program      %%%%%

      EXTERNAL
     &  FCN

      DATA
     &  BIG    /1.9488283820502807912440+29603/
     &, NLONG  /200/
     &, NMAX   / 20/
     &, NSHORT/  4/

 9001 FORMAT(' ***  Fatal error in UTIL routine ZXSIMP'/
     &       ' ***  N (',I3,') > NMAX (',I3,')'/
     &       ' ***  Program Halted')

C-----Make certain N <= current vector dimensions(NMAX)-----------%%%%%
C-----NMAX = *** 20 *** as of 27 Oct 84--------------------------%%%%%

      IF(N .LE. NMAX) GO TO 1
        CALL SKIPOT(OUT, 3)
        WRITE(OUT, 9001) N, NMAX
        CLOSE(OUT, DISP = CRUNCH)
        STOP
    1 CONTINUE

C-----Initial housekeeping

      IP = 0  % First call flag for ZSRCH
      FBEST = BIG
      NP1 = N + 1
```

```
      DO 100 P = 1, N
         XBEST(P) = BIG
  100 CONTINUE

C-----Test NSRCH starting points with NSHORT*NP1 iterations of SIMPLX

      DO 210 S = 1, NSRCH

         CALL ZSRCH (A      , B      , N      , NSRCH  , IP     , X
     &              , M      , IW     , IER    )

         CALL SIMPLX(FCN    , N      , NSIG   , NSHORT*NP1      , OUT
     &              , IR     , A      , B      , X      , F      , WORK   )

         IF(F .GE. FBEST) GO TO 210
           DO 200 P = 1, N
              XBEST(P) = X(P)
  200      CONTINUE
           FBEST = F
  210 CONTINUE

C     CALL SKIPOT(OUT, 1)
C     WRITE(OUT, /) 'FBEST', FBEST, 'XBEST', (XBEST(P), P = 1, N)
C     CALL SKIPOT(OUT, 1)

C-----Optimize best point for NLONG*NP1 iterations of SIMPLX

      DO 300 P = 1, N
         X(P) = XBEST(P)
  300 CONTINUE

      CALL SIMPLX(FCN    , N      , NSIG   , NLONG*NP1       , OUT
     &           , IR     , A      , B      , X      , F      , WORK   )

      END   % ZXSIMP
```

```
#FILE (MARK)MRPEST/FCN ON STUDENTS
$      INCLUDE "UTIL/DBLE/MESSAG"

$      SET OWN

       SUBROUTINE FCN    (
     &  N      , X      , Y      , YPRIME)

       IMPLICIT
     &  LOGICAL(A - Z)

C$      INCLUDE "MRPEST/COMMON"        % When running MDPE           %%%%%
 $      INCLUDE "NONLINWOOD/COMMON"    % When running NONLINWOOD     %%%%%
C$      INCLUDE "NONLINWOOD/SUMMARY/COMMON"   % When running SUMMARY %%%

       INTEGER
     &  C      , J      , N      , S      , P

       DOUBLE PRECISION
     &  X      , Y(N)   , YPRIME(N)
     &, TIME   , TEMP   , OH     , SH     , AQ     , T2TEMP, B(20)     %%%%%
     &, ONE    , RTEMP  , SQRTT  , SQRTOH, SQRTSH, SQRTAQ, TEN    , TEQUIL
     &, K1     , K2     , K3     , K4     , K5     , K6     , K7     , K8
     &, P1     , P2     , P3     , P4     , P5     , P6     , P7     , P8
     &, P9     , P10    , P11    , P12    , P13    , P14    , P15    , P16
     &, P17    , P18    , P19    , P20    , Y1PY2
     &, Y1     , Y2     , Y3     , ALPHA , BETA   , GAMMA , LASTPK, SQRTY3
     &, DWELL , LM     , PKW    , Q      , SPLINE, TAU    , LASTTP, Y3ORD

C-----------------------------------------------------------------------

       ONE = 1D0
       TEN = 1D1

C-----Reaction Kinetics Study
C-----Extract common variables

       TIME   = X
       TEQUIL = EX(EOB, 2) % T(e)
       OH     = EX(EOB, 3)
       SH     = EX(EOB, 4)
       AQ     = EX(EOB, 5)
       T2TEMP = EX(EOB, 6)
       SQRTOH = DSQRT(OH)
       SQRTSH = DSQRT(SH)
       SQRTAQ = DSQRT(AQ)

C-----Modification to Tau to force correct time to (Temp-1oC) [20Jun84]

C-----Temp is a special case because of the rapid heatup schedules.
C-----A good approximation to Temp vs. time is a dwell time plus
C-----first order response model.  This gives:
```

```
C-----T(t) = T(e) + (300-T(e)) exp[(dwell-t) / Tau]; t > dwell,
C-----T(t) = 300; t <= dwell, and
C-----Tau = (t2Temp-dwell) / ln[(T(e)-300) / (T(e) - T(e)-1)],

C-----where T(t) = Temp at time t, T(e) = equilibrium temperature,
C-----T [=] deg. K, dwell, t, t2Temp [=] hr. <dwell = 2min>

      DWELL = 2D0 / 60D0

C     COUNT = COUNT + 1
C     IF(TEQUIL .GT. 300D0) GO TO 91
C       CALL SKIPOT(7, 3)
C       WRITE(7, *//) COUNT, TIME, TEQUIL, OH, SH, AQ, T2TEMP
C       WRITE(7, *//) CHOOSE, ENC, ENODEP, ENOIND, EOB
C       WRITE(7, *//) POINTR, POINTY, N, X, Y, YPRIME
C       WRITE(7, *//) ' JUST BEFORE TAU CALCULATED', TAU
C  91 CONTINUE

      TAU = (T2TEMP - DWELL) / DLOG(TEQUIL - 300D0)

C     IF(TEQUIL .GT. 300D0) GO TO 92
C       CALL SKIPOT(7, 3)
C       WRITE(7, *//) ' JUST AFTER TAU CALCULATED', TAU
C  92 CONTINUE

      IF(TIME .LE. DWELL) TEMP = 300D0
      IF(TIME .GT. DWELL) TEMP =
     &  TEQUIL + (300D0 - TEQUIL) * DEXP((DWELL - TIME) / TAU)

C-----Modification to RTEMP a la Hill's papers [18 Jun 84]

      RTEMP  = ONE / TEMP - ONE / 433D0                              %%%%%

      SQRTT  = DSQRT(TEMP)
      DO 100 J = 1, ENC
        B(J) =  EB(J)
  100 CONTINUE

C-----Temporary Debug Printout
C-----This requires $SET OWN to work

C     COUNT = COUNT + 1
C     IF(COUNT .NE. 1) GO TO 9901
C       CALL MESSAG(6, "   FCN", 0, .FALSE.)
C       WRITE(7, *//) TIME, TEMP, OH, SH, AQ, SQRTT, SQRTOH, SQRTSH
C    &  , SQRTAQ
C       WRITE(7, *//) B

C-----Lignin
C-----S is the subsequent (next) parameter

 9901 CONTINUE
      S = POINTR(1)
      C = CHOOSE(1)
```

```
       P = POINTY(1)
       GO TO(  1,    2,    3,    4,    5,    6,    7,    8,    9,   10
     &     ,  11,   12,   13,   14,   15,   16,   17,   18,   19,   20
     &     ,  21,   22,   23,   24,   25,   26,   27,   28,   29,   30
     &     ,  31,   32,   33,   34,   35,   36,   37,   38,   39,   40
     &     ,  41,   42,   43,   44,   45,   46,   47,   48,   49,   50
     &     ,  51,   52,   53,   54,   55,   56,   57,   58,   59,   60
     &     ,  61,   62), C
          IF(C .NE. 0) CALL MESSAG(6, "   FCN", 9901, .TRUE.)
            GO TO 9902

    1     CONTINUE
          GO TO 9902

    2     CONTINUE
          GO TO 9902

    3     CONTINUE
          P1 = B(S)           % k  OH
          P2 = B(S+1) * 1D4 % AE common
C         P3 = B(S+2) / TEN % YO
          P4 = B(S+3)         % k  SH
          P5 = B(S+4)         % k  AQ
          YPRIME(P) = -Y(P) * SQRTT * SQRTOH
     &              * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                 + SQRTSH * DEXP(P4 - P2 * RTEMP)
     &                 + SQRTAQ * DEXP(P5 - P2 * RTEMP))
          GO TO 9902

    4     CONTINUE
          P1 = B(S   )        % k  OH
          P2 = B(S+1) * 1D4 % AE OH
C         P3 = B(S+2) / TEN % YO
          P4 = B(S+3)         % k  SH
          P5 = B(S+4) * 1D4 % AE SH
          P6 = B(S+5)         % k  AQ
          P7 = B(S+6) * 1D4 % AE AQ
          YPRIME(P) = -Y(P) * SQRTT * SQRTOH
     &              * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                 + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &                 + SQRTAQ * DEXP(P6 - P7 * RTEMP))
          GO TO 9902

    5     CONTINUE
          P1 = B(S   )        % k  OH
          P2 = B(S+1) * 1D4 % AE OH
C         P3 = B(S+2) / TEN % YO
          P4 = B(S+3)         % k  SH
          P5 = B(S+4) * 1D4 % AE SH
          P6 = B(S+5)         % k  AQ
          P7 = B(S+6) * 1D4 % AE AQ
          YPRIME(P) = -(Y(P) * Y(P)) * SQRTT * SQRTOH
     &              * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                 + SQRTSH * DEXP(P4 - P5 * RTEMP)
```

```
     &                    + SQRTAQ * DEXP(P6 - P7 * RTEMP))
      GO TO 9902

    6 CONTINUE
      P1  = B(S   )          % k  OH
      P2  = B(S+1)   * 1D4 % AE OH
C     P3  = B(S+2)   / TEN % YO
      P4  = B(S+3)         % k  SH
      P5  = B(S+4)   * 1D4 % AE SH
      P6  = B(S+5)         % k  AQ
      P7  = B(S+6)   * 1D4 % AE AQ
      P8  = B(S+7)         % Ypower
      P9  = B(S+8)         % OHpower
      P10 = B(S+9)         % OH(SH)power
      P11 = B(S+10)        % SHpower
      P12 = B(S+11)        % OH(AQ)power
      P13 = B(S+12)        % AQpower
      YPRIME(P) = -(Y(P)**P8) * SQRTT
     &          * (  OH**P9             * DEXP(P1 - P2 * RTEMP)
     &             + OH**P10 * SH**P11 * DEXP(P4 - P5 * RTEMP)
     &             + OH**P12 * AQ**P13 * DEXP(P6 - P7 * RTEMP))
      GO TO 9902

    7 CONTINUE
      P1  = B(S   )          % k LDOH
      P2  = B(S+1)   * 1D4 % AELDOH
C     P3  = B(S+2)   / TEN % YO
      P4  = B(S+3)         % k LDSH
      P5  = B(S+4)   * 1D4 % AELDSH
      P6  = B(S+5)         % k LDAQ
      P7  = B(S+6)   * 1D4 % AELDAQ
      P8  = B(S+7)         % k   LR
      P9  = B(S+8)   * 1D4 % AE  LR
      P10 = B(S+9)         % k   RD
      P11 = B(S+10)  * 1D4 % AE  RD
      P12 = B(S+11)        % k   DR
      P13 = B(S+12)  * 1D4 % AE  DR
      K1  = SQRTT * SQRTOH                                   % k'LD
     &    *       (  SQRTOH * DEXP(P1  - P2  * RTEMP)
     &             + SQRTSH * DEXP(P4  - P5  * RTEMP)
     &             + SQRTAQ * DEXP(P6  - P7  * RTEMP))
      K2  = SQRTT * SQRTOH * DEXP(P8  - P9  * RTEMP)         % k'LR
      K3  = SQRTT *     OH * DEXP(P10 - P11 * RTEMP)         % k'RD
      K4  = SQRTT          * DEXP(P12 - P13 * RTEMP)         % k'DR
      Y1  = Y(P)                                             % virgin
      Y2  = Y(P+1)                                           % residual
      Y3  = Y(P+2)                                           % dissolved
      YPRIME(P)   = -(K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 + K4 * Y3 - K3 * Y2
      YPRIME(P+2) = K1 * Y1 + K3 * Y2 - K4 * Y3
      GO TO 9902

    8 CONTINUE
      GO TO 9902
```

```
    9      CONTINUE
           GO TO 9902

   10      CONTINUE
           GO TO 9902

   11      CONTINUE
           P1  = B(S  )          % k   QL
           P2  = B(S+1)  * 1D4 % AE  QL
C          P3  = B(S+2)  / TEN % YO
           P4  = B(S+3)        % k   QR
           P5  = B(S+4)  * 1D4 % AE  QR
           P6  = B(S+5)        % k QDOH
           P7  = B(S+6)  * 1D4 % AEQDOH
           P8  = B(S+7)        % k QDSH
           P9  = B(S+8)  * 1D4 % AEQDSH
           P10 = B(S+9)        % k QDAQ
           P11 = B(S+10) * 1D4 % AEQDAQ
           P12 = B(S+11)       % k   LQ
           P13 = B(S+12) * 1D4 % AE  LQ
           P14 = B(S+13)       % k   DR
           P15 = B(S+14) * 1D4 % AE  DR
           P16 = B(S+15)       % k   RD
           P17 = B(S+16) * 1D4 % AE  RD
           P18 = B(S+17)       % pkalig
           K1  = DEXP(P1  - P2  * RTEMP) * SQRTT        % k'QL
           K2  = DEXP(P4  - P5  * RTEMP) * SQRTT        % k'QR
           K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH   % k'QDOH
           K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SH   % k'QDSH
           K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * AQ   % k'QDAQ
           K6  = DEXP(P12 - P13 * RTEMP) * SQRTT        % k'LQ
           K7  = DEXP(P14 - P15 * RTEMP) * SQRTT        % k'DR
           K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH   % k'RD
           Y1  = Y(P)     % virgin lignin
           Y2  = Y(P+1)   % residual lignin
           Y3  = Y(P+2)   % dissolved lignin
           PKW = LASTPK                                      %%%%%%%%%%%
           IF(TEMP .EQ. LASTTP) GO TO 8011                   % This
              PKW    = SPLINE(TEMP, 10, 11)                  % requires
              LASTPK = PKW                                   % $SET OWN
              LASTTP = TEMP                                  % to work
 8011      CONTINUE                                          %%%%%%%%%%%
           LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
           Q   = K6 * LM / (K1 + K2 + K3 + K4 + K5)
           YPRIME(P)   = K1 * Q - K6 * LM
           YPRIME(P+1) = K2 * Q + K7 * Y3 - K8 * Y2
           YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * Y3
           GO TO 9902

   12      CONTINUE
           P1  = B(S  )          % k   QL
           P2  = B(S+1)  * 1D4 % AE  QL
C          P3  = B(S+2)  / TEN % YO
```

```
      P4  = B(S+3)         % k   QR
      P5  = B(S+4)  * 1D4  % AE  QR
      P6  = B(S+5)         % k QDOH
      P7  = B(S+6)  * 1D4  % AEQDOH
      P8  = B(S+7)         % k QDSH
      P9  = B(S+8)  * 1D4  % AEQDSH
      P10 = B(S+9)         % k QDAQ
      P11 = B(S+10) * 1D4  % AEQDAQ
      P12 = B(S+11)        % k   LQ
      P13 = B(S+12) * 1D4  % AE  LQ
      P14 = B(S+13)        % k   DR
      P15 = B(S+14) * 1D4  % AE  DR
      P16 = B(S+15)        % k   RD
      P17 = B(S+16) * 1D4  % AE  RD
      P18 = B(S+17)        % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT                % k'QL
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT                % k'QR
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH           % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SQRTSH       % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * SQRTAQ       % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT                % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT                % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH           % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                          %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8012                       % This
         PKW    = SPLINE(TEMP, 10, 11)                      % requires
         LASTPK = PKW                                       % $SET OWN
         LASTTP = TEMP                                      % to work
8012  CONTINUE                                              %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K2 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K2 * Q + K7 * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * Y3
      GO TO 9902
--------------------------------------------------------------------------------
13    CONTINUE
      P1  = B(S  )         % k   QL
      P2  = B(S+1)  * 1D4  % AE  QL
C     P3  = B(S+2)  / TEN  % Y0
      P4  = B(S+3)         % k   QR
      P5  = B(S+4)  * 1D4  % AE  QR
      P6  = B(S+5)         % k QDOH
      P7  = B(S+6)  * 1D4  % AEQDOH
      P8  = B(S+7)         % k QDSH
      P9  = B(S+8)  * 1D4  % AEQDSH
      P10 = B(S+9)         % k QDAQ
      P11 = B(S+10) * 1D4  % AEQDAQ
      P12 = B(S+11)        % k   LQ
      P13 = B(S+12) * 1D4  % AE  LQ
      P14 = B(S+13)        % k   DR
```

```
      P15 = B(S+14) * 1D4 % AE   DR
      P16 = B(S+15)       % k    RD
      P17 = B(S+16) * 1D4 % AE   RD
      P18 = B(S+17)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT        % k'QL
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT        % k'QR
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH   % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SH   % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * AQ   % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT        % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT        % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH   % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                           %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8013                        % This
         PKW    = SPLINE(TEMP, 10, 11)                       % requires
         LASTPK = PKW                                        % $SET OWN
         LASTTP = TEMP                                       % to work
 8013 CONTINUE                                               %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K2 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K2 * Q + K7 * (Y1 + Y2) * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * (Y1 + Y2)*Y3
      GO TO 9902

   14 CONTINUE
      P1  = B(S   )       % k    QL
      P2  = B(S+1)  * 1D4 % AE   QL
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k    QR
      P5  = B(S+4)  * 1D4 % AE   QR
      P6  = B(S+5)        % k QDOH
      P7  = B(S+6)  * 1D4 % AEQDOH
      P8  = B(S+7)        % k QDSH
      P9  = B(S+8)  * 1D4 % AEQDSH
      P10 = B(S+9)        % k QDAQ
      P11 = B(S+10) * 1D4 % AEQDAQ
      P12 = B(S+11)       % k    LQ
      P13 = B(S+12) * 1D4 % AE   LQ
      P14 = B(S+13)       % k    DR
      P15 = B(S+14) * 1D4 % AE   DR
      P16 = B(S+15)       % k    RD
      P17 = B(S+16) * 1D4 % AE   RD
      P18 = B(S+17)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT            % k'QL
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT            % k'QR
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH       % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SQRTSH   % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * SQRTAQ   % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT            % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT            % k'DR
```

```
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                        %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8014                     % This
         PKW    = SPLINE(TEMP, 10, 11)                    % requires
         LASTPK = PKW                                     % $SET OWN
         LASTTP = TEMP                                    % to work
8014  CONTINUE                                            %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K2 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K2 * Q + K7 * (Y1 + Y2) * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * (Y1 + Y2)*Y3
      GO TO 9902

  15  CONTINUE
      P1  = B(S  )         % k   QL
      P2  = B(S+1)   * 1D4 % AE  QL
C     P3  = B(S+2)   / TEN % Y0
      P6  = B(S+3)         % k QDOH
      P7  = B(S+4)   * 1D4 % AEQDOH
      P8  = B(S+5)         % k QDSH
      P9  = B(S+6)   * 1D4 % AEQDSH
      P10 = B(S+7)         % k QDAQ
      P11 = B(S+8)   * 1D4 % AEQDAQ
      P12 = B(S+9)         % k   LQ
      P13 = B(S+10)  * 1D4 % AE  LQ
      P14 = B(S+11)        % k   DR
      P15 = B(S+12)  * 1D4 % AE  DR
      P16 = B(S+13)        % k   RD
      P17 = B(S+14)  * 1D4 % AE  RD
      P18 = B(S+15)        % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT         % k'QL
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH    % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SH    % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * AQ    % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT         % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT         % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH    % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                        %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8015                     % This
         PKW    = SPLINE(TEMP, 10, 11)                    % requires
         LASTPK = PKW                                     % $SET OWN
         LASTTP = TEMP                                    % to work
8015  CONTINUE                                            %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K7 * Y3 - K8 * Y2
```

```
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * Y3
      GO TO 9902

 16   CONTINUE
      P1  = B(S  )        % k   QL
      P2  = B(S+1)  * 1D4 % AE  QL
C     P3  = B(S+2)  / TEN % YO
      P6  = B(S+3)        % k QDOH
      P7  = B(S+4)  * 1D4 % AEQDOH
      P8  = B(S+5)        % k QDSH
      P9  = B(S+6)  * 1D4 % AEQDSH
      P10 = B(S+7)        % k QDAQ
      P11 = B(S+8)  * 1D4 % AEQDAQ
      P12 = B(S+9)        % k   LQ
      P13 = B(S+10) * 1D4 % AE  LQ
      P14 = B(S+11)       % k   DR
      P15 = B(S+12) * 1D4 % AE  DR
      P16 = B(S+13)       % k   RD
      P17 = B(S+14) * 1D4 % AE  RD
      P18 = B(S+15)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT          % k'QL
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH     % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT          % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT          % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                    %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8016                 % This
         PKW    = SPLINE(TEMP, 10, 11)                % requires
         LASTPK = PKW                                 % $SET OWN
         LASTTP = TEMP                                % to work
8016  CONTINUE                                        %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K7 * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * Y3
      GO TO 9902

 17   CONTINUE
      P1  = B(S  )        % k   QL
      P2  = B(S+1)  * 1D4 % AE  QL
C     P3  = B(S+2)  / TEN % YO
      P6  = B(S+3)        % k QDOH
      P7  = B(S+4)  * 1D4 % AEQDOH
      P8  = B(S+5)        % k QDSH
      P9  = B(S+6)  * 1D4 % AEQDSH
      P10 = B(S+7)        % k QDAQ
      P11 = B(S+8)  * 1D4 % AEQDAQ
      P12 = B(S+9)        % k   LQ
```

```
      P13 = B(S+10) * 1D4 % AE  LQ
      P14 = B(S+11)       % k   DR
      P15 = B(S+12) * 1D4 % AE  DR
      P16 = B(S+13)       % k   RD
      P17 = B(S+14) * 1D4 % AE  RD
      P18 = B(S+15)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT        % k'QL
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH   % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SH   % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * AQ   % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT        % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT        % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH   % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                       %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8017                    % This
         PKW    = SPLINE(TEMP, 10, 11)                   % requires
         LASTPK = PKW                                    % $SET OWN
         LASTTP = TEMP                                   % to work
8017  CONTINUE                                           %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K7 * (Y1 + Y2) * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * (Y1 + Y2)*Y3
      GO TO 9902

  18  CONTINUE
      P1  = B(S  )         % k   QL
      P2  = B(S+1)   * 1D4 % AE  QL
C     P3  = B(S+2)   / TEN % YO
      P6  = B(S+3)         % k QDOH
      P7  = B(S+4)   * 1D4 % AEQDOH
      P8  = B(S+5)         % k QDSH
      P9  = B(S+6)   * 1D4 % AEQDSH
      P10 = B(S+7)         % k QDAQ
      P11 = B(S+8)   * 1D4 % AEQDAQ
      P12 = B(S+9)         % k   LQ
      P13 = B(S+10)  * 1D4 % AE  LQ
      P14 = B(S+11)        % k   DR
      P15 = B(S+12)  * 1D4 % AE  DR
      P16 = B(S+13)        % k   RD
      P17 = B(S+14)  * 1D4 % AE  RD
      P18 = B(S+15)        % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT            % k'QL
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH       % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SQRTSH   % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * SQRTAQ   % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT            % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT            % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH       % k'RD
      Y1  = Y(P)    % virgin lignin
```

```
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                        %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8018                     % This
         PKW    = SPLINE(TEMP, 10, 11)                    % requires
         LASTPK = PKW                                     % $SET OWN
         LASTTP = TEMP                                    % to work
 8018 CONTINUE                                            %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K3 + K4 + K5)
      YPRIME(P)   = K1 * Q - K6 * LM
      YPRIME(P+1) = K7 * (Y1 + Y2) * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * (Y1 + Y2)*Y3
      GO TO 9902

   19 CONTINUE
      P1  = B(S   )         % k   QL
      P2  = B(S+1)   * 1D4  % AE  QL
C     P3  = B(S+2)   / TEN  % Y0
      P4  = B(S+3)          % k   QR
      P5  = B(S+4)   * 1D4  % AE  QR
      P6  = B(S+5)          % k QDOH
      P7  = B(S+6)   * 1D4  % AEQDOH
      P8  = B(S+7)          % k QDSH
      P9  = B(S+8)   * 1D4  % AEQDSH
      P10 = B(S+9)          % k QDAQ
      P11 = B(S+10)  * 1D4  % AEQDAQ
      P12 = B(S+11)         % k   LQ
      P13 = B(S+12)  * 1D4  % AE  LQ
      P14 = B(S+13)         % k   DR
      P15 = B(S+14)  * 1D4  % AE  DR
      P16 = B(S+15)         % k   RD
      P17 = B(S+16)  * 1D4  % AE  RD
      P18 = B(S+17)         % pkalig
      P19 = B(S+18)         % AQ0
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT          % k'QL
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT          % k'QR
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * OH     % k'QDOH
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT * SH     % k'QDSH
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * AQ / (AQ+P19) % k'QDAQ
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT          % k'LQ
      K7  = DEXP(P14 - P15 * RTEMP) * SQRTT          % k'DR
      K8  = DEXP(P16 - P17 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                        %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8019                     % This
         PKW    = SPLINE(TEMP, 10, 11)                    % requires
         LASTPK = PKW                                     % $SET OWN
         LASTTP = TEMP                                    % to work
 8019 CONTINUE                                            %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P18))))
      Q   = K6 * LM / (K1 + K2 + K3 + K4 + K5)
```

```
      YPRIME(P)    = K1 * Q - K6 * LM
      YPRIME(P+1) = K2 * Q + K7 * (Y1 + Y2) * Y3 - K8 * Y2
      YPRIME(P+2) = (K3 + K4 + K5) * Q + K8 * Y2 - K7 * (Y1 + Y2)*Y3
      GO TO 9902

   20 CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k QR
      P9  = B(S+8)  * 1D4 % AEQR
      P10 = B(S+9)        % k DR
      P11 = B(S+10) * 1D4 % AEDR
      P12 = B(S+11)       % k RD
      P13 = B(S+12) * 1D4 % AERD
      P14 = B(S+13)       % pkalig
      P15 = B(S+14)       % SH0
      P16 = B(S+15)       % AQ0
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH            % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH / (SH+P15) % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ / (AQ+P16) % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT                 % k'QR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT                 % k'DR
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT * OH            % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                        %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8020                     % This
         PKW    = SPLINE(TEMP, 10, 11)                    % requires
         LASTPK = PKW                                     % $SET OWN
         LASTTP = TEMP                                    % to work
 8020 CONTINUE                                            %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P14))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
      YPRIME(P+1) = K4 * Q + K5 * (Y1 + Y2) * Y3 - K6 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * (Y1 + Y2)*Y3
      GO TO 9902

   21 CONTINUE
      GO TO 9902

   22 CONTINUE
      GO TO 9902

   23 CONTINUE
      GO TO 9902

   24 CONTINUE
```

```
      GO TO 9902

25    CONTINUE
      GO TO 9902

26    CONTINUE
      GO TO 9902

27    CONTINUE
      GO TO 9902

28    CONTINUE
      GO TO 9902

29    CONTINUE
      GO TO 9902

30    CONTINUE
      GO TO 9902

31    CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)  * 1D4  % AEQDOH
C     P3  = B(S+2)  / TEN  % YO
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)  * 1D4  % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)  * 1D4  % AEQDAQ
      P8  = B(S+7)         % k   QR
      P9  = B(S+8)  * 1D4  % AE  QR
      P10 = B(S+9)         % k   DR
      P11 = B(S+10) * 1D4  % AE  DR
      P12 = B(S+11)        % k   RD
      P13 = B(S+12) * 1D4  % AE  RD
      P14 = B(S+13)        % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH   % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH   % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ   % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT        % k'QR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT        % k'DR
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT * OH   % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                         %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8031                      % This
         PKW    = SPLINE(TEMP, 10, 11)                     % requires
         LASTPK = PKW                                      % $SET OWN
         LASTTP = TEMP                                     % to work
8031  CONTINUE                                             %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P14))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
      YPRIME(P+1) = K4 * Q + K5 * Y3 - K6 * Y2
```

```
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * Y3
      GO TO 9902

   32 CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)   * 1D4 % AEQDOH
C     P3  = B(S+2)   / TEN % Y0
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)   * 1D4 % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)   * 1D4 % AEQDAQ
      P8  = B(S+7)         % k   QR
      P9  = B(S+8)   * 1D4 % AE  QR
      P10 = B(S+9)         % k   DR
      P11 = B(S+10)  * 1D4 % AE  DR
      P12 = B(S+11)        % k   RD
      P13 = B(S+12)  * 1D4 % AE  RD
      P14 = B(S+13)        % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH     % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT          % k'QR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT          % k'DR
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                    %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8032                 % This
         PKW    = SPLINE(TEMP, 10, 11)                % requires
         LASTPK = PKW                                 % $SET OWN
         LASTTP = TEMP                                % to work
 8032 CONTINUE                                        %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P14))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
      YPRIME(P+1) = K4 * Q + K5 * Y3 - K6 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * Y3
      GO TO 9902

   33 CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)   * 1D4 % AEQDOH
C     P3  = B(S+2)   / TEN % Y0
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)   * 1D4 % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)   * 1D4 % AEQDAQ
      P8  = B(S+7)         % k   QR
      P9  = B(S+8)   * 1D4 % AE  QR
      P10 = B(S+9)         % k   DR
      P11 = B(S+10)  * 1D4 % AE  DR
      P12 = B(S+11)        % k   RD
      P13 = B(S+12)  * 1D4 % AE  RD
```

```
      P14 = B(S+13)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH    % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH    % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ    % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT         % k'QR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT         % k'DR
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT * OH    % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                            %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8033                         % This
         PKW    = SPLINE(TEMP, 10, 11)                        % requires
         LASTPK = PKW                                         % $SET OWN
         LASTTP = TEMP                                        % to work
8033  CONTINUE                                                %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P14))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
      YPRIME(P+1) = K4 * Q + K5 * (Y1 + Y2) * Y3 - K6 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * (Y1+Y2) * Y3
      GO TO 9902

  34  CONTINUE
      P1  = B(S   )       % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % YO
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   QR
      P9  = B(S+8)  * 1D4 % AE  QR
      P10 = B(S+9)        % k   DR
      P11 = B(S+10) * 1D4 % AE  DR
      P12 = B(S+11)       % k   RD
      P13 = B(S+12) * 1D4 % AE  RD
      P14 = B(S+13)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH      % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH  % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ  % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT           % k'QR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT           % k'DR
      K6  = DEXP(P12 - P13 * RTEMP) * SQRTT * OH      % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                            %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8034                         % This
         PKW    = SPLINE(TEMP, 10, 11)                        % requires
         LASTPK = PKW                                         % $SET OWN
         LASTTP = TEMP                                        % to work
8034  CONTINUE                                                %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P14))))
```

```
      Q    = LM
      YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
      YPRIME(P+1) = K4 * Q + K5 * (Y1 + Y2) * Y3 - K6 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * (Y1+Y2) * Y3
      GO TO 9902

 35   CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)  * 1D4 % AE  DR
      P10 = B(S+9)        % k   RD
      P11 = B(S+10) * 1D4 % AE  RD
      P12 = B(S+11)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH   % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH   % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ   % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT        % k'DR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH   % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                        %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8035                     % This
         PKW    = SPLINE(TEMP, 10, 11)                    % requires
         LASTPK = PKW                                     % $SET OWN
         LASTTP = TEMP                                    % to work
8035  CONTINUE                                            %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P12))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * Y3
      GO TO 9902

 36   CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)  * 1D4 % AE  DR
      P10 = B(S+9)        % k   RD
      P11 = B(S+10) * 1D4 % AE  RD
      P12 = B(S+11)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH     % k'QDOH
```

```
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT          % k'DR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                      %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8036                   % This
         PKW    = SPLINE(TEMP, 10, 11)                  % requires
         LASTPK = PKW                                   % $SET OWN
         LASTTP = TEMP                                  % to work
 8036 CONTINUE                                          %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P12))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * Y3
      GO TO 9902

   37 CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % YO
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)  * 1D4 % AE  DR
      P10 = B(S+9)        % k   RD
      P11 = B(S+10) * 1D4 % AE  RD
      P12 = B(S+11)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH   % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH   % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ   % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT        % k'DR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH   % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                      %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8037                   % This
         PKW    = SPLINE(TEMP, 10, 11)                  % requires
         LASTPK = PKW                                   % $SET OWN
         LASTTP = TEMP                                  % to work
 8037 CONTINUE                                          %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P12))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * (Y1 + Y2) * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * (Y1+Y2) * Y3
      GO TO 9902
```

```
   38   CONTINUE
        P1  = B(S   )        % k QDOH
        P2  = B(S+1)  * 1D4 % AEQDOH
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)        % k QDSH
        P5  = B(S+4)  * 1D4 % AEQDSH
        P6  = B(S+5)        % k QDAQ
        P7  = B(S+6)  * 1D4 % AEQDAQ
        P8  = B(S+7)        % k   DR
        P9  = B(S+8)  * 1D4 % AE  DR
        P10 = B(S+9)        % k   RD
        P11 = B(S+10) * 1D4 % AE  RD
        P12 = B(S+11)       % pkalig
        K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH     % k'QDOH
        K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
        K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
        K4  = DEXP(P8  - P9  * RTEMP) * SQRTT          % k'DR
        K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH     % k'RD
        Y1  = Y(P)    % virgin lignin
        Y2  = Y(P+1)  % residual lignin
        Y3  = Y(P+2)  % dissolved lignin
        PKW = LASTPK                                   %%%%%%%%%%%%
        IF(TEMP .EQ. LASTTP) GO TO 8038                % This
           PKW    = SPLINE(TEMP, 10, 11)               % requires
           LASTPK = PKW                                % $SET OWN
           LASTTP = TEMP                               % to work
 8038   CONTINUE                                       %%%%%%%%%%%%
        LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P12))))
        Q   = LM
        YPRIME(P)   = -(K1 + K2 + K3) * Q
        YPRIME(P+1) = K4 * (Y1 + Y2) * Y3 - K5 * Y2
        YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * (Y1+Y2) * Y3
        GO TO 9902

   45   CONTINUE
        P1  = B(S   )        % k QDOH
        P2  = B(S+1)  * 1D4 % AEQDOH
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)        % k QDSH
        P5  = B(S+4)  * 1D4 % AEQDSH
        P6  = B(S+5)        % k QDAQ
        P7  = B(S+6)  * 1D4 % AEQDAQ
        P8  = B(S+7)        % k   DR
        P9  = B(S+8)        % k   RD
        P10 = B(S+9)  * 1D4 % AE  RD
        P11 = B(S+10)       % pkalig
        K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH   % k'QDOH
        K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH   % k'QDSH
        K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ   % k'QDAQ
        K4  = DEXP(P8 -2.5D3 * RTEMP) * SQRTT        % k'DR
        K5  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH   % k'RD
        Y1  = Y(P)    % virgin lignin
        Y2  = Y(P+1)  % residual lignin
        Y3  = Y(P+2)  % dissolved lignin
```

```
      PKW = LASTPK                                               %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8045                            % This
         PKW     = SPLINE(TEMP, 10, 11)                          % requires
         LASTPK  = PKW                                           % $SET OWN
         LASTTP  = TEMP                                          % to work
 8045 CONTINUE                                                   %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * Y3
      GO TO 9902

   46 CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)        % k   RD
      P10 = B(S+9)  * 1D4 % AE  RD
      P11 = B(S+10)       % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH     % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P8 -2.5D3 * RTEMP) * SQRTT          % k'DR
      K5  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                               %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8046                            % This
         PKW     = SPLINE(TEMP, 10, 11)                          % requires
         LASTPK  = PKW                                           % $SET OWN
         LASTTP  = TEMP                                          % to work
 8046 CONTINUE                                                   %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * Y3
      GO TO 9902

   47 CONTINUE
      P1  = B(S   )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
```

```
      P8  = B(S+7)          % k   DR
      P9  = B(S+8)          % k   RD
      P10 = B(S+9)   * 1D4  % AE  RD
      P11 = B(S+10)         % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH   % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SH   % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * AQ   % k'QDAQ
      K4  = DEXP(P8 -2.5D3 * RTEMP) * SQRTT        % k'DR
      K5  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH   % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                           %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8047                        % This
         PKW    = SPLINE(TEMP, 10, 11)                       % requires
         LASTPK = PKW                                        % $SET OWN
         LASTTP = TEMP                                       % to work
 8047 CONTINUE                                               %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * (Y1 + Y2) * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * (Y1+Y2) * Y3
      GO TO 9902

   48 CONTINUE
      P1  = B(S  )          % k QDOH
      P2  = B(S+1)   * 1D4  % AEQDOH
C     P3  = B(S+2)   / TEN  % YO
      P4  = B(S+3)          % k QDSH
      P5  = B(S+4)   * 1D4  % AEQDSH
      P6  = B(S+5)          % k QDAQ
      P7  = B(S+6)   * 1D4  % AEQDAQ
      P8  = B(S+7)          % k   DR
      P9  = B(S+8)          % k   RD
      P10 = B(S+9)   * 1D4  % AE  RD
      P11 = B(S+10)         % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH       % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P8 -2.5D3 * RTEMP) * SQRTT            % k'DR
      K5  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH       % k'RD
      Y1  = Y(P)    % virgin lignin
      Y2  = Y(P+1)  % residual lignin
      Y3  = Y(P+2)  % dissolved lignin
      PKW = LASTPK                                           %%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8048                        % This
         PKW    = SPLINE(TEMP, 10, 11)                       % requires
         LASTPK = PKW                                        % $SET OWN
         LASTTP = TEMP                                       % to work
 8048 CONTINUE                                               %%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
```

```
      YPRIME(P+1) = K4 * (Y1 + Y2) * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K5 * Y2 - K4 * (Y1+Y2) * Y3
      GO TO 9902

 39   CONTINUE
      GO TO 9902

 40   CONTINUE
      GO TO 9902

 41   CONTINUE
      GO TO 9902

 42   CONTINUE
      P1  = B(S  )          % k QDOH
      P2  = B(S+1)   * 1D4  % AEQD
C     P3  = B(S+2)   / TEN  % Y0
      P4  = B(S+3)          % k QDSH
      P5  = B(S+4)          % k QDAQ
      P6  = B(S+5)          % k QR
      P7  = B(S+6)   * 1D4  % AE R
      P8  = B(S+7)          % k DR
      P9  = B(S+8)          % k RD
      P10 = B(S+9)   * 1D4  % AERD
      P11 = B(S+10)         % pkalig
      K1  = DEXP(P1 - P2  * RTEMP) * SQRTT * OH     % k'QDOH
      K2  = DEXP(P4 - P2  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P5 - P2  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P6 - P7  * RTEMP) * SQRTT          % k'QR
      K5  = DEXP(P8 - P7  * RTEMP) * SQRTT          % k'DR
      K6  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                  %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8042               % This
         PKW    = SPLINE(TEMP, 10, 11)              % requires
         LASTPK = PKW                               % $SET OWN
         LASTTP = TEMP                              % to work
 8042 CONTINUE                                      %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
      YPRIME(P+1) = K4 * Q + K5 * Y3 - K6 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * Y3
      GO TO 9902

 43   CONTINUE
      GO TO 9902

 44   CONTINUE
      P1  = B(S  )          % k QDOH
      P2  = B(S+1)   * 1D4  % AEQD
C     P3  = B(S+2)   / TEN  % Y0
```

```
            P4  = B(S+3)        % k QDSH
            P5  = B(S+4)        % k QDAQ
            P6  = B(S+5)        % k QR
            P7  = B(S+6)  * 1D4 % AE R
            P8  = B(S+7)        % k DR
            P9  = B(S+8)        % k RD
            P10 = B(S+9)  * 1D4 % AERD
            P11 = B(S+10)       % pkalig
            K1  = DEXP(P1 - P2  * RTEMP) * SQRTT * OH     % k'QDOH
            K2  = DEXP(P4 - P2  * RTEMP) * SQRTT * SQRTSH % k'QDSH
            K3  = DEXP(P5 - P2  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
            K4  = DEXP(P6 - P7  * RTEMP) * SQRTT          % k'QR
            K5  = DEXP(P8 - P7  * RTEMP) * SQRTT          % k'DR
            K6  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH     % k'RD
            Y1  = Y(P)    % virgin lignin
            Y2  = Y(P+1)  % residual lignin
            Y3  = Y(P+2)  % dissolved lignin
            PKW = LASTPK                                  %%%%%%%%%%%%
            IF(TEMP .EQ. LASTTP) GO TO 8044               % This
               PKW    = SPLINE(TEMP, 10, 11)              % requires
               LASTPK = PKW                               % $SET OWN
               LASTTP = TEMP                              % to work
 8044       CONTINUE                                      %%%%%%%%%%%%
            LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
            Q   = LM
            YPRIME(P)   = -(K1 + K2 + K3 + K4) * Q
            YPRIME(P+1) = K4 * Q + K5 * (Y1 + Y2) * Y3 - K6 * Y2
            YPRIME(P+2) = (K1 + K2 + K3) * Q + K6 * Y2 - K5 * (Y1+Y2) * Y3
            GO TO 9902

   49       CONTINUE
            GO TO 9902

   50       CONTINUE
            P1  = B(S  )        % k QDOH
            P2  = B(S+1)  * 1D4 % AEQDOH
C           P3  = B(S+2)  / TEN % YO
            P4  = B(S+3)        % k QDSH
            P5  = B(S+4)  * 1D4 % AEQDSH
            P6  = B(S+5)        % k QDAQ
            P7  = B(S+6)  * 1D4 % AEQDAQ
            P8  = B(S+7)        % k   DR
            P9  = B(S+8)        % k   RD
            P10 = B(S+9)  * 1D4 % AE  RD
            P11 = B(S+10)       % pkalig
            K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH     % k'QDOH
            K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
            K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
            K4  = DEXP(P8 -2.5D3 * RTEMP) * SQRTT          % k'DR
            K5  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH     % k'RD
            Y1  = Y(P)    % virgin lignin
            Y2  = Y(P+1)  % residual lignin
            Y3  = Y(P+2)  % dissolved lignin
            PKW = LASTPK                                   %%%%%%%%%%%%
```

```
      IF(TEMP .EQ. LASTTP) GO TO 8050                        % This
         PKW    = SPLINE(TEMP, 10, 11)                       % requires
         LASTPK = PKW                                        % $SET OWN
         LASTTP = TEMP                                       % to work
 8050 CONTINUE                                               %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q - K4 * Y3
      GO TO 9902

   51 CONTINUE
      P1  = B(S  )          % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % YO
      P4  = B(S+3)          % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)          % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)          % k   DR
      P9  = B(S+8)  * 1D4 % AE  DR
      P10 = B(S+9)          % k   RD
      P11 = B(S+10) * 1D4 % AE  RD
      P12 = B(S+11)         % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH      % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P8  - P9  * RTEMP) * SQRTT           % k'DR
      K5  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH      % k'RD
      Y1  = Y(P)     % virgin lignin
      Y2  = Y(P+1)   % residual lignin
      Y3  = Y(P+2)   % dissolved lignin
      PKW = LASTPK                                           %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8051                        % This
         PKW    = SPLINE(TEMP, 10, 11)                       % requires
         LASTPK = PKW                                        % $SET OWN
         LASTTP = TEMP                                       % to work
 8051 CONTINUE                                               %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P12))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * Y3 - K5 * Y2
      YPRIME(P+2) = (K1 + K2 + K3) * Q - K4 * Y3
      GO TO 9902

   52 CONTINUE
      P1  = B(S  )          % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  / TEN % YO
      P4  = B(S+3)          % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)          % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
```

```
      P8  = B(S+7)         % k   DR
      P9  = B(S+8)         % k   RD
      P10 = B(S+9)  * 1D4  % AE  RD
      P11 = B(S+10)        % pkalig
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH     % k'QDOH
      K2  = DEXP(P4  - P5  * RTEMP) * SQRTT * SQRTSH % k'QDSH
      K3  = DEXP(P6  - P7  * RTEMP) * SQRTT * SQRTAQ % k'QDAQ
      K4  = DEXP(P8 -2.5D3 * RTEMP) * SQRTT          % k'DR
      K5  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH     % k'RD
      Y1  = Y(P)   % virgin lignin
      Y2  = Y(P+1) % residual lignin
      Y3  = Y(P+2) % dissolved lignin
      PKW = LASTPK                                          %%%%%%%%%%%%
      IF(TEMP .EQ. LASTTP) GO TO 8052                       % This
         PKW    = SPLINE(TEMP, 10, 11)                      % requires
         LASTPK = PKW                                       % $SET OWN
         LASTTP = TEMP                                      % to work
8052  CONTINUE                                              %%%%%%%%%%%%
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -(K1 + K2 + K3) * Q
      YPRIME(P+1) = K4 * DSQRT(Y3) - K5 * Y2
      YPRIME(P+2) = ((K1 + K2 + K3) * Q - K4 * DSQRT(Y3)) / 40.D0
      GO TO 9902

  53  CONTINUE
      P1  = B(S  )         % k QDOH
      P2  = B(S+1)  * 1D4  % AEQDOH
      P3  = B(S+2)  * TEN  % Y0
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)  * 1D4  % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)  * 1D4  % AEQDAQ
      P8  = B(S+7)         % k   DR
      P9  = B(S+8)         % k   RD
      P10 = B(S+9)  * 1D4  % AE  RD
      P11 = B(S+10)        % pkalig
      Y1  = Y(P)   % native lignin
      Y2  = Y(P+1) % residual lignin
      Y3  = Y(P+2) % dissolved lignin
      AQ  = Y(P+3) % anthraquinone
      PKW = (TEMP * 7.19859D-5 - 7.18311D-2) * TEMP + 29.0603D0
      ALPHA = (64.D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &         + 1.24208D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * DSQRT(OH) % k'QD
     &    + DEXP(P4  - P5  * RTEMP) * SQRTT * DSQRT(SH)
     &    + DEXP(P6  - P7  * RTEMP) * SQRTT * DSQRT(AQ)
```

```
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT                % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH           % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT     % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT         % k'Linitial
     &      * (ONE - (ONE / (ONE + OH * 100.D0)))
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -K1 * Q
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * Y3 - K3 * Y2
      YPRIME(P+2) = (K1 * Q - K2 * Y3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

54    CONTINUE
      P1  = B(S  )          % k QDOH
      P2  = B(S+1)   * 1D4 % AEQDOH
      P3  = B(S+2)   * TEN % Y0
      P4  = B(S+3)          % k QDSH
      P5  = B(S+4)   * 1D4 % AEQDSH
      P6  = B(S+5)          % k QDAQ
      P7  = B(S+6)   * 1D4 % AEQDAQ
      P8  = B(S+7)          % k   DR
      P9  = B(S+8)          % k   RD
      P10 = B(S+9)   * 1D4 % AE  RD
      P11 = B(S+10)         % pkalig
      Y1  = Y(P)   % native lignin
      Y2  = Y(P+1) % residual lignin
      Y3  = Y(P+2) % dissolved lignin
      AQ  = Y(P+3) % anthraquinone
      PKW = (TEMP * 7.19859D-5 - 7.18311D-2) * TEMP + 29.0603D0
      ALPHA = (64.D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &        + 1.24208D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      SQRTOH = DSQRT(OH)
      K1  =(DEXP(P1  - P2  * RTEMP) * SQRTOH               % k'QD
     &     + DEXP(P4  - P5  * RTEMP) * DSQRT(SH)
     &     + DEXP(P6  - P7  * RTEMP) * DSQRT(AQ)) * SQRTT * SQRTOH
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT                % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH           % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT     % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT         % k'Linitial
     &      * (ONE - (ONE / (ONE + OH * 100.D0)))
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -K1 * Q
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * Y3 - K3 * Y2
```

```
      YPRIME(P+2) = (K1 * Q - K2 * Y3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

55    CONTINUE
      P1  = B(S  )         % k QDOH
      P2  = B(S+1)  * 1D4  % AEQDOH
      P3  = B(S+2)  * TEN  % Y0
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)  * 1D4  % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)  * 1D4  % AEQDAQ
      P8  = B(S+7)         % k   DR
      P9  = B(S+8)         % k   RD
      P10 = B(S+9)  * 1D4  % AE  RD
      Y1     = Y(P)           % native lignin
      Y2     = Y(P+1)         % residual lignin
      Y3     = Y(P+2)         % dissolved lignin
      Y3     = DMAX1(Y3, 0D0)
      SQRTY3 = DSQRT(Y3)
      AQ     = Y(P+3)         % anthraquinone
      ALPHA =(65.6D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &       + 1.21073D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      SQRTOH = DSQRT(OH)
      K1  =(DEXP(P1  - P2  * RTEMP) * SQRTOH             % k'QD
     &     + DEXP(P4  - P5  * RTEMP) * DSQRT(SH)
     &     + DEXP(P6  - P7  * RTEMP) * DSQRT(AQ)) * SQRTT * SQRTOH
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT              % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH         % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT   % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT       % k'Linitial
     &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)   = -K1 * Y1
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * Y1
      YPRIME(P+1) = K2 * SQRTY3 - K3 * Y2
      YPRIME(P+2) = (K1 * Y1 - K2 * SQRTY3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

56    CONTINUE
      P1  = B(S  )         % k QDOH
      P2  = B(S+1)  * 1D4  % AEQDOH
      P3  = B(S+2)  * TEN  % Y0
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)  * 1D4  % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)  * 1D4  % AEQDAQ
```

```
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)        % k   RD
      P10 = B(S+9)  * 1D4 % AE  RD
      P11 = B(S+10)       % pkalig
      Y1  = Y(P)   % native lignin
      Y2  = Y(P+1) % residual lignin
      Y3  = Y(P+2) % dissolved lignin
      Y3  = DMAX1(Y3, 0D0)
      SQRTY3 = DSQRT(Y3)
      AQ  = Y(P+3) % anthraquinone
      PKW = (TEMP * 7.19859D-5 - 7.18311D-2) * TEMP + 29.0603D0
      ALPHA = (64.D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &       + 1.24208D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * OH          % k'QD
     &     + DEXP(P4  - P5  * RTEMP) * SQRTT * DSQRT(SH)
     &     + DEXP(P6  - P7  * RTEMP) * SQRTT * DSQRT(AQ)
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT               % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH          % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT    % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT        % k'Linitial
     &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -K1 * Q
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * SQRTY3 - K3 * Y2
      YPRIME(P+2) = (K1 * Q - K2 * SQRTY3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

57    CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
      P3  = B(S+2)  * TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)        % k   RD
      P10 = B(S+9)  * 1D4 % AE  RD
      P11 = B(S+10)       % pkalig
      Y1  = Y(P)   % native lignin
      Y2  = Y(P+1) % residual lignin
      Y3  = Y(P+2) % dissolved lignin
      Y3  = DMAX1(Y3, 0D0)
      SQRTY3 = DSQRT(Y3)
```

```
      AQ  = Y(P+3) % anthraquinone
      PKW = (TEMP * 7.19859D-5 - 7.18311D-2) * TEMP + 29.0603D0
      ALPHA = (64.D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
    &       + 1.24208D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      K1  = DEXP(P1  - P2  * RTEMP) * SQRTT * DSQRT(OH) % k'QD
    &     + DEXP(P4  - P5  * RTEMP) * SQRTT * DSQRT(SH)
    &     + DEXP(P6  - P7  * RTEMP) * SQRTT * DSQRT(AQ)
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT              % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH         % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT      % k'Linitial
    &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -K1 * Q
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * SQRTY3 - K3 * Y2
      YPRIME(P+2) = (K1 * Q - K2 * SQRTY3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

58    CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
      P3  = B(S+2)  * TEN % YO
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)        % k   RD
      P10 = B(S+9)  * 1D4 % AE  RD
      P11 = B(S+10)       % pkalig
      Y1  = Y(P)   % native lignin
      Y2  = Y(P+1) % residual lignin
      Y3  = Y(P+2) % dissolved lignin
      Y3  = DMAX1(Y3, 0D0)
      SQRTY3 = DSQRT(Y3)
      AQ  = Y(P+3) % anthraquinone
      PKW = (TEMP * 7.19859D-5 - 7.18311D-2) * TEMP + 29.0603D0
      ALPHA = (64.D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
    &       + 1.24208D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
```

```
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      SQRTOH = DSQRT(OH)
      K1  =(DEXP(P1  - P2  * RTEMP) * SQRTOH             % k'QD
    &      + DEXP(P4  - P5  * RTEMP) * DSQRT(SH)
    &      + DEXP(P6  - P7  * RTEMP) * DSQRT(AQ)) * SQRTOH * SQRTT
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT              % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH         % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT   % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT       % k'Linitial
    &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      LM  = Y1 * (ONE - (ONE / (ONE + OH * TEN ** (PKW - P11))))
      Q   = LM
      YPRIME(P)   = -K1 * Q
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * SQRTY3 - K3 * Y2
      YPRIME(P+2) = (K1 * Q - K2 * SQRTY3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

59    CONTINUE
      P1  = B(S  )         % k QDOH
      P2  = B(S+1)  * 1D4  % AEQDOH
      P3  = B(S+2)  * TEN  % Y0
      P4  = B(S+3)         % k QDSH
      P5  = B(S+4)  * 1D4  % AEQDSH
      P6  = B(S+5)         % k QDAQ
      P7  = B(S+6)  * 1D4  % AEQDAQ
      P8  = B(S+7)         % k   DR
      P9  = B(S+8)         % k   RD
      P10 = B(S+9)  * 1D4  % AE  RD
      P11 = B(S+10) / TEN  % kDRord
      Y1     = Y(P)          % native lignin
      Y2     = Y(P+1)        % residual lignin
      Y3     = Y(P+2)        % dissolved lignin
      Y3     = DMAX1(Y3, 0D0)
      Y3ORD  = Y3 ** P11
      AQ     = Y(P+3)        % anthraquinone
      ALPHA =(65.6D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
    &        + 1.21073D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      SQRTOH = DSQRT(OH)
      K1  =(DEXP(P1  - P2  * RTEMP) * SQRTOH             % k'QD
    &      + DEXP(P4  - P5  * RTEMP) * DSQRT(SH)
    &      + DEXP(P6  - P7  * RTEMP) * DSQRT(AQ)) * SQRTT * SQRTOH
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT              % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH         % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT   % k'AQ
```

```
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT      % k'Linitial
     &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)   = -K1 * Y1
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * Y1
      YPRIME(P+1) = K2 * Y3ORD - K3 * Y2
      YPRIME(P+2) = (K1 * Y1 - K2 * Y3ORD) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

 60   CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
      P3  = B(S+2)  * TEN % Y0
      P4  = B(S+3)        % k QDSH
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)        % k   RD
      P10 = B(S+9)  * 1D4 % AE  RD
      Y1     = Y(P)          % native lignin
      Y2     = Y(P+1)        % residual lignin
      Y3     = Y(P+2)        % dissolved lignin
      AQ     = Y(P+3)        % anthraquinone
      ALPHA =(65.6D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &       + 1.21073D0
      BETA = 30.1D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      SQRTOH = DSQRT(OH)
      K1  =(DEXP(P1   - P2   * RTEMP) * SQRTOH             % k'QD
     &     + DEXP(P4   - P5   * RTEMP) * DSQRT(SH)
     &     + DEXP(P6   - P7   * RTEMP) * DSQRT(AQ)) * SQRTT * SQRTOH
      K2  = DEXP(P8  - 25D2  * RTEMP) * SQRTT              % k'DR
      K3  = DEXP(P9   - P10  * RTEMP) * SQRTT * OH         % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
      K5  = DEXP(7.494D0 - 4738.D0 / TEMP) * SQRTT      % k'Linitial
     &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)   = -K1 * Y1
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * Y1
      YPRIME(P+1) = K2 * Y3 - K3 * Y2
      YPRIME(P+2) = (K1 * Y1 - K2 * Y3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

 61   CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
      P3  = B(S+2)  * TEN % Y0
      P4  = B(S+3)        % k QDSH
```

```
      P5  = B(S+4)  * 1D4 % AEQDSH
      P6  = B(S+5)        % k QDAQ
      P7  = B(S+6)  * 1D4 % AEQDAQ
      P8  = B(S+7)        % k   DR
      P9  = B(S+8)        % k   RD
      P10 = B(S+9)  * 1D4 % AE  RD
      Y1     = Y(P)          % native lignin
      Y2     = Y(P+1)        % residual lignin
      Y3     = Y(P+2)        % dissolved lignin
      AQ     = Y(P+3)        % anthraquinone
      Y1PY2 = Y1 + Y2
      ALPHA =(65.6D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &       + 1.21073D0
      BETA = 30.1D0 - Y1PY2
      IF(Y1PY2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
      OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
      SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      SQRTOH = DSQRT(OH)
      K1  =(DEXP(P1  - P2  * RTEMP) * SQRTOH             % k'QD
     &     + DEXP(P4  - P5  * RTEMP) * DSQRT(SH)
     &     + DEXP(P6  - P7  * RTEMP) * DSQRT(AQ)) * SQRTT * SQRTOH
      K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT              % k'DR
      K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH         % k'RD
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT   % k'AQ
      K5  = DEXP(17.33D0 - 6000.D0 / TEMP)               % k'Lid1
     &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      K6  = DEXP(22.12D0 - 8800.D0 / TEMP)               % k'Lid2
     &     * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)    = -K1 * Y1
      IF(Y1 .GT. P3) YPRIME(P) = YPRIME(P) - K6 * Y1
      IF(Y1 .GT. 26.2D0) YPRIME(P) = YPRIME(P) - K5 * (Y1 - 26.2D0)
      YPRIME(P+1) = K2 * Y3 - K3 * Y2
      YPRIME(P+2) = (K1 * Y1 - K2 * Y3) / 40.D0
      YPRIME(P+3) = -K4 * AQ
      GO TO 9902

 62   CONTINUE
      P1  = B(S  )        % k QDOH
      P2  = B(S+1)  * 1D4 % AEQDOH
C     P3  = B(S+2)  * TEN % Y0
      P4  = B(S+2)        % k QDSH
      P5  = B(S+3)  * 1D4 % AEQDSH
      P6  = B(S+4)        % k QDAQ
      P7  = B(S+5)  * 1D4 % AEQDAQ
      P8  = B(S+6)        % k   DR
      P9  = B(S+7)        % k   RD
      P10 = B(S+8)  * 1D4 % AE  RD
      Y1     = Y(P)          % native lignin
      Y2     = Y(P+1)        % residual lignin
      Y3     = Y(P+2)        % dissolved lignin
      AQ     = Y(P+3)        % anthraquinone
```

```
          Y1PY2 = Y1 + Y2
          ALPHA =(65.6D0 - EX(EOB, 8) - EX(EOB, 9) - EX(EOB, 10)) * 0.4D0
     &          + 1.21073D0
          BETA = 30.1D0 - Y1PY2
          IF(Y1PY2 .GT. P3) ALPHA = ALPHA * BETA / (30.1D0 - P3)
          OH = OH - (0.15D0 * BETA + ALPHA) / 160.D0
          SH = SH - (0.056D0 * (30.1D0 - Y1)) / 224.D0
          OH = DMAX1(OH, 0.D0)
          SH = DMAX1(SH, 0.D0)
          AQ = DMAX1(AQ, 0.D0)
          SQRTOH = DSQRT(OH)
          K1  =(DEXP(P1  - P2  * RTEMP) * SQRTOH            % k'QD
     &        + DEXP(P4  - P5  * RTEMP) * DSQRT(SH)
     &        + DEXP(P6  - P7  * RTEMP) * DSQRT(AQ)) * SQRTT * SQRTOH
          K2  = DEXP(P8 - 25D2 * RTEMP) * SQRTT             % k'DR
          K3  = DEXP(P9  - P10 * RTEMP) * SQRTT * OH        % k'RD
          K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
          K5  = DEXP(17.33D0 - 6000.D0 / TEMP)              % k'Lid1
     &        * (ONE - (ONE / (ONE + OH * 100.D0)))
          K6  = DEXP(22.12D0 - 8800.D0 / TEMP)              % k'Lid2
     &        * (ONE - (ONE / (ONE + OH * 100.D0)))
          YPRIME(P)   = -K1 * Y1
          IF(Y1 .GT. 26.2D0) YPRIME(P) = YPRIME(P) - K5 * (Y1 - 26.2D0)
          IF(Y1 .GT. 22.9D0) YPRIME(P) = YPRIME(P) - K6 * Y1
          YPRIME(P+1) = K2 * Y3 - K3 * Y2
          YPRIME(P+2) = (K1 * Y1 - K2 * Y3) / 40.D0
          YPRIME(P+3) = -K4 * AQ
          GO TO 9902

C-----Celluose

 9902 CONTINUE
      S = POINTR(2)
      C = CHOOSE(2)
      P = POINTY(2)
      GO TO(101, 102, 103, 104, 105, 106, 107, 108, 109, 110
     &    , 111, 112, 113, 114, 115, 116, 117, 118, 119, 120
     &    , 121, 122, 123, 124, 125, 126, 127, 128, 129, 130), C
          IF(C .NE. 0) CALL MESSAG(6, "   FCN", 9902, .TRUE.)
            GO TO 9903

  101     CONTINUE
          P1 = B(S)           % k
          P2 = B(S+1) * 1D4 % AE
C         P3 = B(S+2) / TEN % YO
          YPRIME(P) = -Y(P) * SQRTT * DEXP(P1 - P2 * RTEMP)
     &    * OH
          GO TO 9903

  102     CONTINUE
          P1 = B(S)           % k
          P2 = B(S+1) * 1D4 % AE
C         P3 = B(S+2) / TEN % YO
          YPRIME(P) = -Y(P) * Y(P) * SQRTT * DEXP(P1 - P2 * RTEMP)
```

```
     &   * OH
         GO TO 9903

  103    CONTINUE
         P1 = B(S)          % k  OH
         P2 = B(S+1) * 1D4  % AE common
C        P3 = B(S+2) / TEN  % YO
         P4 = B(S+3)        % k  SH
         P5 = B(S+4)        % k  AQ
         Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
         YPRIME(P) = -Y1 * SQRTT * SQRTOH
     &             * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                + SQRTSH * DEXP(P4 - P2 * RTEMP)
     &                - SQRTAQ * DEXP(P5 - P2 * RTEMP))
         GO TO 9903

  104    CONTINUE
         P1 = B(S  )        % k  OH
         P2 = B(S+1) * 1D4  % AE OH
C        P3 = B(S+2) / TEN  % YO
         P4 = B(S+3)        % k  SH
         P5 = B(S+4) * 1D4  % AE SH
         P6 = B(S+5)        % k  AQ
         P7 = B(S+6) * 1D4  % AE AQ
         Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
         YPRIME(P) = -Y1 * SQRTT * SQRTOH
     &             * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &                - SQRTAQ * DEXP(P6 - P7 * RTEMP))
         GO TO 9903

  105    CONTINUE
         P1 = B(S  )        % k  OH
         P2 = B(S+1) * 1D4  % AE OH
C        P3 = B(S+2) / TEN  % YO
         P4 = B(S+3)        % k  SH
         P5 = B(S+4) * 1D4  % AE SH
         P6 = B(S+5)        % k  AQ
         P7 = B(S+6) * 1D4  % AE AQ
         Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
         YPRIME(P) = -(Y1 * Y1) * SQRTT * SQRTOH
     &             * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &                - SQRTAQ * DEXP(P6 - P7 * RTEMP))
         GO TO 9903

  106    CONTINUE
         P1  = B(S  )        % k  OH
         P2  = B(S+1) * 1D4  % AE OH
C        P3  = B(S+2) / TEN  % YO
         P4  = B(S+3)        % k  SH
         P5  = B(S+4) * 1D4  % AE SH
         P6  = B(S+5)        % k  AQ
         P7  = B(S+6) * 1D4  % AE AQ
```

```
          P8   = B(S+7)          % SHpower                  % 19 Feb 85 %%
          P9   = B(S+8)          % AQpower                  % 19 Feb 85 %%
          Y1   = DMIN1(Y(P), TEN)                           % 31Dec84 %%
          YPRIME(P) = 0D0; IF(Y1 .GT. 0D0)
     &    YPRIME(P) = -Y1 * SQRTT
     &              * (  OH          * DEXP(P1 - P2 * RTEMP)
     &                 + SH**P8      * DEXP(P4 - P5 * RTEMP)
     &                 - AQ**P9 * OH * DEXP(P6 - P7 * RTEMP))
          GO TO 9903

  107     CONTINUE
          P1   = B(S  )          % k peel
          P2   = B(S+1)   * 1D4  % AEpeel
C         P3   = B(S+2)   / TEN  % Y0
C         P4   = B(S+3)   / 1D2  % YSH
C         P5   = B(S+4)   / 1D2  % YAQ
          YPRIME(P)    = -Y(P) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
          GO TO 9903

  108     CONTINUE
          P1   = B(S  )          % k peel
          P2   = B(S+1)   * 1D4  % AEpeel
C         P3   = B(S+2)   / TEN  % Y0
          P4   = B(S+3)          % k stop
          P5   = B(S+4)   * 1D4  % AEstop
          P6   = B(S+5)          % k cleave
          P7   = B(S+6)   * 1D4  % AEcleave
C         P8   = B(S+7)   / 1D2  % YSH
C         P9   = B(S+8)   / 1D2  % YAQ
          K1   = SQRTT * OH * DEXP(P1 - P2 * RTEMP) % kp
          K2   = SQRTT * OH * DEXP(P4 - P5 * RTEMP) % ks
          K3   = SQRTT * OH * DEXP(P6 - P7 * RTEMP) % kc
          Y1   = Y(P)            % virgin
          Y2   = Y(P+1)          % oxidized
          YPRIME(P)    = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9903

  109     CONTINUE
          P1   = B(S  )          % k peel
          P2   = B(S+1)   * 1D4  % AEpeel
C         P3   = B(S+2)   / TEN  % Y0
          P4   = B(S+3)          % OHpower
C         P5   = B(S+4)          % YSHpower
C         P6   = B(S+5)          % YAQpower
C         P7   = B(S+6)   / 1D2  % YSH
C         P8   = B(S+7)   / 1D2  % YAQ
          YPRIME(P) = -Y(P) * SQRTT * OH**P4 * DEXP(P1 - P2 * RTEMP)
          GO TO 9903

  110     CONTINUE
          P1   = B(S  )          % k peel
          P2   = B(S+1)   * 1D4  % AEpeel
C         P3   = B(S+2)   / TEN  % Y0
```

```
          P4  = B(S+3)         % k stop
          P5  = B(S+4)  * 1D4  % AEstop
          P6  = B(S+5)         % k cleave
          P7  = B(S+6)  * 1D4  % AEcleave
          P8  = B(S+7)         % OHpeel power
          P9  = B(S+8)         % OHstop power
          P10 = B(S+9)         % OHcleave power
C         P11 = B(S+10)        % YSHpower
C         P12 = B(S+11)        % YAQpower
C         P13 = B(S+12) / 1D2  % YSH
C         P14 = B(S+13) / 1D2  % YAQ
          K1  = SQRTT * OH**P8  * DEXP(P1 - P2 * RTEMP) % kp
          K2  = SQRTT * OH**P9  * DEXP(P4 - P5 * RTEMP) % ks
          K3  = SQRTT * OH**P10 * DEXP(P6 - P7 * RTEMP) % kc
          Y1  = Y(P)                         % virgin
          Y2  = Y(P+1)                       % oxidized
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9903

  111     CONTINUE
          P1  = B(S  )         % k peel
          P2  = B(S+1)  * 1D4  % AEpeel
C         P3  = B(S+2)  / TEN  % Y0
          P4  = B(S+3)         % k stOH
          P5  = B(S+4)  * 1D4  % AEstOH
          P6  = B(S+5)         % k stAQ
          P7  = B(S+6)  * 1D4  % AEstAQ
          P8  = B(S+7)         % k cleave
          P9  = B(S+8)  * 1D4  % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH             % k'peel
          K2  = DEXP(P4 - P5 * RTEMP) * SQRTT * OH             % k'stop
     &        + DEXP(P6 - P7 * RTEMP) * SQRTT * SQRTAQ
          K3  = DEXP(P8 - P9 * RTEMP) * SQRTT * OH             % k'cleave
          Y1  = DMIN1(Y(P) , TEN)          % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1),TEN)          % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9903

  112     CONTINUE
          P1  = B(S  )         % k peOH
          P2  = B(S+1)  * 1D4  % AEpeOH
C         P3  = B(S+2)  / TEN  % Y0
          P4  = B(S+3)         % k peSH
          P5  = B(S+4)  * 1D4  % AEpeSH
          P6  = B(S+5)         % k stOH
          P7  = B(S+6)  * 1D4  % AEstOH
          P8  = B(S+7)         % k stAQ
          P9  = B(S+8)  * 1D4  % AEstAQ
          P10 = B(S+9)         % k cleave
          P11 = B(S+10) * 1D4  % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH             % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
```

```
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
  &       + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH                % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)           % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

113   CONTINUE
      P1  = B(S  )          % k peOH
      P2  = B(S+1)   * 1D4  % AEpeOH
C     P3  = B(S+2)   / TEN  % Y0
      P4  = B(S+3)          % k peSH
      P5  = B(S+4)   * 1D4  % AEpeSH
      P6  = B(S+5)          % k stOH
      P7  = B(S+6)   * 1D4  % AEstOH
      P8  = B(S+7)          % k stAQ
      P9  = B(S+8)   * 1D4  % AEstAQ
      P10 = B(S+9)          % k cleave
      P11 = B(S+10)  * 1D4  % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
  &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
  &       + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH                % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)           % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

114   CONTINUE
      P1  = B(S  )          % k peOH
      P2  = B(S+1)   * 1D4  % AEpeOH
C     P3  = B(S+2)   / TEN  % Y0
      P4  = B(S+3)          % k peSH
      P5  = B(S+4)   * 1D4  % AEpeSH
      P6  = B(S+5)          % k stOH
      P7  = B(S+6)   * 1D4  % AEstOH
      P8  = B(S+7)          % k stAQ
      P9  = B(S+8)   * 1D4  % AEstAQ
      P10 = B(S+9)          % k cleave
      P11 = B(S+10)  * 1D4  % AEcleave
      P12 = B(S+11)         % SH0
      P13 = B(S+12)         % AQ0
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
  &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SH / (SH + P12)
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
  &       + DEXP(P8 - P9 * RTEMP) * SQRTT * AQ / (AQ + P13)
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH                % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)           % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
```

```
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

 115  CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % YO
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)         % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)         % oxidized % 31Dec84 %%%%%%
      Y3  = DMIN1(Y(P+2), TEN)         % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9903

 116  CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % YO
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)         % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)         % oxidized % 31Dec84 %%%%%%
      Y3  = DMIN1(Y(P+2), TEN)         % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9903

 117  CONTINUE
```

```
            P1  = B(S  )          % k peOH
            P2  = B(S+1)  * 1D4 % AEpeOH
C           P3  = B(S+2)  / TEN % Y0
            P4  = B(S+3)          % k peSH
            P5  = B(S+4)  * 1D4 % AEpeSH
            P6  = B(S+5)          % k stOH
            P7  = B(S+6)  * 1D4 % AEstOH
            P8  = B(S+7)          % k stAQ
            P9  = B(S+8)  * 1D4 % AEstAQ
            P10 = B(S+9)          % k cleave
            P11 = B(S+10) * 1D4 % AEcleave
            K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &          + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
            K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &          + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
            K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
            Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
            Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
            Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
            YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
            YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
            YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
            GO TO 9903

  118       CONTINUE
            P1  = B(S  )          % k peOH
            P2  = B(S+1)  * 1D4 % AEpeOH
C           P3  = B(S+2)  / TEN % Y0
            P4  = B(S+3)          % k peSH
            P5  = B(S+4)  * 1D4 % AEpeSH
            P6  = B(S+5)          % k stOH
            P7  = B(S+6)  * 1D4 % AEstOH
            P8  = B(S+7)          % k stAQ
            P9  = B(S+8)  * 1D4 % AEstAQ
            P10 = B(S+9)          % k cleave
            P11 = B(S+10) * 1D4 % AEcleave
            K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &          + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
            K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &          + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
            K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
            Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
            Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
            Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
            YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
            YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
            YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
            GO TO 9903

  119       CONTINUE
            P1  = B(S  )          % k peOH
            P2  = B(S+1)  * 1D4 % AEpeOH
C           P3  = B(S+2)  / TEN % Y0
            P4  = B(S+3)          % k peSH
```

```
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstOH
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)  * 1D4 % AEstAQ
          P10 = B(S+9)        % k cleave
          P11 = B(S+10) * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &          + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &          + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
          K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
          Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9903

  120     CONTINUE
          P1  = B(S  )        % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstOH
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)  * 1D4 % AEstAQ
          P10 = B(S+9)        % k cleave
          P11 = B(S+10) * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &          + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &          + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
          K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
          Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9903

  121     CONTINUE
          GO TO 9903

  122     CONTINUE
          GO TO 9903

  123     CONTINUE
          GO TO 9903

  124     CONTINUE
          P1  = B(S  )        % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
```

```
      P4   = B(S+3)          % k peSH
      P5   = B(S+4)   * 1D4  % AEpeSH
      P6   = B(S+5)          % k stOH
      P7   = B(S+6)   * 1D4  % AEstop
      P8   = B(S+7)          % k stAQ
      P9   = B(S+8)          % k cleave
      P10  = B(S+9)   * 1D4  % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &     + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
     &     + DEXP(P8 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
      K3   = DEXP(P9 - P10 * RTEMP) * SQRTT * OH              % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)       % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)        % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

125   CONTINUE
      P1   = B(S  )          % k peOH
      P2   = B(S+1)   * 1D4  % AEpeOH
C     P3   = B(S+2)   / TEN  % Y0
      P4   = B(S+3)          % k peSH
      P5   = B(S+4)   * 1D4  % AEpeSH
      P6   = B(S+5)          % k stOH
      P7   = B(S+6)   * 1D4  % AEstop
      P8   = B(S+7)          % k stAQ
      P9   = B(S+8)          % k cleave
      P10  = B(S+9)   * 1D4  % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &     + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
     &     + DEXP(P8 - P7 * RTEMP) * SQRTT * SQRTAQ
      K3   = DEXP(P9 - P10 * RTEMP) * SQRTT * OH              % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)       % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)        % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

126   CONTINUE
      P1   = B(S  )          % k peOH
      P2   = B(S+1)   * 1D4  % AEpeOH
C     P3   = B(S+2)   / TEN  % Y0
      P4   = B(S+3)          % k peSH
      P5   = B(S+4)   * 1D4  % AEpeSH
      P6   = B(S+5)          % k stOH
      P7   = B(S+6)   * 1D4  % AEstop
      P8   = B(S+7)          % k stAQ
      P9   = B(S+8)          % k cleave
      P10  = B(S+9)   * 1D4  % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &     + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
```

```
&         + DEXP(P8 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH              % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)           % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

 127  CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstop
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)        % k cleave
      P10 = B(S+9)  * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
&         + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
&         + DEXP(P8 - P7 * RTEMP) * SQRTT * SQRTAQ
      K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH              % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)           % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9903

 128  CONTINUE
      GO TO 9903

 129  CONTINUE
      GO TO 9903

 130  CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
      P3  = B(S+2)  * TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      Y1  = Y(P)   % native
      Y2  = Y(P+1) % oxidized
      AQ  = Y(P+2) % anthraquinone
      ALPHA = (25.4D0 - EX(EOB, 9) - EX(EOB, 10)) * .4D0
&           + (30.1D0 - EX(EOB, 7)) * .15D0
&           + 1.24208D0
```

```
      IF(P3 .GE. 38.6D0) GO TO 7130
         BETA   = 38.6D0 - Y1 - Y2
         IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (38.6D0 - P3)
         OH = OH - (0.4D0 * BETA + ALPHA) / 160.D0
         GAMMA = 0.056D0 * (30.1D0 - EX(EOB, 7)) / 224.D0
         IF(Y1 + Y2 .GT. P3) GAMMA = GAMMA * BETA / (38.6D0 - P3)
         SH = SH - GAMMA
         GO TO 8130
 7130 CONTINUE
         BETA   = P3 - Y1 - Y2
         OH = OH - (0.4D0 * BETA + ALPHA) / 160.D0
         GAMMA = 0.056D0 * (30.1D0 - EX(EOB, 7)) / 224.D0
         SH = SH - GAMMA
 8130 CONTINUE
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &       + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH            % k'cleave
      K4   = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
      K5   = DEXP(9.251D0 - 4738.D0 / TEMP) * SQRTT      % k'initial
     &       * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      YPRIME(P+2) = -K4 * AQ
      GO TO 9903

C-----Galactoglucomannan [GGM]

 9903 CONTINUE
      S = POINTR(3)
      C = CHOOSE(3)
      P = POINTY(3)
      GO TO(201, 202, 203, 204, 205, 206, 207, 208, 209, 210
     &    , 211, 212, 213, 214, 215, 216, 217, 218, 219, 220
     &    , 221, 222, 223, 224, 225, 226, 227, 228, 229, 230), C
      IF(C .NE. 0) CALL MESSAG(6, "   FCN", 9903, .TRUE.)
        GO TO 9904

  201 CONTINUE
      P1 = B(S)            % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % Y0
      YPRIME(P) = -Y(P) * SQRTT * DEXP(P1 - P2 * RTEMP)
     & * OH
      GO TO 9904

  202 CONTINUE
      P1 = B(S)            % k
      P2 = B(S+1) * 1D4 % AE
```

```
C       P3 = B(S+2) / TEN % YO
        YPRIME(P) = -Y(P) * Y(P) * SQRTT * DEXP(P1 - P2 * RTEMP)
     &  * OH
        GO TO 9904

  203   CONTINUE
        P1 = B(S)          % k  OH
        P2 = B(S+1) * 1D4  % AE common
C       P3 = B(S+2) / TEN  % YO
        P4 = B(S+3)        % k  SH
        P5 = B(S+4)        % k  AQ                         % 23 Jan 85 %%
        Y1 = DMIN1(Y(P), TEN)                              % 31Dec84 %%
        YPRIME(P) = -Y1 * SQRTT * SQRTOH
     &            * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &               + SQRTSH * DEXP(P4 - P2 * RTEMP)
     &               - SQRTAQ * DEXP(P5 - P2 * RTEMP))     % 23 Jan 85 %%
        GO TO 9904

  204   CONTINUE
        P1 = B(S  )        % k  OH
        P2 = B(S+1) * 1D4  % AE OH
C       P3 = B(S+2) / TEN  % YO
        P4 = B(S+3)        % k  SH
        P5 = B(S+4) * 1D4  % AE SH
        P6 = B(S+5)        % k  AQ
        P7 = B(S+6) * 1D4  % AE AQ
        Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
        YPRIME(P) = -Y1 * SQRTT * SQRTOH
     &            * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &               + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &               - SQRTAQ * DEXP(P6 - P7 * RTEMP))
        GO TO 9904

  205   CONTINUE
        P1 = B(S  )        % k  OH
        P2 = B(S+1) * 1D4  % AE OH
C       P3 = B(S+2) / TEN  % YO
        P4 = B(S+3)        % k  SH
        P5 = B(S+4) * 1D4  % AE SH
        P6 = B(S+5)        % k  AQ                         % 23 Jan 85 %%
        P7 = B(S+6) * 1D4  % AE AQ                         % 23 Jan 85 %%
        Y1 = DMIN1(Y(P), TEN)                              % 31Dec84 %%
        YPRIME(P) = -(Y1 * Y1) * SQRTT * SQRTOH
     &            * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &               + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &               - SQRTAQ * DEXP(P6 - P7 * RTEMP))     % 23 Jan 85 %%
        GO TO 9904

  206   CONTINUE
        P1   = B(S  )         % k  OH
        P2   = B(S+1) * 1D4   % AE OH
C       P3   = B(S+2) / TEN   % YO
        P4   = B(S+3)         % k  SH
        P5   = B(S+4) * 1D4   % AE SH
```

```
      P6  = B(S+5)         % k  AQ
      P7  = B(S+6) * 1D4   % AE AQ
      P8  = B(S+7) / TEN   % SHpower
      P9  = B(S+8) / TEN   % AQpower
      Y1  = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
      YPRIME(P) = 0D0; IF(Y1 .GT. 0D0)
     & YPRIME(P) = -Y1 * SQRTT
     &           * (  OH     * DEXP(P1 - P2 * RTEMP)
     &             + SH**P8 * DEXP(P4 - P5 * RTEMP)
     &             - AQ**P9 * DEXP(P6 - P7 * RTEMP))
      GO TO 9904

  207 CONTINUE
      P1  = B(S  )         % k peel
      P2  = B(S+1)  * 1D4  % AEpeel
C     P3  = B(S+2)  / TEN  % Y0
C     P4  = B(S+3)  / 1D2  % YSH
C     P5  = B(S+4)  / 1D2  % YAQ
      YPRIME(P)   = -Y(P) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9904

  208 CONTINUE
      P1  = B(S  )         % k peel
      P2  = B(S+1)  * 1D4  % AEpeel
C     P3  = B(S+2)  / TEN  % Y0
      P4  = B(S+3)         % k stop
      P5  = B(S+4)  * 1D4  % AEstop
      P6  = B(S+5)         % k cleave
      P7  = B(S+6)  * 1D4  % AEcleave
C     P8  = B(S+7)  / 1D2  % YSH
C     P9  = B(S+8)  / 1D2  % YAQ
      K1  = SQRTT * OH * DEXP(P1 - P2 * RTEMP) % kp
      K2  = SQRTT * OH * DEXP(P4 - P5 * RTEMP) % ks
      K3  = SQRTT * OH * DEXP(P6 - P7 * RTEMP) % kc
      Y1  = Y(P)           % virgin
      Y2  = Y(P+1)         % oxidized
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

  209 CONTINUE
      P1  = B(S  )         % k peel
      P2  = B(S+1)  * 1D4  % AEpeel
C     P3  = B(S+2)  / TEN  % Y0
      P4  = B(S+3)         % OHpower
C     P5  = B(S+4)         % YSHpower
C     P6  = B(S+5)         % YAQpower
C     P7  = B(S+6)  / 1D2  % YSH
C     P8  = B(S+7)  / 1D2  % YAQ
      YPRIME(P) = -Y(P) * SQRTT * OH**P4 * DEXP(P1 - P2 * RTEMP)
      GO TO 9904

  210 CONTINUE
      P1  = B(S  )         % k peel
```

```
          P2  = B(S+1)  * 1D4 % AEpeel
C         P3  = B(S+2)  / TEN % YO
          P4  = B(S+3)        % k stop
          P5  = B(S+4)  * 1D4 % AEstop
          P6  = B(S+5)        % k cleave
          P7  = B(S+6)  * 1D4 % AEcleave
          P8  = B(S+7)        % OHpeel power
          P9  = B(S+8)        % OHstop power
          P10 = B(S+9)        % OHcleave power
C         P11 = B(S+10)       % YSHpower
C         P12 = B(S+11)       % YAQpower
C         P13 = B(S+12) / 1D2 % YSH
C         P14 = B(S+13) / 1D2 % YAQ
          K1  = SQRTT * OH**P8  * DEXP(P1 - P2 * RTEMP) % kp
          K2  = SQRTT * OH**P9  * DEXP(P4 - P5 * RTEMP) % ks
          K3  = SQRTT * OH**P10 * DEXP(P6 - P7 * RTEMP) % kc
          Y1  = Y(P)                           % virgin
          Y2  = Y(P+1)                         % oxidized
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9904

  211     CONTINUE
          P1  = B(S  )        % k peel
          P2  = B(S+1)  * 1D4 % AEpeel
C         P3  = B(S+2)  / TEN % YO
          P4  = B(S+3)        % k stOH
          P5  = B(S+4)  * 1D4 % AEstOH
          P6  = B(S+5)        % k stAQ
          P7  = B(S+6)  * 1D4 % AEstAQ
          P8  = B(S+7)        % k cleave
          P9  = B(S+8)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
          K2  = DEXP(P4 - P5 * RTEMP) * SQRTT * OH            % k'stop
     &          + DEXP(P6 - P7 * RTEMP) * SQRTT * SQRTAQ
          K3  = DEXP(P8 - P9 * RTEMP) * SQRTT * OH            % k'cleave
          Y1  = DMIN1(Y(P) , TEN)          % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1),TEN)          % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9904

  212     CONTINUE
          P1  = B(S  )        % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % YO
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstOH
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)  * 1D4 % AEstAQ
          P10 = B(S+9)        % k cleave
          P11 = B(S+10) * 1D4 % AEcleave
```

```
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &       + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)         % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

  213 CONTINUE
      P1   = B(S   )          % k peOH
      P2   = B(S+1)  * 1D4 % AEpeOH
C     P3   = B(S+2)  / TEN % Y0
      P4   = B(S+3)          % k peSH
      P5   = B(S+4)  * 1D4 % AEpeSH
      P6   = B(S+5)          % k stOH
      P7   = B(S+6)  * 1D4 % AEstOH
      P8   = B(S+7)          % k stAQ
      P9   = B(S+8)  * 1D4 % AEstAQ
      P10  = B(S+9)          % k cleave
      P11  = B(S+10) * 1D4 % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &       + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)         % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

  214 CONTINUE
      P1   = B(S   )          % k peOH
      P2   = B(S+1)  * 1D4 % AEpeOH
C     P3   = B(S+2)  / TEN % Y0
      P4   = B(S+3)          % k peSH
      P5   = B(S+4)  * 1D4 % AEpeSH
      P6   = B(S+5)          % k stOH
      P7   = B(S+6)  * 1D4 % AEstOH
      P8   = B(S+7)          % k stAQ
      P9   = B(S+8)  * 1D4 % AEstAQ
      P10  = B(S+9)          % k cleave
      P11  = B(S+10) * 1D4 % AEcleave
      P12  = B(S+11)         % SH0
      P13  = B(S+12)         % AQ0
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SH / (SH + P12)
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &       + DEXP(P8 - P9 * RTEMP) * SQRTT * AQ / (AQ + P13)
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)         % virgin   % 31Dec84 %%%%%%
```

```
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

  215 CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9904

  216 CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9904
```

```
217     CONTINUE
        P1  = B(S  )          % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)          % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)          % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
        P8  = B(S+7)          % k stAQ
        P9  = B(S+8)  * 1D4 % AEstAQ
        P10 = B(S+9)          % k cleave
        P11 = B(S+10) * 1D4 % AEcleave
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
        Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
        Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
        Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
        YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
        YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
        YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
        GO TO 9904

218     CONTINUE
        P1  = B(S  )          % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)          % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)          % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
        P8  = B(S+7)          % k stAQ
        P9  = B(S+8)  * 1D4 % AEstAQ
        P10 = B(S+9)          % k cleave
        P11 = B(S+10) * 1D4 % AEcleave
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
        Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
        Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
        Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
        YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
        YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
        YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
        GO TO 9904

219     CONTINUE
        P1  = B(S  )          % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
```

```
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)        % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)        % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
        P8  = B(S+7)        % k stAQ
        P9  = B(S+8)  * 1D4 % AEstAQ
        P10 = B(S+9)        % k cleave
        P11 = B(S+10) * 1D4 % AEcleave
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH             % k'cleave
        Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
        Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
        YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
        YPRIME(P+1) = K2 * Y1 - K3 * Y2
        GO TO 9904

 220    CONTINUE
        P1  = B(S  )        % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)        % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)        % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
        P8  = B(S+7)        % k stAQ
        P9  = B(S+8)  * 1D4 % AEstAQ
        P10 = B(S+9)        % k cleave
        P11 = B(S+10) * 1D4 % AEcleave
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH             % k'cleave
        Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
        Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
        YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
        YPRIME(P+1) = K2 * Y1 - K3 * Y2
        GO TO 9904

 221    CONTINUE
        P1  = B(S  )        % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % YO
        P4  = B(S+3)        % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
        YPRIME(P)   = -K1 * Y(P)
        GO TO 9904
```

```
 222     CONTINUE
         P1   = B(S   )          % k peOH
         P2   = B(S+1)   * 1D4 % AEpeOH
C        P3   = B(S+2)   / TEN % YO
         P4   = B(S+3)          % k peSH
         P5   = B(S+4)   * 1D4 % AEpeSH
         K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH             % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
         YPRIME(P)    = -K1 * Y(P)
         GO TO 9904

 223     CONTINUE
         P1   = B(S   )          % k peel
         P2   = B(S+1)   * 1D4 % AEpeel
C        P3   = B(S+2)   / TEN % YO
         P4   = B(S+3)          % k stOH
         P5   = B(S+4)   * 1D4 % AEstOH
         P6   = B(S+5)          % k stAQ
         P7   = B(S+6)   * 1D4 % AEstAQ
         P8   = B(S+7)          % k cleave
         P9   = B(S+8)   * 1D4 % AEcleave
         K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH             % k'peel
         K2   = DEXP(P4 - P5 * RTEMP) * SQRTT * OH             % k'stop
     &        + DEXP(P6 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
         K3   = DEXP(P8 - P9 * RTEMP) * SQRTT * OH             % k'cleave
         Y1   = DMIN1(Y(P) , TEN)          % virgin   % 31Dec84 %%%%%%
         Y2   = DMIN1(Y(P+1),TEN)          % oxidized % 31Dec84 %%%%%%
         YPRIME(P)    = K3 * Y2 - (K1 + K2) * Y1
         YPRIME(P+1) = K2 * Y1 - K3 * Y2
         GO TO 9904

 224     CONTINUE
         P1   = B(S   )          % k peOH
         P2   = B(S+1)   * 1D4 % AEpeOH
C        P3   = B(S+2)   / TEN % YO
         P4   = B(S+3)          % k peSH
         P5   = B(S+4)   * 1D4 % AEpeSH
         P6   = B(S+5)          % k stOH
         P7   = B(S+6)   * 1D4 % AEstop
         P8   = B(S+7)          % k stAQ
         P9   = B(S+8)          % k cleave
         P10  = B(S+9)   * 1D4 % AEcleave
         K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH             % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
         K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH             % k'stop
     &        + DEXP(P8 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
         K3   = DEXP(P9 - P10 * RTEMP) * SQRTT * OH            % k'cleave
         Y1   = DMIN1(Y(P)   , TEN)        % virgin   % 31Dec84 %%%%%%
         Y2   = DMIN1(Y(P+1), TEN)         % oxidized % 31Dec84 %%%%%%
         YPRIME(P)    = K3 * Y2 - (K1 + K2) * Y1
         YPRIME(P+1) = K2 * Y1 - K3 * Y2
         GO TO 9904

 225     CONTINUE
```

```
          P1  = B(S  )         % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstop
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)        % k cleave
          P10 = B(S+9)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &        + DEXP(P8 - P7 * RTEMP) * SQRTT * SQRTAQ
          K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH             % k'cleave
          Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9904

  226     CONTINUE
          P1  = B(S  )         % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstop
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)        % k cleave
          P10 = B(S+9)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &        + DEXP(P8 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
          K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH             % k'cleave
          Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9904

  227     CONTINUE
          P1  = B(S  )         % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstop
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)        % k cleave
          P10 = B(S+9)  * 1D4 % AEcleave
```

```
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
     &       + DEXP(P8 - P7 * RTEMP) * SQRTT * SQRTAQ
      K3   = DEXP(P9 - P10 * RTEMP) * SQRTT * OH                 % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

 228  CONTINUE
      P1   = B(S  )        % k peOH
      P2   = B(S+1) * 1D4  % AEpeOH
C     P3   = B(S+2) / TEN  % YO
      P4   = B(S+3)        % k peSH
      P5   = B(S+4) * 1D4  % AEpeSH
      P6   = B(S+5)        % k stop
      P7   = B(S+6) * 1D4  % AEstop
      P8   = B(S+7)        % k cleave
      P9   = B(S+8) * 1D4  % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
      K3   = DEXP(P8 - P9 * RTEMP) * SQRTT * OH                  % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

 229  CONTINUE
      P1   = B(S  )        % k peOH
      P2   = B(S+1) * 1D4  % AEpeOH
C     P3   = B(S+2) / TEN  % YO
      P4   = B(S+3)        % k peSH
      P5   = B(S+4) * 1D4  % AEpeSH
      P6   = B(S+5)        % k stop
      P7   = B(S+6) * 1D4  % AEstop
      P8   = B(S+7)        % k cleave
      P9   = B(S+8) * 1D4  % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
      K3   = DEXP(P8 - P9 * RTEMP) * SQRTT * OH                  % k'cleave
      Y1   = DMIN1(Y(P)   , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)           % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9904

 230  CONTINUE
      P1   = B(S  )        % k peOH
      P2   = B(S+1) * 1D4  % AEpeOH
```

```
      P3  = B(S+2)  * TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      Y1  = Y(P)   % native
      Y2  = Y(P+1) % oxidized
      AQ  = Y(P+2) % anthraquinone
      ALPHA = (47.3D0 - EX(EOB, 8) - EX(EOB, 10)) * .4D0
     &        + (30.1D0 - EX(EOB, 7)) * .15D0
     &        + 1.24208D0
      BETA  = 16.7D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (16.7D0 - P3)
      OH = OH - (0.4D0 * BETA + ALPHA) / 160.D0
      GAMMA = 0.056D0 * (30.1D0 - EX(EOB, 7)) / 224.D0
      IF(Y1 + Y2 .GT. P3) GAMMA = GAMMA * BETA / (16.7D0 - P3)
      SH = SH - GAMMA
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
      K5  = DEXP(9.251D0 - 4738.D0 / TEMP) * SQRTT      % k'initial
     &    * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      YPRIME(P+2) = -K4 * AQ
      GO TO 9904

C-----Arabinoxylan

 9904 CONTINUE
      S = POINTR(4)
      C = CHOOSE(4)
      P = POINTY(4)
      GO TO(301, 302, 303, 304, 305, 306, 307, 308, 309, 310
     &    , 311, 312, 313, 314, 315, 316, 317, 318, 319, 320
     &    , 321, 322, 323, 324, 325, 326, 327, 328, 329, 330
     &    , 331, 332), C
      IF(C .NE. 0) CALL MESSAG(6, "   FCN", 9904, .TRUE.)
        GO TO 9905

  301 CONTINUE
      P1 = B(S)         % k
      P2 = B(S+1) * 1D4 % AE
```

```
C         P3 = B(S+2) / TEN % YO
          YPRIME(P) = -Y(P) * SQRTT * DEXP(P1 - P2 * RTEMP)
     &    * OH
          GO TO 9905

  302     CONTINUE
          P1 = B(S)          % k
          P2 = B(S+1) * 1D4  % AE
C         P3 = B(S+2) / TEN  % YO
          YPRIME(P) = -Y(P) * Y(P) * SQRTT * DEXP(P1 - P2 * RTEMP)
     &    * OH
          GO TO 9905

  303     CONTINUE
          P1 = B(S)          % k  OH
          P2 = B(S+1) * 1D4  % AE common
C         P3 = B(S+2) / TEN  % YO
          P4 = B(S+3)        % k  SH
          P5 = B(S+4)        % k  AQ
          Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
          YPRIME(P) = -Y1 * SQRTT * SQRTOH
     &              * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                 + SQRTSH * DEXP(P4 - P2 * RTEMP)
     &                 - SQRTAQ * DEXP(P5 - P2 * RTEMP))
          GO TO 9905

  304     CONTINUE
          P1 = B(S  )        % k  OH
          P2 = B(S+1) * 1D4  % AE OH
C         P3 = B(S+2) / TEN  % YO
          P4 = B(S+3)        % k  SH
          P5 = B(S+4) * 1D4  % AE SH
          P6 = B(S+5)        % k  AQ
          P7 = B(S+6) * 1D4  % AE AQ
          Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
          YPRIME(P) = -Y1 * SQRTT * SQRTOH
     &              * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                 + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &                 - SQRTAQ * DEXP(P6 - P7 * RTEMP))
          GO TO 9905

  305     CONTINUE
          P1 = B(S  )        % k  OH
          P2 = B(S+1) * 1D4  % AE OH
C         P3 = B(S+2) / TEN  % YO
          P4 = B(S+3)        % k  SH
          P5 = B(S+4) * 1D4  % AE SH
          P6 = B(S+5)        % k  AQ
          P7 = B(S+6) * 1D4  % AE AQ
          Y1 = DMIN1(Y(P), TEN)                          % 31Dec84 %%%%%%
          YPRIME(P) = -(Y1 * Y1) * SQRTT * SQRTOH
     &              * (  SQRTOH * DEXP(P1 - P2 * RTEMP)
     &                 + SQRTSH * DEXP(P4 - P5 * RTEMP)
     &                 - SQRTAQ * DEXP(P6 - P7 * RTEMP))
```

```
      GO TO 9905

 306  CONTINUE
      P1  = B(S  )         % k  OH
      P2  = B(S+1) * 1D4   % AE OH
C     P3  = B(S+2) / TEN   % YO
      P4  = B(S+3)         % k  SH
      P5  = B(S+4) * 1D4   % AE SH
      P6  = B(S+5)         % k  AQ
      P7  = B(S+6) * 1D4   % AE AQ
      P8  = B(S+7)         % Ypower
      P9  = B(S+8)         % OHpower
      P10 = B(S+9)         % OH(SH)power
      P11 = B(S+10)        % SHpower
      P12 = B(S+11)        % OH(AQ)power
      P13 = B(S+12)        % AQpower
      Y1  = DMIN1(Y(P), TEN)                        % 31Dec84 %%%%%%
      YPRIME(P) = 0D0; IF(Y1 .GT. 0D0)
   &  YPRIME(P) = -(Y1**P8) * SQRTT
   &             * (  OH**P9             * DEXP(P1 - P2 * RTEMP)
   &                + OH**P10 * SH**P11  * DEXP(P4 - P5 * RTEMP)
   &                - OH**P12 * AQ**P13  * DEXP(P6 - P7 * RTEMP))
      GO TO 9905

 307  CONTINUE
      P1  = B(S  )         % k peel
      P2  = B(S+1)  * 1D4  % AEpeel
C     P3  = B(S+2)  / TEN  % YO
C     P4  = B(S+3)  / 1D2  % YSH
C     P5  = B(S+4)  / 1D2  % YAQ
      YPRIME(P)   = -Y(P) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9905

 308  CONTINUE
      P1  = B(S  )         % k peel
      P2  = B(S+1)  * 1D4  % AEpeel
C     P3  = B(S+2)  / TEN  % YO
      P4  = B(S+3)         % k stop
      P5  = B(S+4)  * 1D4  % AEstop
      P6  = B(S+5)         % k cleave
      P7  = B(S+6)  * 1D4  % AEcleave
C     P8  = B(S+7)  / 1D2  % YSH
C     P9  = B(S+8)  / 1D2  % YAQ
      K1  = SQRTT * OH * DEXP(P1 - P2 * RTEMP) % kp
      K2  = SQRTT * OH * DEXP(P4 - P5 * RTEMP) % ks
      K3  = SQRTT * OH * DEXP(P6 - P7 * RTEMP) % kc
      Y1  = Y(P)           % virgin
      Y2  = Y(P+1)         % oxidized
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9905

 309  CONTINUE
      P1  = B(S  )         % k peel
```

```
          P2  = B(S+1)  * 1D4 % AEpeel
C         P3  = B(S+2)  / TEN % YO
          P4  = B(S+3)        % OHpower
C         P5  = B(S+4)        % YSHpower
C         P6  = B(S+5)        % YAQpower
C         P7  = B(S+6)  / 1D2 % YSH
C         P8  = B(S+7)  / 1D2 % YAQ
          YPRIME(P) = -Y(P) * SQRTT * OH**P4 * DEXP(P1 - P2 * RTEMP)
          GO TO 9905

  310     CONTINUE
          P1  = B(S  )        % k peel
          P2  = B(S+1)  * 1D4 % AEpeel
C         P3  = B(S+2)  / TEN % YO
          P4  = B(S+3)        % k stop
          P5  = B(S+4)  * 1D4 % AEstop
          P6  = B(S+5)        % k cleave
          P7  = B(S+6)  * 1D4 % AEcleave
          P8  = B(S+7)        % OHpeel power
          P9  = B(S+8)        % OHstop power
          P10 = B(S+9)        % OHcleave power
C         P11 = B(S+10)       % YSHpower
C         P12 = B(S+11)       % YAQpower
C         P13 = B(S+12) / 1D2 % YSH
C         P14 = B(S+13) / 1D2 % YAQ
          K1  = SQRTT * OH**P8  * DEXP(P1 - P2 * RTEMP) % kp
          K2  = SQRTT * OH**P9  * DEXP(P4 - P5 * RTEMP) % ks
          K3  = SQRTT * OH**P10 * DEXP(P6 - P7 * RTEMP) % kc
          Y1  = Y(P)                          % virgin
          Y2  = Y(P+1)                        % oxidized
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9905

  311     CONTINUE
          P1  = B(S  )        % k peel
          P2  = B(S+1)  * 1D4 % AEpeel
C         P3  = B(S+2)  / TEN % YO
          P4  = B(S+3)        % k stOH
          P5  = B(S+4)  * 1D4 % AEstOH
          P6  = B(S+5)        % k stAQ
          P7  = B(S+6)  * 1D4 % AEstAQ
          P8  = B(S+7)        % k cleave
          P9  = B(S+8)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH           % k'peel
          K2  = DEXP(P4 - P5 * RTEMP) * SQRTT * OH           % k'stop
     &          + DEXP(P6 - P7 * RTEMP) * SQRTT * SQRTAQ
          K3  = DEXP(P8 - P9 * RTEMP) * SQRTT * OH           % k'cleave
          Y1  = DMIN1(Y(P) , TEN)           % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1),TEN)           % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9905
```

```
 312    CONTINUE
        P1  = B(S  )         % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % Y0
        P4  = B(S+3)         % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)         % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
        P8  = B(S+7)         % k stAQ
        P9  = B(S+8)  * 1D4 % AEstAQ
        P10 = B(S+9)         % k cleave
        P11 = B(S+10) * 1D4 % AEcleave
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
        Y1  = DMIN1(Y(P)  , TEN)        % virgin   % 31Dec84 %%%%%%
        Y2  = DMIN1(Y(P+1), TEN)        % oxidized % 31Dec84 %%%%%%
        YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
        YPRIME(P+1) = K2 * Y1 - K3 * Y2
        GO TO 9905

 313    CONTINUE
        P1  = B(S  )         % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % Y0
        P4  = B(S+3)         % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)         % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
        P8  = B(S+7)         % k stAQ
        P9  = B(S+8)  * 1D4 % AEstAQ
        P10 = B(S+9)         % k cleave
        P11 = B(S+10) * 1D4 % AEcleave
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
        Y1  = DMIN1(Y(P)  , TEN)        % virgin   % 31Dec84 %%%%%%
        Y2  = DMIN1(Y(P+1), TEN)        % oxidized % 31Dec84 %%%%%%
        YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
        YPRIME(P+1) = K2 * Y1 - K3 * Y2
        GO TO 9905

 314    CONTINUE
        P1  = B(S  )         % k peOH
        P2  = B(S+1)  * 1D4 % AEpeOH
C       P3  = B(S+2)  / TEN % Y0
        P4  = B(S+3)         % k peSH
        P5  = B(S+4)  * 1D4 % AEpeSH
        P6  = B(S+5)         % k stOH
        P7  = B(S+6)  * 1D4 % AEstOH
```

```
      P8  = B(S+7)          % k stAQ
      P9  = B(S+8)   * 1D4  % AEstAQ
      P10 = B(S+9)          % k cleave
      P11 = B(S+10)  * 1D4  % AEcleave
      P12 = B(S+11)         % SH0
      P13 = B(S+12)         % AQ0
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SH / (SH + P12)
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * AQ / (AQ + P13)
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9905

315   CONTINUE
      P1  = B(S  )          % k peOH
      P2  = B(S+1)   * 1D4  % AEpeOH
C     P3  = B(S+2)   / TEN  % Y0
      P4  = B(S+3)          % k peSH
      P5  = B(S+4)   * 1D4  % AEpeSH
      P6  = B(S+5)          % k stOH
      P7  = B(S+6)   * 1D4  % AEstOH
      P8  = B(S+7)          % k stAQ
      P9  = B(S+8)   * 1D4  % AEstAQ
      P10 = B(S+9)          % k cleave
      P11 = B(S+10)  * 1D4  % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH              % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      Y3  = DMIN1(Y(P+2), TEN)          % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9905

316   CONTINUE
      P1  = B(S  )          % k peOH
      P2  = B(S+1)   * 1D4  % AEpeOH
C     P3  = B(S+2)   / TEN  % Y0
      P4  = B(S+3)          % k peSH
      P5  = B(S+4)   * 1D4  % AEpeSH
      P6  = B(S+5)          % k stOH
      P7  = B(S+6)   * 1D4  % AEstOH
      P8  = B(S+7)          % k stAQ
      P9  = B(S+8)   * 1D4  % AEstAQ
      P10 = B(S+9)          % k cleave
      P11 = B(S+10)  * 1D4  % AEcleave
```

```
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
&          + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
&          + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
      Y1   = DMIN1(Y(P)  , TEN)         % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)         % oxidized % 31Dec84 %%%%%%
      Y3   = DMIN1(Y(P+2), TEN)         % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9905

317   CONTINUE
      P1  = B(S   )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
&          + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
&          + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
      Y1   = DMIN1(Y(P)  , TEN)         % virgin   % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)         % oxidized % 31Dec84 %%%%%%
      Y3   = DMIN1(Y(P+2), TEN)         % cleaved  % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9905

318   CONTINUE
      P1  = B(S   )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
&          + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2   = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
&          + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
```

```
      K3   = DEXP(P10 - P11 * RTEMP) * SQRTT * OH               % k'cleave
      Y1   = DMIN1(Y(P)  , TEN)          % virgin    % 31Dec84 %%%%%%
      Y2   = DMIN1(Y(P+1), TEN)          % oxidized  % 31Dec84 %%%%%%
      Y3   = DMIN1(Y(P+2), TEN)          % cleaved   % 31Dec84 %%%%%%
      YPRIME(P)   = -(K1 + K2) * (Y1 + Y3)  +  K3 * Y2
      YPRIME(P+1) = K2 * (Y1 + Y3)  -  K3 * Y2
      YPRIME(P+2) = -(K1 + K2) * Y3  +  K3 * (Y1 + Y3)
      GO TO 9905

319   CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH                % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin    % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized  % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9905

320   CONTINUE
      P1  = B(S  )        % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
C     P3  = B(S+2)  / TEN % Y0
      P4  = B(S+3)        % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)        % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)        % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)        % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH                  % k'peel
     &    + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH                  % k'stop
     &    + DEXP(P8 - P9 * RTEMP) * SQRTT * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH                % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin    % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized  % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9905
```

```
  321    CONTINUE
         GO TO 9905

  322    CONTINUE
         GO TO 9905

  323    CONTINUE
         P1  = B(S  )         % k peel
         P2  = B(S+1)  * 1D4 % AEpeel
C        P3  = B(S+2)  / TEN % Y0
         P4  = B(S+3)        % k stOH
         P5  = B(S+4)  * 1D4 % AEstOH
         P6  = B(S+5)        % k stAQ
         P7  = B(S+6)  * 1D4 % AEstAQ
         P8  = B(S+7)        % k cleave
         P9  = B(S+8)  * 1D4 % AEcleave
         K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
         K2  = DEXP(P4 - P5 * RTEMP) * SQRTT * OH              % k'stop
     &       + DEXP(P6 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
         K3  = DEXP(P8 - P9 * RTEMP) * SQRTT * OH              % k'cleave
         Y1  = DMIN1(Y(P) , TEN)          % virgin   % 31Dec84 %%%%%%
         Y2  = DMIN1(Y(P+1),TEN)          % oxidized % 31Dec84 %%%%%%
         YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
         YPRIME(P+1) = K2 * Y1 - K3 * Y2
         GO TO 9905

  324    CONTINUE
         P1  = B(S  )         % k peOH
         P2  = B(S+1)  * 1D4 % AEpeOH
C        P3  = B(S+2)  / TEN % Y0
         P4  = B(S+3)        % k peSH
         P5  = B(S+4)  * 1D4 % AEpeSH
         P6  = B(S+5)        % k stOH
         P7  = B(S+6)  * 1D4 % AEstop
         P8  = B(S+7)        % k stAQ
         P9  = B(S+8)        % k cleave
         P10 = B(S+9)  * 1D4 % AEcleave
         K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &       + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
         K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &       + DEXP(P8 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
         K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH             % k'cleave
         Y1  = DMIN1(Y(P)   , TEN)        % virgin   % 31Dec84 %%%%%%
         Y2  = DMIN1(Y(P+1), TEN)         % oxidized % 31Dec84 %%%%%%
         YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
         YPRIME(P+1) = K2 * Y1 - K3 * Y2
         GO TO 9905

  325    CONTINUE
         P1  = B(S  )         % k peOH
         P2  = B(S+1)  * 1D4 % AEpeOH
C        P3  = B(S+2)  / TEN % Y0
         P4  = B(S+3)        % k peSH
```

```
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstop
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)        % k cleave
          P10 = B(S+9)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &        + DEXP(P8 - P7 * RTEMP) * SQRTT * SQRTAQ
          K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH             % k'cleave
          Y1  = DMIN1(Y(P)  , TEN)        % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1), TEN)        % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9905

  326     CONTINUE
          P1  = B(S  )        % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstop
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)        % k cleave
          P10 = B(S+9)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &        + DEXP(P8 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
          K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH             % k'cleave
          Y1  = DMIN1(Y(P)  , TEN)        % virgin   % 31Dec84 %%%%%%
          Y2  = DMIN1(Y(P+1), TEN)        % oxidized % 31Dec84 %%%%%%
          YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
          YPRIME(P+1) = K2 * Y1 - K3 * Y2
          GO TO 9905

  327     CONTINUE
          P1  = B(S  )        % k peOH
          P2  = B(S+1)  * 1D4 % AEpeOH
C         P3  = B(S+2)  / TEN % Y0
          P4  = B(S+3)        % k peSH
          P5  = B(S+4)  * 1D4 % AEpeSH
          P6  = B(S+5)        % k stOH
          P7  = B(S+6)  * 1D4 % AEstop
          P8  = B(S+7)        % k stAQ
          P9  = B(S+8)        % k cleave
          P10 = B(S+9)  * 1D4 % AEcleave
          K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH              % k'peel
     &        + DEXP(P4 - P5 * RTEMP) * SQRTT * SH
          K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH              % k'stop
     &        + DEXP(P8 - P7 * RTEMP) * SQRTT * SQRTAQ
```

```
      K3  = DEXP(P9 - P10 * RTEMP) * SQRTT * OH              % k'cleave
      Y1  = DMIN1(Y(P)  , TEN)          % virgin   % 31Dec84 %%%%%%
      Y2  = DMIN1(Y(P+1), TEN)          % oxidized % 31Dec84 %%%%%%
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      GO TO 9905

328   CONTINUE
      GO TO 9905

329   CONTINUE
      GO TO 9905

330   CONTINUE
      P1  = B(S  )          % k peOH
      P2  = B(S+1)  * 1D4 % AEpeOH
      P3  = B(S+2)  * TEN % Y0
      P4  = B(S+3)          % k peSH
      P5  = B(S+4)  * 1D4 % AEpeSH
      P6  = B(S+5)          % k stOH
      P7  = B(S+6)  * 1D4 % AEstOH
      P8  = B(S+7)          % k stAQ
      P9  = B(S+8)  * 1D4 % AEstAQ
      P10 = B(S+9)          % k cleave
      P11 = B(S+10) * 1D4 % AEcleave
      Y1  = Y(P)    % native
      Y2  = Y(P+1)  % oxidized
      AQ  = Y(P+2) % anthraquinone
      ALPHA = (55.3D0 - EX(EOB, 8) - EX(EOB, 9)) * .4D0
     &        + (30.1D0 - EX(EOB, 7)) * .15D0
     &        + 1.24208D0
      BETA  = 8.7D0 - Y1 - Y2
      IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (8.7D0 - P3)
      OH = OH - (0.4D0 * BETA + ALPHA) / 160.D0
      GAMMA = 0.056D0 * (30.1D0 - EX(EOB, 7)) / 224.D0
      IF(Y1 + Y2 .GT. P3) GAMMA = GAMMA * BETA / (8.7D0 - P3)
      SH = SH - GAMMA
      OH = DMAX1(OH, 0.D0)
      SH = DMAX1(SH, 0.D0)
      AQ = DMAX1(AQ, 0.D0)
      K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH               % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
      K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH               % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
      K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH             % k'cleave
      K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
      K5  = DEXP(9.251D0 - 4738.D0 / TEMP) * SQRTT      % k'initial
     &      * (ONE - (ONE / (ONE + OH * 100.D0)))
      YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
      IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
      YPRIME(P+1) = K2 * Y1 - K3 * Y2
      YPRIME(P+2) = -K4 * AQ
      GO TO 9905
```

```
331     CONTINUE
        P1   = B(S  )          % k peOH
        P2   = B(S+1)   * 1D4  % AEpeOH
        P3   = B(S+2)   * TEN  % YO
        P4   = B(S+3)          % k stOH
        P5   = B(S+4)   * 1D4  % AEstOH
        P6   = B(S+5)          % k stAQ
        P7   = B(S+6)   * 1D4  % AEstAQ
        P8   = B(S+7)          % k cleave
        P9   = B(S+8)   * 1D4  % AEcleave
        Y1   = Y(P)   % native
        Y2   = Y(P+1) % oxidized
        AQ   = Y(P+2) % anthraquinone
        ALPHA = (55.3D0 - EX(EOB, 8) - EX(EOB, 9)) * .4D0
   &          + (30.1D0 - EX(EOB, 7)) * .15D0
   &          + 1.24208D0
        BETA  = 8.7D0 - Y1 - Y2
        IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (8.7D0 - P3)
        OH = OH - (0.4D0 * BETA + ALPHA) / 160.D0
        GAMMA = 0.056D0 * (30.1D0 - EX(EOB, 7)) / 224.D0
        IF(Y1 + Y2 .GT. P3) GAMMA = GAMMA * BETA / (8.7D0 - P3)
        SH = SH - GAMMA
        OH = DMAX1(OH, 0.D0)
        SH = DMAX1(SH, 0.D0)
        AQ = DMAX1(AQ, 0.D0)
        K1   = DEXP(P1 - P2 * RTEMP) * SQRTT * OH          % k'peel
        K2   = DEXP(P4 - P5 * RTEMP) * SQRTT * OH          % k'stop
   &        + DEXP(P6 - P7 * RTEMP) * SQRTT * OH * SQRTAQ
        K3   = DEXP(P8 - P9 * RTEMP) * SQRTT * OH          % k'cleave
        K4   = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
        K5   = DEXP(9.251D0 - 4738.D0 / TEMP) * SQRTT      % k'initial
   &        * (ONE - (ONE / (ONE + OH * 100.D0)))
        YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
        IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * BETA
        YPRIME(P+1) = K2 * Y1 - K3 * Y2
        YPRIME(P+2) = -K4 * AQ
        GO TO 9905

332     CONTINUE
        P1   = B(S  )          % k peOH
        P2   = B(S+1)   * 1D4  % AEpeOH
        P3   = B(S+2)   * TEN  % YO
        P4   = B(S+3)          % k peSH
        P5   = 2500.D+0        % AEpeSH
        P6   = B(S+4)          % k stOH
        P7   = B(S+5)   * 1D4  % AEstOH
        P8   = B(S+6)          % k stAQ
        P9   = B(S+7)   * 1D4  % AEstAQ
        P10  = B(S+8)          % k cleave
        P11  = B(S+9)   * 1D4  % AEcleave
        Y1   = Y(P)   % native
        Y2   = Y(P+1) % oxidized
        AQ   = Y(P+2) % anthraquinone
        ALPHA = (55.3D0 - EX(EOB, 8) - EX(EOB, 9)) * .4D0
```

```
     &         + (30.1D0 - EX(EOB, 7)) * .15D0
     &         + 1.24208D0
        BETA  = 8.7D0 - Y1 - Y2
        IF(Y1 + Y2 .GT. P3) ALPHA = ALPHA * BETA / (8.7D0 - P3)
        OH = OH - (0.4D0 * BETA + ALPHA) / 160.D0
        GAMMA = 0.056D0 * (30.1D0 - EX(EOB, 7)) / 224.D0
        IF(Y1 + Y2 .GT. P3) GAMMA = GAMMA * BETA / (8.7D0 - P3)
        SH = SH - GAMMA
        OH = DMAX1(OH, 0.D0)
        SH = DMAX1(SH, 0.D0)
        AQ = DMAX1(AQ, 0.D0)
        K1  = DEXP(P1 - P2 * RTEMP) * SQRTT * OH            % k'peel
     &      + DEXP(P4 - P5 * RTEMP) * SQRTT * SQRTSH
        K2  = DEXP(P6 - P7 * RTEMP) * SQRTT * OH            % k'stop
     &      + DEXP(P8 - P9 * RTEMP) * SQRTT * OH * SQRTAQ
        K3  = DEXP(P10 - P11 * RTEMP) * SQRTT * OH          % k'cleave
        K4  = DEXP(20.4828D0 - 10692.9D0 / TEMP) * SQRTT  % k'AQ
        K5  = DEXP(9.251D0 - 4738.D0 / TEMP) * SQRTT      % k'initial
     &      * (ONE - (ONE / (ONE + OH * 100.D0)))
        YPRIME(P)   = K3 * Y2 - (K1 + K2) * Y1
        IF(Y1 + Y2 .GT. P3) YPRIME(P) = YPRIME(P) - K5 * Y1
        YPRIME(P+1) = K2 * Y1 - K3 * Y2
        YPRIME(P+2) = -K4 * AQ
        GO TO 9905

C-----Extractives

 9905 CONTINUE
      S = POINTR(5)
      C = CHOOSE(5)
      P = POINTY(5)
      GO TO(401, 402, 403, 404, 405, 406, 407, 408, 409, 410
     &    , 411, 412, 413, 414, 415, 416, 417, 418, 419, 420
     &    , 421, 422), C
        IF(C .NE. 0) CALL MESSAG(6, "   FCN", 9905, .TRUE.)
          GO TO 9906

  401   CONTINUE
        P1 = B(S)
        P2 = B(S+1) * 1D4
        YPRIME(P) = -Y(P) * SQRTT
     &  * DEXP(P1 - P2 * RTEMP)
        GO TO 9906

  402   CONTINUE
        P1 = B(S)
        P2 = B(S+1) * 1D4
        YPRIME(P) = -Y(P) * Y(P) * SQRTT
     &  * DEXP(P1 - P2 * RTEMP)
        GO TO 9906

  403   CONTINUE
        P1 = B(S)
        P2 = B(S+1) * 1D4
```

```
      YPRIME(P) = -Y(P) * SQRTT
     &  * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 404  CONTINUE
      P1 = B(S  )         % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      P4 = B(S+3) *1D-1 % powrOH
      YPRIME(P) = -Y(P) * SQRTT * OH ** P4
     &            *  DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 405  CONTINUE
      P1 = B(S  )         % k  OH
      P2 = B(S+1) * 1D4 % AE OH
C     P3 = B(S+2) / TEN % YO
      P4 = B(S+3)         % k  SH
      P5 = B(S+4) * 1D4 % AE SH
      P6 = B(S+5)         % k  AQ
      P7 = B(S+6) * 1D4 % AE AQ
      P8 = B(S+7) *1D-1 % Yinfinity
      YPRIME(P) = (P8 - Y(P)) * SQRTT
     &            * (                   OH * DEXP(P1 - P2 * RTEMP)
     &              - SQRTOH * SQRTSH * DEXP(P4 - P5 * RTEMP)
     &              - SQRTOH * SQRTAQ * DEXP(P6 - P7 * RTEMP))
      GO TO 9906

 406  CONTINUE
      P1  = B(S  )        % k  OH
      P2  = B(S+1) * 1D4 % AE OH
C     P3  = B(S+2) / TEN % YO
      P4  = B(S+3)        % k  SH
      P5  = B(S+4) * 1D4 % AE SH
      P6  = B(S+5)        % k  AQ
      P7  = B(S+6) * 1D4 % AE AQ
      P8  = B(S+7) *1D-1 % Yinfinity
      P9  = B(S+8)        % Ypower
      P10 = B(S+9)        % OHpower
      P11 = B(S+10)       % OH(SH)power
      P12 = B(S+11)       % SHpower
      P13 = B(S+12)       % OH(AQ)power
      P14 = B(S+13)       % AQpower
      YPRIME(P) = 0D0; IF(Y(P) .GT. P8)
     & YPRIME(P) = -((Y(P) - P8) ** P9) * SQRTT
     &            * (  OH**P10             * DEXP(P1 - P2 * RTEMP)
     &              + OH**P11 * SH**P12 * DEXP(P4 - P5 * RTEMP)
     &              + OH**P13 * AQ**P14 * DEXP(P6 - P7 * RTEMP))
      GO TO 9906

 407  CONTINUE
      GO TO 9906

 408  CONTINUE
```

```
      GO TO 9906

 409  CONTINUE
      GO TO 9906

 410  CONTINUE
      GO TO 9906

 411  CONTINUE
      P1 = B(S  )       % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      YPRIME(P) = -Y(P) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 412  CONTINUE
      P1 = B(S  )       % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      YPRIME(P) = -(Y(P) ** 2) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 413  CONTINUE
      P1 = B(S  )       % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      YPRIME(P) = -(Y(P) ** 3) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 414  CONTINUE
      P1 = B(S  )       % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      YPRIME(P) = -(Y(P) ** 4) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 415  CONTINUE
      P1 = B(S  )       % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      Y1 = Y(P) -1.268D0 / 42.095D0
      YPRIME(P) = -Y1 * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 416  CONTINUE
      P1 = B(S  )       % k
      P2 = B(S+1) * 1D4 % AE
C     P3 = B(S+2) / TEN % YO
      Y1 = Y(P) - 1.268D0 %/ 42.095D0
      YPRIME(P) = -(Y1 ** 2) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
      GO TO 9906

 417  CONTINUE
      P1 = B(S  )       % k
```

```
            P2 = B(S+1) * 1D4 % AE
C           P3 = B(S+2) / TEN % YO
            Y1 = Y(P) -1.268D0 / 42.095D0
            YPRIME(P) = -(Y1 ** 3) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
            GO TO 9906

  418       CONTINUE
            P1 = B(S   )        % k
            P2 = B(S+1) * 1D4 % AE
C           P3 = B(S+2) / TEN % YO
            Y1 = Y(P) -1.268D0 / 42.095D0
            YPRIME(P) = -(Y1 ** 4) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
            GO TO 9906

  419       CONTINUE
            P1 = B(S   )        % k
            P2 = B(S+1) * 1D4 % AE
C           P3 = B(S+2) / TEN % YO
            Y1 = Y(P) -1.268D0 / 42.095D0
            YPRIME(P) = -Y1 * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
            GO TO 9906

  420       CONTINUE
            P1 = B(S   )        % k
            P2 = B(S+1) * 1D4 % AE
C           P3 = B(S+2) / TEN % YO
            Y1 = Y(P) -1.268D0 / 42.095D0
            YPRIME(P) = -(Y1 ** 2) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
            GO TO 9906

  421       CONTINUE
            P1 = B(S   )        % k
            P2 = B(S+1) * 1D4 % AE
C           P3 = B(S+2) / TEN % YO
            Y1 = Y(P) -1.268D0 / 42.095D0
            YPRIME(P) = -(Y1 ** 3) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
            GO TO 9906

  422       CONTINUE
            P1 = B(S   )        % k
            P2 = B(S+1) * 1D4 % AE
C           P3 = B(S+2) / TEN % YO
            Y1 = Y(P) -1.268D0 / 42.095D0
            YPRIME(P) = -(Y1 ** 4) * SQRTT * OH * DEXP(P1 - P2 * RTEMP)
            GO TO 9906

 9906 CONTINUE
      IF(.NOT. FHFACT) GO TO 9999
         P = POINTY(6)
         YPRIME(P) = DEXP(43.2D0 - 16113.D0 / TEMP)
      GO TO 9999

 9999 CONTINUE
      RETURN
      END       % FCN
```

```
#FILE (MARK)MDPE/DATA ON STUDENTS
 0, 13, 37,  1, 6, 5,                        % NF, NM, NR, NT, NV, NW
 3, 3, 5, 7, 7, 9, 5, 9, 8, 14, 9,11,11,% NP    [models]
 4, 1, 2,                                    % SW, ZOPT, IOPT
 2, 2,                                       % METH, MITER
 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11,12,13,% PM
 1, 1, 1, 1, 1, 1, 1, 2, 1,  2, 2, 2, 2,% FS
 1, 1, 1, 1, 1, 1, 1, 2, 1,  2, 2, 2, 2,% FT
 24   , 190   , 3    ,1   , 10   , 22,       % BVHI  [variables]
   .5, 130   ,  .5   ,0   ,  0   ,  0,       % BVLO  [variables]
  5   , 160   , 1.2   , .3,    .1, 20,       % BVOPT [variables]
 F,                                          % DOC
 F,                                          % DOD
 T, T, T, T, T, T, T, T, T,  T, T, T, T,% BEOLD [models]
 1, 1, 1, 1, 1, 1, 1, 1, 1,  1, 1, 1, 1,
-3.90444, 1.66144, 8.25810,
-2.96290, 1.63052, 9.47123,
-3.86662, 1.72128, 8.42143, -3.97109, -5.71134,
-3.73366, 1.91611, 8.43164, -6.04277, .340908, -6.56476, 2.61378,
-2.85332, 1.84239, 10.0059, -5.41381, .0985124, -5.68169, 2.02887,
-2.14748, 2.02469, 9.94019, -.321812, 1.51824, -4.81137, 2.28535,
  2.88204, 1.63491, 1.26482, 7.48882, .984311, .0542400,
-3.42910, 1.73989, 12.2884, -52.8431, 9.26329D-7,
-2.42961, 1.35405, 11.2756, -3.04743, .734387, -4.51641, .822044,
  -25.6742, 5.96788,
-3.19580, 2.13410, 12.7440, 1.33234, 3.61746D-8, .823876, -5.29342,
  .634090,
-4.24333, 1.69402, 10.0468, -5.24570, 1.48091, -6.08732, 1.59604,
  1.11976, .962314, 1.20731, .118783, 7.51205D-6, -22.7476, .0175199,
-1.80607, 1.42207, 10.9674, -2.70310, .679818, -3.32771, 1.33783,
  -3.98987, 1.21949,
-2.23539, 1.25839, 11.6311, -6.72146, 1.47744D-3, -3.04547, .506178,
  -4.60503, 1.66014, -4.28622, .974461,
-2.13937, 1.56045, 10.8315, -4.96625, 1.12583, -3.00669, .861331,
  -3.65685, 1.46634, -4.05272, 1.18162,
A-1
 2.390, 150, 1.002,  .198,  1.007, 17, % AV[variables x runs]
A-2
 2.390, 150, 1.002;  .198,  1.005, 17, % all runs here except C1-C3
A-3
 2.390, 150, 1.002,  .198,   .999, 17, % Cel = Glu - 1/3 Man
A-4
 2.390, 150, 1.002,  .198,  1.006, 17, % GGM = Gal + 4/3 Man
B-1
 1.367, 130,  .501,  .099,   .502, 16,
B-2
 4.228, 130, 2.007,  .099,  2.007, 16,
B-3
 4.228, 130,  .511,  .402,   .507, 16,
B-4
 1.367, 130, 2.018,  .402,  1.989, 16,
C-1
 1.326, 170,  .501,  .099,  2.004, 18,
```

```
C-2
 1.326, 170, 2.007,  .099,   .512, 18,
C-3
 4.299, 170,  .511,  .402,  1.995, 18,
C-4
 4.299, 170, 2.018,  .402,   .507, 18,
E-1
 0.473, 130, 1.002,  .198,  1.029, 19,
E-2
23.993, 136, 1.002,  .198,  1.013, 20,
E-3
 0.473, 190, 1.002,  .198,  1.007, 21,
E-4
 0.506, 170, 1.002,  .198,   .992, 19,
F-1
 0.604, 190, 1.002,  .198,  1.013, 22,
F-2
 0.477, 176, 1.002,  .198,  1.018, 24,
F-3
 2.459, 130, 1.002,  .198,   .993, 24,
F-4
24.008, 138, 1.002,  .198,   .999, 27,
I-1
 7.767, 190,  .420,  .099,  3.208, 18,
I-2
24.005, 130,  .128, 1.002,  0.   , 19,
I-3
 0.628, 190,  .094, 0.   ,  6.404, 18,
I-4
 3.396, 190,  .229,  .798,   .101, 18,
J-1
 5.824, 130, 1.085,  .662,  6.679, 21,
J-2
48.730, 130, 2.798, 0.   ,  9.780, 20,
J-3
48.730, 130, 3.001,  .846, 10.010, 20,
J-4
26.800, 130, 3.001,  .846, 10.012, 24,
K-1
19.994, 180, 2.482, 1.002,  9.999, 20,
K-2
20.018, 180, 2.475, 0.   ,  0.   , 20,
K-3
20.018, 180, 2.482, 1.002,  0.   , 20,
K-4
20.018, 180, 2.475, 0.   ,  9.996, 20,
M-1
22.054, 190,  .505, 1.002, 10.008, 19,
M-2
22.054, 190,  .505, 1.002,  0.   , 19,
M-3
22.054, 190,  .498, 0.   ,  9.999, 19,
M-4
22.054, 190,  .498, 0.   ,  0.   , 19,
```

```
N-1
21.180, 130,  .511, 0.    ,  9.990, 17,
N-2
 3.736, 144, 2.999, 0.    ,  0.    , 19,
N-3
 5.147, 152, 1.950, 0.005,  2.627, 17,
N-4
23.985, 144, 2.999, 0.    ,  0.    , 16,
A-1
 6.192, 33.803,  6.007, 2.984,  .165,  % AY[responses x runs]
A-2
 5.528, 35.784,  6.821, 3.071,  .129,
A-3
 5.611, 38.927,  7.195, 3.496,  .166,
A-4
 5.929, 39.572,  7.292, 3.676,  .180,
B-1
17.625, 38.363,  7.163, 7.296,  .249,
B-2
14.242, 35.402,  8.796, 5.123,  .282,
B-3
16.222, 38.698,  7.178, 7.000,  .174,
B-4
19.946, 37.065,  9.797, 6.273,  .556,
C-1
 4.270, 47.259,  7.788, 4.383,  .132,
C-2
 1.840, 51.703,  5.892,  .688,  .064,
C-3
 1.806, 53.147,  7.789, 3.659,  .074,
C-4
  .177, 21.046,  1.632,  .184,  .034,
E-1
26.214, 42.228,  9.956, 8.032, 0.248,
E-2
 1.669, 35.269,  5.944, 2.682, 0.055,
E-3
 1.132, 36.939,  5.589, 1.160, 0.089,
E-4
 9.856, 40.329,  7.833, 3.752, 0.219,
F-1
 0.494, 33.998,  4.895, 1.002, 0.087,
F-2
 5.015, 37.467,  6.353, 2.309, 0.154,
F-3
19.694, 36.500,  9.028, 5.301,  .273,
F-4
 1.181, 36.226,  5.960, 2.537,  .087,
I-1
  .174, 19.719,  2.002, 0.426,  .184,
I-2
13.441, 34.146,  7.993, 4.412,  .313,
I-3
12.421, 39.845,  4.665, 7.151,  .544,
```

```
I-4
  .404, 28.391,  2.624, 1.726,  .129,
J-1
 7.863, 35.779,  8.260, 4.743, 0.    ,
J-2
  .522, 31.923,  4.664,  .740, 0.    ,
J-3
  .424, 32.191,  2.888,  .601, 0.    ,
J-4
  .304, 34.519,  3.320,  .711, 0.    ,
K-1
 0.   , 0.    , 0.    , 0.   , 0.    ,
K-2
 0.   , 0.    , 0.    , 0.   , 0.    ,
K-3
 0.   , 0.    , 0.    , 0.   , 0.    ,
K-4
 0.   , 0.    , 0.    , 0.   , 0.    ,
M-1
  .028,   .348,   .025,  .003, 0.    ,
M-2
  .483,  4.260,   .332,  .067, 0.    ,
M-3
  .893,  8.849,   .685,  .086, 0.    ,
M-4
  .211,  6.842,   .461,  .011, 0.    ,
N-1
 9.791, 34.095,  9.258, 5.387, 3.600,
N-2
14.151, 32.998,  7.174, 3.255,  .290,
N-3
 1.187, 33.962,  5.100,  .931,  .091,
N-4
  .756, 22.067,  2.163,  .472,  .046,
1                               , % LABMDA
1.188E-4, 5.831E-3, 1.426E-3, 2.030E-3, 4.374E-5, % BSSW(responses)
.01                             , % H
0                               , % DEBUGG
8                               , % NSIG
T                               , % FIRST (calc prob from scratch)
F                               , % SECOND (do fix temp CMAX trials)
386.2,                            % DMAX
8.766D-3, 4.379D-9, 2.101D-25, 2.276D-19, % EMAX
190,                              % BTRIAL [trials]
C-1
 1.326, 170,  .501,  .099,  2.004, 18,
C-2
 1.326, 170, 2.007,  .099,   .512, 18,
C-3
 4.299, 170,  .511,  .402,  1.995, 18,
C-1
 4.270, 47.259,  7.788, 4.383,  .132,
C-2
 1.840, 51.703,  5.892,  .688,  .064,
C-3
 1.806, 53.147,  7.789, 3.659,  .074,
```

```
#FILE (MARK)MDPE/RESULTS/40/AX/PRELIM/2/2 ON STUDENTS

 enter INITAL

NF=0  NM=13  NR=40  NT=1  NV=6  NW=5

 NP , 3, 3, 5, 7, 7, 13, 5, 9, 8, 14, 9, 11, 11,

SW=4   (Arabinoxylan)

ZOPT=1   (ZXMWD)

METH=2   (Gear(stiff))
MITER=2   (Internal, full Jacobian)

 PM, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,

 FS, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 2,

 FT, 1, 1, 1, 1, 1, 1, 1, 2, 1, 2, 2, 2, 2,

 UPPER BOUNDS ON VARIABLES [Variables]

24.0, 190.0, 3.0, 1.0, 10.0, 22.0,

 LOWER BOUNDS ON VARIABLES [Variables]

0.5, 130.0, 0.5, 0.0, 0.0, 0.0,

 DEFAULT VALUES OF UNOPTIMIZED VARIABLES [Variables]

5.0, 160.0, 1.2, 0.3, 0.1, 20.0,

DOC=F

DOD=F

 READ IN PARAMETER ESTIMATION MAXIMA? [models]

T, T, T, T, T, T, T, T, T, T, T, T, T,

 PARAMETER ESTIMATION MAXIMA [models]

1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,

 PARAMETERS [Models x Parameters]

1, -3.90444, 1.66144, 8.2581,
2, -2.9629, 1.63052, 9.47123,
3, -3.86662, 1.72128, 8.42143, -3.97109, -5.71134,
```

4, -3.73366, 1.91611, 8.43164, -6.04277, 0.340908, -6.56476, 2.61378,
5, -2.85332, 1.84239, 10.0059, -5.41381, 0.0985124, -5.68169, 2.02887,
6, -2.14748, 2.02469, 9.94019, -0.321812, 1.51824, -4.81137, 2.28535,
6, 2.88204, 1.63491, 1.26482, 7.48882, 0.984311, 0.05424,
7, -3.4291, 1.73989, 12.2884, -52.8431, 9.26329D-7,
8, -2.42961, 1.35405, 11.2756, -3.04743, 0.734387, -4.51641, 0.822044,
8, -25.6742, 5.96788,
9, -3.1958, 2.1341, 12.744, 1.33234, 3.61746D-8, 0.823876, -5.29342,
9, 0.63409,
10, -4.24333, 1.69402, 10.0468, -5.2457, 1.48091, -6.08732, 1.59604,
10, 1.11976, 0.962314, 1.20731, 0.118783, 7.51205D-6, -22.7476,
0.0175199,
11, -1.80607, 1.42207, 10.9674, -2.7031, 0.679818, -3.32771, 1.33783,
11, -3.98987, 1.21949,
12, -2.23539, 1.25839, 11.6311, -6.72146, 0.00147744, -3.04547,
0.506178,
12, -4.60503, 1.66014, -4.28622, 0.974461,
13, -2.13937, 1.56045, 10.8315, -4.96625, 1.12583, -3.00669, 0.861331,
13, -3.65685, 1.46634, -4.05272, 1.18162,

VARIABLES [Runs x Variables]

A-1
1, 2.39, 150.0, 1.002, 0.198, 1.007, 17.0,
A-2
2, 2.39, 150.0, 1.002, 0.198, 1.005, 17.0,
A-3
3, 2.39, 150.0, 1.002, 0.198, 0.999, 17.0,
A-4
4, 2.39, 150.0, 1.002, 0.198, 1.006, 17.0,
B-1
5, 1.367, 130.0, 0.501, 0.099, 0.502, 16.0,
B-2
6, 4.228, 130.0, 2.007, 0.099, 2.007, 16.0,
B-3
7, 4.228, 130.0, 0.511, 0.402, 0.507, 16.0,
B-4
8, 1.367, 130.0, 2.018, 0.402, 1.989, 16.0,
C-1
9, 1.326, 170.0, 0.501, 0.099, 2.004, 18.0,
C-2
10, 1.326, 170.0, 2.007, 0.099, 0.512, 18.0,
C-3
11, 4.299, 170.0, 0.511, 0.402, 1.995, 18.0,
C-4
12, 4.299, 170.0, 2.018, 0.402, 0.507, 18.0,
E-1
13, 0.473, 130.0, 1.002, 0.198, 1.029, 19.0,
E-2
14, 23.993, 136.0, 1.002, 0.198, 1.013, 20.0,
E-3
15, 0.473, 190.0, 1.002, 0.198, 1.007, 21.0,
E-4
16, 0.506, 170.0, 1.002, 0.198, 0.992, 19.0,

F-1
17, 0.604, 190.0, 1.002, 0.198, 1.013, 22.0,
F-2
18, 0.477, 176.0, 1.002, 0.198, 1.018, 24.0,
F-3
19, 2.459, 130.0, 1.002, 0.198, 0.993, 24.0,
F-4
20, 24.008, 138.0, 1.002, 0.198, 0.999, 27.0,
I-1
21, 7.767, 190.0, 0.42, 0.099, 3.208, 18.0,
I-2
22, 24.005, 130.0, 0.128, 1.002, 0.0, 19.0,
I-3
23, 0.628, 190.0, 0.094, 0.0, 6.404, 18.0,
I-4
24, 3.396, 190.0, 0.229, 0.798, 0.101, 18.0,
J-1
25, 5.824, 130.0, 1.085, 0.662, 6.679, 21.0,
J-2
26, 48.73, 130.0, 2.798, 0.0, 9.78, 20.0,
J-3
27, 48.73, 130.0, 3.001, 0.846, 10.01, 20.0,
J-4
28, 26.8, 130.0, 3.001, 0.846, 10.012, 24.0,
K-1
29, 19.994, 180.0, 2.482, 1.002, 9.999, 20.0,
K-2
30, 20.018, 180.0, 2.475, 0.0, 0.0, 20.0,
K-3
31, 20.018, 180.0, 2.482, 1.002, 0.0, 20.0,
K-4
32, 20.018, 180.0, 2.475, 0.0, 9.996, 20.0,
M-1
33, 22.054, 190.0, 0.505, 1.002, 10.008, 19.0,
M-2
34, 22.054, 190.0, 0.505, 1.002, 0.0, 19.0,
M-3
35, 22.054, 190.0, 0.498, 0.0, 9.999, 19.0,
M-4
36, 22.054, 190.0, 0.498, 0.0, 0.0, 19.0,
N-1
37, 21.18, 130.0, 0.511, 0.0, 9.99, 17.0,
N-2
38, 3.736, 144.0, 2.999, 0.0, 0.0, 19.0,
N-3
39, 5.147, 152.0, 1.95, 0.005, 2.627, 17.0,
N-4
40, 23.985, 144.0, 2.999, 0.0, 0.0, 16.0,

EXPERIMENTAL DATA [Runs x Responses]

A-1
1, 6.192, 33.803, 6.007, 2.984, 0.165,
A-2

2, 5.528, 35.784, 6.821, 3.071, 0.129,
A-3
3, 5.611, 38.927, 7.195, 3.496, 0.166,
A-4
4, 5.929, 39.572, 7.292, 3.676, 0.18,
B-1
5, 17.625, 38.363, 7.163, 7.296, 0.249,
B-2
6, 14.242, 35.402, 8.796, 5.123, 0.282,
B-3
7, 16.222, 38.698, 7.178, 7.0, 0.174,
B-4
8, 19.946, 37.065, 9.797, 6.273, 0.556,
C-1
9, 4.27, 47.259, 7.788, 4.383, 0.132,
C-2
10, 1.84, 51.703, 5.892, 0.688, 0.064,
C-3
11, 1.806, 53.147, 7.789, 3.659, 0.074,
C-4
12, 0.177, 21.046, 1.632, 0.184, 0.034,
E-1
13, 26.214, 42.228, 9.956, 8.032, 0.248,
E-2
14, 1.669, 35.269, 5.944, 2.682, 0.055,
E-3
15, 1.132, 36.939, 5.589, 1.16, 0.089,
E-4
16, 9.856, 40.329, 7.833, 3.752, 0.219,
F-1
17, 0.494, 33.998, 4.895, 1.002, 0.087,
F-2
18, 5.015, 37.467, 6.353, 2.309, 0.154,
F-3
19, 19.694, 36.5, 9.028, 5.301, 0.273,
F-4
20, 1.181, 36.226, 5.96, 2.537, 0.087,
I-1
21, 0.174, 19.719, 2.002, 0.426, 0.184,
I-2
22, 13.441, 34.146, 7.993, 4.412, 0.313,
I-3
23, 12.421, 39.845, 4.665, 7.151, 0.544,
I-4
24, 0.404, 28.391, 2.624, 1.726, 0.129,
J-1
25, 7.863, 35.779, 8.26, 4.743, 0.0,
J-2
26, 0.522, 31.923, 4.664, 0.74, 0.0,
J-3
27, 0.424, 32.191, 2.888, 0.601, 0.0,
J-4
28, 0.304, 34.519, 3.32, 0.711, 0.0,
K-1

```
29, 0.0, 0.0, 0.0, 0.0, 0.0,
 K-2
30, 0.0, 0.0, 0.0, 0.0, 0.0,
 K-3
31, 0.0, 0.0, 0.0, 0.0, 0.0,
 K-4
32, 0.0, 0.0, 0.0, 0.0, 0.0,
 M-1
33, 0.028, 0.348, 0.025, 0.003, 0.0,
 M-2
34, 0.483, 4.26, 0.332, 0.067, 0.0,
 M-3
35, 0.893, 8.849, 0.685, 0.086, 0.0,
 M-4
36, 0.211, 6.842, 0.461, 0.011, 0.0,
 N-1
37, 9.791, 34.095, 9.258, 5.387, 3.6,
 N-2
38, 14.151, 32.998, 7.174, 3.255, 0.29,
 N-3
39, 1.187, 33.962, 5.1, 0.931, 0.091,
 N-4
40, 0.756, 22.067, 2.163, 0.472, 0.046,

LAMBDA=1.0

 RESPONSE VARIANCE

1.188D-4, 0.005831, 0.001426, 0.00203, 4.374D-5,

H=0.01

 DEBUG LEVEL

DEBUGG=0

 # SIGNIFICANT DIGITS

NSIG=4

 CALCULATE PROBABILITIES FROM SCRATCH?

FIRST=T

 DO FIXED TEMPERATURE CMAX TRIALS INSTEAD?

SECOND=F

 exit INITAL


 TRANSFORMED DATA ---------------------------------------------------
```

UPPER BOUNDS ON VARIABLES [Variables]

24.0, 463.0, 3.0, 1.0, 10.0, 0.3667,

LOWER BOUNDS ON VARIABLES [Variables]

0.5, 403.0, 0.5, 0.0, 0.0, 0.0,

DEFAULT VALUES OF UNOPTIMIZED VARIABLES [Variables]

5.0, 433.0, 1.2, 0.3, 0.1, 0.3333,

VARIABLES [Runs x Variables]

A-1
2.39, 423.0, 0.916, 0.198, 1.007, 0.2833,
A-2
2.39, 423.0, 0.9213, 0.198, 1.005, 0.2833,
A-3
2.39, 423.0, 0.931, 0.198, 0.999, 0.2833,
A-4
2.39, 423.0, 0.934, 0.198, 1.006, 0.2833,
B-1
1.367, 403.0, 0.4665, 0.099, 0.502, 0.2667,
B-2
4.228, 403.0, 1.956, 0.099, 2.007, 0.2667,
B-3
4.228, 403.0, 0.4733, 0.402, 0.507, 0.2667,
B-4
1.367, 403.0, 1.99, 0.402, 1.989, 0.2667,
C-1
1.326, 443.0, 0.4503, 0.099, 2.004, 0.3,
C-2
1.326, 443.0, 1.948, 0.099, 0.512, 0.3,
C-3
4.299, 443.0, 0.4668, 0.402, 1.995, 0.3,
C-4
4.299, 443.0, 1.87, 0.402, 0.507, 0.3,
E-1
0.473, 403.0, 1.006, 0.198, 1.029, 0.3167,
E-2
23.99, 409.0, 0.9078, 0.198, 1.013, 0.3333,
E-3
0.473, 463.0, 0.906, 0.198, 1.007, 0.35,
E-4
0.506, 443.0, 0.9467, 0.198, 0.992, 0.3167,
F-1
0.604, 463.0, 0.8954, 0.198, 1.013, 0.3667,
F-2
0.477, 449.0, 0.9212, 0.198, 1.018, 0.4,
F-3
2.459, 403.0, 0.9677, 0.198, 0.993, 0.4,
F-4
24.01, 411.0, 0.9086, 0.198, 0.999, 0.45,

I-1
7.767, 463.0, 0.27, 0.099, 3.208, 0.3,
I-2
24.0, 403.0, 0.06842, 1.002, 0.0, 0.3167,
I-3
0.628, 463.0, 0.04424, 0.0, 6.404, 0.3,
I-4
3.396, 463.0, 0.105, 0.798, 0.101, 0.3,
J-1
5.824, 403.0, 1.017, 0.662, 6.679, 0.35,
J-2
48.73, 403.0, 2.685, 0.0, 9.78, 0.3333,
J-3
48.73, 403.0, 2.884, 0.846, 10.01, 0.3333,
J-4
26.8, 403.0, 2.891, 0.846, 10.01, 0.4,
K-1
19.99, 453.0, 2.278, 1.002, 9.999, 0.3333,
K-2
20.02, 453.0, 2.271, 0.0, 0.0, 0.3333,
K-3
20.02, 453.0, 2.278, 1.002, 0.0, 0.3333,
K-4
20.02, 453.0, 2.271, 0.0, 9.996, 0.3333,
M-1
22.05, 463.0, 0.3024, 1.002, 10.01, 0.3167,
M-2
22.05, 463.0, 0.3138, 1.002, 0.0, 0.3167,
M-3
22.05, 463.0, 0.3197, 0.0, 9.999, 0.3167,
M-4
22.05, 463.0, 0.3125, 0.0, 0.0, 0.3167,
N-1
21.18, 403.0, 0.4479, 0.0, 9.99, 0.2833,
N-2
3.736, 417.0, 2.934, 0.0, 0.0, 0.3167,
N-3
5.147, 425.0, 1.845, 0.005, 2.627, 0.2833,
N-4
23.99, 417.0, 2.857, 0.0, 0.0, 0.2667,

EXPERIMENTAL DATA [Runs x Responses]

A-1
0.2216, 0.9402, 0.3837, 0.4009, 0.05012,
A-2
0.1979, 0.9953, 0.4357, 0.4125, 0.03919,
A-3
0.2008, 1.083, 0.4595, 0.4696, 0.05043,
A-4
0.2122, 1.101, 0.4657, 0.4938, 0.05468,
B-1
0.6309, 1.067, 0.4575, 0.9801, 0.07564,
B-2

0.5098, 0.9847, 0.5618, 0.6882, 0.08566,
B-3
0.5806, 1.076, 0.4585, 0.9404, 0.05286,
B-4
0.7139, 1.031, 0.6257, 0.8427, 0.1689,
C-1
0.1528, 1.315, 0.4974, 0.5888, 0.0401,
C-2
0.06586, 1.438, 0.3763, 0.09242, 0.01944,
C-3
0.06464, 1.478, 0.4975, 0.4915, 0.02248,
C-4
0.006335, 0.5854, 0.1042, 0.02472, 0.01033,
E-1
0.9383, 1.175, 0.6359, 1.079, 0.07533,
E-2
0.05974, 0.981, 0.3796, 0.3603, 0.01671,
E-3
0.04052, 1.027, 0.357, 0.1558, 0.02704,
E-4
0.3528, 1.122, 0.5003, 0.504, 0.06652,
F-1
0.01768, 0.9456, 0.3126, 0.1346, 0.02643,
F-2
0.1795, 1.042, 0.4058, 0.3102, 0.04678,
F-3
0.7049, 1.015, 0.5766, 0.7121, 0.08293,
F-4
0.04227, 1.008, 0.3807, 0.3408, 0.02643,
I-1
0.006228, 0.5485, 0.1279, 0.05723, 0.05589,
I-2
0.4811, 0.9498, 0.5105, 0.5927, 0.09508,
I-3
0.4446, 1.108, 0.2979, 0.9606, 0.1652,
I-4
0.01446, 0.7897, 0.1676, 0.2319, 0.03919,
J-1
0.2814, 0.9952, 0.5276, 0.6372, 0.0,
J-2
0.01868, 0.8879, 0.2979, 0.09941, 0.0,
J-3
0.01518, 0.8954, 0.1845, 0.08074, 0.0,
J-4
0.01088, 0.9601, 0.212, 0.09551, 0.0,
K-1
0.0, 0.0, 0.0, 0.0, 0.0,
K-2
0.0, 0.0, 0.0, 0.0, 0.0,
K-3
0.0, 0.0, 0.0, 0.0, 0.0,
K-4
0.0, 0.0, 0.0, 0.0, 0.0,
M-1

0.001002, 0.00968, 0.001597, 4.03D-4, 0.0,
M-2
0.01729, 0.1185, 0.0212, 0.009001, 0.0,
M-3
0.03196, 0.2461, 0.04375, 0.01155, 0.0,
M-4
0.007552, 0.1903, 0.02944, 0.001478, 0.0,
N-1
0.3505, 0.9483, 0.5913, 0.7237, 1.094,
N-2
0.5065, 0.9178, 0.4582, 0.4373, 0.08809,
N-3
0.04249, 0.9446, 0.3257, 0.1251, 0.02764,
N-4
0.02706, 0.6138, 0.1381, 0.06341, 0.01397,
JFEVAL=0

PT = 0: 0: 2.07 IO = 0: 0: 1.12

MODEL VARIANCE [Models]

JFEVAL=40
JFEVAL=80
JFEVAL=120
JFEVAL=160
JFEVAL=200
JFEVAL=240
JFEVAL=280
JFEVAL=320
JFEVAL=360
JFEVAL=400
JFEVAL=440
JFEVAL=480
JFEVAL=520
0.01557, 0.01172, 0.01487, 0.01347, 0.008172, 0.005897, 0.01684,
0.007252, 0.05497, 0.02247, 0.006202, 0.006997, 0.004769,

PT = 0: 5:45.00 IO = 0: 0: 1.17

MODEL LIKELYHOODS [Runs x Models]

0, 0.07692, 0.07692, 0.07692, 0.07692, 0.07692, 0.07692, 0.07692,
0.07692, 0.07692, 0.07692, 0.07692, 0.07692, 0.07692,
JFEVAL=533
1, 0.04236, 0.08019, 0.06062, 0.04326, 0.07201, 0.09136, 0.04272,
0.124, 0.0239, 0.02636, 0.137, 0.1182, 0.138,
0.4009, 0.5887, 0.5167, 0.5508, 0.5809, 0.5241, 0.501, 0.5911, 0.4688,
0.7364, 0.6544, 0.4596, 0.4758, 0.4644,
JFEVAL=546
2, 0.01885, 0.06583, 0.03784, 0.01994, 0.0556, 0.09035, 0.0192, 0.1542,
0.005732, 0.007278, 0.1878, 0.1416, 0.1958,
0.4125, 0.5879, 0.5154, 0.5495, 0.5797, 0.5227, 0.4992, 0.5894, 0.468,

0.7335, 0.6535, 0.4585, 0.4751, 0.4634,
JFEVAL=559
3, 0.009625, 0.0521, 0.0242, 0.01093, 0.04826, 0.09655, 0.009691,
0.1589, 0.001327, 0.002345, 0.2035, 0.1479, 0.2347,
0.4696, 0.5857, 0.5129, 0.5472, 0.5774, 0.5201, 0.4958, 0.5862, 0.4665,
0.7282, 0.6519, 0.4566, 0.4739, 0.4614,
JFEVAL=572
4, 0.005642, 0.04325, 0.01689, 0.006928, 0.0456, 0.1068, 0.005573,
0.156, 3.404D-4, 8.895D-4, 0.2029, 0.15, 0.2592,
0.4938, 0.5851, 0.5122, 0.5466, 0.5769, 0.5194, 0.4948, 0.5852, 0.4662,
0.7266, 0.6514, 0.4563, 0.4736, 0.4611,
JFEVAL=585
5, 0.001882, 0.03056, 0.006364, 0.002553, 0.04208, 0.1146, 0.003498,
0.1531, 8.461D-5, 3.342D-4, 0.2076, 0.1384, 0.299,
0.9801, 0.8151, 0.9168, 0.8294, 0.829, 0.9561, 0.9722, 1.043, 0.9963,
1.214, 0.8254, 0.957, 1.02, 0.9905,
JFEVAL=598
6, 0.001498, 0.0256, 0.005135, 0.002156, 0.04276, 0.08442, 0.001696,
0.1604, 1.572D-5, 2.173D-4, 0.2034, 0.1523, 0.3204,
0.6882, 0.6868, 0.6436, 0.7034, 0.6999, 0.664, 0.6013, 0.8201, 0.6593,
1.003, 0.7307, 0.635, 0.6747, 0.639,
JFEVAL=611
7, 0.001332, 0.03649, 0.004373, 0.001655, 0.04465, 0.2087, 0.00212,
0.09603, 8.673D-6, 1.836D-4, 0.127, 0.1484, 0.3291,
0.9404, 0.7897, 0.85, 0.7856, 0.7775, 0.8161, 0.9219, 0.8395, 0.7814,
1.187, 0.7791, 0.7897, 0.8138, 0.8251,
JFEVAL=624
8, 9.149D-4, 0.03135, 0.003139, 0.00121, 0.04468, 0.2192, 0.001554,
0.08341, 1.527D-6, 1.066D-4, 0.08725, 0.1509, 0.3763,
0.8427, 0.7815, 0.8301, 0.79, 0.7853, 0.8356, 0.8071, 0.8273, 0.7831,
1.159, 0.7697, 0.7532, 0.8134, 0.8114,
JFEVAL=637
9, 0.00156, 0.02958, 0.004417, 0.0018, 0.02945, 0.2148, 0.001992,
0.0201, 1.569D-6, 1.965D-4, 0.1333, 0.04613, 0.5166,
0.5888, 0.4949, 0.4258, 0.4637, 0.4706, 0.4154, 0.4507, 0.4513, 0.3722,
0.4496, 0.6074, 0.4784, 0.3849, 0.4756,
JFEVAL=650
10, 0.001179, 0.02236, 0.003297, 0.001357, 0.02737, 0.2151, 0.001295,
0.00811, 6.12D-7, 7.708D-5, 0.1386, 0.02496, 0.5563,
0.09242, 0.09031, 0.1506, 0.05916, 0.04725, 0.129, 0.1356, 0.02625,
0.225, 9.703D-4, 0.2474, 0.1242, 0.2022, 0.133,
JFEVAL=663
11, 4.357D-4, 0.01067, 1.398D-4, 1.408D-4, 0.002656, 0.1114, 9.387D-5,
0.002991, 9.444D-7, 6.331D-4, 0.2392, 0.01382, 0.6178,
0.4915, 0.1199, 0.1686, 0.03718, 0.08781, 0.1555, 0.2402, 0.03525,
0.2108, 0.0253, 0.3121, 0.2783, 0.2278, 0.2797,
JFEVAL=676
12, 2.848D-4, 0.00786, 9.316D-5, 9.777D-5, 0.002291, 0.09644, 5.929D-5,
0.002621, 3.469D-7, 3.08D-4, 0.2279, 0.0128, 0.6492,
0.02472, 3.909D-4, 0.04856, 2.253D-5, 3.486D-5, 0.04027, 0.07093,
1.733D-6, 0.05311, 3.312D-7, 0.1094, 0.006732, 0.03123, 0.009323,
JFEVAL=689
13, 3.315D-5, 0.003147, 1.287D-5, 1.197D-5, 0.001385, 0.05782,
3.511D-5, 0.002409, 1.249D-7, 4.81D-5, 0.2108, 0.01379, 0.7105,

1.079, 0.8207, 0.9322, 0.8362, 0.8358, 0.9759, 0.9771, 0.9851, 1.026,
1.222, 0.8137, 1.021, 1.087, 1.037,
JFEVAL=702
14, 2.058D-5, 0.001836, 6.859D-6, 6.99D-6, 7.471D-4, 0.05486, 1.668D-5,
0.002092, 3.991D-8, 1.907D-5, 0.1972, 0.01231, 0.7309,
0.3603, 0.3274, 0.2831, 0.2773, 0.2909, 0.2652, 0.3695, 0.2606, 0.3435,
0.4709, 0.4842, 0.3637, 0.3556, 0.3562,
JFEVAL=715
15, 1.885D-5, 0.001786, 8.693D-6, 9.247D-6, 0.001091, 0.08276,
1.957D-5, 0.002469, 2.327D-8, 6.403D-6, 0.2001, 0.008372, 0.7034,
0.1558, 0.2597, 0.2566, 0.1643, 0.1642, 0.2037, 0.213, 0.1256, 0.239,
0.01633, 0.3927, 0.2539, 0.286, 0.2556,
JFEVAL=728
16, 1.158D-5, 0.001616, 6.893D-6, 6.987D-6, 0.001202, 0.103, 1.201D-5,
0.003059, 1.055D-8, 2.789D-6, 0.1609, 0.01061, 0.7195,
0.504, 0.6222, 0.5643, 0.5779, 0.5926, 0.5457, 0.4664, 0.621, 0.4875,
0.617, 0.6719, 0.4127, 0.494, 0.4339,
JFEVAL=741
17, 1.139D-5, 0.001615, 6.281D-6, 6.582D-6, 0.001552, 0.1374, 9.364D-6,
0.002973, 4.977D-9, 1.308D-6, 0.1618, 0.006623, 0.6881,
0.1346, 0.1503, 0.1908, 0.0753, 0.07523, 0.1473, 0.1747, 0.04677,
0.2139, 0.001832, 0.3035, 0.2122, 0.2545, 0.218,
JFEVAL=754
18, 1.478D-6, 4.016D-4, 1.724D-6, 1.376D-6, 4.596D-4, 0.1082, 1.999D-6,
0.001624, 2.427D-9, 1.295D-7, 0.189, 0.002825, 0.6975,
0.3102, 0.5896, 0.5259, 0.5346, 0.5468, 0.4946, 0.4238, 0.5624, 0.4532,
0.4974, 0.6477, 0.3915, 0.467, 0.4034,
JFEVAL=767
19, 1.866D-6, 3.683D-4, 2.117D-6, 1.768D-6, 3.506D-4, 0.04264,
8.342D-7, 0.00182, 3.023D-10, 1.427D-7, 0.3281, 0.00295, 0.6237,
0.7121, 0.7862, 0.8404, 0.7951, 0.7912, 0.85, 0.8772, 0.9274, 0.8172,
1.176, 0.7908, 0.7719, 0.8229, 0.8273,
JFEVAL=780
20, 1.101D-6, 2.068D-4, 9.307D-7, 9.135D-7, 1.82D-4, 0.04347, 3.069D-7,
0.001595, 1.154D-10, 6.78D-8, 0.3152, 0.002675, 0.6367,
0.3408, 0.2671, 0.246, 0.2149, 0.2366, 0.2344, 0.331, 0.1913, 0.3026,
0.3564, 0.4411, 0.313, 0.3084, 0.3075,
JFEVAL=793
21, 7.277D-7, 1.671D-4, 6.252D-7, 6.357D-7, 1.684D-4, 0.03987, 1.97D-7,
0.00159, 4.519D-11, 3.753D-8, 0.3164, 0.002691, 0.6391,
0.05723, 1.076D-5, 0.0368, 5.17D-6, 2.758D-4, 0.03294, 0.1082,
-6.156D-6, 0.06268, 3.661D-6, 0.1289, 0.08728, 0.06804, 0.1024,
JFEVAL=806
22, 2.003D-7, 1.161D-5, 3.503D-7, 4.291D-7, 1.707D-4, 0.05232,
1.467D-7, 0.001093, 8.305D-13, 1.55D-8, 0.0377, 0.003307, 0.9054,
0.5927, 0.795, 0.8633, 0.7191, 0.6938, 0.6447, 0.5914, 0.6635, 0.6954,
1.204, 0.7629, 0.7907, 0.5962, 0.5946,
JFEVAL=819
23, 3.735D-8, 2.36D-6, 2.368D-7, 3.025D-7, 1.505D-4, 0.05219, 8.532D-8,
0.001008, 2.417D-13, 8.909D-9, 0.03287, 0.002315, 0.9115,
0.9606, 0.7474, 0.7695, 0.9322, 0.9886, 0.975, 0.9752, 1.028, 0.9758,
1.132, 0.9552, 1.007, 0.8867, 0.9938,
JFEVAL=832
24, 2.051D-8, 1.702D-6, 3.428D-8, 8.667D-8, 8.931D-5, 0.05328,

1.959D-8, 6.465D-4, 6.687D-14, 3.773D-9, 0.02305, 0.001632, 0.9213,
0.2319, 0.1436, 0.1872, 0.001152, 0.06157, 0.1396, 0.2331, 0.032,
0.1472, 0.04101, 0.3563, 0.3085, 0.1576, 0.2662,
JFEVAL=845
25, 1.42D-8, 1.378D-6, 2.504D-8, 7.527D-8, 9.967D-5, 0.06765, 1.59D-8,
7.277D-4, 4.45D-15, 2.36D-9, 0.01984, 0.001638, 0.91,
0.6372, 0.7236, 0.7073, 0.715, 0.6762, 0.6248, 0.6458, 0.6635, 0.6665,
1.111, 0.7224, 0.7157, 0.6934, 0.7044,
JFEVAL=858
26, 8.15D-9, 9.867D-7, 1.609D-8, 5.075D-8, 7.457D-5, 0.05253, 8.154D-9,
6.097D-4, 1.504D-15, 5.714D-10, 0.01837, 0.001418, 0.927,
0.09941, 0.03994, 0.1081, 0.116, 0.08985, 0.1464, 0.1553, 0.01735,
0.07142, 0.02973, 0.2972, 0.09314, 0.07901, 0.0937,
JFEVAL=871
27, 4.78D-9, 6.898D-7, 9.307D-9, 2.889D-8, 6.142D-5, 0.04885, 4.29D-9,
4.986D-4, 5.074D-16, 2.041D-10, 0.01685, 0.00122, 0.9325,
0.08074, 0.03191, 0.1014, 0.02209, 0.01008, 0.07679, 0.07126, 0.007637,
0.04877, 0.01953, 0.2204, 0.07864, 0.06258, 0.07359,
JFEVAL=884
28, 6.704D-9, 9.434D-7, 1.39D-8, 4.489D-8, 1.119D-4, 0.1076, 5.934D-9,
7.768D-4, 4.131D-16, 1.042D-10, 0.01457, 0.001401, 0.8755,
0.09551, 0.1388, 0.17, 0.1147, 0.07454, 0.1317, 0.09777, 0.06061,
0.166, 0.1285, 0.3053, 0.2183, 0.1985, 0.208,
JFEVAL=897
29, 4.209D-9, 6.697D-7, 8.903D-9, 3.002D-8, 9.224D-5, 0.1003, 3.598D-9,
6.715D-4, 1.441D-16, 5.546D-11, 0.01338, 0.001228, 0.8843,
0.0, 2.25D-7, 0.003847, -6.264D-10, -1.394D-6, 0.003167, 0.007295,
1.015D-10, -3.763D-6, -1.048D-11, -2.239D-5, 1.561D-6, 1.888D-6,
2.103D-6,
JFEVAL=910
30, 2.643D-9, 4.755D-7, 5.706D-9, 2.009D-8, 7.604D-5, 0.09261,
2.182D-9, 5.806D-4, 5.028D-17, 2.952D-11, 0.01228, 0.001076, 0.8934,
0.0, 2.351D-7, 0.003855, -3.892D-9, -9.017D-10, 0.002751, 0.0147,
-3.887D-9, 1.894D-6, -6.604D-15, -2.007D-5, -1.23D-6, -5.765D-8,
3.557D-7,
JFEVAL=923
31, 1.657D-9, 3.371D-7, 3.65D-9, 1.342D-8, 6.259D-5, 0.08622, 1.321D-9,
5.012D-4, 1.751D-17, 1.568D-11, 0.01126, 9.421D-4, 0.901,
0.0, 2.18D-7, 0.003843, -1.656D-10, -9.031D-7, 0.002677, 0.007229,
1.012D-10, -4.317D-6, -1.009D-11, -2.257D-5, 1.797D-6, -1.203D-5,
3.021D-7,
JFEVAL=936
32, 1.039D-9, 2.39D-7, 2.336D-9, 8.964D-9, 5.152D-5, 0.07939,
7.999D-10, 4.327D-4, 6.101D-18, 8.332D-12, 0.01032, 8.249D-4, 0.909,
0.0, 2.351D-7, 0.003855, -3.016D-8, -3.615D-9, 0.003221, 0.0152,
-3.887D-9, -1.983D-6, -6.823D-15, -2.007D-5, 1.558D-6, -3.776D-7,
2.183D-6,
JFEVAL=949
33, 6.526D-10, 1.691D-7, 1.498D-9, 5.809D-9, 4.215D-5, 0.07259,
4.853D-10, 3.742D-4, 2.13D-18, 4.363D-12, 0.00947, 7.231D-4, 0.9168,
4.03D-4, -6.047D-7, 0.01159, 1.594D-8, 0.03208, 0.0135, 0.01963,
-6.066D-6, 0.002871, -3.615D-8, 0.02887, 0.004655, 0.003637, 0.005685,
JFEVAL=962
34, 4.097D-10, 1.204D-7, 9.594D-10, 3.884D-9, 3.483D-5, 0.06767,

2.943D-10, 3.234D-4, 7.441D-19, 2.313D-12, 0.00867, 6.325D-4, 0.9227,
0.009001, 4.396D-7, 0.01116, -1.612D-8, 1.816D-8, 0.007258, 0.01866,
1.437D-8, 0.002007, -7.534D-9, 0.02618, -1.694D-6, 1.225D-5, -2.122D-6,
JFEVAL=975
35, 2.576D-10, 8.593D-8, 6.175D-10, 2.611D-9, 2.887D-5, 0.05711,
1.787D-10, 2.798D-4, 2.606D-19, 1.226D-12, 0.007963, 5.549D-4, 0.9341,
0.01155, 5.452D-7, 0.01097, 0.007003, 0.0145, 0.0127, 0.05295,
2.443D-8, 0.002788, 1.448D-7, 0.03216, 0.002871, 0.002765, 0.005229,
JFEVAL=988
36, 1.62D-10, 6.093D-8, 3.963D-10, 1.749D-9, 2.38D-5, 0.04709,
1.085D-10, 2.423D-4, 9.106D-20, 6.394D-13, 0.007321, 4.872D-4, 0.9448,
0.001478, 4.196D-7, 0.01121, -2.852D-6, 1.365D-7, 0.007351, 0.04648,
2.541D-8, 0.002653, -1.071D-6, 0.03419, -1.266D-6, 1.512D-5, -1.209D-6,
JFEVAL=1001
37, 1.089D-10, 3.137D-8, 2.542D-10, 1.389D-9, 2.32D-5, 0.04979,
3.31D-11, 2.464D-4, 8.726D-21, 2.395D-13, 0.006814, 4.189D-4, 0.9427,
0.7237, 0.6635, 0.6072, 0.7994, 0.7232, 0.7327, 0.7515, 0.9037, 0.7363,
1.132, 0.8836, 0.7746, 0.665, 0.7736,
JFEVAL=1014
38, 2.527D-10, 5.0D-8, 6.394D-10, 3.861D-9, 4.199D-5, 0.01678,
6.287D-11, 9.216D-4, 1.075D-20, 6.33D-13, 0.007905, 0.001275, 0.9731,
0.4373, 0.3111, 0.2755, 0.3221, 0.3353, 0.2963, 0.2273, 0.2845, 0.3632,
0.199, 0.525, 0.2788, 0.3408, 0.2835,
JFEVAL=1027
39, 2.111D-10, 4.384D-8, 5.145D-10, 3.438D-9, 4.01D-5, 0.01729,
4.654D-11, 4.173D-4, 4.613D-21, 2.014D-13, 0.008186, 8.451D-4, 0.9732,
0.1251, 0.1305, 0.1708, 0.1697, 0.1292, 0.1785, 0.1799, 0.06769,
0.2567, 0.02953, 0.3231, 0.1774, 0.2262, 0.1886,
JFEVAL=1040
40, 1.496D-10, 3.912D-8, 3.705D-10, 2.563D-9, 4.161D-5, 0.01969,
3.194D-11, 4.044D-4, 1.956D-21, 1.058D-13, 0.007685, 7.966D-4, 0.9714,
0.06341, 0.001319, 0.05557, 0.001483, 0.001885, 0.06002, 0.08658,
6.188D-5, 0.01698, -1.431D-6, 0.1726, 0.004909, 0.01005, 0.006018,

PT = 0:11:29.03 IO = 0: 0: 1.43

APPENDIX VII

CHIP MODEL

```
#FILE (MARK)CHIP/S ON STUDENTS
REMOVE CHIP/COMMON
GET CHIP/COMMON/GENERIC AS CHIP/COMMON
REPLACE .NGPTX.  .5.  15300-END
REPLACE .NGPTY.  .5.  15300-END
REPLACE .NGPTZ.  .5.  15300-END
REPLACE .NPRINT. .40. 15300-END
REPLACE .PDHI.   .16. 15300-END
REPLACE .PDLO.   .11. 15300-END
REPLACE .POHI.   .18. 15300-END
REPLACE .PYHI.   .10. 15300-END
REPLACE .PYLO.   .1.  15300-END
SA
REMOVE CHIP/SOURCE/DECLARE
GET CHIP/SOURCE/DECLARE/GENERIC AS CHIP/SOURCE/DECLARE
REPLACE .NGPTX.  .5.
REPLACE .NGPTY.  .5.
REPLACE .NGPTZ.  .5.
REPLACE .NMETH.  .2.
REPLACE .NMITER. .3.
REPLACE .NWORK.  .23463.
REPLACE .ODE.    .2133.
REPLACE .PDHI.   .16.
REPLACE .PDLO.   .11.
REPLACE .POHI.   .18.
REPLACE .PYLO.   .1.
SA
REMOVE CHIP/BLOCK/DATA
GET CHIP/BLOCK/DATA/GENERIC AS CHIP/BLOCK/DATA
REPLACE .NX. .5.
REPLACE .NY. .5.
REPLACE .NZ. .5.
REPLACE .NP. .40.
SA
START ST/C("CHIP/SOURCE")
```

```
#FILE (MARK)CHIP/DUDT ON STUDENTS
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE DWOOD
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates chip yield and diffusion rates in-----
C     bulk liquor & liquor saturated wood.
C-----Creation date: 23 Mar 82---Last update: 18 Aug 85----------------

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'

      DOUBLE PRECISION
     & A     , ALPHAX, ALPHAY, ALPHAZ, BETA  , DNAQ  , DNLD  , DNDS
     &, DNOH  , DNSH  , DTPA  , DTPB  , DTPC
     &, ECCSLA, ECCSLB, ECCSLC, ECCSLD, ECCSLE, ECCSLF, ECCSLG, ECCSLH
     &, ECCSRA, ECCSRB, ECCSRC, ECCSRD, ECCSRE, ECCSRF, ECCSRG, ECCSRH
     &, ECCSTA, ECCSTB, ECCSTC, ECCSTD, ECCSTE, ECCSTF, ECCSTG, ECCSTH
     &, EX    , EY    , EZ    , OH    , ONE   , P65   , RVOY  , SH
     &, SOLIDS, TP    , TPDVS , TPMB  , VS
     &, VSA   , VSB   , VSC   , VSD   , VSE   , VSF   , VSG   , VSH
     &, VSW   , VSJ   , VSK   , YIELD , ZERO  , TPHI  , TPLO
     &, VOTP  , VOHDMW

      INTEGER
     & COMP  , X     , Y     , Z

      LOGICAL
     & FN10

      DATA
     & ONE  /+1.D+0  /
     &, P65  /+6.5D-1 /
     &, TPHI /+645.D+0/
     &, TPLO /+273.D+0/
     &, ZERO /+0.D+0  /

C-------Diffusivities normalized to 298degK using Stokes-Einstein Eq.
C       DN   = D(298) * Vis(water at 298) / 298
C-------D(T) = DN * T / Vis(T)

     &, DNOH /+2.280D-8/ % NaOH [=] cP m**2 / hr / degK
     &, DNSH /+1.624D-8/ % NaSH
     &, DNAQ /+6.992D-9/ % Anthraquinone
     &, DNLD /+1.979D-9/ % dissolved lignin
     &, DNDS /+7.239D-9/ % dissolved solids (approximated as glucose)

C-----Constants for viscosity equations

     &, VSA /+2.260D-2/ % VS(H2O) = f(T)
     &, VSB /+285.5D+0/
     &, VSC /+9854.D+0/
```

```
     &, VSD /+1.422D+0/
     &, VSG /+10.63D+0/ % VS(liquor)=f(Vis(H2O),dissolved solids)
     &, VSH /+1.302D+0/
     &, VSJ /+27.31D+0/
     &, VSK /+4.108D+0/

C-----Constants for thermal diffusivity equation

     &, DTPA /-6.149D-9/   % [=] m**2 / hr
     &, DTPB /+5.219D-6/
     &, DTPC /-4.856D-4/

C-----Constants for ECCSA equations

     &, ECCSLA /+1.446D-1/ % ECCSA = f(yield), 100 >= yield >= 65
     &, ECCSLB /+4.565D-1/
     &, ECCSTA /-7.850D-1/
     &, ECCSTB /+9.550D-1/
     &, ECCSRA /-6.056D-1/
     &, ECCSRB /+7.620D-1/
     &, ECCSLC /+1.286D+0/ % ECCSA = f(yield), 65 > yield >= 0
     &, ECCSLD /-1.528D+0/
     &, ECCSTC /+1.065D-1/
     &, ECCSTD /-9.235D-1/
     &, ECCSRC /+5.633D-1/
     &, ECCSRD /-1.338D+0/
     &, ECCSLE /+6.08D-1 / % ECCSA = f([NaOH])
     &, ECCSLF /-1.39D-1 /
     &, ECCSLG /+1.75D-1 /
     &, ECCSTE /+5.D-2   /
     &, ECCSTF /+1.299D-1/
     &, ECCSTG /+5.5D-1  /
     &, ECCSRE /+5.D-2   /
     &, ECCSRF /+1.299D-1/
     &, ECCSRG /+5.5D-1  /
     &, ECCSLH /+2.030D+0/ % ECCSA = f(yield, [NaOH])
     &, ECCSTH /+1.108D+0/
     &, ECCSRH /+1.108D+0/

C-----------------------------------------------------------------------

DEBUG MONITOR(6) PTP, VO(PTP), VU(PTP), VOTP, TP, TPMB, VSW, RRHOL
     &          , SOLIDS
C     WRITE(P, *//) 'ENTER DWOOD', VU
C     WRITE(P, /) ' '

C-----DWOOD calculates the diffusivities of Temp, NaOH, NaSH, AQ,
C     dissolved solids, and dissolved lignin in liquor-saturated wood
C     as a function of Temp, [NaOH], and yield.
C     Note that the components are divided up into reactants which
C     contribute to yield (PYLO to PYHI), diffusing species (PDLO to
C     PDHI), reactants which don't contribute to yield (PNLO to PNHI),
C     and others (POLO to POHI).  Examples:
C     PY:  native lignin
```

```
C     PD:  sodium hydroxide
C     PN:  pulp viscosity
C     PO:  yield
C-----Note: PYLO <= PYHI < PDLO <= PDHI < PNLO <= PNHI < POLO <= POHI

      IF(DEBUGG .GE. 1) WRITE(6, /) '       enter DWOOD'

C-----Calculate total yield for each grid point

      RVOY = ONE / VO(PY)
      DO 100 Z = 1, NGPTZ
         DO 100 Y = 1, NGPTY
            DO 100 X = 1, NGPTX
               U(PY, X, Y, Z) = ZERO

               DO 110 COMP = PYLO, PYHI
                  U(PY, X, Y, Z) = U(COMP, X, Y, Z) * VO(COMP)
     &                           + U(PY, X, Y, Z)
  110          CONTINUE

               U(PY, X, Y, Z) = U(PY, X, Y, Z) * RVOY
  100 CONTINUE

C-----Bulk liquor diffusivities (VA) 5 Aug 85

      VOTP   = VO(PTP)
      TP     = VU(PTP) * VOTP
      TP     = DMAX1(TP, TPLO)
      TP     = DMIN1(TP, TPHI)
      TPMB   = TP - VSB
      VSW = ONE / (VSA * (TPMB + DSQRT(TPMB * TPMB + VSC)) - VSD)

      VOHDMW = VO(POH) * 25.D-3                           % convert to mol/l
      SOLIDS = VU(PDS)
      SOLIDS = DMIN1(ONE, SOLIDS)
      SOLIDS = DMAX1(SOLIDS, ZERO)

      VSE = (VSG * SOLIDS + VSH) * SOLIDS * SOLIDS + ONE
      VSF = (VSJ * SOLIDS * SOLIDS + VSK) * SOLIDS

      IF(VSW .LE. ZERO) WRITE(6, /) VSW, NFEVAL, TIME, VOTP, TP
     &, SOLIDS, X, Y, Z, 'VU(PTP)', VU(PTP)
      VS  = DEXP(DLOG(VSW) * VSE + VSF)

      IF(VS .LE. ZERO) WRITE(6, /) VS, VSW, NFEVAL, TIME, VOTP, TP
     &, SOLIDS, X, Y, Z, 'VU(PTP)', VU(PTP)
      TPDVS = TP / VS
      VA(POH) = DNOH * TPDVS
      VA(PSH) = DNSH * TPDVS
      VA(PAQ) = DNAQ * TPDVS
      VA(PLD) = DNLD * TPDVS
      VA(PDS) = DNDS * TPDVS

      VA(PTP) = (DTPA * TP + DTPB) * TP + DTPC
```

```
C-----Diffusion rates through liquor saturated wood (A) 5 Aug 85

      DO 200 Z = 1, NGPTZ
         DO 200 Y = 1, NGPTY
            DO 200 X = 1, NGPTX
               TP     = U(PTP, X, Y, Z) * VOTP
               TP     = DMAX1(TP, TPLO)
               TP     = DMIN1(TP, TPHI)
               YIELD = U(PY , X, Y, Z)
               OH     = DMAX1(U(POH, X, Y, Z), ZERO) * VOHDMW

               IF(YIELD .LT. P65) GO TO 210
                  ALPHAX = ECCSLA * YIELD + ECCSLB
                  ALPHAY = ECCSTA * YIELD + ECCSTB
                  ALPHAZ = ECCSRA * YIELD + ECCSRB
                  GO TO 220
  210          CONTINUE
                  ALPHAX = (ECCSLC * YIELD - ECCSLD) * YIELD + ONE
                  ALPHAY = (ECCSTC * YIELD - ECCSTD) * YIELD + ONE
                  ALPHAZ = (ECCSRC * YIELD - ECCSRD) * YIELD + ONE
  220          CONTINUE

               BETA = OH ** ECCSLG * ECCSLF + ECCSLE
               EX   = ONE - (ONE - ALPHAX) * (ONE - BETA) * ECCSLH

               BETA = OH ** ECCSTG * ECCSTF + ECCSTE
               EY   = ONE - (ONE - ALPHAY) * (ONE - BETA) * ECCSTH

               BETA = OH ** ECCSRG * ECCSRF + ECCSRE
               EZ   = ONE - (ONE - ALPHAZ) * (ONE - BETA) * ECCSRH

               TPMB  = TP - VSB
               VSW = ONE/((DSQRT(TPMB * TPMB + VSC) + TPMB) * VSA - VSD)

               SOLIDS = U(PDS, X, Y, Z)
               SOLIDS = DMIN1(ONE, SOLIDS)
               SOLIDS = DMAX1(SOLIDS, ZERO)

               VSE = (VSG * SOLIDS + VSH) * SOLIDS * SOLIDS + ONE
               VSF = (VSJ * SOLIDS * SOLIDS + VSK) * SOLIDS

               IF(VSW .LE. ZERO) WRITE(6, /)VSW, NFEVAL, TIME, VOTP
     &          , TP, SOLIDS, X, Y, Z, 'U(PTP)',U(PTP,X,Y,Z)
               VS  = DEXP(DLOG(VSW) * VSE + VSF)

               IF(VS .LE. ZERO) WRITE(6, /) VS, VSW, NFEVAL, TIME
     &          , VOTP, TP, SOLIDS, X, Y, Z, 'U(PTP)',U(PTP,X,Y,Z)
               TPDVS = TP / VS
               A     = DNOH * TPDVS
               AX(POH, X, Y, Z) = A * EX
               AY(POH, X, Y, Z) = A * EY
               AZ(POH, X, Y, Z) = A * EZ
```

```
            A = DNSH * TPDVS
            AX(PSH, X, Y, Z) = A * EX
            AY(PSH, X, Y, Z) = A * EY
            AZ(PSH, X, Y, Z) = A * EZ

            A = DNAQ * TPDVS
            AX(PAQ, X, Y, Z) = A * EX
            AY(PAQ, X, Y, Z) = A * EY
            AZ(PAQ, X, Y, Z) = A * EZ

            A = DNLD * TPDVS
            AX(PLD, X, Y, Z) = A * EX
            AY(PLD, X, Y, Z) = A * EY
            AZ(PLD, X, Y, Z) = A * EZ

            A = DNDS * TPDVS
            AX(PDS, X, Y, Z) = A * EX
            AY(PDS, X, Y, Z) = A * EY
            AZ(PDS, X, Y, Z) = A * EZ

            A = (DTPA * TP + DTPB) * TP + DTPC
            AX(PTP, X, Y, Z) = A
            AY(PTP, X, Y, Z) = A
            AZ(PTP, X, Y, Z) = A
  200   CONTINUE

      IF(DEBUGG .GE. 2) WRITE(6, /) ' VU', VU
      IF(DEBUGG .GE. 2) WRITE(6, /) '       exit  DWOOD'

      RETURN
      END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE REACT
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates the reaction rates
C-----Creation date: 23 Mar 82---Last update: 18 Oct 85

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'

      DOUBLE PRECISION
     &  MLN    , MLR    , MLD    , MCN    , MCX    , MGN    , MGX    , MXN
     &, MXX    , MEX    , MAG    , MOH    , MSH    , MAQ    , MVP    , MTP
     &, RMWOH  , RMWSH  , RMWAQ  , SQRTSH, SQRTAQ, SQRTTP, NTP    , ALPHA
     &, BETA   , GAMMA  , DELTA  , KBULK  , KCONDE, KRESID, DLN    , DLR
     &, HUNDRE, NR433  , ONE    , TEN    , ZERO   , PKL    , PKWA   , PKWB
     &, PKWC   , KLA    , KLB    , KLC    , KLD    , KLE    , KLF    , KLG
     &, KLH    , KLJ    , KLK    , KCA    , KCB    , KCC    , KCD    , KCE
     &, KCF    , KCG    , KCH    , KCJ    , KCK    , KGA    , KGB    , KGC
     &, KGD    , KGE    , KGF    , KGG    , KGH    , KXA    , KXB    , KXC
     &, KGJ    , KGK    , KXJ    , KXK    , AQ     , SQRTOH
     &, KXD    , KXE    , KXF    , KXG    , KXH    , KINA   , KINB   , KINC
```

```
&, KIND  , KVPA  , KVPB  , KVPC  , KOHA  , KOHB  , KOHC  , KOHD
&, KSHA  , KAQA  , KAQB  , KTPA  , RVTRAP, USLA
&, USLB  , USC   , USG   , USX   , VOLN  , VOLR  , VOCN  , VOCX
&, VOGN  , VOGX  , VOXN  , VOXX  , VOEX  , VOAG  , VOOH  , VOSH
&, VOAQ  , KNDSA , VOTP  , RTP   , VOY
&, DLD   , LL    , KPEEL , KSTOP , KCLEAV, DCN   , DCX   , DGN
&, DGX   , DXN   , DXX   , DEX   , DAG   , DWLIGN, DWCARB, DWEX
&, DWAG  , DOH   , DSH   , DAQ   , DDS   , DTP   , DVP   , PKW
&, KNOHA , KNOHB , KNOHC , KNOHD , KNSHA , KNTPA , KNVPA , KNVPB
&, TPHI  , TPLO  , SQRTLD, KNLG  , HALF  , LDORD

 INTEGER
& X      , Y     , Z

 DATA
& HALF   /+5.D-1/
&, HUNDRE /+1.D+2/
&, NR433  /-2.30946882217090069284lD-3/  % -1/433
&, ONE    /+1.D+0/
&, TEN    /+1.D+1/
&, TPHI   /+645.D+0/
&, TPLO   /+273.D+0/
&, ZERO   /+0.D+0/

&, KLA /-4.15518D+0/ % lignin kbulk               LIG 62LN 18 OCT 85
&, KLB /-19608.7D+0/
&, KLC /-3.24838D+0/
&, KLD /-10821.8D+0/
&, KLE /-3.83789D+0/
&, KLF /-15897.1D+0/

&, KLG /-5.97332D+0/ % kcondense
&, KLH /-2500.00D+0/

&, KLJ /-6.00188D+0/ % kresidual
&, KLK /-9991.03D+0/

&, KCA /-5.04455D+0/ % cellulose kpeel             CEL 30    5 OCT 85
&, KCB /-6696.65D+0/
&, KCC /-6.68305D+0/
&, KCD /-8013.12D+0/

&, KCE /-2.32248D+0/ % kstop
&, KCF /-9611.41D+0/
&, KCG /-4.62038D+0/
&, KCH /-16718.2D+0/

&, KCJ /-3.31930D+0/  % kcleave
&, KCK /-22844.2D+0/

&, KGA /-.857458D+0/ % glucomannan kpeel           GGM 30   13 OCT 85
&, KGB /-6681.58D+0/
&, KGC /-4.69499D+0/
&, KGD /-12067.8D+0/
```

```
     &, KGE /-1.51020D+0/ % kstop
     &, KGF /-3125.76D+0/
     &, KGG /-2.35942D+0/
     &, KGH /-10770.8D+0/

     &, KGJ /-4.76589D+0/ % kcleave
     &, KGK /-12636.8D+0/

     &, KXA /-2.30326D+0/ % xylan kpeel                 AX  32   13 OCT 85
     &, KXB /-15595.1D+0/
     &, KXC /-6.31659D+0/
     &, KXD /-2500.00D+0/

     &, KXE /-2.73932D+0/ % kstop
     &, KXF /-8860.34D+0/
     &, KXG /-6.33924D+0/
     &, KXH /-22304.0D+0/

     &, KXJ /-3.74624D+0/ % kcleave
     &, KXK /-12112.0D+0/

     &, KINC /+9.251D+0/  % carbohydrates initial phase
     &, KIND /-4.738D+3/

     &, KVPA /-1.268D+0/   % pulp viscosity
     &, KVPB /-6.39855D+0/
     &, KVPC /-19106.9D+0/

     &, KOHA /+1.5D-1  /   % NaOH  (g NAOH/g dissolved)
     &, KOHB /+4.D-1   /
     &, KOHC /+1.D-1   /
     &, KOHD /+6.77484D-1/

     &, KSHA /+5.6D-2  /   % NaSH

     &, KAQA /+20.4828D+0/ % AQ
     &, KAQB /-10692.9D+0/

     &, KTPA /-3.33625D-1/ % temperature
C-----------------------------------------------------------------------

      IF(DEBUGG .GE. 1) WRITE(6, /) '        enter REACT'

      RMWOH  = ONE / 40D0                                         % g/mol
      RMWSH  = ONE / 56D0                                         % g/mol
      RMWAQ  = ONE / 208D-3                                       % g/mmol
      USLA   = 0.87D+0
      USLB   = 0.76D+0
      USC    = VUSLOW(PCN) + VUSLOW(PCX)
      USG    = VUSLOW(PGN) + VUSLOW(PGX)
      USX    = VUSLOW(PXN) + VUSLOW(PXX)
      VOLN   = VO(PLN)
```

```
      VOLR    = VO(PLR)
      VOCN    = VO(PCN)
      VOCX    = VO(PCX)
      VOGN    = VO(PGN)
      VOGX    = VO(PGX)
      VOXN    = VO(PXN)
      VOXX    = VO(PXX)
      VOEX    = VO(PEX)
      VOAG    = VO(PAG)
      VOOH    = VO(POH)
      VOSH    = VO(PSH)
      VOAQ    = VO(PAQ)
      VOY     = VO(PY)
      VOTP    = VO(PTP)
      KNLG    = KLG %- DLOG(VOLN * HUNDRE / VOY) * HALF
      RVTRAP = ONE / (VOY / RHOC - VOY / RHOW)  % 1 / trapped liquor vol
      KNOHA   = KOHA * RVTRAP / VOOH
      KNOHB   = KOHB * RVTRAP / VOOH
      KNOHC   = KOHC * RVTRAP / VOOH
      KNOHD   = KOHD * RVTRAP / VOOH
      KNSHA   = ZERO
      IF(VOSH .GT. ZERO) KNSHA = KSHA * RVTRAP / VOSH
      KNDSA   = RVTRAP / VO(PDS)
      KNTPA   = KTPA * VOOH / VOTP
      KNVPA   = KVPA / VO(PVP)
      KNVPB   = KVPB + DLOG(42.1D0)

      DO 100 Z = 1, NGPTZ
         DO 100 Y = 1, NGPTY
            DO 100 X = 1, NGPTX

C--------------Set up mnemonic phrases for reactants.

               MOH = U(POH, X, Y, Z)   % NaOH
               IF(MOH .GT. ZERO) GO TO 101
                  DLN = ZERO; DLR = ZERO; DLD = ZERO;
                  DCN = ZERO; DCX = ZERO; DGN = ZERO; DGX = ZERO
                  DXN = ZERO; DXX = ZERO; DEX = ZERO; DAG = ZERO
                  DOH = ZERO; DSH = ZERO; DAQ = ZERO; DDS = ZERO
                  DVP = ZERO; DTP = ZERO;
                  GO TO 199
  101          CONTINUE
               MLN = U(PLN, X, Y, Z)   % native lignin
               MLR = U(PLR, X, Y, Z)   % residual lignin
               MLD = U(PLD, X, Y, Z)   % dissolved lignin
               MCN = U(PCN, X, Y, Z)   % native cellulose
               MCX = U(PCX, X, Y, Z)   % oxidized cellulose
               MGN = U(PGN, X, Y, Z)   % native (galacto)glucomannan
               MGX = U(PGX, X, Y, Z)   % oxidized glucomannan
               MXN = U(PXN, X, Y, Z)   % native (arabino)xylan
               MXX = U(PXX, X, Y, Z)   % oxidized xylan
               MEX = U(PEX, X, Y, Z)   % extractives
               MAG = U(PAG, X, Y, Z)   % acetyl groups
               MSH = U(PSH, X, Y, Z)   % NaSH
```

```
                    AQ  = U(PAQ, X, Y, Z)    % anthraquinone
                    MVP = U(PVP, X, Y, Z)    % pulp viscosity
                    MTP = U(PTP, X, Y, Z)    % temperature [=] deg K

                    MLD    = DMAX1(MLD, ZERO)
C                   MOH    = DMAX1(MOH, ZERO)
C                   MOH    = DMIN1(MOH, ONE)
                    MSH    = DMAX1(MSH, ZERO)
C                   MSH    = DMIN1(MSH, ONE)
                    AQ     = DMAX1(AQ,  ZERO)
C                   MAQ    = DMIN1(MAQ, ONE)
                    MOH    = MOH * VOOH * RMWOH
                    MSH    = MSH * VOSH * RMWSH
                    MAQ    = AQ  * VOAQ * RMWAQ
                    MTP    = MTP * VOTP
                    MTP    = DMAX1(MTP, TPLO)
                    MTP    = DMIN1(MTP, TPHI)
                    LDORD  = MLD
                    SQRTOH = DSQRT(MOH)
                    SQRTSH = DSQRT(MSH)
                    SQRTAQ = DSQRT(MAQ)
                    SQRTTP = DSQRT(MTP)
                    RTP    = ONE / MTP
                    NTP    = RTP + NR433
                    ALPHA  = MOH * SQRTTP
                    BETA   = (ONE - (ONE / (HUNDRE * MOH + ONE)))
                    GAMMA  = DEXP(KIND * RTP + KINC) * SQRTTP * BETA

C-------------dlignin/dt

                    KBULK  =(DEXP(KLB * NTP + KLA) * SQRTOH
     &                     + DEXP(KLD * NTP + KLC) * SQRTSH
     &                     + DEXP(KLF * NTP + KLE) * SQRTAQ)* SQRTTP * SQRTOH

                    KCONDE = DEXP(KLH * NTP + KNLG) * SQRTTP

                    KRESID = DEXP(KLK * NTP + KLJ) * ALPHA

                    DLN = -KBULK * MLN
                    DLR = KCONDE * LDORD - KRESID * MLR
                    DLD = (KBULK * MLN - KCONDE * LDORD) * RVTRAP

                    KINA = ZERO
                    KINB = ZERO
                    IF(MLN .LE. USLB) GO TO 110
                       KINB = DEXP(-8800D0 * RTP + 22.12D0) * MLN * BETA
                       IF(MLN .LE. USLA) GO TO 110
                          KINA = DEXP(-6000D0 * RTP + 17.33D0) * (MLN - USLA)
     &                          * BETA
  110               CONTINUE
                    DLN = DLN - KINA - KINB

C-------------dcelluose/dt
```

```
            KPEEL  =(DEXP(KCB * NTP + KCA) * MOH
     &             + DEXP(KCD * NTP + KCC) * SQRTSH) * SQRTTP

            KSTOP  =(DEXP(KCF * NTP + KCE)
     &             + DEXP(KCH * NTP + KCG) * SQRTAQ) * ALPHA

            KCLEAV = DEXP(KCK * NTP + KCJ) * ALPHA

            DCN = KCLEAV * MCX - (KPEEL + KSTOP) * MCN
            DCX = KSTOP * MCN - KCLEAV * MCX

            IF(MCN + MCX .GT. USC)
     &         DCN = DCN - MCN * GAMMA

C--------------dgalactoglucomannan/dt

            KPEEL  =(DEXP(KGB * NTP + KGA) * MOH
     &             + DEXP(KGD * NTP + KGC) * SQRTSH) * SQRTTP

            KSTOP  =(DEXP(KGF * NTP + KGE)
     &             + DEXP(KGH * NTP + KGG) * SQRTAQ) * ALPHA

            KCLEAV = DEXP(KGK * NTP + KGJ) * ALPHA

            DGN = KCLEAV * MGX - (KPEEL + KSTOP) * MGN
            DGX = KSTOP * MGN - KCLEAV * MGX

            IF(MGN + MGX .GT. USG)
     &         DGN = DGN - MGN * GAMMA

C--------------darabinoxylan/dt

            KPEEL  =(DEXP(KXB * NTP + KXA) * MOH
     &             + DEXP(KXD * NTP + KXC) * SQRTSH) * SQRTTP

            KSTOP  =(DEXP(KXF * NTP + KXE)
     &             + DEXP(KXH * NTP + KXG) * SQRTAQ) * ALPHA

            KCLEAV = DEXP(KXK * NTP + KXJ) * ALPHA

            DXN = KCLEAV * MXX - (KPEEL + KSTOP) * MXN
            DXX = KSTOP * MXN - KCLEAV * MXX

            IF(MXN + MXX .GT. USX)
     &         DXN = DXN - MXN * GAMMA

C--------------dextractives/dt & dacetyl groups/dt

            DEX = -MEX * GAMMA
            DAG = -MAG * GAMMA

C--------------dNAOH/dt

            DWLIGN = VOLN * DLN + VOLR * DLR
```

```
            DWCARB = VOCN * DCN + VOCX * DCX
     &               + VOGN * DGN + VOGX * DGX
     &               + VOXN * DXN + VOXX * DXX
            DWEX   = VOEX * DEX
            DWAG   = VOAG * DAG

            DOH = KNOHA * DWLIGN + KNOHB * DWCARB + KNOHC * DWEX
     &            + KNOHD * DWAG

C--------------dNASH/dt

            DSH = ZERO
            IF(MSH .GT. ZERO) DSH = KNSHA * VOLN * DLN

C--------------dAQ/dt

            DAQ = -DEXP(KAQB * RTP + KAQA) * SQRTTP * AQ

C--------------ddissolved solids/dt

            DDS = -(DWLIGN + DWCARB + DWEX + DWAG) * KNDSA

C--------------dTemperature/dt  (degrees K/hr)

            DTP = KNTPA * DOH

C--------------dpulp viscosity/dt

            DVP = +DEXP(KVPC * NTP + KNVPB) * (MVP + KNVPA)**2
     &            * SQRTTP * MOH
            DVP = -DSIGN(DVP, MVP + KNVPA)

C           IF(X.EQ.1.AND.Y.EQ.1.AND.Z.EQ.1)WRITE(P, *//)DVP,MVP
C    &       , KNVPA, KNVPB,KVPC,MTP,NTP,SQRTTP,MOH,X,Y,Z,NFEVAL,TIME
C           IF(X.EQ.1.AND.Y.EQ.1.AND.Z.EQ.1)WRITE(P, /) ' '

C--------------Assign reaction rates to rate array [R]

  199       CONTINUE
            R(PLN, X, Y, Z) = DLN
            R(PLR, X, Y, Z) = DLR
            R(PLD, X, Y, Z) = DLD
            R(PCN, X, Y, Z) = DCN
            R(PCX, X, Y, Z) = DCX
            R(PGN, X, Y, Z) = DGN
            R(PGX, X, Y, Z) = DGX
            R(PXN, X, Y, Z) = DXN
            R(PXX, X, Y, Z) = DXX
            R(PEX, X, Y, Z) = DEX
            R(PAG, X, Y, Z) = DAG
            R(POH, X, Y, Z) = DOH
            R(PSH, X, Y, Z) = DSH
            R(PAQ, X, Y, Z) = DAQ
            R(PDS, X, Y, Z) = DDS
```

```
            R(PVP, X, Y, Z) = DVP
            R(PTP, X, Y, Z) = DTP
  100 CONTINUE

      IF(DEBUGG .GE. 2) WRITE(6, /) ' VU', VU
      IF(DEBUGG .GE. 2) WRITE(6, /) '        exit  REACT'

      RETURN
      END
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE DUDT
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates the total rate of change G = dU/dt
C-----Creation date: 23 Mar 82---Last Update: 17 Aug 85

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'

      DOUBLE PRECISION
     & BULK   , GBULK , K      , QX     , QY     , QZ     , TWO    , ZERO
     &, DADX  , DADY  , DADZ
     &, DUDX  , DUDY  , DUDZ
     &, D2UDX2, D2UDY2, D2UDZ2

      INTEGER
     & COMP   , J     , X      , Y      , Z

      DATA
     & TWO    /+2.D+0/
     &, ZERO  /+0.D+0/

C-----------------------------------------------------------------------

DEBUG MONITOR(6) PTP, VO(PTP), VU(PTP)
C     WRITE(P, *//) 'ENTER DUDT', VU

C-----Note that the components are divided up into reactants which
C     contribute to yield (PYLO to PYHI), diffusing species (PDLO to
C     PDHI), reactants which don't contribute to yield (PNLO to PNHI),
C     and others (POLO to POHI).  Examples:
C     PY:  native lignin
C     PD:  sodium hydroxide
C     PN:  pulp viscosity
C     PO:  yield
C-----PYLO <= PYHI < PDLO <= PDHI < PNLO <= PNHI < POLO <= POHI

      IF(DEBUGG .GE. 1) WRITE(6, /) '       enter DUDT'

      NFEVAL = NFEVAL + 1

      IF(NFEVAL .GT. 1) DEBUGG = 0
```

```
C-----Yield [U] and diffusivities [AX, AY, AZ, & VA]

      CALL DWOOD

C-----Rate of change due to reaction [R]

      CALL REACT

C-----Rate of change of bulk liquor concentration
C-----due to mass transfer between liquor to chip [VG]

      CALL DIGEST

C-----Total rate of change  dU/dt = r(U)  [G = R]
C-----for non-diffusing species

      DO 100 Z = 1, NGPTZ
         DO 100 Y = 1, NGPTY
            DO 100 X = 1, NGPTX
               DO 110 COMP = PYLO, PYHI
                  G(COMP, X, Y, Z) = R(COMP, X, Y, Z)
  110          CONTINUE

               DO 120 COMP = PNLO, PNHI
                  G(COMP, X, Y, Z) = R(COMP, X, Y, Z)
  120          CONTINUE
  100 CONTINUE

C-----dU/dt  = Ax(d2U/dx2) + (dAx/dx)(dU/dx)
C            + Ay(d2U/dy2) + (dAy/dy)(dU/dy)
C            + Az(d2U/dz2) + (dAz/dz)(dU/dz)
C            + R(U)
C-----for diffusing species

      DO 200 Z = 1, NGPTZ
         DO 200 Y = 1, NGPTY
            DO 200 X = 1, NGPTX
               DO 200 COMP = PDLO, PDS
                  BULK = VU(COMP)
                  K    = VA(COMP) * KNORM
                  QX   = ((U(COMP, 1, Y, Z) - BULK)
     &                    / AX(COMP, 1, Y, Z)) * K
                  QY   = ((U(COMP, X, 1, Z) - BULK)
     &                    / AY(COMP, X, 1, Z)) * K
                  QZ   = ((U(COMP, X, Y, 1) - BULK)
     &                    / AZ(COMP, X, Y, 1)) * K

                  DADX = ZERO
                  DO 210 J = LOXA(X), HIXA(X)
                     DADX = AX(COMP, J, Y, Z) * CXA(J, X) + DADX
  210             CONTINUE

                  DUDX = CX1(0, X) * QX
                  DO 220 J = LOX1(X), HIX1(X)
```

```
                DUDX = U(COMP, J, Y, Z) * CX1(J, X) + DUDX
220          CONTINUE

             D2UDX2 = CX2(0, X) * QX
             DO 230 J = LOX2(X), HIX2(X)
                D2UDX2 = U(COMP, J, Y, Z) * CX2(J, X) + D2UDX2
230          CONTINUE

             DADY = ZERO
             DO 240 J = LOYA(Y), HIYA(Y)
                DADY = AY(COMP, X, J, Z) * CYA(J, Y) + DADY
240          CONTINUE

             DUDY = CY1(0, Y) * QY
             DO 250 J = LOY1(Y), HIY1(Y)
                DUDY = U(COMP, X, J, Z) * CY1(J, Y) + DUDY
250          CONTINUE

             D2UDY2 = CY2(0, Y) * QY
             DO 260 J = LOY2(Y), HIY2(Y)
                D2UDY2 = U(COMP, X, J, Z) * CY2(J, Y) + D2UDY2
260          CONTINUE

             DADZ = ZERO
             DO 270 J = LOZA(Z), HIZA(Z)
                DADZ = AZ(COMP, X, Y, J) * CZA(J, Z) + DADZ
270          CONTINUE

             DUDZ = CZ1(0, Z) * QZ
             DO 280 J = LOZ1(Z), HIZ1(Z)
                DUDZ = U(COMP, X, Y, J) * CZ1(J, Z) + DUDZ
280          CONTINUE

             D2UDZ2 = CZ2(0, Z) * QZ
             DO 290 J = LOZ2(Z), HIZ2(Z)
                D2UDZ2 = U(COMP, X, Y, J) * CZ2(J, Z) + D2UDZ2
290          CONTINUE

             G(COMP, X, Y, Z) =
     &         AX(COMP, X, Y, Z) * D2UDX2 + DADX * DUDX
     &       + AY(COMP, X, Y, Z) * D2UDY2 + DADY * DUDY
     &       + AZ(COMP, X, Y, Z) * D2UDZ2 + DADZ * DUDZ
     &       + R (COMP, X, Y, Z)

C            IF(COMP.EQ.POH .AND. X.EQ.1 .AND. Y.EQ.1 .AND. Z.EQ.1
C    &          .AND. NFEVAL .EQ. 261)  WRITE(P, /)
C    &          'NFEVAL', NFEVAL
C    &        , 'G'     , ROUND(G (COMP, X, Y, Z), NSIG)
C    &        , 'AX'    , ROUND(AX(COMP, X, Y, Z), NSIG)
C    &        , 'D2UDX2', ROUND(D2UDX2, NSIG)
C    &        , 'DADX'  , ROUND(DADX, NSIG)
C    &        , 'DUDX'  , ROUND(DUDX, NSIG)
C    &        , 'AY'    , ROUND(AY(COMP, X, Y, Z), NSIG)
C    &        , 'D2UDY2', ROUND(D2UDY2, NSIG)
```

```
C     &               , 'DADY'  , ROUND(DADY, NSIG)
C     &               , 'DUDY'  , ROUND(DUDY, NSIG)
C     &               , 'AZ'    , ROUND(AZ(COMP, X, Y, Z), NSIG)
C     &               , 'D2UDZ2', ROUND(D2UDZ2, NSIG)
C     &               , 'DADZ'  , ROUND(DADZ, NSIG)
C     &               , 'DUDZ'  , ROUND(DUDZ, NSIG)
C     &               , 'R'     , ROUND(R (COMP, X, Y, Z), NSIG)
C     &               , 'LOX1'  , LOX1(X)
C     &               , 'HIX1'  , HIX1(X)
C     &               , 'CX1(1)',(ROUND(CX1(J,X),NSIG),J=LOX1(X),HIX1(X))
C     &               , 'CX1(0)', ROUND(CX1(0,X),NSIG)
C     &               , 'BULK'  , ROUND(BULK, NSIG)
C     &               , 'U'     , ROUND( U(COMP, X, Y, Z), NSIG)
  200 CONTINUE
      DO 300 Z = 2, NGPTZ
         DO 300 Y = 2, NGPTY
            DO 300 X = 2, NGPTX
               COMP = PTP
               BULK = VU(COMP)
               K    = VA(COMP) * KNORM
               QX   = ((U(COMP, 1, Y, Z) - BULK)
     &                  / AX(COMP, 1, Y, Z)) * K
               QY   = ((U(COMP, X, 1, Z) - BULK)
     &                  / AY(COMP, X, 1, Z)) * K
               QZ   = ((U(COMP, X, Y, 1) - BULK)
     &                  / AZ(COMP, X, Y, 1)) * K
               DADX = ZERO
               DO 310 J = LOTXA(X), HITXA(X)
                  DADX = AX(COMP, J, Y, Z) * CTXA(J, X) + DADX
  310          CONTINUE

               DUDX = CTX1(0, X) * QX
               DO 320 J = LOTX1(X), HITX1(X)
                  DUDX = U(COMP, J, Y, Z) * CTX1(J, X) + DUDX
  320          CONTINUE

               D2UDX2 = CTX2(0, X) * QX
               DO 330 J = LOTX2(X), HITX2(X)
                  D2UDX2 = U(COMP, J, Y, Z) * CTX2(J, X) + D2UDX2
  330          CONTINUE

               DADY = ZERO
               DO 340 J = LOTYA(Y), HITYA(Y)
                  DADY = AY(COMP, X, J, Z) * CTYA(J, Y) + DADY
  340          CONTINUE

               DUDY = CTY1(0, Y) * QY
               DO 350 J = LOTY1(Y), HITY1(Y)
                  DUDY = U(COMP, X, J, Z) * CTY1(J, Y) + DUDY
  350          CONTINUE

               D2UDY2 = CTY2(0, Y) * QY
               DO 360 J = LOTY2(Y), HITY2(Y)
                  D2UDY2 = U(COMP, X, J, Z) * CTY2(J, Y) + D2UDY2
```

```
 360          CONTINUE

              DADZ = ZERO
              DO 370 J = LOTZA(Z), HITZA(Z)
                 DADZ = AZ(COMP, X, Y, J) * CTZA(J, Z) + DADZ
 370          CONTINUE

              DUDZ = CTZ1(0, Z) * QZ
              DO 380 J = LOTZ1(Z), HITZ1(Z)
                 DUDZ = U(COMP, X, Y, J) * CTZ1(J, Z) + DUDZ
 380          CONTINUE

              D2UDZ2 = CTZ2(0, Z) * QZ
              DO 390 J = LOTZ2(Z), HITZ2(Z)
                 D2UDZ2 = U(COMP, X, Y, J) * CTZ2(J, Z) + D2UDZ2
 390          CONTINUE

              G(COMP, X, Y, Z) =
     &          AX(COMP, X, Y, Z) * D2UDX2 + DADX * DUDX
     &        + AY(COMP, X, Y, Z) * D2UDY2 + DADY * DUDY
     &        + AZ(COMP, X, Y, Z) * D2UDZ2 + DADZ * DUDZ
     &        + R (COMP, X, Y, Z)
 300 CONTINUE

     GBULK = VG(PTP)

     DO 400 Z = 1, NGPTZ
        DO 400 Y = 1, NGPTY
           G(PTP, 1, Y, Z) = GBULK
 400 CONTINUE

     DO 500 Z = 1, NGPTZ
        DO 500 X = 1, NGPTX
           G(PTP, X, 1, Z) = GBULK
 500 CONTINUE

     DO 600 Y = 1, NGPTY
        DO 600 X = 1, NGPTX
           G(PTP, X, Y, 1) = GBULK
 600 CONTINUE

     IF(DEBUGG .GE. 2) WRITE(6, /) '        exit  DUDT'

     RETURN
     END
```

```
#FILE (MARK)CHIP/UTIL/FD ON STUDENTS
C-----------------------------------------------------------------------

C   Chip routine name   - FD

C   Creation date       - 16 Sep 85
C   Latest revision     - 17 Sep 85

C   Author              - Mark A. Burazin

C   Purpose             - Calculate finite difference formula to
C                           approximate an arbitrary order partial
C                           differential equation.

C   Hardware            - Burroughs Double Precision

C   Usage               - CALL FD(N, NP, JX, NC, JDER, POINTS, JAVAIL,
C                        &        JHI, JLO, COEF)

C   Arguments     N     - # of points in to be found including JX.
C                           (INPUT)
C                 NP    - # of points to be searched.
C                           NP >= N  (INPUT)
C                 NC    - Dimension variable for COEF.  (INPUT)
C                 JX    - Grid point where the FD approximation is
C                           desired.  (INPUT)
C                 JDER  - Order of derivative to be estimated.  (INPUT)
C                 POINTS - Vector of dimension NP containing the loca-
C                           tions of the points to be searched.
C                           POINTS(j) <= POINTS(j+1) for j = 1 to NP-1.
C                           (INPUT)
C                 JAVAIL - Vector of dimension NP containing the number
C                           of available values at each point in POINTS
C                           (INPUT)
C                 JHI   - High point of FD formula.  (OUTPUT)
C                 JLO   - Low point of FD formula.  (OUTPUT)
C                 COEF  - Array dimensioned (0:NC, NC) containing the
C                           FD formula coefficients.  (OUTPUT)
C                           On output, the coefficients are stored in
C                           COEF(JLO, JX) through COEF(JHI, JX).

C   Common        CHIP  - All global variables used by chip model.

C   Routines called     - FINITE, GAUSS, LOCATE

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FD    (N      , NP     , NC     , JX     , JDER   , POINTS
     &, JAVAIL, JHI    , JLO    , COEF   )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'
```

```
      DOUBLE PRECISION
     &  POINTS, COEF
     &, LOC    , C
     &, HD10   , TENTH  , ZERO   , HX

      INTEGER
     &  N      , NP     , NC     , JX     , JDER   , JAVAIL, JHI    , JLO
     &, J      , K      , L      , NCP1   , NC2
     &, ORDER  , REMAIN

      DIMENSION
     &  C      (15)
     &, COEF   (0:NC, NC)
     &, JAVAIL(NP)
     &, LOC    (15)
     &, ORDER  (15)
     &, POINTS(NP)
     &, REMAIN(15)

      DATA
     &  TENTH  /1.D-1/
     &, ZERO   /0.D+0/

C-----------------------------------------------------------------------
C-----initial housekeeping

      NCP1 = NC + 1
      NC2  = NC + NC
      DO 10 J = 1, NC
         C(J) = ZERO
   10 CONTINUE

C-----calculate FD formula for JDERth space derivative at NXth grid
C     point.  Find and use N points for FD formula.
C
C     Centerplane boundary conditions may be used to generate the FD
C     formulas but do not explicitly appear in the C vectors.  This is
C     because the first derivative at chip center = 0, so that term
C     drops out.  Since the formulas are valid for any value of first
C     derivative, this results in no loss of accuracy.
C
C     Note the distinction between real space and shadow space.
C     The one-eighth of the chip (surface to center in each dimension)
C     enclosed by the FD grid constitutes real space.  The rest of the
C     chip is shadow space.  Points in real and shadow space are used in
C     FD approximations.  A FD approximation consisting entirely of real
C     space points is then constructed by folding the shadow points onto
C     the corresponding real points through application of the symmetry
C     condition about the chip centerplanes.  Using shadow space in this
C-----fashion greatly increases the attainable accuracy.

C-----note: FINDIF expects grid spacing to either be constant or to
C-----geometrically INCREASE towards the chip center.
```

```
      HX   = POINTS(2) - POINTS(1)
      HD10 = HX * TENTH

      CALL LOCATE(N, NP, POINTS, POINTS(JX), JAVAIL, LOC, REMAIN)

      DO 100 K = 1, NC
         J = NCP1 - K
         IF(DABS(POINTS(J) - LOC(1)) .LT. HD10) JLO = J
  100 CONTINUE

      ORDER(1) = 0
      DO 200 J = 2, N
         ORDER(J) = 0
         IF(DABS(LOC(J-1) - LOC(J)) .LT. HD10) ORDER(J) = 1
  200 CONTINUE

      DO 300 J = 1, N
         LOC(J) = (LOC(J) - POINTS(JX)) / HX
  300 CONTINUE

      CALL FINITE(N, LOC, JDER, ORDER, C)

      JHI = JLO - 1
      DO 400 J = 1, N
         C(J) = C(J) * HX ** (ORDER(J) - JDER)
         IF(ORDER(J) .EQ. 0) JHI = JHI + 1
  400 CONTINUE

C-----handle real surface boundary condition if used

      COEF(0, JX) = ZERO
      IF(ORDER(2) .EQ. 1 .AND. JLO .EQ. 1) COEF(0, JX) = C(2)

      J = JLO - 1
      DO 500 K = 1, N
         IF(ORDER(K) .NE. 0) GO TO 500
         J = J + 1
         L = NC2 - J
         IF(J .LE. NC) COEF(J, JX) = C(K)
         IF(J .GT. NC) COEF(L, JX) = COEF(L, JX) + C(K)
  500 CONTINUE

      JHI = MIN0(JHI, NC)

      RETURN
      END
```

```
#FILE (MARK)CHIP/UTIL/AVE ON STUDENTS
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION AVE   (
     & LOCX   , LOCY   , LOCZ   , U      , BULK   , CX1    , CX2    , CY1
     &, CY2   , CZ1    , CZ2    , HIX1   , HIX2   , HIY1   , HIY2   , HIZ1
     &, HIZ2  , LOX1   , LOX2   , LOY1   , LOY2   , LOZ1   , LOZ2   , COMPHI
     &, COMPLO, NGPTX  , NGPTY  , NGPTZ  , COMP   , WHEREX, WHEREY, WHEREZ)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This function calculates 1D, 2D and 3D integral averages of a 4D
C     (comp, x, y, z) grid with arbitrary grid spacing (comp fixed)
C-----Creation date: 23 Oct 84---Last Update: 19 Sep 85

      IMPLICIT COMPLEX(A - Z)

      INTEGER
     & HIX1   , HIX2   , HIY1   , HIY2   , HIZ1   , HIZ2   , LOX1   , LOX2
     &, LOY1  , LOY2   , LOZ1   , LOZ2   , COMPHI, COMPLO
     &, NGPTX , NGPTY  , NGPTZ  , COMP   , WHEREX, WHEREY, WHEREZ
     &, INDEX , J      , NXM1   , NYM1   , NZM1   , X      , Y      , Z

      DOUBLE PRECISION
     & LOCX   , LOCY   , LOCZ   , U      , BULK   , CX1    , CX2    , CY1
     &, CY2   , CZ1    , CZ2
     &, AX    , AY     , AZ     , BX     , BY     , BZ     , C      , DUDX
     &, DUDY  , DUDZ   , H      , TEMP   , UX     , UY     , UZ
     &, D2UDX2, D2UDY2, D2UDZ2, XP1    , YP1    , ZERO   , ZP1    , R2
     &, R10   , R120

      DIMENSION
     & LOCX   (NGPTX)
     &, LOCY  (NGPTY)
     &, LOCZ  (NGPTZ)
     &, U     (COMPLO:COMPHI, NGPTX, NGPTY, NGPTZ)
     &, CX1   (0:NGPTX, NGPTX)
     &, CX2   (0:NGPTX, NGPTX)
     &, CY1   (0:NGPTY, NGPTY)
     &, CY2   (0:NGPTY, NGPTY)
     &, CZ1   (0:NGPTZ, NGPTZ)
     &, CZ2   (0:NGPTZ, NGPTZ)
     &, LOX1  (NGPTX)
     &, LOX2  (NGPTX)
     &, LOY1  (NGPTY)
     &, LOY2  (NGPTY)
     &, LOZ1  (NGPTZ)
     &, LOZ2  (NGPTZ)
     &, HIX1  (NGPTX)
     &, HIX2  (NGPTX)
     &, HIY1  (NGPTY)
     &, HIY2  (NGPTY)
     &, HIZ1  (NGPTZ)
     &, HIZ2  (NGPTZ)
     &, DUDX  (16)
```

```
     &, DUDY   (16)
     &, DUDZ   (16)
     &, D2UDX2 (16)
     &, D2UDY2 (16)
     &, D2UDZ2 (16)
     &, UX     (16)
     &, UY     (16)
     &, UZ     (16)

      DATA
     &  R2     /+5.D-1/
     &  R10    /+1.D-1/
     &, R120   /+8.3333333333333333333D-3/
     &, ZERO   /+0.D+0/

C-----Calculate (1D, 2D, 3D) integral average.  This is done by first
C     integrating the (line, rectangle, cuboid) grid the corrected
C     trapezoidal rule, then dividing the (line, surface, space)
C     integral by the grid (length, area, volume).
C
C     There are seven cases to consider: line integrals in X, Y, & Z;
C     surface integrals on the XY, XZ, & YZ planes; and a space integral
C     over XYZ.  WHEREX, WHEREY & WHEREZ indicate the dimension and
C     location of the integral.  A zero value indicates the dimension
C     is to be averaged; a nonzero value fixes the integral location
C     (WHEREX = 1 means chip surface, WHEREX = NX means chip center).
C
C     case  description WHEREX   WHEREY   WHEREZ (N = non-zero)
C      1        'OD'        N        N        N (degenerate case)
C      2        1D X        0        N        N
C      3        1D Y        N        0        N
C      4        1D Z        N        N        0
C      5        2D XY       0        0        N
C      6        2D XZ       0        N        0
C      7        2D YZ       N        0        0
C      8        3D XYZ      0        0        0
C
C     The corrected trapezoidal rule uses the function values and first
C-----space derivatives calculated from finite difference formulas.

      AX = LOCX(1)
      AY = LOCY(1)
      AZ = LOCZ(1)
      BX = LOCX(NGPTX)
      BY = LOCY(NGPTY)
      BZ = LOCZ(NGPTZ)
      NXM1 = NGPTX - 1
      NYM1 = NGPTY - 1
      NZM1 = NGPTZ - 1

      INDEX = 1
      IF(WHEREX .EQ. 0) INDEX = INDEX + 4
      IF(WHEREY .EQ. 0) INDEX = INDEX + 2
      IF(WHEREZ .EQ. 0) INDEX = INDEX + 1
```

```
      GO TO (1, 4, 3, 7, 2, 6, 5, 8), INDEX

C-----degerate case ('0D average')

    1 CONTINUE
         AVE = U(COMP, WHEREX, WHEREY, WHEREZ)
      GO TO 9

C-----1D average in X direction

    2 CONTINUE
         DO 200 X = 1, NGPTX
            UX(X) = U(COMP, X, WHEREY, WHEREZ)
  200    CONTINUE
         DO 210 X = 1, NGPTX
            TEMP = ZERO
            DO 220 J = LOX1(X), HIX1(X)
               TEMP = CX1(J, X) * UX(J) + TEMP
  220       CONTINUE
            DUDX(X) = TEMP
            TEMP = ZERO
            DO 230 J = LOX2(X), HIX2(X)
               TEMP = CX2(J, X) * UX(J) + TEMP
  230       CONTINUE
            D2UDX2(X) = TEMP
  210    CONTINUE

         TEMP = ZERO
         DO 240 X = 1, NXM1
            XP1  = X + 1
            H    = LOCX(XP1) - LOCX(X)
            TEMP =(UX(X) + UX(XP1)) * H * R2
     &           +(DUDX(X) - DUDX(XP1)) * H * H * R10
     &           +(D2UDX2(X) + D2UDX2(XP1)) * H * H * H * R120 + TEMP
  240    CONTINUE

         AVE = TEMP / (BX - AX)
      GO TO 9

C-----1D average in Y direction

    3 CONTINUE
         DO 300 Y = 1, NGPTY
            UY(Y) = U(COMP, WHEREX, Y, WHEREZ)
  300    CONTINUE
         DO 310 Y = 1, NGPTY
            TEMP = ZERO
            DO 320 J = LOY1(Y), HIY1(Y)
               TEMP = CY1(J, Y) * UY(J) + TEMP
  320       CONTINUE
            DUDY(Y) = TEMP
            TEMP = ZERO
            DO 330 J = LOY2(Y), HIY2(Y)
               TEMP = CY2(J, Y) * UY(J) + TEMP
```

```
  330         CONTINUE
              D2UDY2(Y) = TEMP
  310     CONTINUE

          TEMP = ZERO
          DO 340 Y = 1, NYM1
             YP1  = Y + 1
             H    = LOCY(YP1) - LOCY(Y)
             TEMP =(UY(Y) + UY(YP1)) * H * R2
     &            +(DUDY(Y) - DUDY(YP1)) * H * H * R10
     &            +(D2UDY2(Y) + D2UDY2(YP1)) * H * H * H * R120 + TEMP
  340     CONTINUE

          AVE = TEMP / (BY - AY)
      GO TO 9

C-----1D average in Z direction

    4 CONTINUE
          DO 400 Z = 1, NGPTZ
             UZ(Z) = U(COMP, WHEREX, WHEREY, Z)
  400     CONTINUE
          DO 410 Z = 1, NGPTZ
             TEMP = ZERO
             DO 420 J = LOZ1(Z), HIZ1(Z)
                TEMP = CZ1(J, Z) * UZ(J) + TEMP
  420        CONTINUE
             DUDZ(Z) = TEMP
             TEMP = ZERO
             DO 430 J = LOZ2(Z), HIZ2(Z)
                TEMP = CZ2(J, Z) * UZ(J) + TEMP
  430        CONTINUE
             D2UDZ2(Z) = TEMP
  410     CONTINUE

          TEMP = ZERO
          DO 440 Z = 1, NZM1
             ZP1  = Z + 1
             H    = LOCZ(ZP1) - LOCZ(Z)
             TEMP =(UZ(Z) + UZ(ZP1)) * H * R2
     &            +(DUDZ(Z) - DUDZ(ZP1)) * H * H * R10
     &            +(D2UDZ2(Z) + D2UDZ2(ZP1)) * H * H * H * R120 + TEMP
  440     CONTINUE

          AVE = TEMP / (BZ - AZ)
      GO TO 9

C-----2D average on a XY plane

    5 CONTINUE
          DO 500 Y = 1, NGPTY
             DO 510 X = 1, NGPTX
                UX(X) = U(COMP, X, Y, WHEREZ)
  510        CONTINUE
```

```
            DO 520 X = 1, NGPTX
               TEMP = ZERO
               DO 530 J = LOX1(X), HIX1(X)
                  TEMP = CX1(J, X) * UX(J) + TEMP
  530          CONTINUE
               DUDX(X) = TEMP
               TEMP = ZERO
               DO 540 J = LOX2(X), HIX2(X)
                  TEMP = CX2(J, X) * UX(J) + TEMP
  540          CONTINUE
               D2UDX2(X) = TEMP
  520       CONTINUE

            TEMP = ZERO
            DO 550 X = 1, NXM1
               XP1  = X + 1
               H    = LOCX(XP1) - LOCX(X)
               TEMP =(UX(X) + UX(XP1)) * H * R2
     &             +(DUDX(X) - DUDX(XP1)) * H * H * R10
     &             +(D2UDX2(X) + D2UDX2(XP1)) * H * H * H * R120 + TEMP
  550       CONTINUE
            UY(Y) = TEMP
  500    CONTINUE
         DO 560 Y = 1, NGPTY
            TEMP = ZERO
            DO 570 J = LOY1(Y), HIY1(Y)
               TEMP = CY1(J, Y) * UY(J) + TEMP
  570       CONTINUE
            DUDY(Y) = TEMP
            TEMP = ZERO
            DO 580 J = LOY2(Y), HIY2(Y)
               TEMP = CY2(J, Y) * UY(J) + TEMP
  580       CONTINUE
            D2UDY2(Y) = TEMP
  560    CONTINUE

         TEMP = ZERO
         DO 590 Y = 1, NYM1
            YP1  = Y + 1
            H    = LOCY(YP1) - LOCY(Y)
            TEMP =(UY(Y) + UY(YP1)) * H * R2
     &          +(DUDY(Y) - DUDY(YP1)) * H * H * R10
     &          +(D2UDY2(Y) + D2UDY2(YP1)) * H * H * H * R120 + TEMP
  590    CONTINUE

         AVE = TEMP / ((BX - AX) * (BY - AY))
      GO TO 9

C-----2D average on a XZ plane

    6 CONTINUE
         DO 600 Z = 1, NGPTZ
            DO 610 X = 1, NGPTX
               UX(X) = U(COMP, X, WHEREY, Z)
```

```
 610       CONTINUE
           DO 620 X = 1, NGPTX
              TEMP = ZERO
              DO 630 J = LOX1(X), HIX1(X)
                 TEMP = CX1(J, X) * UX(J) + TEMP
 630          CONTINUE
              DUDX(X) = TEMP
              TEMP = ZERO
              DO 640 J = LOX2(X), HIX2(X)
                 TEMP = CX2(J, X) * UX(J) + TEMP
 640          CONTINUE
              D2UDX2(X) = TEMP
 620       CONTINUE

           TEMP = ZERO
           DO 650 X = 1, NXM1
              XP1  = X + 1
              H    = LOCX(XP1) - LOCX(X)
              TEMP =(UX(X) + UX(XP1)) * H * R2
     &           +(DUDX(X) - DUDX(XP1)) * H * H * R10
     &           +(D2UDX2(X) + D2UDX2(XP1)) * H * H * H * R120 + TEMP
 650       CONTINUE
           UZ(Z) = TEMP
 600    CONTINUE
        DO 660 Z = 1, NGPTZ
           TEMP = ZERO
           DO 670 J = LOZ1(Z), HIZ1(Z)
              TEMP = CZ1(J, Z) * UZ(J) + TEMP
 670       CONTINUE
           DUDZ(Z) = TEMP
           TEMP = ZERO
           DO 680 J = LOZ2(Z), HIZ2(Z)
              TEMP = CZ2(J, Z) * UZ(J) + TEMP
 680       CONTINUE
           D2UDZ2(Z) = TEMP
 660    CONTINUE

        TEMP = ZERO
        DO 690 Z = 1, NZM1
           ZP1  = Z + 1
           H    = LOCZ(ZP1) - LOCZ(Z)
           TEMP =(UZ(Z) + UZ(ZP1)) * H * R2
     &          +(DUDZ(Z) - DUDZ(ZP1)) * H * H * R10
     &          +(D2UDZ2(Z) + D2UDZ2(ZP1)) * H * H * H * R120 + TEMP
 690    CONTINUE

        AVE = TEMP / ((BX - AX) * (BZ - AZ))
      GO TO 9

C-----2D average on a YZ plane

    7 CONTINUE
        DO 700 Z = 1, NGPTZ
           DO 710 Y = 1, NGPTY
```

```
            UY(Y) = U(COMP, WHEREX, Y, Z)
  710     CONTINUE
          DO 720 Y = 1, NGPTY
             TEMP = ZERO
             DO 730 J = LOY1(Y), HIY1(Y)
                TEMP = CY1(J, Y) * UY(J) + TEMP
  730        CONTINUE
             DUDY(Y) = TEMP
             TEMP = ZERO
             DO 740 J = LOY2(Y), HIY2(Y)
                TEMP = CY2(J, Y) * UY(J) + TEMP
  740        CONTINUE
             D2UDY2(Y) = TEMP
  720     CONTINUE

          TEMP = ZERO
          DO 750 Y = 1, NYM1
             YP1  = Y + 1
             H    = LOCY(YP1) - LOCY(Y)
             TEMP =(UY(Y) + UY(YP1)) * H * R2
     &             +(DUDY(Y) - DUDY(YP1)) * H * H * R10
     &             +(D2UDY2(Y) + D2UDY2(YP1)) * H * H * H * R120 + TEMP
  750     CONTINUE
          UZ(Z) = TEMP
  700  CONTINUE
       DO 760 Z = 1, NGPTZ
          TEMP = ZERO
          DO 770 J = LOZ1(Z), HIZ1(Z)
             TEMP = CZ1(J, Z) * UZ(J) + TEMP
  770     CONTINUE
          DUDZ(Z) = TEMP
          TEMP = ZERO
          DO 780 J = LOZ2(Z), HIZ2(Z)
             TEMP = CZ2(J, Z) * UZ(J) + TEMP
  780     CONTINUE
          D2UDZ2(Z) = TEMP
  760  CONTINUE

       TEMP = ZERO
       DO 790 Z = 1, NZM1
          ZP1  = Z + 1
          H    = LOCZ(ZP1) - LOCZ(Z)
          TEMP =(UZ(Z) + UZ(ZP1)) * H * R2
     &          +(DUDZ(Z) - DUDZ(ZP1)) * H * H * R10
     &          +(D2UDZ2(Z) + D2UDZ2(ZP1)) * H * H * H * R120 + TEMP
  790  CONTINUE

       AVE = TEMP / ((BY - AY) * (BZ - AZ))
    GO TO 9

C-----3D average

    8 CONTINUE
       DO 800 Z = 1, NGPTZ
```

```
          DO 805 Y = 1, NGPTY
             DO 810 X = 1, NGPTX
                UX(X) = U(COMP, X, Y, Z)
 810         CONTINUE
             DO 815 X = 1, NGPTX
                TEMP = ZERO
                DO 820 J = LOX1(X), HIX1(X)
                   TEMP = CX1(J, X) * UX(J) + TEMP
 820            CONTINUE
                DUDX(X) = TEMP
                TEMP = ZERO
                DO 825 J = LOX2(X), HIX2(X)
                   TEMP = CX2(J, X) * UX(J) + TEMP
 825            CONTINUE
                D2UDX2(X) = TEMP
 815         CONTINUE

             TEMP = ZERO
             DO 830 X = 1, NXM1
                XP1  = X + 1
                H    = LOCX(XP1) - LOCX(X)
                TEMP =(UX(X) + UX(XP1)) * H * R2
     &               +(DUDX(X) - DUDX(XP1)) * H * H * R10
     &               +(D2UDX2(X) + D2UDX2(XP1)) * H*H*H * R120 + TEMP
 830         CONTINUE
             UY(Y) = TEMP
 805      CONTINUE
          DO 835 Y = 1, NGPTY
             TEMP = ZERO
             DO 840 J = LOY1(Y), HIY1(Y)
                TEMP = CY1(J, Y) * UY(J) + TEMP
 840         CONTINUE
             DUDY(Y) = TEMP
             TEMP = ZERO
             DO 845 J = LOY2(Y), HIY2(Y)
                TEMP = CY2(J, Y) * UY(J) + TEMP
 845         CONTINUE
             D2UDY2(Y) = TEMP
 835      CONTINUE

          TEMP = ZERO
          DO 850 Y = 1, NYM1
             YP1  = Y + 1
             H    = LOCY(YP1) - LOCY(Y)
             TEMP =(UY(Y) + UY(YP1)) * H * R2
     &            +(DUDY(Y) - DUDY(YP1)) * H * H * R10
     &            +(D2UDY2(Y) + D2UDY2(YP1)) * H * H * H * R120 + TEMP
 850      CONTINUE
          UZ(Z) = TEMP
 800   CONTINUE
       DO 855 Z = 1, NGPTZ
          TEMP = ZERO
          DO 860 J = LOZ1(Z), HIZ1(Z)
             TEMP = CZ1(J, Z) * UZ(J) + TEMP
```

```
860         CONTINUE
            DUDZ(Z) = TEMP
            TEMP = ZERO
            DO 865 J = LOZ2(Z), HIZ2(Z)
               TEMP = CZ2(J, Z) * UZ(J) + TEMP
865         CONTINUE
            D2UDZ2(Z) = TEMP
855      CONTINUE

         TEMP = ZERO
         DO 870 Z = 1, NZM1
            ZP1  = Z + 1
            H    = LOCZ(ZP1) - LOCZ(Z)
            TEMP =(UZ(Z) + UZ(ZP1)) * H * R2
   &            +(DUDZ(Z) - DUDZ(ZP1)) * H * H * R10
   &            +(D2UDZ2(Z) + D2UDZ2(ZP1)) * H * H * H * R120 + TEMP
870      CONTINUE

         AVE = TEMP / ((BX - AX) * (BY - AY) * (BZ - AZ))
      GO TO 9

    9 CONTINUE
      RETURN
      END
```

```
#FILE (MARK)CHIP/UTIL/DUMP ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name   - DUMP

C   Creation date       - 20 Sep 85
C   Latest revision     - 20 Sep 85

C   Author              - Mark A. Burazin

C   Purpose             - Write 4D array A to file F

C   Hardware            - Burroughs Double Precision

C   Usage               - CALL DUMP(A, JF, NSIG, QLABEL, LABEL, JCHI
C                          &        , JCLO, NX, NY, NZ)

C   Arguments      A    - An array dimensioned (JCLO:JCHI, NX, NY, NZ)
C                           which is to be printed.  (INPUT)
C                  JF   - Unit number of output file.  (INPUT)
C                  NSIG  - Number of significant digits desired.  (INPUT)
C                  QLABEL - Number associated with this printout.  (INPUT)
C                  LABEL - A one to SIX character string associated with
C                           this printout.  (INPUT)

C   Routines called     - ROUND

C   Reference           - none

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE DUMP  (A      , F      , NSIG   , QLABEL, LABEL , CHI
     &                        , CLO    , NX     , NY     , NZ     )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      INTEGER
     &  F      , NSIG   , LABEL  , CHI    , CLO    , NX     , NY     , NZ
     &, COMP   , X      , Y      , Z

      DOUBLE PRECISION
     &  A      , QLABEL
     &, ROUND

      DIMENSION
     &  A(CLO:CHI, NX, NY, NZ)

 9001 FORMAT(1X,A6)
C-----------------------------------------------------------------------

      WRITE(F, 9001) LABEL
      WRITE(F, /) ROUND(QLABEL, NSIG), CLO, CHI, NX, NY, NZ
      DO 100 Z = 1, NZ
```

```
          DO 100 Y = 1, NY
             DO 100 X = 1, NX
                WRITE(F, /) (ROUND(A(COMP, X, Y, Z), NSIG)
     &                       , COMP = CLO, CHI)
  100 CONTINUE

      RETURN
      END
```

```
#FILE (MARK)CHIP/UTIL/GRID ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name   - GRID

C   Creation date       - 13 Aug 84
C   Latest revision     - 19 Aug 85

C   Author              - Mark A. Burazin

C   Purpose             - Calculates grid spacings as a geometric
C                         progression.

C   Usage               - CALL GRID  (N, A, D, B)

C   Arguments     N     - Number of intervals between chip edge and
C                           chip center. (INPUT)
C                 A     - Length of interval adjoining chip edge.
C                           (INPUT)
C                 D     - Distance between chip edge and chip center.
C                           (INPUT)
C                 B     - A vector of length N.  On output, B contains
C                           the distances of the grid points from the
C                           edge of the chip.  (OUTPUT)

C   Routines called     - None

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE GRID(N, A, D, B)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      INTEGER
     &  N
     &, J

      DOUBLE PRECISION
     &  A     , D     , B(1)
     &, ALPHA , ERROR , NRATIO, ORATIO, SUM   , TOL

      DATA
     &  ALPHA  /1.25D-01/
     &, TOL    /1.00D-21/

C-----------------------------------------------------------------------

      ORATIO = D ** (1.DO / (N - 2.DO))

    1 CONTINUE
      SUM = 0.DO
      DO 100 J = 0, N-3
         SUM = ORATIO**J + SUM
```

```
100 CONTINUE

    NRATIO = (D - A) / (SUM * A)
    ORATIO = (1.D0 - ALPHA) * ORATIO + ALPHA * NRATIO
    ERROR  = DABS(NRATIO - ORATIO) * 2.D0 / (NRATIO + ORATIO)
    IF(ERROR .GE. TOL) GO TO 1

    B(1) = 0.D0
    B(2) = A
    B(N) = D
    DO 200 J = 3, N-1
       B(J) = (B(J-1) - B(J-2)) * ORATIO + B(J-1)
200 CONTINUE

    END       % GRID
```

```
#FILE (MARK)CHIP/UTIL/ERRFD ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name    - ERRFD

C   Creation date        -  1 Sep 84
C   Latest revision      - 17 Sep 85

C   Author               - Mark A. Burazin

C   Purpose              - Find error of FD approximation of space
C                             derivatives.  Hardwired for CHIP.

C     IDER      IDER    N
C    d    f / dx      = sum   COEF(k, JX0) f[x(k)]
C                      k=1

C   Hardware             - Burroughs Double Precision

C   Usage                - E = ERRFD(JX0, JDER, JHI, JLO, JHIX, JLOX
C                          &        , FLOCX, COEF, KDA)

C   Arguments    JX0     - Evaluation point for derivative. (INPUT)
C                JDER    - Order of derivative. (INPUT)
C                JHI     - Upper bound of vectors. (INPUT)
C                JLO     - Lower bound of first index of COEF. (INPUT)
C                JHIX    - Upper bound of FD formula. (INPUT)
C                JLOX    - Lower bound of FD formula. (INPUT)
C                FLOCX   - Vector of length JHI containing the positions
C                            of the grid points relative to the chip
C                            surface. (INPUT)
C                COEF    - Vector dimensioned (JLO:JHI, JHI) containing
C                            the finite difference formula coefficients.
C                            (INPUT)
C                KDA     - Bi * Achip / Vchip. (INPUT)

C   Routines called      - none

C   Reference            - none

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION ERRFD  (X0     , DER    , HI     , LO
     &                           , HIX    , LOX    , LOCX   , COEF   , KDA    )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      INTEGER
     &  X0     , DER    , HI     , LO     , HIX    , LOX
     &, J

      DOUBLE PRECISION
     &  LOCX   , COEF   , KDA
```

```
     &, ACTUAL, CENTER, DERIV , DUDXS , NORMAL, UB     , US     , ZERO

      DIMENSION
     &  LOCX(HI)
     &, COEF(LO:HI, HI)

      DATA
     &  ZERO   /+0.D+0/

C-----------------------------------------------------------------------

      CENTER = LOCX(HI)

      IF(DER .EQ. 1) NORMAL = DABS(DSINH(LOCX(1)  - CENTER))
      IF(DER .EQ. 2) NORMAL = DABS(DCOSH(LOCX(1)  - CENTER))
      IF(DER .EQ. 1) ACTUAL = DSINH(LOCX(X0) - CENTER) / NORMAL
      IF(DER .EQ. 2) ACTUAL = DCOSH(LOCX(X0) - CENTER) / NORMAL

      DERIV  = ZERO
      IF(LO .NE. 0 .OR. KDA .EQ. ZERO) GO TO 1
         DUDXS = DSINH(LOCX(1) - CENTER) / NORMAL     % du/dx at surface
         DERIV = COEF(0, X0) * DUDXS
    1 CONTINUE
      DO 100 J = LOX, HIX
         DERIV = DCOSH(LOCX(J) - CENTER) * COEF(J, X0) / NORMAL + DERIV
  100 CONTINUE

      ERRFD = DERIV - ACTUAL

      RETURN
      END
```

```
#FILE (MARK)CHIP/UTIL/GAUSS ON STUDENTS
C----------------------------------------------------------------------

C   Util routine name    - GAUSS

C   Creation date        - 23 Aug 84
C   Latest revision      - 24 Aug 84

C   Author               - Mark A. Burazin

C   Purpose              - Direct solution of a linear system of
C                            equations by Gaussian elimination with
C                            row and column pivoting.

C   Hardware             - Burroughs Double Precision

C   Usage                - CALL GAUSS (A, X, N, JA, KWK1, KWK2, WORK)

C   Arguments      A     - N by NP1 input matrix containing the
C                            coefficient matrix of the equation
C                            AX = B. (INPUT)
C                  X     - Solution vector of length N. (OUTPUT)
C                  N     - Number of equations and unknowns. (INPUT)
C                  JA    - Row dimension of A exactly as specified in
C                            the dimension statement of the calling
C                            program. (INPUT)
C           KWK1, KWK2   - Work vectors of length >= NP1.
C                  WORK  - Work vector of length >= N.

C   Routines called      - None

C   References           - "Numerical Analysis", 2nd ed.  Ch. 6
C                            R.L. Burden, J.D. Faires, A.C. Reynolds
C                            Prindle, Weber, & Schmidt
C                            Boston (1981).

C                          "Numerical Methods, Software, and Analysis:
C                            IMSL Reference Edition", Ch. 6
C                            J. R. Rice
C                            McGraw-Hill
C                            New York (1983).

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE GAUSS (
     & A      , X      , N      , JA     , NROW   , NCOL   , WORK   )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      DOUBLE PRECISION
     & A      , X      , WORK
     &, BUFFER, EPS    , PIVOT  , ROUND  , TEMP
```

```
      INTEGER
     &  N      , JA     , NROW   , NCOL
     &, C      , NM1    , NP1    , OUT    , PCOL   , PROW   , R      , S

      DIMENSION
     &  A(JA, 1)        , X(1)   , WORK(1)
     &, NROW(1)         , NCOL(1)

      DATA
C    &  EPS    /01301000000000000/                   % single precision
     &  EPS    /01451000000000000000000000000000/    % double precision
     &, OUT    /     6/

 9001 FORMAT(///' Matrix is numerically singular, pivot = ',1PD12.4//
     &           '    ***  PROGRAM HALTED  ***')

C-----Initial housekeeping---------------------------------------------

      BUFFER = EPS * N            % Rice p137
      NM1 = N - 1
      NP1 = N + 1
      NCOL(NP1) = NP1
      DO 100 S = 1, N
         NCOL(S) = S
         NROW(S) = S
  100 CONTINUE

C-----Elimination process (S => stage, R => row, C => column)

      DO 200 S = 1, NM1

C--------Find pivot(A(PROW, PCOL))

         PROW = 0
         PCOL = 0
         PIVOT = BUFFER

         DO 300 C = S, N
            DO 300 R = S, N
               TEMP = DABS(A(NROW(R), NCOL(C)))
               IF(TEMP .LE. PIVOT) GO TO 300
                 PROW = R
                 PCOL = C
                 PIVOT = TEMP
  300    CONTINUE

C--------Check for numerical singularity

         IF(PIVOT .GT. BUFFER) GO TO 40
           GO TO 99999
   40    CONTINUE

C--------Simulate row and/or column interchange if necessary
```

```
      IF(NROW(S) .EQ. NROW(PROW)) GO TO 51
        TEMP = NROW(S)
        NROW(S) = NROW(PROW)
        NROW(PROW) = TEMP
   51 CONTINUE

      IF(NCOL(S) .EQ. NCOL(PCOL)) GO TO 52
        TEMP = NCOL(S)
        NCOL(S) = NCOL(PCOL)
        NCOL(PCOL) = TEMP
   52 CONTINUE

C--------Eliminate non-zero entries below diagonal in column S

      DO 600 R = S+1, N
         TEMP = A(NROW(R), NCOL(S)) / A(NROW(S), NCOL(S))
         DO 600 C = 1, NP1
            A(NROW(R), NCOL(C)) = A(NROW(R), NCOL(C))
     &                          - A(NROW(S), NCOL(C)) * TEMP
  600 CONTINUE
  200 CONTINUE

C-----Last check for numerical singularity

      PIVOT = DABS(A(NROW(N), NCOL(N)))
      IF(PIVOT .GT. BUFFER) GO TO 90
        GO TO 99999
   90 CONTINUE

C-----Backwards substitution

      X(N) = A(NROW(N), NP1) / A(NROW(N), NCOL(N))

      DO 1100 S = 1, NM1
         R = N - S
         X(R) = A(NROW(R), NP1)
         DO 1110 C = R+1, N
            X(R) = X(R) - A(NROW(R), NCOL(C)) * X(C)
 1110    CONTINUE
         X(R) = X(R) / A(NROW(R), NCOL(R))
 1100 CONTINUE

C-----Reorder solution vector X using column order vector NCOL

      DO 1200 C = 1, N
         WORK(C) = X(C)
 1200 CONTINUE
      DO 1210 C = 1, N
         X(NCOL(C)) = WORK(C)
 1210 CONTINUE

      RETURN

C-----Array is numerically singular. Report & halt
```

```
99999 CONTINUE
      WRITE(OUT, 9001) PIVOT
      DO 1300 R = 1, N
         WRITE(OUT, /) 'A row ', R, (ROUND(A(R, C), 4), C = 1, NP1)
 1300 CONTINUE
      WRITE(OUT, /) ' X     ', (ROUND(X(R), 4), R = 1, N)
      WRITE(OUT, /) ' N = ', N, 'JA = ', JA
      WRITE(OUT, /) 'NROW  ', (NROW(R), R = 1, N)
      WRITE(OUT, /) 'NCOL  ', (NCOL(C), C = 1, NP1)
      WRITE(OUT, /) 'WORK  ', (ROUND(WORK(R), 4), R = 1, N)
      STOP
      END
```

```
#FILE (MARK)CHIP/UTIL/ROUND ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name    - ROUND

C   Creation date        - 15 Nov 83
C   Latest revision      - 27 Nov 84

C   Author               - Mark A. Burazin

C   Purpose              - Round double precision variable to specified
C                            number of significant digits

C   Usage                - R = ROUND(U, N)

C   Arguments      U     - Number to be rounded. (INPUT)
C                  N     - Number of significant digits desired in
C                            rounded number. (INPUT)

C   Routines called      - None

C   Reference            - None

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      DOUBLE PRECISION FUNCTION ROUND (
     &  UNROUN, SIG   )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      DOUBLE PRECISION
     &  UNROUN

      DOUBLE PRECISION
     &  AUNRND, FACTOR, TEMPOR

      INTEGER
     &  SIG
     &, SCALE , SIGNIF

C-----Handle UNROUN = 0 case--------------------------------------------

      IF(UNROUN .EQ. 0D0) ROUND = UNROUN
      IF(UNROUN .EQ. 0D0) GO TO 99

C-----Force 1 <= SIGNIF <= 20

      SIGNIF = SIG
      IF(SIG .LT.  1) SIGNIF =  1
      IF(SIG .GT. 20) SIGNIF = 20

C-----Round off UNROUN(ed) to SIGNIF significant figures
```

```
      AUNRND = DABS(UNROUN)
      SCALE  = DLOG10(AUNRND)

      IF(AUNRND .GE. 1) SCALE = SIGNIF - SCALE - 1
      IF(AUNRND .LT. 1) SCALE = SIGNIF - SCALE

      FACTOR = 1D1 ** SCALE
      TEMPOR = IDINT(AUNRND * FACTOR + 5D-1)
      TEMPOR = DSIGN(TEMPOR, UNROUN)

      ROUND  = TEMPOR / FACTOR

   99 CONTINUE
      RETURN
      END
```

```
#FILE (MARK)CHIP/UTIL/TIMER ON STUDENTS
$     OPT = 0
$     SET OWN        % Set if Opt = 0 or -1, reset if Opt = 1
$     SET OWNARRAYS % Always set

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE TIMER (
     &  HOUR  , MINUTE, SECOND, TYPE  )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine gets elapsed processor or I/O time---------------
C-----Creation date: 06 Aug 82----Last update: 27 Oct 82---------------

      IMPLICIT COMPLEX(A - Z)

      DOUBLE PRECISION
     &  SECOND

      REAL
     &  TIM

      INTEGER
     &  HOUR, MINUTE, TYPE

C-----Calculate hours, minutes & seconds as per TYPE------------------
C     using Burroughs TIME intrinsic
C     2 => processor time, 3 => input/output time
C-----(both in units of 1/60th of a second)---------------------------

      IF(TYPE .NE. 2 .AND. TYPE .NE. 3) RETURN

      TIM    = TIME(TYPE) / 6E1 % convert to seconds
      HOUR   = TIM / 36E2
      MINUTE = (TIM - HOUR * 36E2) / 6E1
      SECOND = TIM - HOUR * 36E2 - MINUTE * 6E1

      END
```

```
#FILE (MARK)CHIP/UTIL/FINITE ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name    - FINITE

C   Creation date        -  1 Sep 84
C   Latest revision      - 21 Mar 85

C   Author               - Mark A. Burazin

C   Purpose              - Calculation of finite difference formulas
C                            to approximate space derivatives.

C      IDER       IDER     N              JDER(k)                JDER(k)
C     d     f / dx      = sum    COEF(k) d         f[x(k)] / dx
C                         k=1

C   Hardware             - Burroughs Double Precision

C   Usage                - CALL FINITE(N, X, IDER, JDER, C)

C   Arguments      N     - # of points in finite difference formula.
C                            N <= 15. (INPUT)
C                  X     - Vector of dimension N containing the distance
C                            of each grid point relative to the point
C                            where the derivative is evaluated. (INPUT)
C                  IDER  - Order of derivative w/respect to X.
C                            IDER <= N-1. (INPUT)
C                  JDER  - Vector of dimension N containing the deriva-
C                            tive order of each grid point(0 for function
C                            value, 1 for 1st derivative, etc.)
C                            JDER <= IDER. (INPUT)
C                  C     - Vector of dimension N containing the finite
C                            difference formula coefficients. (OUTPUT)

C   Routines called      - GAUSS

C   Reference            - "Applied Numerical Analysis", Appendix B
C                            C.F. Gerald
C                            Addison-Wesley
C                            Reading, Mass. (1978).

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FINITE(
     &  N      , X      , DER    , XDER   , COEF   )
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      DOUBLE PRECISION
     &  X      , COEF
     &, A      , WORK
     &, FACT   , ONE    , XC     , ZERO
```

```
      INTEGER
     &  N       , DER    , XDER
     &, JA      , JWK1   , JWK2
     &, C       , D      , DEBUGG, OUT    , R      , XDERC

      DIMENSION
     &  A   (15, 16)
     &, COEF(1)
     &, FACT(0:14)
     &, JWK1(16)
     &, JWK2(16)
     &, WORK(15)
     &, X   (1)
     &, XDER(1)

      DATA
     &  DEBUGG / 0/
     &, FACT   /          1.D0
     &         ,          1.D0
     &         ,          2.D0
     &         ,          6.D0
     &         ,         24.D0
     &         ,         12.D1
     &         ,         72.D1
     &         ,        504.D1
     &         ,       4032.D1
     &         ,      36288.D1
     &         ,      36288.D2
     &         ,     399168.D2
     &         ,    4790016.D2
     &         ,   62270208.D2
     &         ,  871782912.D2 /
     &, JA     /15   /
     &, ONE    / 1.D0/
     &, OUT    / 6   /
     &, ZERO   / 0.D0/

C-----Initial housekeeping--------------------------------------------

      DO 100 R = 1, N
         A(R, N+1) = ZERO
  100 CONTINUE

C-----Constuct system of equations so that finite difference formula
C     exactly satisifies the DERth derivative of the set of basis
C-----functions  X**j, j = 0, N-1

      DO 200 C = 1, N
         XDERC = XDER(C)
         XC    = X(C)
         DO 210 R = 1, XDERC
            A(R, C) = ZERO
  210    CONTINUE
```

```
         DO 200 R = XDERC + 1, N
            D       = R - XDERC - 1
            A(R, C) = FACT(R-1) / FACT(D) * XC**D
  200 CONTINUE

      A(DER+1, N+1) = FACT(DER)

C-----Solve linear system of equations for finite difference formula
C-----coefficients (COEF)

      IF(DEBUGG .LT. 10) GO TO 1
         WRITE(OUT, /) ' FINITE just before GAUSS'
         WRITE(OUT, *//) N, DER
         DO 300 R = 1, N
            WRITE(OUT, /) ' A row ', R, (ROUND(A(R, C), 4), C = 1, N+1)
  300    CONTINUE
         WRITE(OUT, /) ' X     ', (ROUND(X(R), 4), R = 1, N)
         WRITE(OUT, /) ' XDER  ', (XDER(R), R = 1, N)
         WRITE(OUT, /) ' '
    1 CONTINUE

      CALL GAUSS (A     , COEF  , N     , JA    , JWK1  , JWK2
     &            , WORK  )

      IF(DEBUGG .LT. 10) GO TO 2
         WRITE(OUT, /) ' FINITE just after GAUSS'
         WRITE(OUT, *//) N, DER
         DO 400 R = 1, N
            WRITE(OUT, /) ' A row ', R, (ROUND(A(R, C), 4), C = 1, N+1)
  400    CONTINUE
         WRITE(OUT, /) ' COEF  ', (ROUND(COEF(R), 4), R = 1, N)
         WRITE(OUT, /) ' NROW  ', (JWK1(R), R = 1, N+1)
         WRITE(OUT, /) ' NCOL  ', (JWK2(C), C = 1, N+1)
         WRITE(OUT, /) ' '
    2 CONTINUE

      RETURN
      END
```

```
#FILE (MARK)CHIP/UTIL/LOCATE ON STUDENTS
C-----------------------------------------------------------------------

C   Util routine name   - LOCATE

C   Creation date       - 22 Mar 85
C   Latest revision     - 19 Aug 85

C   Author              - Mark A. Burazin

C   Purpose             - Search for N closest values to X in POINTS.
C                           handles multiple values at a single point.

C    +                                                                +
C    +-+---+-------X--------------+-------------------------------+

C   Hardware            - Burroughs Double Precision

C   Usage               - CALL LOCATE(N, NP, POINTS, X, IAVAIL, LOC,
C                                        IWORK)

C   Arguments     N       - # of points in to be found including X.
C                           (INPUT)
C                 NP      - # of points to be searched.
C                           NP >= N  (INPUT)
C                 POINTS  - Vector of dimension NP containing the loca-
C                           tions of the points to be searched.
C                           POINTS(j) <= POINTS(j+1) for j = 1 to NP-1.
C                           (INPUT)
C                 X       - location of the point for which the nearest
C                           neighbors are to be found. (INPUT)
C                 IAVAIL  - Vector of dimension NP containing the number
C                           of available values at each point in POINTS
C                           (INPUT)
C                 LOC     - Vector of dimension N containing the distance
C                           of the N chosen points from X.  (OUTPUT)
C                 IWORK   - Work vector of dimension NP.

C   Routines called     - VSRTAD

C   Reference           - None

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE LOCATE(
     &  N     , NP     , POINTS, X      , AVAIL , LOC    , REMAIN)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

      DOUBLE PRECISION
     &  POINTS, X      , LOC

      INTEGER
```

```
     &  AVAIL , N      , NP     , REMAIN

      DOUBLE PRECISION
     &  CLOSE , DISTAN, FAR    , XP

      INTEGER
     &  DEBUGG, J      , K      , L      , NPP1   , OUT    , P

      LOGICAL
     &  TOP

      DIMENSION
     &  AVAIL (1)
     &, LOC   (1)
     &, POINTS(1)
     &, REMAIN(1)

      DATA
     &  DEBUGG / 0/
     &, OUT    / 6/

C-----------------------------------------------------------------------
C-----Initial housekeeping

      FAR   = DABS(POINTS(NP) - POINTS(1)) * 2
      NPP1  = NP + 1
      TOP   = .FALSE.

      DO 100 J = 1, NP
         REMAIN(J) = AVAIL(J)
  100 CONTINUE

      IF(DEBUGG .LT. 10) GO TO 1
         WRITE(OUT, /) ' Enter LOCATE'
         WRITE(OUT, /) ' N', N, 'NP', NP, 'X', ROUND(X, 4)
         WRITE(OUT, /) ' POINTS', (ROUND(POINTS(J), 4), J = 1, NP)
         WRITE(OUT, /) ' AVAIL ', (AVAIL(J), J = 1, NP)
    1 CONTINUE

C-----Find N points nearest X on POINTS(including X)

      DO 200 L = 1, N
         CLOSE = FAR

         DO 210 K = 1, NP
            J = K
            IF(TOP) J = NPP1 - K
            DISTAN = DABS(POINTS(J) - X)
            IF(DISTAN .GE. CLOSE  .OR.  REMAIN(J) .LE. 0) GO TO 210
               CLOSE = DISTAN
               P     = J
               XP    = POINTS(J)
  210    CONTINUE
```

```
          LOC(L) = XP
          REMAIN(P) = REMAIN(P) - 1
          TOP = XP .LT. X
  200 CONTINUE

C-----Sort LOC

      CALL VSRTAD(LOC, N)

      IF(DEBUGG .LT. 10) GO TO 2
         WRITE(OUT, /) ' Exit LOCATE'
         WRITE(OUT, /) ' N', N, 'NP', NP, 'X', ROUND(X, 4)
         WRITE(OUT, /) ' POINTS', (ROUND(POINTS(J), 4), J = 1, NP)
         WRITE(OUT, /) ' AVAIL ', (AVAIL(J), J = 1, NP)
         WRITE(OUT, /) ' LOC   ', (ROUND(LOC(J), 4), J = 1, N)
         WRITE(OUT, /) ' REMAIN', (REMAIN(J), J = 1, NP)
    2 CONTINUE

      RETURN
      END
```

```
#FILE (MARK)CHIP/BLOCK/DATA ON STUDENTS
      BLOCK DATA

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE "CHIP/COMMON"

      DATA
     &  ERRTOL  /  1.D-6   /   % [=] -  1.D-6
     &, H       / -1.D+0   /   % [=] m (manual geometric if H > 0,      -1
C                                        auto geometric if H = 0,
C                                        auto even      if H < 0)
     &, LENX    / 33.D-3   /   % .0165
     &, LENY    / 33.D-3   /   % .0177
     &, LENZ    / 12.D-3   /   % .0043
     &, LIQ2WD  /  4.D+0   /   % [=] -    4
     &, TIEND   /  5.25D+0/    % [=] hr  5.25
     &, TIFRST  /  0.375D+0/   % .375
     &, TIMPRG  /  0.D+0    /  % .50
     &, TINTER  /  0.125D+0/   % .125
     &, TI2TP   /  1.D+0    /  % 2.
     &, TPCOOK  /443.D+0   /   % [=] deg K 446
     &, TPMPRG  /300.D+0   /   % 353.
     &, EA      / 17.05D+0/    % [=] %ODW as Na2O % 18
     &, SULFID  / 30.00D+0/    % [=] sulfidity, Na2O basis % 25
     &, AQCHAR  /  0.D+0     / % [=] %ODW as AQ   % .1
     &, HTCOEF  / 17.00D+0 /   % [=] 1 / hr   17

      DATA
     &  D       /1/            % disk file # [=] -
     &, DEBUGG  /0/
     &, ORDERH  /4/
     &, NGPTX   /5/
     &, NGPTY   /5/
     &, NGPTZ   /5/
     &, NPRINT  /40/
     &, NSIG    /4/            % # sig digits in output
     &, P       /6/            % printer file #

      DATA
     &  FLOOPP  /T/            % T -> generate TPRINT from scratch
C                              % F -> initialize TPRINT in BLOCK DATA
     &, FMPREG  /T/            % T -> chip impregnated with liquor
C                              % F -> chip impregnated with water

      DATA
     &  BIOT    /  10.D+0        /  % [=] -
     &, DWARF   /   1.392D-14790/
     &, ERREPS  /   2.647D-23    /
     &, GIANT   /   4.414D+14801/
     &, RHOC    / 358.D+0        /  % [=] g / 1                 417, 430
     &, RHOL    /1000.D+0        /
     &, RHOW    /1530.D+0        /
```

```
&, VO      / 388.D+0          % CN [=] g / kg wood          401, 402
&,           388.D+0          % CX
&,           193.D+0          % GN                          177, 167
&,           193.D+0          % GX
&,            98.D+0          % XN                           77,  87
&,            98.D+0          % XX
&,            40.D+0          % EX                                33
&,            13.D+0          % AG
&,           273.D+0          % LN                          284, 301
&,           273.D+0          % LR
&,           273.D+0          % LD
&,             0.00D+0        % OH          % if 0, use EA,
&,             0.00D+0        % SH          %            S,
&,             0.0000D+0      % AQ          %        & AQC instead
&,          1000.D+0          % DS
&,             0.D+0          % TP
&,           170.0D+0         % VP                              57.7
&,          1000.D+0       /  % Y
&, VUSLOW /1000.D-3           % CN [=] kg / kg wood
&,             0.D-3          % CX
&,           752.D-3          % GN
&,             0.D-3          % GX
&,           868.D-3          % XN
&,             0.D-3          % XX
&,             0.D-3          % EX
&,             0.D-3          % AG
&,             0.D-3          % LN
&,             0.D-3       /  % LR

 DATA
&  PAG     / 8/               % [=] -
&, PAQ     /14/
&, PCN     / 1/
&, PCX     / 2/
&, PDHI    /16/
&, PDLO    /11/
&, PDS     /15/
&, PEX     / 7/
&, PGF     /17/
&, PGN     / 3/
&, PGX     / 4/
&, PHF     /18/
&, PLD     /11/
&, PLN     / 9/
&, PLR     /10/
&, PNHI    /17/
&, PNLO    /17/
&, POH     /12/
&, POHI    /18/
&, POLO    /18/
&, PSH     /13/
&, PTP     /16/
&, PVP     /17/
&, PXN     / 5/
```

```
      &, PXX      / 6/
      &, PY       /18/
      &, PYHI     /10/
      &, PYLO     / 1/

       END
C     &, TPRINT /     .01667          % printout times (if FLOOPP is false)
C     &,              .03333          % [=] hr
C     &,              .3167
C     &,              .979
C     &,            26.8
C     &,            48.73
C     &,              .08333
C     &,              .1667
C     &,              .25
C     &,              .5
C     &,             1.
C     &,             2.
C     &,             4.
C     &,             8.
C     &,            16.
C     &,            32.
C     &,              .9
C     &,            26.
C     &,            27.
C     &,            48.
C     &,            49.
C     &,              .7071
C     &,             1.414
C     &,             2.828
C     &,             5.657
C     &,            11.31
C     &,            22.63
C     &,            45.25
C     &,              .3536
C     &,            50.       /
```

```
#FILE (MARK)CHIP/BLOCK/DATA/GENERIC ON STUDENTS
      BLOCK DATA

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE "CHIP/COMMON"

      DATA
     & ERRTOL  /  1.D-6  /    % [=] -  1.D-6
     &, H      / -1.D+0  /    % [=] m (manual geometric if H > 0,     -1
C                                      auto geometric if H = 0,
C                                      auto even      if H < 0)
     &, LENX   / 33.D-3  /    % .0165
     &, LENY   / 33.D-3  /    % .0177
     &, LENZ   / 12.D-3  /    % .0043
     &, LIQ2WD /  4.D+0  /    % [=] -    4
     &, TIEND  /  5.25D+0/    % [=] hr  5.25
     &, TIFRST /  0.375D+0/   % .375
     &, TIMPRG /  0.D+0   /   % .50
     &, TINTER /  0.125D+0/   % .125
     &, TI2TP  /  1.D+0   /   % 2.
     &, TPCOOK /443.D+0  /    % [=] deg K 446
     &, TPMPRG /300.D+0  /    % 353.
     &, EA     / 17.05D+0/    % [=] %ODW as Na2O % 18
     &, SULFID / 30.00D+0/    % [=] sulfidity, Na2O basis % 25
     &, AQCHAR /  0.D+0    /  % [=] %ODW as AQ   % .1
     &, HTCOEF / 17.00D+0 /   % [=] 1 / hr    17

      DATA
     & D       /1/            % disk file # [=] -
     &, DEBUGG /0/
     &, ORDERH /4/
     &, NGPTX  /NX/
     &, NGPTY  /NY/
     &, NGPTZ  /NZ/
     &, NPRINT /NP/
     &, NSIG   /4/            % # sig digits in output
     &, P      /6/            % printer file #

      DATA
     & FLOOPP  /T/            % T -> generate TPRINT from scratch
C                             % F -> initialize TPRINT in BLOCK DATA
     &, FMPREG /T/            % T -> chip impregnated with liquor
C                             % F -> chip impregnated with water

      DATA
     & BIOT   /  10.D+0       /  % [=] -
     &, DWARF /   1.392D-14790/
     &, ERREPS /  2.647D-23   /
     &, GIANT /   4.414D+14801/
     &, RHOC  / 358.D+0       /  % [=] g / 1                    417, 430
     &, RHOL  /1000.D+0       /
     &, RHOW  /1530.D+0       /
```

```
     &, VO      / 388.D+0            % CN [=] g / kg wood              401, 402
     &,           388.D+0            % CX
     &,           193.D+0            % GN
     &,           193.D+0            % GX                              177, 167
     &,            98.D+0            % XN
     &,            98.D+0            % XX                               77,  87
     &,            40.D+0            % EX
     &,            13.D+0            % AG                                    33
     &,           273.D+0            % LN
     &,           273.D+0            % LR                              284, 301
     &,           273.D+0            % LD
     &,             0.00D+0          % OH         % if 0, use EA,
     &,             0.00D+0          % SH         %             S,
     &,             0.0000D+0        % AQ         %           & AQC instead
     &,          1000.D+0            % DS
     &,             0.D+0            % TP
     &,           170.0D+0           % VP
     &,          1000.D+0        /   % Y                                  57.7
     &, VUSLOW /1000.D-3             % CN [=] kg / kg wood
     &,             0.D-3            % CX
     &,           752.D-3            % GN
     &,             0.D-3            % GX
     &,           868.D-3            % XN
     &,             0.D-3            % XX
     &,             0.D-3            % EX
     &,             0.D-3            % AG
     &,             0.D-3            % LN
     &,             0.D-3        /   % LR

      DATA
     &  PAG       / 8/               % [=] -
     &, PAQ       /14/
     &, PCN       / 1/
     &, PCX       / 2/
     &, PDHI      /16/
     &, PDLO      /11/
     &, PDS       /15/
     &, PEX       / 7/
     &, PGF       /17/
     &, PGN       / 3/
     &, PGX       / 4/
     &, PHF       /18/
     &, PLD       /11/
     &, PLN       / 9/
     &, PLR       /10/
     &, PNHI      /17/
     &, PNLO      /17/
     &, POH       /12/
     &, POHI      /18/
     &, POLO      /18/
     &, PSH       /13/
     &, PTP       /16/
     &, PVP       /17/
     &, PXN       / 5/
```

```
     &, PXX      / 6/
     &, PY       /18/
     &, PYHI     /10/
     &, PYLO     / 1/

      END
C    &, TPRINT /    .01667         % printout times (if FLOOPP is false)
C    &,             .03333         % [=] hr
C    &,             .3167
C    &,             .979
C    &,           26.8
C    &,           48.73
C    &,             .08333
C    &,             .1667
C    &,             .25
C    &,             .5
C    &,            1.
C    &,            2.
C    &,            4.
C    &,            8.
C    &,           16.
C    &,           32.
C    &,             .9
C    &,           26.
C    &,           27.
C    &,           48.
C    &,           49.
C    &,             .7071
C    &,            1.414
C    &,            2.828
C    &,            5.657
C    &,           11.31
C    &,           22.63
C    &,           45.25
C    &,             .3536
C    &,           50.        /
```

```
#FILE (MARK)CHIP/COMMON ON STUDENTS
$   RESET LIST
      DOUBLE PRECISION                                                  % 3D
     &  AX      % effective diffusivity [m**2 / hr]
     &, AY
     &, AZ
     &, G       % total time derivative [kg / m**3 / hr]
     &, R       % reaction time derivative [kg / m**3 / hr]
     &, U       % concentration [kg / m**3]
C
      DOUBLE PRECISION                                                  % 2D
     &  CTXA    % finite difference formula coefficients [-]
     &, CTX1
     &, CTX2
     &, CTYA
     &, CTY1
     &, CTY2
     &, CTZA
     &, CTZ1
     &, CTZ2
     &, CXA
     &, CX1
     &, CX2
     &, CYA
     &, CY1
     &, CY2
     &, CZA
     &, CZ1
     &, CZ2
C
      DOUBLE PRECISION                                                  % 1D
     &  LOCX    % location of grid points relative to chip surface [m]
     &, LOCY
     &, LOCZ
     &, TPRINT  % printout times desired [hr]
     &, VA      % bulk liquor diffusivity [m**2 / hr]
     &, VG      % bulk total time derivative [kg / m**3 / hr]
     &, VU      % bulk concentrations [kg / m**3]
     &, VUSLOW  % fraction remaining after initial phase [-]
     &, V0      % initial concentrations [kg / m**3]
C
      DOUBLE PRECISION                                                  % 0D
     &  BIOT    % k L / D (mass transfer:diffusion) [-]
     &, DP      % effective chip diameter [m]
     &, DWARF   % square root of smallest # > 0 available on machine [-]
     &, ERREPS  % smallest # > 0 such that  1-erreps < 1 < 1+erreps [-]
     &, ERRTOL  % error tolerance [-]
     &, GIANT   % square root of largest # > 0 available on machine [-]
     &, H       % distance between surface and first interior point [m]
C               % if H is input as zero, H is chosen by program
     &, HTCOEF  % [=] 1 / hr
     &, KNORM   % 2Bi * (Achip / Vchip) (Vchip / Vtrapped)
     &, LENX    % chip size [m]
```

```
      &, LENY
      &, LENZ
      &, LIQ2WD  % liquor:wood [-]
      &, TIEND   % quitting time [hr]
      &, TIFRST  % first printout time [hr]
      &, TIME    % current time [hr]
      &, TIMPRG  % impregnation time [hr]
      &, TINTER  % printout interval [hr]
      &, TI2TP   % time to temperature [hr]
      &, TPMPRG  % impregnation temperature [deg. K]
      &, TPCOOK  % cook temperature [deg. K]
      &, TPCRIT  % critical point of water [deg. K]
      &, TPMELT  % melting (freezing) point of water [deg. K]
      &, YID100  % YINITL / 100 [kg / m**3]
      &, YINITL  % initial yield [kg / m**3]
      &, EA      % effective alkali %ODW as Na2O
      &, SULFID  % sulfidity % as Na2O
      &, AQCHAR  % AQ charge, %ODW as AQ
      &, RHOC    % chip density [=] kg/m**3
      &, RHOL    % liquor density
      &, RHOW    % wood substance density
C
       INTEGER                                                    % 1D
      &  HITXA   % upper point of finite difference formulas
      &, HITX1
      &, HITX2
      &, HITYA
      &, HITY1
      &, HITY2
      &, HITZA
      &, HITZ1
      &, HITZ2
      &, HIXA
      &, HIX1
      &, HIX2
      &, HIYA
      &, HIY1
      &, HIY2
      &, HIZA
      &, HIZ1
      &, HIZ2
      &, LOTXA   % lower point of finite difference formulas
      &, LOTX1
      &, LOTX2
      &, LOTYA
      &, LOTY1
      &, LOTY2
      &, LOTZA
      &, LOTZ1
      &, LOTZ2
      &, LOXA
      &, LOX1
      &, LOX2
      &, LOYA
```

```
      &, LOY1
      &, LOY2
      &, LOZA
      &, LOZ1
      &, LOZ2
C
       INTEGER                                                    % OD
      & D        % disk file number (disk file used for output only)
      &, DEBUGG  % diagnostic output increases with DEBUGG
      &, ORDERH  % truncation error of FD formulas [O(h ** ORDERH)]
      &, NFEVAL  % number of function evaluations
      &, NGPTX   % number of grid points between chip surface and chip
      &, NGPTY   %   center (including surface and center points)
      &, NGPTZ
      &, NPRINT  % # printouts desired
      &, NSIG    % # significant digits for rounding purposes
      &, P       % printer file number
      &, PAG     % acetyl group pointer
      &, PAQ     % anthraquinone pointer
      &, PCN     % native cellulose pointer
      &, PCX     % oxidized cellulose pointer
      &, PDHI    % pointers to range of diffusing components in U, etc.
      &, PDLO
      &, PDS     % dissolved solids pointer
      &, PEX     % extractives pointer
      &, PGF     % G factor pointer
      &, PGN     % native (galacto)glucomannan pointer
      &, PGX     % oxidized glucomannan pointer
      &, PHF     % H factor pointer
      &, PLD     % dissolved lignin pointer
      &, PLN     % native lignin pointer
      &, PLR     % residual lignin pointer
      &, PNHI    % pointers to range of components which react but don't
      &, PNLO    %    contribute to yield
      &, POH     % NaOH pointer
      &, POHI    % pointers to range of 'other' components (they aren't
      &, POLO    %    integrated by GEAR)
      &, PSH     % NaSH pointer
      &, PTP     % temperature pointer
      &, PVP     % pulp viscosity pointer
      &, PXN     % native (arabino)xylan pointer
      &, PXX     % oxidized xylan pointer
      &, PY      % yield pointer
      &, PYHI    % pointers to range of yield contributing components
      &, PYLO
C
       LOGICAL                                                    % OD
      & FLOOPP   % true  -> load TPRINT using TIFRST, TINTER, & TIEND
C                % false -> use TPRINT as initialized in BLOCK DATA
      &, FMPREG  % true  -> chip impregnated with liquor
C                % false -> chip impregnated with water
C
C-----note: 1 <= 10 < 11 <= 16 < PNLO <= PNHI < POLO <= 18
C
```

```
      DIMENSION
     &  AX     (11:16, 5 , 5 , 5)
     &, AY     (11:16, 5 , 5 , 5)
     &, AZ     (11:16, 5 , 5 , 5)
     &, G      (1:18, 5 , 5 , 5)
     &, R      (1:18, 5 , 5 , 5)
     &, U      (1:18, 5 , 5 , 5)
C
     &, CTXA  (0:5, 5)
     &, CTX1  (0:5, 5)
     &, CTX2  (0:5, 5)
     &, CTYA  (0:5, 5)
     &, CTY1  (0:5, 5)
     &, CTY2  (0:5, 5)
     &, CTZA  (0:5, 5)
     &, CTZ1  (0:5, 5)
     &, CTZ2  (0:5, 5)
     &, CXA   (0:5, 5)
     &, CX1   (0:5, 5)
     &, CX2   (0:5, 5)
     &, CYA   (0:5, 5)
     &, CY1   (0:5, 5)
     &, CY2   (0:5, 5)
     &, CZA   (0:5, 5)
     &, CZ1   (0:5, 5)
     &, CZ2   (0:5, 5)
C
     &, LOCX  (5)
     &, LOCY  (5)
     &, LOCZ  (5)
     &, TPRINT(40)
     &, VA    (11:16)
     &, VG    (11:18)
     &, VU    (11:18)
     &, VUSLOW(1:10)
     &, VO    (1:18)
C
     &, HITXA (5)
     &, HITX1 (5)
     &, HITX2 (5)
     &, HITYA (5)
     &, HITY1 (5)
     &, HITY2 (5)
     &, HITZA (5)
     &, HITZ1 (5)
     &, HITZ2 (5)
     &, HIXA  (5)
     &, HIX1  (5)
     &, HIX2  (5)
     &, HIYA  (5)
     &, HIY1  (5)
     &, HIY2  (5)
     &, HIZA  (5)
     &, HIZ1  (5)
```

```
     &, HIZ2  (5)
     &, LOTXA (5)
     &, LOTX1 (5)
     &, LOTX2 (5)
     &, LOTYA (5)
     &, LOTY1 (5)
     &, LOTY2 (5)
     &, LOTZA (5)
     &, LOTZ1 (5)
     &, LOTZ2 (5)
     &, LOXA  (5)
     &, LOX1  (5)
     &, LOX2  (5)
     &, LOYA  (5)
     &, LOY1  (5)
     &, LOY2  (5)
     &, LOZA  (5)
     &, LOZ1  (5)
     &, LOZ2  (5)
C
      COMMON
     &  /COMCAX/ AX
     &  /COMCAY/ AY
     &  /COMCAZ/ AZ
     &  /COMG   / G
     &  /COMR   / R
     &  /COMU   / U
C
      COMMON
     &  /CMCTXA/ CTXA
     &  /CMCTX1/ CTX1
     &  /CMCTX2/ CTX2
     &  /CMCTYA/ CTYA
     &  /CMCTY1/ CTY1
     &  /CMCTY2/ CTY2
     &  /CMCTZA/ CTZA
     &  /CMCTZ1/ CTZ1
     &  /CMCTZ2/ CTZ2
     &  /COMCXA/ CXA
     &  /COMCX1/ CX1
     &  /COMCX2/ CX2
     &  /COMCYA/ CYA
     &  /COMCY1/ CY1
     &  /COMCY2/ CY2
     &  /COMCZA/ CZA
     &  /COMCZ1/ CZ1
     &  /COMCZ2/ CZ2
C
      COMMON
     &  /CMLOCX/ LOCX
     &  /CMLOCY/ LOCY
     &  /CMLOCZ/ LOCZ
     &  /CMTPRI/ TPRINT
     &  /COMVA / VA
```

```
     &  /COMVG / VG
     &  /COMVU / VU
     &  /CVUSLO/ VUSLOW
     &  /COMVO / VO
C
      COMMON
     &  /CHITXA/ HITXA
     &  /CHITX1/ HITX1
     &  /CHITX2/ HITX2
     &  /CHITYA/ HITYA
     &  /CHITY1/ HITY1
     &  /CHITY2/ HITY2
     &  /CHITZA/ HITZA
     &  /CHITZ1/ HITZ1
     &  /CHITZ2/ HITZ2
     &  /CMHIXA/ HIXA
     &  /CMHIX1/ HIX1
     &  /CMHIX2/ HIX2
     &  /CMHIYA/ HIYA
     &  /CMHIY1/ HIY1
     &  /CMHIY2/ HIY2
     &  /CMHIZA/ HIZA
     &  /CMHIZ1/ HIZ1
     &  /CMHIZ2/ HIZ2
     &  /CLOTXA/ LOTXA
     &  /CLOTX1/ LOTX1
     &  /CLOTX2/ LOTX2
     &  /CLOTYA/ LOTYA
     &  /CLOTY1/ LOTY1
     &  /CLOTY2/ LOTY2
     &  /CLOTZA/ LOTZA
     &  /CLOTZ1/ LOTZ1
     &  /CLOTZ2/ LOTZ2
     &  /CMLOXA/ LOXA
     &  /CMLOX1/ LOX1
     &  /CMLOX2/ LOX2
     &  /CMLOYA/ LOYA
     &  /CMLOY1/ LOY1
     &  /CMLOY2/ LOY2
     &  /CMLOZA/ LOZA
     &  /CMLOZ1/ LOZ1
     &  /CMLOZ2/ LOZ2

      COMMON /COMDP /
$       INCLUDE "CHIP/COMMON" 4200-7200

      COMMON /COMINT/
$       INCLUDE "CHIP/COMMON" 11300-15000

      COMMON /COMLOG/
$       INCLUDE "CHIP/COMMON" 15300-15600
$     POP LIST
```

```
#FILE (MARK)CHIP/COMMON/GENERIC ON STUDENTS
$   RESET LIST
      DOUBLE PRECISION                                                      % 3D
     &  AX      % effective diffusivity [m**2 / hr]
     &, AY
     &, AZ
     &, G       % total time derivative [kg / m**3 / hr]
     &, R       % reaction time derivative [kg / m**3 / hr]
     &, U       % concentration [kg / m**3]
C
      DOUBLE PRECISION                                                      % 2D
     &  CTXA    % finite difference formula coefficients [-]
     &, CTX1
     &, CTX2
     &, CTYA
     &, CTY1
     &, CTY2
     &, CTZA
     &, CTZ1
     &, CTZ2
     &, CXA
     &, CX1
     &, CX2
     &, CYA
     &, CY1
     &, CY2
     &, CZA
     &, CZ1
     &, CZ2
C
      DOUBLE PRECISION                                                      % 1D
     &  LOCX    % location of grid points relative to chip surface [m]
     &, LOCY
     &, LOCZ
     &, TPRINT  % printout times desired [hr]
     &, VA      % bulk liquor diffusivity [m**2 / hr]
     &, VG      % bulk total time derivative [kg / m**3 / hr]
     &, VU      % bulk concentrations [kg / m**3]
     &, VUSLOW  % fraction remaining after initial phase [-]
     &, V0      % initial concentrations [kg / m**3]
C
      DOUBLE PRECISION                                                      % 0D
     &  BIOT    % k L / D (mass transfer:diffusion) [-]
     &, DP      % effective chip diameter [m]
     &, DWARF   % square root of smallest # > 0 available on machine [-]
     &, ERREPS  % smallest # > 0 such that  1-erreps < 1 < 1+erreps [-]
     &, ERRTOL  % error tolerance [-]
     &, GIANT   % square root of largest # > 0 available on machine [-]
     &, H       % distance between surface and first interior point [m]
C               % if H is input as zero, H is chosen by program
     &, HTCOEF  % [=] 1 / hr
     &, KNORM   % 2Bi * (Achip / Vchip) (Vchip / Vtrapped)
     &, LENX    % chip size [m]
```

```
     &, LENY
     &, LENZ
     &, LIQ2WD  % liquor:wood [-]
     &, TIEND   % quitting time [hr]
     &, TIFRST  % first printout time [hr]
     &, TIME    % current time [hr]
     &, TIMPRG  % impregnation time [hr]
     &, TINTER  % printout interval [hr]
     &, TI2TP   % time to temperature [hr]
     &, TPMPRG  % impregnation temperature [deg. K]
     &, TPCOOK  % cook temperature [deg. K]
     &, TPCRIT  % critical point of water [deg. K]
     &, TPMELT  % melting (freezing) point of water [deg. K]
     &, YID100  % YINITL / 100 [kg / m**3]
     &, YINITL  % initial yield [kg / m**3]
     &, EA      % effective alkali %ODW as Na2O
     &, SULFID  % sulfidity % as Na2O
     &, AQCHAR  % AQ charge, %ODW as AQ
     &, RHOC    % chip density [=] kg/m**3
     &, RHOL    % liquor density
     &, RHOW    % wood substance density
C
      INTEGER                                                   % 1D
     & HITXA    % upper point of finite difference formulas
     &, HITX1
     &, HITX2
     &, HITYA
     &, HITY1
     &, HITY2
     &, HITZA
     &, HITZ1
     &, HITZ2
     &, HIXA
     &, HIX1
     &, HIX2
     &, HIYA
     &, HIY1
     &, HIY2
     &, HIZA
     &, HIZ1
     &, HIZ2
     &, LOTXA   % lower point of finite difference formulas
     &, LOTX1
     &, LOTX2
     &, LOTYA
     &, LOTY1
     &, LOTY2
     &, LOTZA
     &, LOTZ1
     &, LOTZ2
     &, LOXA
     &, LOX1
     &, LOX2
     &, LOYA
```

```
     &, LOY1
     &, LOY2
     &, LOZA
     &, LOZ1
     &, LOZ2
C
      INTEGER                                                       % OD
     &  D       % disk file number (disk file used for output only)
     &, DEBUGG  % diagnostic output increases with DEBUGG
     &, ORDERH  % truncation error of FD formulas [O(h ** ORDERH)]
     &, NFEVAL  % number of function evaluations
     &, NGPTX   % number of grid points between chip surface and chip
     &, NGPTY   %   center (including surface and center points)
     &, NGPTZ
     &, NPRINT  % # printouts desired
     &, NSIG    % # significant digits for rounding purposes
     &, P       % printer file number
     &, PAG     % acetyl group pointer
     &, PAQ     % anthraquinone pointer
     &, PCN     % native cellulose pointer
     &, PCX     % oxidized cellulose pointer
     &, PDHI    % pointers to range of diffusing components in U, etc.
     &, PDLO
     &, PDS     % dissolved solids pointer
     &, PEX     % extractives pointer
     &, PGF     % G factor pointer
     &, PGN     % native (galacto)glucomannan pointer
     &, PGX     % oxidized glucomannan pointer
     &, PHF     % H factor pointer
     &, PLD     % dissolved lignin pointer
     &, PLN     % native lignin pointer
     &, PLR     % residual lignin pointer
     &, PNHI    % pointers to range of components which react but don't
     &, PNLO    %    contribute to yield
     &, POH     % NaOH pointer
     &, POHI    % pointers to range of 'other' components (they aren't
     &, POLO    %    integrated by GEAR)
     &, PSH     % NaSH pointer
     &, PTP     % temperature pointer
     &, PVP     % pulp viscosity pointer
     &, PXN     % native (arabino)xylan pointer
     &, PXX     % oxidized xylan pointer
     &, PY      % yield pointer
     &, PYHI    % pointers to range of yield contributing components
     &, PYLO
C
      LOGICAL                                                       % OD
     &  FLOOPP  % true  -> load TPRINT using TIFRST, TINTER, & TIEND
C               % false -> use TPRINT as initialized in BLOCK DATA
     &, FMPREG  % true  -> chip impregnated with liquor
C               % false -> chip impregnated with water
C
C-----note: PYLO <= PYHI < PDLO <= PDHI < PNLO <= PNHI < POLO <= POHI
C
```

```
      DIMENSION
     &  AX     (PDLO:PDHI, NGPTX , NGPTY , NGPTZ)
     &, AY     (PDLO:PDHI, NGPTX , NGPTY , NGPTZ)
     &, AZ     (PDLO:PDHI, NGPTX , NGPTY , NGPTZ)
     &, G      (PYLO:POHI, NGPTX , NGPTY , NGPTZ)
     &, R      (PYLO:POHI, NGPTX , NGPTY , NGPTZ)
     &, U      (PYLO:POHI, NGPTX , NGPTY , NGPTZ)
C
     &, CTXA  (0:NGPTX, NGPTX)
     &, CTX1  (0:NGPTX, NGPTX)
     &, CTX2  (0:NGPTX, NGPTX)
     &, CTYA  (0:NGPTY, NGPTY)
     &, CTY1  (0:NGPTY, NGPTY)
     &, CTY2  (0:NGPTY, NGPTY)
     &, CTZA  (0:NGPTZ, NGPTZ)
     &, CTZ1  (0:NGPTZ, NGPTZ)
     &, CTZ2  (0:NGPTZ, NGPTZ)
     &, CXA   (0:NGPTX, NGPTX)
     &, CX1   (0:NGPTX, NGPTX)
     &, CX2   (0:NGPTX, NGPTX)
     &, CYA   (0:NGPTY, NGPTY)
     &, CY1   (0:NGPTY, NGPTY)
     &, CY2   (0:NGPTY, NGPTY)
     &, CZA   (0:NGPTZ, NGPTZ)
     &, CZ1   (0:NGPTZ, NGPTZ)
     &, CZ2   (0:NGPTZ, NGPTZ)
C
     &, LOCX  (NGPTX)
     &, LOCY  (NGPTY)
     &, LOCZ  (NGPTZ)
     &, TPRINT(NPRINT)
     &, VA    (PDLO:PDHI)
     &, VG    (PDLO:POHI)
     &, VU    (PDLO:POHI)
     &, VUSLOW(PYLO:PYHI)
     &, VO    (PYLO:POHI)
C
     &, HITXA (NGPTX)
     &, HITX1 (NGPTX)
     &, HITX2 (NGPTX)
     &, HITYA (NGPTY)
     &, HITY1 (NGPTY)
     &, HITY2 (NGPTY)
     &, HITZA (NGPTZ)
     &, HITZ1 (NGPTZ)
     &, HITZ2 (NGPTZ)
     &, HIXA  (NGPTX)
     &, HIX1  (NGPTX)
     &, HIX2  (NGPTX)
     &, HIYA  (NGPTY)
     &, HIY1  (NGPTY)
     &, HIY2  (NGPTY)
     &, HIZA  (NGPTZ)
     &, HIZ1  (NGPTZ)
```

```
     &, HIZ2  (NGPTZ)
     &, LOTXA (NGPTX)
     &, LOTX1 (NGPTX)
     &, LOTX2 (NGPTX)
     &, LOTYA (NGPTY)
     &, LOTY1 (NGPTY)
     &, LOTY2 (NGPTY)
     &, LOTZA (NGPTZ)
     &, LOTZ1 (NGPTZ)
     &, LOTZ2 (NGPTZ)
     &, LOXA  (NGPTX)
     &, LOX1  (NGPTX)
     &, LOX2  (NGPTX)
     &, LOYA  (NGPTY)
     &, LOY1  (NGPTY)
     &, LOY2  (NGPTY)
     &, LOZA  (NGPTZ)
     &, LOZ1  (NGPTZ)
     &, LOZ2  (NGPTZ)
C
      COMMON
     &  /COMCAX/ AX
     &  /COMCAY/ AY
     &  /COMCAZ/ AZ
     &  /COMG   / G
     &  /COMR   / R
     &  /COMU   / U
C
      COMMON
     &  /CMCTXA/ CTXA
     &  /CMCTX1/ CTX1
     &  /CMCTX2/ CTX2
     &  /CMCTYA/ CTYA
     &  /CMCTY1/ CTY1
     &  /CMCTY2/ CTY2
     &  /CMCTZA/ CTZA
     &  /CMCTZ1/ CTZ1
     &  /CMCTZ2/ CTZ2
     &  /COMCXA/ CXA
     &  /COMCX1/ CX1
     &  /COMCX2/ CX2
     &  /COMCYA/ CYA
     &  /COMCY1/ CY1
     &  /COMCY2/ CY2
     &  /COMCZA/ CZA
     &  /COMCZ1/ CZ1
     &  /COMCZ2/ CZ2
C
      COMMON
     &  /CMLOCX/ LOCX
     &  /CMLOCY/ LOCY
     &  /CMLOCZ/ LOCZ
     &  /CMTPRI/ TPRINT
     &  /COMVA / VA
```

```
     & /COMVG / VG
     & /COMVU / VU
     & /CVUSLO/ VUSLOW
     & /COMVO / VO
C
      COMMON
     & /CHITXA/ HITXA
     & /CHITX1/ HITX1
     & /CHITX2/ HITX2
     & /CHITYA/ HITYA
     & /CHITY1/ HITY1
     & /CHITY2/ HITY2
     & /CHITZA/ HITZA
     & /CHITZ1/ HITZ1
     & /CHITZ2/ HITZ2
     & /CMHIXA/ HIXA
     & /CMHIX1/ HIX1
     & /CMHIX2/ HIX2
     & /CMHIYA/ HIYA
     & /CMHIY1/ HIY1
     & /CMHIY2/ HIY2
     & /CMHIZA/ HIZA
     & /CMHIZ1/ HIZ1
     & /CMHIZ2/ HIZ2
     & /CLOTXA/ LOTXA
     & /CLOTX1/ LOTX1
     & /CLOTX2/ LOTX2
     & /CLOTYA/ LOTYA
     & /CLOTY1/ LOTY1
     & /CLOTY2/ LOTY2
     & /CLOTZA/ LOTZA
     & /CLOTZ1/ LOTZ1
     & /CLOTZ2/ LOTZ2
     & /CMLOXA/ LOXA
     & /CMLOX1/ LOX1
     & /CMLOX2/ LOX2
     & /CMLOYA/ LOYA
     & /CMLOY1/ LOY1
     & /CMLOY2/ LOY2
     & /CMLOZA/ LOZA
     & /CMLOZ1/ LOZ1
     & /CMLOZ2/ LOZ2

      COMMON /COMDP /
$        INCLUDE "CHIP/COMMON" 4200-7200

      COMMON /COMINT/
$        INCLUDE "CHIP/COMMON" 11300-15000

      COMMON /COMLOG/
$        INCLUDE "CHIP/COMMON" 15300-15600
$     POP LIST
```

```
#FILE (MARK)CHIP/DIGEST ON STUDENTS
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE DIGEST
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

C-----This subroutine calculates the bulk liquor rates of change
C     from chip rates of change
C-----Creation date: 23 Mar 82; Last update:  8 Oct 85

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE "CHIP/COMMON"

      DOUBLE PRECISION
     &  ALPHA , GFA    , GFB    , HFA    , HFB    , TARGET, TWO    , UBULK
     &, USURFX, USURFY, USURFZ, VCHIP  , VOTP   , VTDVB  , VTRAP  , ZERO
     &, AQA    , AQB

      INTEGER
     &  COMP   , X      , Y      , Z

      DATA
     &  AQA    /+20.4828D+0/
     &, AQB    /-10692.9D+0/
     &, GFA    /+57.71D+0/
     &, GFB    /-21527D+0/
     &, HFA    /+43.20D+0/
     &, HFB    /-16113D+0/
     &, TWO    /+2.D+0 /
     &, ZERO   /+0.D+0 /

C---------------------------------------------------------------------

      IF(DEBUGG .GE. 1) WRITE(P, /) '        enter DIGEST'

C-----calculate Achip/Vbulk as the product of Achip/Vchip and
C-----Vchip/Vbulk Vtrapped/Vchip (Vtrapped/Vchip denormalizes KNORM)

      VCHIP  = LENX * LENY * LENZ
      VTRAP  = (RHOL / RHOC) - (RHOL / RHOW)
      VTDVB  = VTRAP / (LIQ2WD - VTRAP)
      ALPHA  = (TWO / VCHIP) * VTDVB * KNORM

C-----Calculate dUbulk/dt = k(Usurface - Ubulk) Achip / Vbulk

      DO 100 COMP = PDLO, PDS
         UBULK  = VU(COMP)
         USURFX = AVE(LOCX   , LOCY   , LOCZ   , U      , UBULK  , CTX1
     &                , CTX2   , CTY1   , CTY2   , CTZ1   , CTZ2   , HITX1
     &                , HITX2  , HITY1  , HITY2  , HITZ1  , HITZ2  , LOTX1
     &                , LOTX2  , LOTY1  , LOTY2  , LOTZ1  , LOTZ2  , POHI
     &                , PYLO   , NGPTX  , NGPTY  , NGPTZ  , COMP
     &                , 1      , 0      , 0      )
```

```
      USURFY = AVE(LOCX   , LOCY   , LOCZ   , U      , UBULK  , CTX1
     &                    , CTX2   , CTY1   , CTY2   , CTZ1   , CTZ2   , HITX1
     &                    , HITX2  , HITY1  , HITY2  , HITZ1  , HITZ2  , LOTX1
     &                    , LOTX2  , LOTY1  , LOTY2  , LOTZ1  , LOTZ2  , POHI
     &                    , PYLO   , NGPTX  , NGPTY  , NGPTZ  , COMP
     &                    , 0      , 1      , 0      )

      USURFZ = AVE(LOCX   , LOCY   , LOCZ   , U      , UBULK  , CTX1
     &                    , CTX2   , CTY1   , CTY2   , CTZ1   , CTZ2   , HITX1
     &                    , HITX2  , HITY1  , HITY2  , HITZ1  , HITZ2  , LOTX1
     &                    , LOTX2  , LOTY1  , LOTY2  , LOTZ1  , LOTZ2  , POHI
     &                    , PYLO   , NGPTX  , NGPTY  , NGPTZ  , COMP
     &                    , 0      , 0      , 1      )

      VG(COMP) =
     &     ((USURFX - UBULK) * LENY * LENZ
     &    + (USURFY - UBULK) * LENX * LENZ
     &    + (USURFZ - UBULK) * LENX * LENY)
     &    * VA(COMP) * ALPHA
  100 CONTINUE

      IF(TIME .LE. TIMPRG) GO TO 1
      IF(TIME .GE. TI2TP)  GO TO 3
      GO TO 2

    1 CONTINUE
         TARGET = TPMPRG
      GO TO 4

    2 CONTINUE
         TARGET = ((TIME - TIMPRG) / (TI2TP - TIMPRG))
     &          * (TPCOOK - TPMPRG) + TPMPRG
      GO TO 4

    3 CONTINUE
         TARGET = TPCOOK
      GO TO 4

    4 CONTINUE
      VOTP  = VO(PTP)
      UBULK = VU(PTP) * VOTP
      ALPHA = DEXP(AQB / UBULK + AQA) * DSQRT(UBULK) * VU(PAQ)
      VG(PAQ) = VG(PAQ) - ALPHA
      VG(PTP) = (TARGET - UBULK) * HTCOEF / VOTP
      VG(PGF) = DEXP(GFB / UBULK + GFA)
      VG(PHF) = DEXP(HFB / UBULK + HFA)

      IF(DEBUGG .GE. 2) WRITE(P, /) '        exit  DIGEST'

      END
```

```
#FILE (MARK)CHIP/SOURCE ON STUDENTS
FILE  1(KIND = DISK)
FILE  6(KIND = PRINTER)
$      SET $
$      SET ERRLIST
$    RESET FREE
$      LIMIT = 0
$      SET LINEINFO
$      SET LIST
$    RESET LONG
$      SET NOBINDINFO
$      SET OMITDEBUG
$      OPT = 1  RESET OWN  SET OWNARRAYS
$      SET OWN
$      SET OWNARRAYS
$    RESET XREF

$    RESET LIST
$      INCLUDE '*IMSL/UGETIO'
$      INCLUDE '*IMSL/USPKD'
$      INCLUDE '*IMSL/UERTST'

$      INCLUDE 'IMSL/DBLE/DCSQDU'
$      INCLUDE 'IMSL/DBLE/IQHSCU'
$      INCLUDE 'IMSL/DBLE/VSRTAD'

$      INCLUDE 'IMSL/DBLE/LUELMF'
$      INCLUDE 'IMSL/DBLE/LEQT1B'
$      INCLUDE 'IMSL/DBLE/LUDATF'
$      INCLUDE 'IMSL/DBLE/DGRCS'
$      POP LIST
$      INCLUDE 'IMSL/DBLE/DGRPS'
$      INCLUDE 'IMSL/DBLE/DGRIN'
$      INCLUDE 'IMSL/DBLE/DGRST'
$      INCLUDE 'IMSL/DBLE/DGEAR'

$      INCLUDE 'CHIP/BLOCK/DATA'
$      INCLUDE 'CHIP/UTIL/AVE'
$      INCLUDE 'CHIP/UTIL/DUMP'
$      INCLUDE 'CHIP/UTIL/ERRFD'
$      INCLUDE 'CHIP/UTIL/GAUSS'
$      INCLUDE 'CHIP/UTIL/GRID'
$      INCLUDE 'CHIP/UTIL/ROUND'
$      INCLUDE 'CHIP/UTIL/TIMER'
$      INCLUDE 'CHIP/UTIL/FINITE'
$      INCLUDE 'CHIP/UTIL/LOCATE'
$      INCLUDE 'CHIP/UTIL/FD'
$      INCLUDE 'CHIP/DIGEST'
$      INCLUDE 'CHIP/DUDT'

C-------------------------------------------------------------------

C   Chip routine name   - FINDIF
```

```
C     Creation date         -  7 Sep 84
C     Latest revision       - 16 Sep 85

C     Author                - Mark A. Burazin

C     Purpose               - Calculate finite difference formulas to
C                               approximate first order and second order
C                               partial differential equations.

C     Hardware              - Burroughs Double Precision

C     Usage                 - CALL FINDIF

C     Arguments             - None.

C     Common        CHIP    - All global variables used by chip model.

C     Routines called       - FD, FINITE, GAUSS, LOCATE

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE FINDIF
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'

      DOUBLE PRECISION
     &  COEF   , HD10  , LOC    , POINTS, TWODLX, TWODLY, TWODLZ
     &, ONE    , TENTH , TWO    , ZERO

      INTEGER
     &  AVAIL  , J     , K      , L      , M      , NP     , NX2    , NX2M1
     &, NX2M2  , NY2   , NY2M1  , NY2M2  , NZ2    , NZ2M1  , NZ2M2  , OHP1
     &, OHP2   , ORDER , WORK   , X      , Y      , Z
     &, NXP1   , NYP1  , NZP1

      DIMENSION
     &  AVAIL(29)
     &, COEF(15)
     &, LOC(15)
     &, ORDER(15)
     &, POINTS(29)
     &, WORK(15)

      DATA
     &  ONE    /+1.D+0/
     &, TENTH  /+1.D-1/
     &, TWO    /+2.D+0/
     &, ZERO   /+0.D+0/

 9000 FORMAT(' ')
 9001 FORMAT(// 1X, 72('*') / ' FATAL ERROR IN ROUTINE FINDIF' /
```

```
     &, ' OHP1 (= ', I2, ') is too big for number of grid points '
     &, 'available' / ' NX2 = ', I2, ', NY2 = ', I2, ', and NZ2 = ', I2
     &/ ' SIMULATION HALTED' / 1X, 72('*'))

C-----------------------------------------------------------------------
C-----initial housekeeping

      NXP1    = NGPTX + 1
      NX2     = NGPTX + NGPTX
      NX2M1   = NX2 - 1
      NX2M2   = NX2 - 2
      NYP1    = NGPTY + 1
      NY2     = NGPTY + NGPTY
      NY2M1   = NY2 - 1
      NY2M2   = NY2 - 2
      NZP1    = NGPTZ + 1
      NZ2     = NGPTZ + NGPTZ
      NZ2M1   = NZ2 - 1
      NZ2M2   = NZ2 - 2
      OHP1    = ORDERH + 1
      OHP2    = ORDERH + 2

C-----calculate FD formulas for dU/d(x,y,z) (CX1, CY1, CZ1),
C     d2U/d(x,y,z)2 (CX2, CY2, CZ2), and dA/d(x,y,z) (CAX1, CAY1,
C     CAZ1) for all x, y, and z.
C     Find and use ORDERH+N nearest points for FD formula.
C
C     'U' formulas may make use of the surface boundary condition
C     (dU/d(x, y, z) at surface). 'A' formulas may not.
C
C     Centerplane boundary conditions may be used to generate the FD
C     formulas but do not explicitly appear in the C vectors.  This is
C     because the first derivative at chip center = 0, so that term
C     drops out.  Since the formulas are valid for any value of first
C     derivative, this results in no loss of accuracy.
C
C     Note the distinction between real space and shadow space.
C     The one-eighth of the chip (surface to center in each dimension)
C     enclosed by the FD grid constitutes real space.  The rest of the
C     chip is shadow space.  Points in real and shadow space are used in
C     FD approximations.  A FD approximation consisting entirely of real
C     space points is then constructed by folding the shadow points onto
C     the corresponding real points through application of the symmetry
C     condition about the chip centerplanes.  Using shadow space in this
C-----fashion greatly increases the attainable accuracy.

C-----make certain enough grid points are available for the FD formulas

      IF(MINO(NX2, NY2, NZ2) .GT. OHP1) GO TO 1
         WRITE(P, 9001) NX2, NY2, NZ2, OHP1
         STOP
    1 CONTINUE

C-----dU/dx (CX1), d2U/dx2 (CX2), and dA/dx (CAX1)
```

```
C-----note: FINDIF expects grid spacing to either be constant or to
C-----geometrically INCREASE towards the chip center.

      DO 100 K = 1, NGPTX
         J = NX2 - K
         AVAIL(J)  = 1
         AVAIL(K)  = 1
         POINTS(J) = LENX - LOCX(K)
         POINTS(K) = LOCX(K)
  100 CONTINUE

      DO 200 X = 1, NGPTX

C--------dU/dx (CX1)

         IF(X .EQ. 1 .OR. X .EQ. NGPTX) GO TO 2
            AVAIL(1)     = 2                                  % 1
            AVAIL(NGPTX) = 2
            AVAIL(NX2M1) = 1

            CALL FD(OHP1, NX2M1, NGPTX, X, 1, POINTS, AVAIL, HIX1(X)
     &             , LOX1(X), CX1)
    2    CONTINUE

C--------dTemperature/dx (CTX1)

         IF(X .EQ. NGPTX) GO TO 5
            AVAIL(1)     = 1
            AVAIL(NGPTX) = 2
            AVAIL(NX2M1) = 1

            CALL FD(OHP1, NX2M1, NGPTX, X, 1, POINTS, AVAIL, HITX1(X)
     &             , LOTX1(X), CTX1)
    5    CONTINUE

C--------d2U/dx2 (CX2)

         AVAIL(1)     = 2                   % 1; IF(X .EQ. 1) AVAIL(1) = 2
         AVAIL(NGPTX) = 2
         AVAIL(NX2M1) = 1

         CALL FD(OHP2, NX2M1, NGPTX, X, 2, POINTS, AVAIL, HIX2(X)
     &          , LOX2(X), CX2)

C--------d2Temperature/dx2 (CTX2)

         AVAIL(1)     = 1
         AVAIL(NGPTX) = 2
         AVAIL(NX2M1) = 1

         CALL FD(OHP2, NX2M1, NGPTX, X, 2, POINTS, AVAIL, HITX2(X)
     &          , LOTX2(X), CTX2)
```

```
C--------dA/dx (CXA)

         IF(X .EQ. NGPTX) GO TO 4
            AVAIL(1)     = 1
            AVAIL(NGPTX) = 2
            AVAIL(NX2M1) = 1

            CALL FD(OHP1, NX2M1, NGPTX, X, 1, POINTS, AVAIL, HIXA(X)
     &             , LOXA(X), CXA)
    4    CONTINUE

C--------dA(Temperature)/dx (CTXA)

         IF(X .EQ. NGPTX) GO TO 8
            AVAIL(1)     = 1
            AVAIL(NGPTX) = 2
            AVAIL(NX2M1) = 1

            CALL FD(OHP1, NX2M1, NGPTX, X, 1, POINTS, AVAIL, HITXA(X)
     &             , LOTXA(X), CTXA)
    8    CONTINUE
  200 CONTINUE

C-----optimize formulas for centerpoint 1st derivatives (CX1 & CXA)
C-----and surface 1st derivative (CX1)

      LOX1(1) = 1
      HIX1(1) = 1
      CX1(0, 1) = ONE
      CX1(1, 1) = ZERO

      LOX1(NGPTX) = NGPTX
      HIX1(NGPTX) = NGPTX
      CX1(0, NGPTX) = ZERO
      CX1(NGPTX, NGPTX) = ZERO

      LOTX1(NGPTX) = NGPTX
      HITX1(NGPTX) = NGPTX
      CTX1(0, NGPTX) = ZERO
      CTX1(NGPTX, NGPTX) = ZERO

      LOXA(NGPTX) = NGPTX
      HIXA(NGPTX) = NGPTX
      CXA(NGPTX, NGPTX) = ZERO

      LOTXA(NGPTX) = NGPTX
      HITXA(NGPTX) = NGPTX
      CTXA(NGPTX, NGPTX) = ZERO

C-----dU/dy (CY1), d2U/dy2 (CY2), and dA/dy (CAY1)

      DO 1100 K = 1, NGPTY
         J = NY2 - K
         AVAIL(J)  = 1
```

```
        AVAIL(K)   = 1
        POINTS(J) = LENY - LOCY(K)
        POINTS(K) = LOCY(K)
 1100 CONTINUE

      DO 1200 Y = 1, NGPTY

C--------dU/dy (CY1)

        IF(Y .EQ. 1 .OR. Y .EQ. NGPTY) GO TO 12
           AVAIL(1)     = 2                                       % 1
           AVAIL(NGPTY) = 2
           AVAIL(NY2M1) = 1

           CALL FD(OHP1, NY2M1, NGPTY, Y, 1, POINTS, AVAIL, HIY1(Y)
     &            , LOY1(Y), CY1)
   12   CONTINUE

C--------dTemperature/dy (CTY1)

        IF(Y .EQ. NGPTY) GO TO 15
           AVAIL(1)     = 1
           AVAIL(NGPTY) = 2
           AVAIL(NY2M1) = 1

           CALL FD(OHP1, NY2M1, NGPTY, Y, 1, POINTS, AVAIL, HITY1(Y)
     &            , LOTY1(Y), CTY1)
   15   CONTINUE

C--------d2U/dy2 (CY2)

        AVAIL(1)     = 2                     % 1; IF(Y .EQ. 1) AVAIL(1) = 2
        AVAIL(NGPTY) = 2
        AVAIL(NY2M1) = 1

        CALL FD(OHP2, NY2M1, NGPTY, Y, 2, POINTS, AVAIL, HIY2(Y)
     &         , LOY2(Y), CY2)

C--------d2Temperature/dy2 (CTY2)

        AVAIL(1)     = 1
        AVAIL(NGPTY) = 2
        AVAIL(NY2M1) = 1

        CALL FD(OHP2, NY2M1, NGPTY, Y, 2, POINTS, AVAIL, HITY2(Y)
     &         , LOTY2(Y), CTY2)

C--------dA/dy (CYA)

        IF(Y .EQ. NGPTY) GO TO 14
           AVAIL(1)     = 1
           AVAIL(NGPTX) = 2
           AVAIL(NY2M1) = 1
```

```
          CALL FD(OHP1, NY2M1, NGPTY, Y, 1, POINTS, AVAIL, HIYA(Y)
     &              , LOYA(Y), CYA)
   14    CONTINUE

C--------dA(Temperature)/dy (CTYA)

         IF(Y .EQ. NGPTY) GO TO 18
            AVAIL(1)     = 1
            AVAIL(NGPTY) = 2
            AVAIL(NY2M1) = 1

            CALL FD(OHP1, NY2M1, NGPTY, Y, 1, POINTS, AVAIL, HITYA(Y)
     &              , LOTYA(Y), CTYA)
   18    CONTINUE
 1200 CONTINUE

C-----optimize formulas for centerpoint 1st derivatives (CY1 & CYA)
C-----and surface 1st derivative (CY1)

      LOY1(1) = 1
      HIY1(1) = 1
      CY1(0, 1) = ONE
      CY1(1, 1) = ZERO

      LOY1(NGPTY) = NGPTY
      HIY1(NGPTY) = NGPTY
      CY1(0, NGPTY) = ZERO
      CY1(NGPTY, NGPTY) = ZERO

      LOTY1(NGPTY) = NGPTY
      HITY1(NGPTY) = NGPTY
      CTY1(0, NGPTY) = ZERO
      CTY1(NGPTY, NGPTY) = ZERO

      LOYA(NGPTY) = NGPTY
      HIYA(NGPTY) = NGPTY
      CYA(NGPTY, NGPTY) = ZERO

      LOTYA(NGPTY) = NGPTY
      HITYA(NGPTY) = NGPTY
      CTYA(NGPTY, NGPTY) = ZERO

C-----dU/dz (CZ1), d2U/dz2 (CZ2), and dA/dz (CAZ1)

      DO 2100 K = 1, NGPTZ
         J = NZ2 - K
         AVAIL(J)  = 1
         AVAIL(K)  = 1
         POINTS(J) = LENZ - LOCZ(K)
         POINTS(K) = LOCZ(K)
 2100 CONTINUE

      DO 2200 Z = 1, NGPTZ
```

```
C--------dU/dz (CZ1)

         IF(Z .EQ. 1 .OR. Z .EQ. NGPTZ) GO TO 22
            AVAIL(1)     = 2                                       % 1
            AVAIL(NGPTZ) = 2
            AVAIL(NZ2M1) = 1

            CALL FD(OHP1, NZ2M1, NGPTZ, Z, 1, POINTS, AVAIL, HIZ1(Z)
     &             , LOZ1(Z), CZ1)
   22    CONTINUE

C--------dTemperature/dz (CTZ1)

         IF(Z .EQ. NGPTZ) GO TO 25
            AVAIL(1)     = 1
            AVAIL(NGPTZ) = 2
            AVAIL(NZ2M1) = 1

            CALL FD(OHP1, NZ2M1, NGPTZ, Z, 1, POINTS, AVAIL, HITZ1(Z)
     &             , LOTZ1(Z), CTZ1)
   25    CONTINUE

C--------d2U/dz2 (CZ2)

         AVAIL(1)     = 2                  % 1; IF(Z .EQ. 1) AVAIL(1) = 2
         AVAIL(NGPTZ) = 2
         AVAIL(NZ2M1) = 1

         CALL FD(OHP2, NZ2M1, NGPTZ, Z, 2, POINTS, AVAIL, HIZ2(Z)
     &          , LOZ2(Z), CZ2)

C--------d2Temperature/dz2 (CTZ2)

         AVAIL(1)     = 1
         AVAIL(NGPTZ) = 2
         AVAIL(NZ2M1) = 1

         CALL FD(OHP2, NZ2M1, NGPTZ, Z, 2, POINTS, AVAIL, HITZ2(Z)
     &          , LOTZ2(Z), CTZ2)

C--------dA/dz (CZA)

         IF(Z .EQ. NGPTZ) GO TO 24
            AVAIL(1)     = 1
            AVAIL(NGPTZ) = 2
            AVAIL(NZ2M1) = 1

            CALL FD(OHP1, NZ2M1, NGPTZ, Z, 1, POINTS, AVAIL, HIZA(Z)
     &             , LOZA(Z), CZA)
   24    CONTINUE

C--------dA(Temperature)/dz (CTZA)

         IF(Z .EQ. NGPTZ) GO TO 28
```

```
          AVAIL(1)      = 1
          AVAIL(NGPTZ) = 2
          AVAIL(NZ2M1) = 1

          CALL FD(OHP1, NZ2M1, NGPTZ, Z, 1, POINTS, AVAIL, HITZA(Z)
     &              , LOTZA(Z), CTZA)
   28     CONTINUE
 2200 CONTINUE

C-----optimize formulas for centerpoint 1st derivatives (CZ1 & CZA)
C-----and surface 1st derivative (CZ1)

      LOZ1(1) = 1
      HIZ1(1) = 1
      CZ1(0, 1) = ONE
      CZ1(1, 1) = ZERO

      LOZ1(NGPTZ) = NGPTZ
      HIZ1(NGPTZ) = NGPTZ
      CZ1(0, NGPTZ) = ZERO
      CZ1(NGPTZ, NGPTZ) = ZERO

      LOTZ1(NGPTZ) = NGPTZ
      HITZ1(NGPTZ) = NGPTZ
      CTZ1(0, NGPTZ) = ZERO
      CTZ1(NGPTZ, NGPTZ) = ZERO

      LOZA(NGPTZ) = NGPTZ
      HIZA(NGPTZ) = NGPTZ
      CZA(NGPTZ, NGPTZ) = ZERO

      LOTZA(NGPTZ) = NGPTZ
      HITZA(NGPTZ) = NGPTZ
      CTZA(NGPTZ, NGPTZ) = ZERO

      WRITE(P, /) 'ERROR CXA ', (X, ROUND(ERRFD(X, 1, NGPTX, 0, HIXA(X)
     &, LOXA(X), LOCX, CXA, KNORM), NSIG), X = 1, NGPTX-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CX1 ', (X, ROUND(ERRFD(X, 1, NGPTX, 0, HIX1(X)
     &, LOX1(X), LOCX, CX1, KNORM), NSIG), X = 1, NGPTX-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CX2 ', (X, ROUND(ERRFD(X, 2, NGPTX, 0, HIX2(X)
     &, LOX2(X), LOCX, CX2, KNORM), NSIG), X = 1, NGPTX)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CYA ', (Y, ROUND(ERRFD(Y, 1, NGPTY, 0, HIYA(Y)
     &, LOYA(Y), LOCY, CYA, KNORM), NSIG), Y = 1, NGPTY-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CY1 ', (Y, ROUND(ERRFD(Y, 1, NGPTY, 0, HIY1(Y)
     &, LOY1(Y), LOCY, CY1, KNORM), NSIG), Y = 1, NGPTY-1)
      WRITE(P, 9000)
```

```
      WRITE(P, /) 'ERROR CY2 ', (Y, ROUND(ERRFD(Y, 2, NGPTY, 0, HIY2(Y)
     &, LOY2(Y), LOCY, CY2, KNORM), NSIG), Y = 1, NGPTY)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CZA ', (Z, ROUND(ERRFD(Z, 1, NGPTZ, 0, HIZA(Z)
     &, LOZA(Z), LOCZ, CZA, KNORM), NSIG), Z = 1, NGPTZ-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CZ1 ', (Z, ROUND(ERRFD(Z, 1, NGPTZ, 0, HIZ1(Z)
     &, LOZ1(Z), LOCZ, CZ1, KNORM), NSIG), Z = 1, NGPTZ-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CZ2 ', (Z, ROUND(ERRFD(Z, 2, NGPTZ, 0, HIZ2(Z)
     &, LOZ2(Z), LOCZ, CZ2, KNORM), NSIG), Z = 1, NGPTZ)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTXA', (X, ROUND(ERRFD(X, 1, NGPTX, 0, HITXA(X)
     &, LOTXA(X), LOCX, CTXA, KNORM), NSIG), X = 1, NGPTX-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTX1', (X, ROUND(ERRFD(X, 1, NGPTX, 0, HITX1(X)
     &, LOTX1(X), LOCX, CTX1, KNORM), NSIG), X = 1, NGPTX-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTX2', (X, ROUND(ERRFD(X, 2, NGPTX, 0, HITX2(X)
     &, LOTX2(X), LOCX, CTX2, KNORM), NSIG), X = 1, NGPTX)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTYA', (Y, ROUND(ERRFD(Y, 1, NGPTY, 0, HITYA(Y)
     &, LOTYA(Y), LOCY, CTYA, KNORM), NSIG), Y = 1, NGPTY-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTY1', (Y, ROUND(ERRFD(Y, 1, NGPTY, 0, HITY1(Y)
     &, LOTY1(Y), LOCY, CTY1, KNORM), NSIG), Y = 1, NGPTY-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTY2', (Y, ROUND(ERRFD(Y, 2, NGPTY, 0, HITY2(Y)
     &, LOTY2(Y), LOCY, CTY2, KNORM), NSIG), Y = 1, NGPTY)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTZA', (Z, ROUND(ERRFD(Z, 1, NGPTZ, 0, HITZA(Z)
     &, LOTZA(Z), LOCZ, CTZA, KNORM), NSIG), Z = 1, NGPTZ-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTZ1', (Z, ROUND(ERRFD(Z, 1, NGPTZ, 0, HITZ1(Z)
     &, LOTZ1(Z), LOCZ, CTZ1, KNORM), NSIG), Z = 1, NGPTZ-1)
      WRITE(P, 9000)

      WRITE(P, /) 'ERROR CTZ2', (Z, ROUND(ERRFD(Z, 2, NGPTZ, 0, HITZ2(Z)
     &, LOTZ2(Z), LOCZ, CTZ2, KNORM), NSIG), Z = 1, NGPTZ)
      WRITE(P, 9000)

C     WRITE(P, *//) CXA
```

```
C     WRITE(P, *//) CX1
C     WRITE(P, *//) CX2
C     WRITE(P, *//) CYA
C     WRITE(P, *//) CY1
C     WRITE(P, *//) CY2
C     WRITE(P, *//) CZA
C     WRITE(P, *//) CZ1
C     WRITE(P, *//) CZ2
C     WRITE(P, *//) CTXA
C     WRITE(P, *//) CTX1
C     WRITE(P, *//) CTX2
C     WRITE(P, *//) CTYA
C     WRITE(P, *//) CTY1
C     WRITE(P, *//) CTY2
C     WRITE(P, *//) CTZA
C     WRITE(P, *//) CTZ1
C     WRITE(P, *//) CTZ2
      WRITE(P, 9000)

      RETURN
      END

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE SWAP  (
     &  TOVECT, N      , TIGEAR, V      , VPRIME)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'

      LOGICAL
     &  TOVECT

      INTEGER
     &  N
     &, J      , K
     &, X      , Y      , Z      , COMP

      DOUBLE PRECISION
     &  TIGEAR, V(N)  , VPRIME(N)

C------------------------------------------------------------------------

DEBUG MONITOR(6) PTP, VO(PTP), VU(PTP)
C     WRITE(P, *//) 'ENTER SWAP', VU
C     WRITE(P, *//) TOVECT, N, TIGEAR, PDLO, POHI, PYLO, PNHI
C     WRITE(P, /) 'V', ((J, V(J)), J = 1, 8)
C     WRITE(P, /) ' '

C-----note: PYLO <= PYHI < PDLO <= PDHI < PNLO <= PNHI < POLO <= POHI

      IF(TOVECT) GO TO 2
        J = 1
```

```
          DO 100 K = PDLO, POHI
C            WRITE(6, /)'J',J,'K',K,'VU(K)',VU(K),'V(J)',V(J)
C            WRITE(6, *//) VU
             VU(K)  = V(J)
C            WRITE(6, /)'J',J,'K',K,'VU(K)',VU(K),'V(J)',V(J)
C            WRITE(6, *//) VU
             VG(K)  = VPRIME(J)
C            WRITE(6, /)'J',J,'K',K,'VU(K)',VU(K),'V(J)',V(J)
C            WRITE(6, *//) VU
C            WRITE(6, /) ' '
             J = J + 1
  100     CONTINUE
C         WRITE(6, *//) VU

          DO 110 Z = 1, NGPTZ
             DO 110 Y = 1, NGPTY
                DO 110 X = 1, NGPTX
                   DO 110 COMP = PYLO, PNHI
                      U(COMP, X, Y, Z) = V(J)
                      G(COMP, X, Y, Z) = VPRIME(J)
                      J = J + 1
  110     CONTINUE
       GO TO 9

    2 CONTINUE
          J = 1
          DO 200 K = PDLO, POHI
             V(J) = VU(K)
             VPRIME(J) = VG(K)
             J = J + 1
  200     CONTINUE

          DO 210 Z = 1, NGPTZ
             DO 210 Y = 1, NGPTY
                DO 210 X = 1, NGPTX
                   DO 210 COMP = PYLO, PNHI
                      V(J)      = U(COMP, X, Y, Z)
                      VPRIME(J) = G(COMP, X, Y, Z)
                      J = J + 1
  210     CONTINUE
       GO TO 9

    9 CONTINUE
C      WRITE(P, *//) 'EXIT SWAP', VU
C      WRITE(P, /) ' '
       RETURN
       END

C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
       SUBROUTINE FCN    (
     & N       , TIGEAR, V      , VPRIME)
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

       IMPLICIT COMPLEX(A - Z)
```

```
      INTEGER
     & N

      DOUBLE PRECISION
     & TIGEAR, V(N)  , VPRIME(N)

$     INCLUDE 'CHIP/COMMON'

C-----------------------------------------------------------------------

DEBUG MONITOR(6) PTP, VO(PTP), VU(PTP)
C     WRITE(P, *//) 'ENTER FCN', VU
C     WRITE(P, /) ' '

C-----download V(*) to U(*,*,*,*)

      CALL SWAP(.FALSE., N, TIGEAR, V, VPRIME)

      TIME = TIGEAR

      CALL DUDT

C-----upload U(*,*,*,*) to V(*)

      CALL SWAP(.TRUE. , N, TIGEAR, V, VPRIME)

C     WRITE(P, *//) 'ENTER FCN', VU
C     WRITE(P, /) ' '
      RETURN
      END

$     OPT = 0  SET OWN  SET OWNARRAYS
C-----------------------------------------------------------------------

C    Chip routine name    - INITAL

C    Creation date        -  3 Jun 82
C    Latest revision      - 21 Aug 85

C    Author               - Mark A. Burazin

C    Purpose              - Initialize common block variables.

C    Hardware             - Burroughs Double Precision

C    Usage                - CALL INITAL

C    Arguments            - None.

C    Common       CHIP    - All global variables used by chip model.

C    Routines called      - SWAP
```

```
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
      SUBROUTINE INITAL
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'

      DOUBLE PRECISION
     &  ALPHA , ONE    , ZERO   , INITTP, INITDS, INITLD, TIGEAR
     &, BETA  , HALF   , HX     , HY    , HZ    , TWO
     &, LXD2  , LYD2   , LZD2

      INTEGER
     &  COMP  , J      , X      , Y     , Z

      DATA
     &  HALF   /+5.D-1/
     &, ONE    /+1.D+0/
     &, TWO    /+2.D+0/
     &, ZERO   /+0.D+0/

 9000 FORMAT(' ')
 9100 FORMAT(/ ' *** FATAL ERROR: NGPTX (= ', I2, '), NGPTY (= ', I2
     &      , '), and NGPTZ (= ', I2, '), must all be >= 3. ***'
     &      / ' *** PROGRAM HALTED ***')
 9200 FORMAT(// ' *** FATAL ERROR ***'
     &      / ' NaOH (= ', 1PD14.4, ') must be greater than zero and'
     &      / ' NaSH (= ', 1PD14.4, ') & AQ (= ', 1PD14.4, ') must be'
     &      / ' non-negative.  *** PROGRAM HALTED ***')

C-----------------------------------------------------------------------

DEBUG MONITOR(6) PTP, VO(PTP), VU(PTP)

C-----Note that the components are divided up into reactants which
C     contribute to yield (PYLO to PYHI), diffusing species (PDLO to
C     PDHI), reactants which don't contribute to yield (PNLO to PNHI),
C     and others (POLO to POHI).  Examples:
C     PY:  native lignin
C     PD:  sodium hydroxide
C     PN:  pulp viscosity
C     PO:  yield
C-----PYLO <= PYHI < PDLO <= PDHI < PNLO <= PNHI < POLO <= POHI

      IF(DEBUGG .LT. 1) GO TO 1
         WRITE(P, /) '    enter INITAL'
         WRITE(P, 9000)
    1 CONTINUE

C-----make certain NGPTX, NGPTY, or NGPTZ >= 3

      IF(NGPTX .GE. 3 .AND. NGPTY .GE. 3 .AND. NGPTZ .GE. 3) GO TO 2
         WRITE(P, 9100) NGPTX, NGPTY, NGPTZ
```

```
         STOP
       2 CONTINUE

C-----calculate grid spacings
C     If H < 0, automatically use even grid spacing for all three axes
C     If H > 0, use as grid spacing between surface and first interior
C        grid point for all three axes and calculate the rest of the
C        grid spacings using a different geometric progression for each
C        axis
C-----If H = 0, generate a value for H then proceed as for H > 0 case

      LXD2 = LENX * HALF
      LYD2 = LENY * HALF
      LZD2 = LENZ * HALF

      IF(H) 10, 20, 30
   10    CONTINUE
            DO 11 X = 1, NGPTX
               LOCX(X) = LXD2 * DFLOAT(X - 1) / DFLOAT(NGPTX - 1)
   11       CONTINUE
            DO 12 Y = 1, NGPTY
               LOCY(Y) = LYD2 * DFLOAT(Y - 1) / DFLOAT(NGPTY - 1)
   12       CONTINUE
            DO 13 Z = 1, NGPTZ
               LOCZ(Z) = LZD2 * DFLOAT(Z - 1) / DFLOAT(NGPTZ - 1)
   13       CONTINUE
         GO TO 40
   20    CONTINUE
            HX = LXD2 / ((TWO ** (NGPTX-1)) - ONE)
            HY = LYD2 / ((TWO ** (NGPTY-1)) - ONE)
            HZ = LZD2 / ((TWO ** (NGPTZ-1)) - ONE)
            H  = DMIN1(HX, HY, HZ)
         GO TO 30
   30    CONTINUE
            H = DMIN1(LXD2 / DFLOAT(NGPTX - 1)
     &                , LYD2 / DFLOAT(NGPTY - 1)
     &                , LZD2 / DFLOAT(NGPTZ - 1)
     &                , H)
            CALL GRID(NGPTX, H, LXD2, LOCX)
            CALL GRID(NGPTY, H, LYD2, LOCY)
            CALL GRID(NGPTZ, H, LZD2, LOCZ)
         GO TO 40
   40 CONTINUE

C-----round off grid locations to NSIG significant digits

      DO 52 X = 1, NGPTX
         LOCX(X) = ROUND(LOCX(X), NSIG)
   52 CONTINUE

      DO 54 Y = 1, NGPTY
         LOCY(Y) = ROUND(LOCY(Y), NSIG)
   54 CONTINUE
```

```
      DO 56 Z = 1, NGPTZ
         LOCZ(Z) = ROUND(LOCZ(Z), NSIG)
   56 CONTINUE

C-----generate finite difference formulas
C     KNORM is normalized so that du/dx(s) = KNORM (us - ub) D / Dx
C     and dub/dt = KNORM D (us - ub) (Ac / Vb) (Vt / Vc)
C-----KNORM = 2 Bi (Ac / Vc) (Vc / Vt)

      KNORM = TWO * BIOT * (LENX * LENY + LENX * LENZ + LENY * LENZ)
     &                   / (LENX * LENY * LENZ)
     &       *(RHOL / RHOC)
     &       /(RHOL / RHOC - RHOL / RHOW)

      CALL FINDIF

C-----calculate initial concentrations of diffusing species

      IF(VO(POH) .GT. ZERO) GO TO 3
         ALPHA = EA * RHOL / 31.D2 / LIQ2WD
         BETA  = SULFID / (2.D2 - SULFID)
         VO(POH) = 40.DO * ALPHA
         VO(PSH) = 56.DO * ALPHA * BETA
         VO(PAQ) = AQCHAR * RHOL / 1.D2 / LIQ2WD
    3 CONTINUE

C-----initialize chip interior conditions

      TIGEAR = ZERO
      INITDS = (VO(POH) + VO(PSH) + VO(PAQ)) / VO(PDS)
      INITTP = TPMPRG / TPCOOK

      DO 100 Z = 1, NGPTZ
         DO 100 Y = 1, NGPTY
            DO 100 X = 1, NGPTX
               U(PCN, X, Y, Z) = ONE
               U(PCX, X, Y, Z) = ZERO
               U(PGN, X, Y, Z) = ONE
               U(PGX, X, Y, Z) = ZERO
               U(PXN, X, Y, Z) = ONE
               U(PXX, X, Y, Z) = ZERO
               U(PEX, X, Y, Z) = ONE
               U(PAG, X, Y, Z) = ONE
               U(PLN, X, Y, Z) = ONE
               U(PLR, X, Y, Z) = ZERO
               U(PLD, X, Y, Z) = ZERO
               DO 120 COMP = POH, PAQ
                  U(COMP, X, Y, Z) = ZERO
                  IF(FMPREG) U(COMP, X, Y, Z) = ONE
  120          CONTINUE
               U(PDS, X, Y, Z) = ZERO
               IF(FMPREG) U(PDS, X, Y, Z) = INITDS
               U(PTP, X, Y, Z) = INITTP
               U(PVP, X, Y, Z) = ONE
```

```
            U(PY , X, Y, Z) = ONE
  100 CONTINUE

      ALPHA  = LIQ2WD / (LIQ2WD + RHOL / RHOW - RHOL / RHOC)

      VU(PLD) = ZERO
      VU(POH) = ONE
      VU(PSH) = ONE
      VU(PAQ) = ONE
      VU(PDS) = INITDS

      DO 200 COMP = PLD, PDS
         IF(.NOT. FMPREG) VU(COMP) = ALPHA * VU(COMP)
  200 CONTINUE
      VU(PTP)     = INITTP
      VU(PGF)     = ZERO
      VU(PGF)     = ZERO
      VU(PHF)     = ZERO
      VO(PTP)     = TPCOOK
C     WRITE(P, *//) 'EXIT INITAL', VU

      IF(VO(POH) .GT. ZERO .AND. VO(PSH) .GE. ZERO .AND.
     &   VO(PAQ) .GE. ZERO) GO TO 4
         WRITE(P, 9200) VO(POH), VO(PSH), VO(PAQ)
         STOP
    4 CONTINUE

C-----set up print times

      IF(.NOT. FLOOPP) GO TO 9
         DO 300 J = 1, NPRINT
            TPRINT(J) = DFLOAT(J - 1) * TINTER + TIFRST
  300    CONTINUE
    9 CONTINUE
      DO 400 J = 1, NPRINT
         TPRINT(J) = ROUND(TPRINT(J), NSIG)
  400 CONTINUE
      CALL VSRTAD(TPRINT, NPRINT)

      RETURN
      END


      SUBROUTINE FCNJ  (N, X, Y, PD)
      INTEGER N
      DOUBLE PRECISION Y(N), PD(N, N), X
      RETURN
      END

C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     MAIN PROGRAM
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

```
      IMPLICIT COMPLEX(A - Z)

$     INCLUDE 'CHIP/COMMON'
$     INCLUDE 'CHIP/SOURCE/DECLARE'

      INTEGER
     &  COMP   , IDUMMY, J      , NQUSED, NSTEP , NFE    , NJE    , GFACT
     &, HFACT  , IER    , INDEX  , Z
     &, PTHR   , PTMIN  , IOHR   , IOMIN , K     , X      , Y      , JPRINT

      DOUBLE PRECISION
     &  DUMMY  , HUSED  , HGEAR  , XEND   , TIGEAR
     &, PTSEC  , IOSEC  , KAPPA  , YIELD  , ZERO   , BLK
     &, CEL    , GM     , XYLAN  , EXTRAC, ACETYL, LIGNIN, CARBOS, VISC

      REAL
     &  DUM    , DUMJ   , SDUMMY

      DIMENSION
     &  DUMMY(48)
     &, IDUMMY(38)
     &, SDUMMY(4)

      COMMON
     & /GEAR   / DUMMY  , SDUMMY, IDUMMY

      DATA
     &  ZERO    /+0.D+0/

 9000 FORMAT(' ')
 9001 FORMAT('1')
 9010 FORMAT(' Processor Time =',I3,':',I2,':',F5.2
     &  ,'  Input/Output Time =',I3,':',I2,':',F5.2)

C-----------------------------------------------------------------------

DEBUG MONITOR(6) PTP, VO(PTP), VU(PTP)

      HGEAR  = ROUND(DSQRT(ERREPS), NSIG)
      INDEX  = 1 % first call to DGEAR

C-----initialize COMMON

      CALL INITAL

C-----initialize GEAR's 'Y' vector (V)

      J = 1
      DO 100 K = PDLO, POHI
         V(J) = VU(K)
         J = J + 1
  100 CONTINUE

      DO 110 Z = 1, NGPTZ
```

```
            DO 110 Y = 1, NGPTY
               DO 110 X = 1, NGPTX
                  DO 110 COMP = PYLO, PNHI
                     V(J) = U(COMP, X, Y, Z)
                     J = J + 1
  110 CONTINUE

         DO 200 JPRINT = 1, NPRINT
            XEND = TPRINT(JPRINT)
            IF(XEND .GT. TIEND) STOP

C--------print out initial conditions

            IF(TIGEAR .EQ. ZERO) GO TO 2

    1          CONTINUE
               CALL DGEAR(
     &           N      , DUM    , DUMJ   , TIGEAR, HGEAR  , V      , XEND
     &         , ERRTOL, METH   , MITER  , INDEX  , IWORK  , WORK   , IER    )

               IF(IER .GT. 128) STOP
    2       CONTINUE

C--------dump U to disk file D

            CALL DUMP(U     , D      , NSIG   , TIME   ,'U     '
     &                , POHI  , PYLO   , NGPTX  , NGPTY  , NGPTZ )

C--------write current values of Y

            HUSED  = ROUND(DUMMY(8), NSIG)
            NQUSED = IDUMMY(6)
            NSTEP  = IDUMMY(7)
            NFE    = IDUMMY(8)
            NJE    = IDUMMY(9)

            IF(TIGEAR .GT. ZERO) GO TO 3
               WRITE(P, *//) NGPTX, NGPTY, NGPTZ, LOCX, LOCY, LOCZ
               WRITE(P, 9000)
               WRITE(P, *//) N, TIGEAR, HGEAR, XEND, ERRTOL, METH, MITER
     &                     , INDEX
               WRITE(P, 9000)
               WRITE(P, *//) ERRTOL, H, LENX, LENY, LENZ, LIQ2WD, TIEND
     &                     , TIFRST, TIMPRG, TINTER, TI2TP, TPCOOK, TPMPRG
     &                     , EA, SULFID, AQCHAR, DEBUGG, ORDERH, FMPREG
     &                     , FLOOPP, BIOT, RHOC, RHOL, RHOW, HTCOEF, VO
               WRITE(P, 9000)
               WRITE(P, *//) NPRINT, TPRINT
               WRITE(P, 9000)
    3       CONTINUE

            WRITE(P, *//) TIME, TIGEAR, HUSED, NQUSED, NSTEP, NFE,NJE,INDEX
            WRITE(P, 9000)
```

```
      CALL TIMER(PTHR  , PTMIN , PTSEC , 2     )
      CALL TIMER(IOHR  , IOMIN , IOSEC , 3     )

      WRITE(P, 9010) PTHR, PTMIN, PTSEC, IOHR, IOMIN, IOSEC
      WRITE(P, 9000)

      DO 10 COMP = PDLO, PDHI
         BULK (COMP) = ROUND(VU(COMP), NSIG)
         GBULK(COMP) = ROUND(VG(COMP), NSIG)
10    CONTINUE
      BULK(PTP) = ROUND(VU(PTP) * VO(PTP) - 273D0, NSIG)

      WRITE(P, *//) PCN, PCX, PGN, PGX, PXN, PXX, PEX, PAG, PLN
     &             , PLR, PLD, POH, PSH, PAQ, PDS, PTP, PVP, PY
     &             , PGF, PHF
         WRITE(P, 9000)
      WRITE(P, *//) BULK
      WRITE(P, *//) GBULK
      WRITE(P, 9000)

      DO 20 COMP = PYLO, POHI
         BLK = ZERO
         IF(COMP .GE. PDLO .AND. COMP .LE. PDHI) BLK = VU(COMP)
         AVERAG(COMP) = AVE(LOCX  , LOCY  , LOCZ  , U     , BLK
     &                    , CTX1  , CTX2  , CTY1  , CTY2  , CTZ1
     &                    , CTZ2  , HITX1 , HITX2 , HITY1 , HITY2
     &                    , HITZ1 , HITZ2 , LOTX1 , LOTX2 , LOTY1
     &                    , LOTY2 , LOTZ1 , LOTZ2 , POHI  , PYLO
     &                    , NGPTX , NGPTY , NGPTZ , COMP  , 0, 0, 0)
         CENTER(COMP) = U(COMP, NGPTX, NGPTY, NGPTZ)
         CORNER(COMP) = U(COMP,  1,  1,  1)
         BLK = ZERO
         IF(COMP .GE. PDLO .AND. COMP .LE. PDHI) BLK = VG(COMP)
         GAVERG(COMP) = AVE(LOCX  , LOCY  , LOCZ  , G     , BLK
     &                    , CTX1  , CTX2  , CTY1  , CTY2  , CTZ1
     &                    , CTZ2  , HITX1 , HITX2 , HITY1 , HITY2
     &                    , HITZ1 , HITZ2 , LOTX1 , LOTX2 , LOTY1
     &                    , LOTY2 , LOTZ1 , LOTZ2 , POHI  , PYLO
     &                    , NGPTX , NGPTY , NGPTZ , COMP  , 0, 0, 0)
         GCENTR(COMP) = G(COMP, NGPTX, NGPTY, NGPTZ)
         GCORNR(COMP) = G(COMP,  1,  1,  1)
         GSURF (COMP) = G(COMP, NGPTX, NGPTY,  1)
         SURFAC(COMP) = U(COMP, NGPTX, NGPTY,  1)
20    CONTINUE
      CORNER(PTP) = CORNER(PTP) * VO(PTP) - 273D0
      CENTER(PTP) = CENTER(PTP) * VO(PTP) - 273D0
      SURFAC(PTP) = SURFAC(PTP) * VO(PTP) - 273D0
      AVERAG(PTP) = AVERAG(PTP) * VO(PTP) - 273D0

      CEL    = (AVERAG(PCN) * VO(PCN) + AVERAG(PCX) * VO(PCX))
     &         / VO(PY) * 1D2
      GM     = (AVERAG(PGN) * VO(PGN) + AVERAG(PGX) * VO(PGX))
     &         / VO(PY) * 1D2
      XYLAN  = (AVERAG(PXN) * VO(PXN) + AVERAG(PXX) * VO(PXX))
```

```
     &          / VO(PY) * 1D2
          EXTRAC = AVERAG(PEX) * VO(PEX) / VO(PY) * 1D2
          ACETYL = AVERAG(PAG) * VO(PAG) / VO(PY) * 1D2
          LIGNIN = (AVERAG(PLN) * VO(PLN) + AVERAG(PLR) * VO(PLR))
     &          / VO(PY) * 1D2
          CARBOS = CEL + GM + XYLAN
          YIELD  = CARBOS + EXTRAC + ACETYL + LIGNIN
          KAPPA  = LIGNIN / YIELD / 15D-4
          VISC   = AVERAG(PVP) * VO(PVP)

          YIELD  = ROUND(YIELD , NSIG)
          KAPPA  = ROUND(KAPPA, NSIG)
          CARBOS = ROUND(CARBOS, NSIG)
          CEL    = ROUND(CEL   , NSIG)
          GM     = ROUND(GM    , NSIG)
          XYLAN  = ROUND(XYLAN , NSIG)
          EXTRAC = ROUND(EXTRAC, NSIG)
          ACETYL = ROUND(ACETYL, NSIG)
          LIGNIN = ROUND(LIGNIN, NSIG)
          VISC   = ROUND(VISC  , NSIG)
          GFACT  = VU(PGF) + 5D-1
          HFACT  = VU(PHF) + 5D-1

          DO 30 COMP = PYLO, POHI
             AVERAG(COMP) = ROUND(AVERAG(COMP), NSIG)
             CENTER(COMP) = ROUND(CENTER(COMP), NSIG)
             CORNER(COMP) = ROUND(CORNER(COMP), NSIG)
             GAVERG(COMP) = ROUND(GAVERG(COMP), NSIG)
             GCENTR(COMP) = ROUND(GCENTR(COMP), NSIG)
             GCORNR(COMP) = ROUND(GCORNR(COMP), NSIG)
             GSURF (COMP) = ROUND(GSURF (COMP), NSIG)
             SURFAC(COMP) = ROUND(SURFAC(COMP), NSIG)
   30     CONTINUE

          WRITE(P, *//) HFACT , GFACT, YIELD , KAPPA , CARBOS, VISC
          WRITE(P, *//) CEL   , GM   , XYLAN , EXTRAC, ACETYL, LIGNIN
          WRITE(P, 9000)
          WRITE(P, *//) AVERAG
          WRITE(P, 9000)
          WRITE(P, *//) CENTER
          WRITE(P, 9000)
          WRITE(P, *//) CORNER
          WRITE(P, 9000)
          WRITE(P, *//) SURFAC
          WRITE(P, 9000)
          WRITE(P, *//) GAVERG
          WRITE(P, 9000)
          WRITE(P, *//) GCENTR
          WRITE(P, 9000)
          WRITE(P, *//) GCORNR
          WRITE(P, 9000)
          WRITE(P, *//) GSURF
          WRITE(P, 9000)
C         DO 210 COMP = PDLO, PDHI
```

```
C           DO 220 Z = 1, NGPTZ
C              DO 220 Y = 1, NGPTY
C                 DO 220 X = 1, NGPTX
C                    UC(X, Y, Z) = ROUND(AX(COMP, X, Y, Z), NSIG)
C 220       CONTINUE
C           WRITE(P, *//) 'AX diffusivities', COMP
C           WRITE(P, 9000)
C           WRITE(P, *//) UC
C           WRITE(P, 9000)
C 210    CONTINUE
C        DO 211 COMP = PDLO, PDHI
C           DO 221 Z = 1, NGPTZ
C              DO 221 Y = 1, NGPTY
C                 DO 221 X = 1, NGPTX
C                    UC(X, Y, Z) = ROUND(AY(COMP, X, Y, Z), NSIG)
C 221       CONTINUE
C           WRITE(P, *//) 'AY diffusivities', COMP
C           WRITE(P, 9000)
C           WRITE(P, *//) UC
C           WRITE(P, 9000)
C 211    CONTINUE
C        DO 212 COMP = PDLO, PDHI
C           DO 222 Z = 1, NGPTZ
C              DO 222 Y = 1, NGPTY
C                 DO 222 X = 1, NGPTX
C                    UC(X, Y, Z) = ROUND(AZ(COMP, X, Y, Z), NSIG)
C 222       CONTINUE
C           WRITE(P, *//) 'AZ diffusivities', COMP
C           WRITE(P, 9000)
C           WRITE(P, *//) UC
C           WRITE(P, 9000)
C 212    CONTINUE
         DO 230 Z = 1, NGPTZ
            DO 230 Y = 1, NGPTY
               DO 230 X = 1, NGPTX
                  DO 230 COMP = PYLO, POHI
                     IF(U(COMP, X, Y, Z) .LT. -1D0) WRITE(P, *//)
     &                  'FATAL ERROR IN U', X, Y, Z, U(COMP, X, Y, Z)
                     IF(U(COMP, X, Y, Z) .LT. -1D0) STOP
  230    CONTINUE

         WRITE(P, *//) VA
C        WRITE(P,9000)
         WRITE(P,9001)

C-------after initial conditons printed, go back and call DGEAR

         IF(TIGEAR .EQ. ZERO) GO TO 1
  200 CONTINUE
      END
```

```
#FILE (MARK)CHIP/SOURCE/DECLARE ON STUDENTS
      INTEGER
     &  IWORK , N      , METH   , MITER

      DOUBLE PRECISION
     &  V      , WORK
     &, AVERAG, BULK   , CENTER, CORNER, GAVERG, GBULK , GCENTR, GCORNR
     &, GSURF , SURFAC, UC

      DIMENSION
     &  AVERAG(1:18)
     &, BULK  (11:16)
     &, CENTER(1:18)
     &, CORNER(1:18)
     &, GAVERG(1:18)
     &, GBULK (11:16)
     &, GCENTR(1:18)
     &, GCORNR(1:18)
     &, GSURF (1:18)
     &, IWORK (2133)
     &, SURFAC(1:18)
     &, UC    (5, 5, 5)
     &, V     (2133)
     &, WORK  (23463)

      DATA
     &  METH  /2/
     &, MITER /3/
     &, N     /2133/
```

```
#FILE (MARK)CHIP/SOURCE/DECLARE/GENERIC ON STUDENTS
      INTEGER
     &  IWORK , N     , METH   , MITER

      DOUBLE PRECISION
     &  V     , WORK
     &, AVERAG, BULK  , CENTER, CORNER, GAVERG, GBULK , GCENTR, GCORNR
     &, GSURF , SURFAC, UC

      DIMENSION
     &  AVERAG(PYLO:POHI)
     &, BULK  (PDLO:PDHI)
     &, CENTER(PYLO:POHI)
     &, CORNER(PYLO:POHI)
     &, GAVERG(PYLO:POHI)
     &, GBULK (PDLO:PDHI)
     &, GCENTR(PYLO:POHI)
     &, GCORNR(PYLO:POHI)
     &, GSURF (PYLO:POHI)
     &, IWORK (ODE)
     &, SURFAC(PYLO:POHI)
     &, UC    (NGPTX, NGPTY, NGPTZ)
     &, V     (ODE)
     &, WORK  (NWORK)

      DATA
     &  METH  /NMETH/
     &, MITER /NMITER/
     &, N     /ODE/
```