DOE/BC/14600-42
Distribution Category UC-122

Multivariate Optimization of Production Systems-- The Time Dimension

SUPRI TR 90

DOE/BC/14600--42

DE93 000131

By
Niranjan Ravindran
Roland N. Horne

April 1993

Work Performed Under Contract No. DE-FG22-90BC14600

Prepared for
U.S. Department of Energy
Assistant Secretary for Fossil Energy

Thomas B. Reid, Project Manager
Bartlesville Project Office
P.O. Box 1398
Bartlesville, OK 74005

Prepared by
Stanford University
Petroleum Research Institute
Stanford, CA 94305-4042

Contents

List of Tables

List of Figures

Abstract

Traditional analysis of oil and gas production systems treats individual nodes one at a time. This only calculates a feasible solution which is not necessarily optimal. Multivariate optimization is able to determine the most profitable configuration, including all variables simultaneously. The optimization can also find the optimal recovery over a period of time, rather than just at a single instant as in traditional methods..This report describes the development of multivariate optimization for situations in which the decision variables may change as a function of time. For example, instead of estimating a tubing size which is optimal over the life of the project, this approach determines a series of optimal tubing sizes which may change from year to year. Examples show that under an optimal strategy, tubing size can be changed only infrequently while still increasing profitability of a project. The methods used in this work considered the special requirements of objectives which are not smooth functions of their decision variables. The physical problems considered included artificial lift production systems.

1. Introduction

1.1 Production Optimization

When a new well is drilled, the production engineer is given the task of designing the "best" completion for the well. Usually the engineer will use nodal analysis to find the "best" size of tubing, the "best" operating pressure for the production separator, etc. (decision variables) Any type of optimization is limited to varying the decision variables on a trial and error basis to find a suitable combination of the variables.

Nodal analysis is not an optimization technique. It is a method to find the stabilized flow rate for a given set of conditions (decision variables). Further, nodal analysis does not have a time dimension. The solution is for an instant in time.

There are numerical algorithms that are designed to find the minimum or maximum of nonlinear functions. These algorithms can be used to optimize production from a single completion. To accomplish the optimization an objective function has to be chosen along with the decision variables -- they are the variables that the engineer chooses to optimize. These optimization algorithms simultaneously vary all the decision variables, and proceed to find the extreme point without the use of a trial and error procedure.

Carroll [1990] was one of the first to apply nonlinear optimization algorithms to production systems. Others have used linear programming, integer programming, dynamic programming and other techniques to optimization problems in petroleum engineering. However, Carroll [1990] focused on optimization of well production. The reader is referred to the work of Carroll [1990] for a history of optimization studies in petroleum engineering.

In this study a model of the reservoir is coupled to a model of the tubing, and a model of a two stage separator. These models are then coupled with the nonlinear optimization algorithms and the objective function. The goal of the optimization study was to find the optimum values of the decision variables that gave the maximum value of the objective function -- in this case the present value of the hydrocarbon production. The production model included a time dimension; the oil and gas rates are given as a function of time.

The well performance was modeled using a tank model for the reservoir, and several multiphase correlations for the flow through the tubing string. The sequence of production separators was modeled using a two phase flash process and an equation of state. The well performance model was used to generate a production profile for oil and gas from the reservoir. This production profile was used to generate the present value for

the production. The present value of the oil and gas produced was used as the objective function for the nonlinear optimization algorithms.

Chapter 2 of this report describes the reservoir model used, Chapter 3 the vertical multiphase flow correlations used and Chapter 4 the separator model developed. Chapter 5 describes the nonlinear optimization algorithms used and Chapter 6 the results of the study.

Carroll [1990] studied the optimization of decision variables that were fixed with time. This particular study concentrates on the optimization of decision variables that change with time, i.e., variables that are optimized for a single or group of time steps.

2 Reservoir Model

2.1 Introduction

The behavior of the petroleum reservoir and the inflow performance relationship was simulated by using a model developed by Borthne [1986] at the Norwegian Institute of Technology. This chapter is a summary of the detailed analysis that appears in Borthne's thesis. Readers are referred to his thesis and that of Carroll [1990] for further details.

Developed for use in production optimization of a recently discovered reservoir, the model requires only very basic field parameters, and is able to compute a production forecast very rapidly.

Based on a material balance equation, which is coupled with a pseudopressure based inflow relationship, the model assumes that the reservoir is a "tank." The production from the reservoir is constrained by a minimum flowing wellhead pressure, and both, maximum and minimum flow rates of the preferred phase (oil).

A series of simplifying assumptions are made in deriving the model. They are:

- The reservoir is homogenous, isotropic, horizontal, cylindrical, and of uniform thickness.
- The reservoir is a single cell with no-flow boundaries.
- The reservoir is saturated with a single hydrocarbon phase, and an immobile connate water phase.
- The drive mechanism for the oil reservoir is solution-drive.
- During a time step, production occurs under pseudosteady state conditions, at a constant rate.
- Capillary pressure, gravity effects, and coning forces are negligible, and are therefore ignored.
- All wells are assumed to be equivalent, and are assumed to produce at the same average reservoir conditions, and the same inflow performance relationships.

2.2 Reservoir Material Balance

The mass conservation equation for flow in a porous medium can be written as:

$$\frac{\partial m_p}{\partial t} + \tilde{q}_p = -\nabla\left(\frac{k\rho\nabla p}{\mu}\right)V_b \qquad (2.1)$$

Where
- p = pressure
- k = permeability
- ρ = density
- μ = viscosity
- V_b = reservoir bulk volume
- $\tilde{q}_b$ = mass production rate from reservoir
- m_b = mass of oil and gas in reservoir
- t = time

If the reservoir is considered to be a single cell, then there is no mass flux across the boundaries and the right hand side of Equation 2.1 is zero. Discretizing the equation gives the following:

$$\Delta m_p + \tilde{q}_p \Delta t = 0 \qquad (2.2)$$

In order to use a realistic pressure behavior in the production system, the black oil model assumptions were assumed. The black oil model for phase behavior assumes that the reservoir oil will produce both oil and gas at the surface. Similarly, the reservoir gas is assumed to produce gas, as well as oil at the surface. The black oil PVT formulation is shown schematically below.

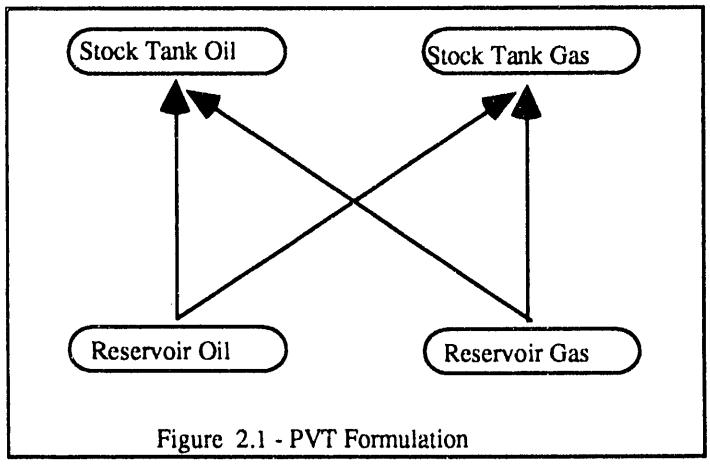

Figure 2.1 - PVT Formulation

The following definitions are used in the derivation of the material balance equations.

$$B_o = \frac{V_o^R}{V_{oo}^S} \tag{2.3}$$

$$B_g = \frac{V_g^R}{V_{gg}^S} \tag{2.4}$$

$$R_s = \frac{V_{go}^S}{V_{oo}^S} \tag{2.5}$$

$$r_s = \frac{V_{og}^S}{V_{gg}^S} \tag{2.6}$$

where
- B_o = formation volume factor for oil
- B_g = formation volume factor for gas
- R_s = solution gas / oil ratio in reservoir oil
- r_s = solution oil / gas ratio in reservoir gas
- V_o^R = volume of reservoir oil
- V_g^R = volume of reservoir gas
- V_{oo}^S = stock tank oil volume from flash separation of reservoir oil
- V_{og}^S = stock tank oil volume from flash separation of reservoir gas
- V_{gg}^S = stock tank gas volume from flash separation of reservoir gas
- V_{go}^S = stock tank gas volume from flash separation of reservoir oil

If all the oil and gas in the reservoir is separated at stock tank conditions (STC), the mass of oil will be m_o, and the mass of gas m_g. As all this oil and gas originates from the reservoir, the total oil and gas mass may be expressed as follows:

$$m_o = m_{oo} + m_{og} \tag{2.7}$$

and

$$m_g = m_{gg} + m_{go} \tag{2.8}$$

where
- m_g = total mass of gas at stock tank conditions
- m_o = total mass of oil at stock tank conditions
- m_{oo} = mass of oil at STC from separation of reservoir oil
- m_{og} = mass of oil at STC from separation of reservoir gas
- m_{gg} = mass of gas at STC from separation of reservoir gas
- m_{go} = mass of gas at STC from separation of reservoir oil

These masses can be expressed in terms of reservoir variables and fluid properties.

$$m_{oo} = \frac{\phi S_o \rho_{oo}^S}{B_o} V_b \tag{2.9}$$

$$m_{og} = \frac{\phi S_g r_s \rho_{og}^S}{B_g} V_b \tag{2.10}$$

$$m_{gg} = \frac{\phi S_g \rho_{gg}^S}{B_g} V_b \tag{2.11}$$

$$m_{go} = \frac{\phi S_o R_s \rho_{go}^S}{B_o} V_b \tag{2.12}$$

where S_x = saturation of the appropriate phase
ρ_{xx}^S = density of the appropriate phase at STC
ϕ = porosity

The oil production mass flux (mass per unit time) can be defined as follows.

$$\tilde{q}_o = \tilde{q}_{oo} + \tilde{q}_{og} \tag{2.13}$$

where $\tilde{q}_o$ = total mass flux of oil at the surface
$\tilde{q}_{oo}$ = mass flux of oil originating from reservoir oil
$\tilde{q}_{og}$ = mass flux of oil originating from solution in reservoir gas

In terms of volumetric flow rates and solubility,

$$\tilde{q}_{oo} = q_{oo} \rho_{oo}^S \tag{2.14}$$

$$\tilde{q}_{og} = q_{gg} r_s \rho_{og}^S \tag{2.15}$$

where q_{oo} = oil production rate from the reservoir oil, at standard conditions
q_{gg} = gas production rate from the reservoir gas, at standard conditions.

Equivalent equations can be written for the gas production rates. The total mass flux equation for gas production is:

$$\tilde{q}_g = \tilde{q}_{gg} + \tilde{q}_{go} \tag{2.16}$$

where $\tilde{q}_o$ = total mass flux of gas at the surface
$\tilde{q}_{gg}$ = mass flux of gas originating from reservoir gas
$\tilde{q}_{go}$ = mass flux of gas originating from solution in reservoir oil.

Substituting the appropriate equations and the PVT formulations into Equation 2.2, four mass balance equations can be derived.

$$\Delta\left(\frac{\phi S_o \rho_{oo}^S}{B_o}\right) + q_{oo}'' \rho_{oo}^S \Delta t = 0 \tag{2.17}$$

$$\Delta\left(\frac{\phi S_g r_s \rho_{og}^S}{B_g}\right) + q_{gg}'' r_s \rho_{og}^S \Delta t = 0 \tag{2.18}$$

$$\Delta\left(\frac{\phi S_g \rho_{gg}^S}{B_g}\right) + q_{gg}'' \rho_{gg}^S \Delta t = 0 \tag{2.19}$$

$$\Delta\left(\frac{\phi S_g R_s \rho_{og}^S}{B_o}\right) + q_{oo}'' R_s \rho_{go}^S \Delta t = 0 \tag{2.20}$$

where the double quote (") indicates that the variable has been divided by the reservoir bulk volume. For example, $q'' = q / V_b$.

Summing the two oil equations (2.17 and 2.18) and rearranging we obtain the following expression for oil production at the surface.

$$\Delta\left[\phi\left(\frac{S_o}{B_o} + \frac{S_g r_s}{B_g}\frac{\rho_{og}^S}{\rho_{oo}^S}\right)\right] + q_{oo}'' \Delta t + q_{gg}'' \Delta t\, r_s \frac{\rho_{og}^S}{\rho_{oo}^S} = 0 \tag{2.21}$$

Similarly, rearranging Equations 2.19 and 2.20, we obtain the following expression for gas production at the surface.

$$\Delta\left[\phi\left(\frac{S_g}{B_g} + \frac{S_o R_s}{B_o}\frac{\rho_{go}^S}{\rho_{gg}^S}\right)\right] + q_{gg}'' \Delta t + q_{oo}'' \Delta t\, R_s \frac{\rho_{go}^S}{\rho_{gg}^S} = 0 \tag{2.22}$$

In Equations 2.21 and 2.22 the oil and gas rates appear. Using Darcy's law the oil and gas rates can be written as:

$$q_{gg}'' B_g = C_1 \frac{k_{rg}}{\mu_g} \tag{2.23}$$

$$q''_{oo}B_o = C_2\frac{k_{ro}}{\mu_o} \tag{2.24}$$

where C_1 and C_2 are constants. The two constants are equal, if the capillary pressure is zero. Taking the gas - oil ratio (dividing Equation 2.23 by 2.24) and multiplying the result by Δt gives the following equation.

$$\frac{\Delta G''_p}{\Delta N''_p} = \frac{k_{rg}\mu_o B_o}{k_{ro}\mu_g B_g} \tag{2.25}$$

where $\Delta G''_p = q''_{gg}\Delta t$ which is the incremental gas produced during the last time step from reservoir gas, per unit volume of the reservoir.

and $\Delta N''_p = q''_{oo}\Delta t$ which is the incremental oil produced during the last time step from reservoir oil, per unit volume of the reservoir.

Substituting Equation 2.25 into Equations 2.21 and 2.22 gives the following,

$$\Delta\left[\phi\left(\frac{S_o}{B_o}+\frac{S_g r_s}{B_g}\frac{\rho^S_{og}}{\rho^S_{oo}}\right)\right]+\Delta N''_p\left[1+r_s\left(\frac{\rho^S_{og}}{\rho^S_{oo}}\right)\frac{k_{rg}\mu_o B_o}{k_{ro}\mu_g B_g}\right]=0 \tag{2.26}$$

$$\Delta\left[\phi\left(\frac{S_g}{B_g}+\frac{S_o R_s}{B_o}\frac{\rho^S_{go}}{\rho^S_{gg}}\right)\right]+\Delta N''_p\left[\frac{k_{rg}\mu_o B_o}{k_{ro}\mu_g B_g}+R_s\frac{\rho^S_{go}}{\rho^S_{gg}}\right]=0 \tag{2.27}$$

For a given time step Δt, Equations 2.26 and 2.27 can be solved using trial and error procedure to find the incremental oil produced during the time step.

2.3 Solution Procedure

1. Specify the oil rate q_o and time step length Δt.

2. Calculate incremental total oil production $\Delta N''_p$.

3. Assume average reservoir pressure, and calculate the pressure dependent properties - formation volume factors, solution gas ratios, viscosities and densities.

4. Calculate oil saturation from Equation 2.26. The term beginning with in Equation 2.26 is calculated using the properties calculated at the previous time step.

5. Calculate the gas saturation.

6. Calculate the relative permeability ratio as a function of gas saturation from the input table.

7. Calculate incremental oil and gas production.

8. Calculate the material balance error.

9. If the material balance is not satisfied then go to step 3 and try a new estimate for the average pressure.

2.4 Inflow Performance Relationship

A inflow performance relationships is used to calculate the flowing bottomhole pressure (FBHP) for the well. The calculations are based on equations which relate the average reservoir pressure, and fluid properties with the FBHP.

Darcy's law can be written as

$$q_{oo} = \frac{2\pi rhkk_{ro}}{\mu_o B_o}\frac{\partial p}{\partial r} \qquad (2.28)$$

separating and integrating Equation 2.28 gives:

$$q_{oo} = \frac{2\pi kh}{\ln(r_e/r_w)}\int_{p_w}^{p_e}\frac{k_{ro}}{\mu_o B_o}dp \qquad (2.29)$$

where
- k = absolute permeability
- k_{ro} = relative permeability to oil
- r_e = radius of the reservoir
- r_w = radius of the well
- h = height of the reservoir
- p_e = pressure at the external boundary of the reservoir
- p_{wf} = flowing bottomhole pressure
- q_{oo} = oil production rate from reservoir oil

Including the skin factor and assuming pseudosteady state flow conditions, Equation 2.28 can be written in terms of pseudopressure ($m(p)$) as:

$$q_{oo} = \frac{2\pi kh}{\ln(r_e/r_w) - 0.75 + S + Dq_{oo}}\left[m(p_r) - m(p_{wf})\right] \qquad (2.30)$$

where S = skin

D = rate dependent skin

and $$m(p) = \int_0^p \frac{k_{ro}}{\mu_o B_o} dp \tag{2.31}$$

By redefining the pseudopressure, Equation 2.30 can be expressed in terms of the oil production measured at the surface q_o:

$$q_o = \frac{2\pi kh}{\ln(r_e/r_w) - 0.75 + S + Dq_o}\left[m(p_r) - m(p_{wf})\right] \tag{2.32}$$

where $$m(p) = \int_0^p \left(\frac{k_{ro}}{\mu_o B_o} + \frac{k_{rg} r_s}{\mu_g B_g} \right) dp \tag{2.33}$$

A similar equation can be written for gas production.

$$q_g = \frac{2\pi kh}{\ln(r_e/r_w) - 0.75 + S + Dq_g}\left[m(p_r) - m(p_{wf})\right] \tag{2.34}$$

where $$m(p) = \int_0^p \left(\frac{k_{rg}}{\mu_g B_g} + \frac{k_{ro} R_s}{\mu_o B_o} \right) dp \tag{2.35}$$

The derivation above relates the flow equations to the pseudopressure functions. The pseudopressure is a function of both pressure and saturations. Assuming a constant producing gas - oil ratio, we can derive a relationship between saturation and pressure. During the discussion of the reservoir material balance equations, the producing gas - oil ratio R_p was approximated by:

$$R_p = \frac{\Delta G_p}{\Delta N_p} \tag{2.36}$$

Using the material balance equations to substitute for ΔN_p and ΔG_p, Equation 2.36 can be written as

$$R_p = \frac{\Delta N''_{oo}\left(M + R_s \frac{\rho^S_{go}}{\rho^S_{gg}} \right)}{\Delta N''_{oo}\left(1 + r_s \frac{\rho^S_{og}}{\rho^S_{oo}} M \right)} \tag{2.37}$$

where M is the mobility ratio. Solving for M gives

$$M = \frac{R_p - R_s \frac{\rho_{go}^s}{p_{gg}^s}}{1 - R_p r_s \frac{\rho_{og}^s}{\rho_{oo}^s}} \tag{2.38}$$

The mobility ratio is calculated from the material balance equations. For the mobility ratio, we can calculate the relative permeability ratio. As the permeability ratio is a function of saturation, we can calculate the phase saturation as a function of pressure for each time step. With the phase saturation, we can calculate the relative permeability of each phase. and use it in the calculation of the pseudopressure function.

The deliverability Equation (2.31), in terms of pseudopressure can be written as

$$\int_{p_{wf}}^{p_i} \left(\frac{k_{ro}}{\mu_o B_o} + \frac{k_{rg} r_s}{\mu_g B_g} \right) dp = q_o \frac{\ln(r_e/r_w) - 0.75 - S + Dq_o}{2\pi kh} \tag{2.39}$$

In this equation, the right hand side is a constant during a time step. The above equation is solved to find the FBHP (p_{wf}), which gives the value for the integral, equal to the value of the constant on the right hand side of the equation.

3 Tubing Model

3.1 Introduction

The flow of fluids from the reservoir to the wellhead through the completion string is modeled using the vertical multiphase flow correlations. For a more complete analysis of multiphase vertical flow the readers are referred to Brown [1977], Govier and Aziz [1972] and to Beggs [1992].

The law of conservation of energy is used as the theoretical basis for most flow correlations. For a steady state system, the energy balance equation may be written as

$$U_1 + p_1V_1 + \frac{mv_1^2}{2g_c} + \frac{mgZ_1}{g_c} + q' + W_s = U_2 + p_2V_2 + \frac{mv_2^2}{2g_c} + \frac{mgZ_2}{g_c} \qquad (3.1)$$

where
- U = internal energy
- pV = energy of expansion or compression
- q' = heat energy added to the system
- W_S = work done by the fluid
- mv^2 = kinetic energy
- mgZ = potential energy

Equation 3.1 can be modified to a pressure gradient form by using thermodynamic relations. For a fluid flowing in a pipe, the law of conservation of energy can be written as

$$\frac{dp}{dL} = \frac{g}{g_c}\rho\sin\theta + \frac{\rho v dv}{g_c dL} + \left(\frac{dp}{dL}\right)_f \qquad (3.2)$$

where
- ρ = density of the fluid
- θ = the angle of inclination from the horizontal
- g = gravitational acceleration
- g_c = gravitational constant
- v = fluid velocity
- L = length of the pipe
- $\left(\frac{dp}{dL}\right)_f$ is the pressure loss due to friction

As Equation 3.2 indicates, the total pressure drop can be thought of as composed of three distinct components - hydrostatic component, kinetic energy component, and a component due to friction. Symbolically this can be expressed as

$$\left(\frac{dp}{dL}\right)_{total} = \left(\frac{dp}{dL}\right)_{hydrostatic} + \left(\frac{dp}{dL}\right)_{kinetic} + \left(\frac{dp}{dL}\right)_{friction} \tag{3.3}$$

3.2 Two Phase Flow

When considering multiphase flow, the introduction of another phase into the flow steam complicates the analysis of the pressure gradient equation. The pressure gradient is changed by the presence of different phases, which in turn change the fluid properties such as densities, viscosities, surface tensions and flow velocities. To take into account these changes a set of variables are defined. A brief description of these variables follow.

Liquid Holdup H_L , defined as the fraction of the pipe element that is occupied by liquid

$$H_L = \frac{\text{Volume of Liquid in a Pipe Element}}{\text{Volume of the Pipe Element}} \tag{3.4}$$

The gas holdup is defined as

$$H_G = 1 - H_L \tag{3.5}$$

No-Slip Liquid Holdup λ_L, is defined as the ratio of the volume of the liquid in a pipe element that would exist if the gas and liquid traveled at the same velocity (no slippage) divided by the volume of the pipe element. It can be calculated from the gas and liquid velocities as

$$\lambda_L = \frac{q_L}{q_L + q_G} \tag{3.6}$$

Similarly, the no slip gas holdup factor is defined as

$$\lambda_G = 1 - \lambda_L \tag{3.7}$$

Density ρ_m , is used in calculating the kinetic and potential energy changes. Several different equations have been used to calculate the appropriate fluid density. The most commonly used equation is

$$\rho_m = \rho_L H_L + \rho_G H_G \tag{3.8}$$

in this case ρ_L is defined to be

$$\rho_L = \rho_o f_o + \rho_w f_w \tag{3.9}$$

where f_x represents the fractional flow of water or oil.

Velocity υ Many of the two phase flow correlations are based on the superficial velocity of a phase. The superficial velocity is defined as the velocity of the phase, if the phase was flowing through the total cross sectional area of the pipe. The superficial velocity of gas is defined as

$$v_{SG} = \frac{q_G}{A} \tag{3.10}$$

where v_{SG} = superficial velocity of gas
A = cross sectional area of the pipe
q_G = flowrate of gas

The actual gas velocity is calculated from

$$v_G = \frac{q_G}{AH_G} \tag{3.11}$$

Similar definitions can be applied to calculate the superficial velocity, and the actual velocity of the liquid phase.

The viscosity and surface tension of the fluid is calculated in a similar manner to the density calculation - (Equation 3.8), with the density terms replaced by the appropriate viscosity or surface tension terms.

There are many empirical multiphase correlations that have been developed to predict the pressure drop due to two phase flow in pipes. Most of these correlations use liquid holdup to accurately predict the pressure loss due to the potential energy term. This is accomplished by using flow regime maps. Depending on the superficial velocity of gas and liquid, distinct flow patterns are observed. Each of these flow patterns provides a different pressure drop associated with the potential energy and friction energy terms. The flow regimes are based on the fluid distribution in the pipe.

Descriptions of the major flow regimes are as follows:

Bubble Flow - The pipe is almost completely filled with liquid and the free gas phase is present as small bubbles. The bubbles flow at different velocities, and except for the density, have little effect on the pressure gradient.

Slug Flow - The gas phase is more pronounced. The gas bubbles coalesce and form plugs or slugs which almost fill the pipe cross section. The gas bubble velocity is greater than the liquid velocity. Both phases have a significant effect on the pressure drop.

Transitional Flow - Change from liquid phase to a continuous gas phase occurs. Gas bubbles join together, and the liquid may be entrained in the bubbles. Although the liquid phase effects are significant, the gas phase effects are dominant.

Mist Flow - The gas phase is continuous, and the liquid is entrained as droplets in the gas phase. Gas phase controls the pressure gradient.

3.3 Hagedorn and Brown Correlation

Hagedorn and Brown [1964] proposed a generalized correlation for multiphase vertical flow. This correlation was generated using data measured for a wide range of gas-liquid ratios, tubing sizes, and differing fluid properties.

The pressure drop equation can be written as

$$144\frac{\Delta p}{\Delta h}=\frac{g}{g_c}\left[\rho_L H_L+\rho_g(1-H_L)\right]\left\{1+\frac{f(v_{SL}+v_{SG})^2}{2g_c d}+\Delta\frac{\left[\frac{(v_{SL}+v_{SG})^2}{2g}\right]}{\Delta h}\right\} \quad (3.12)$$

where Δh = depth increment
d = pipe diameter
f = Moody friction factor

This correlation does not take into account the different flow regimes. In this case, the liquid hold up is used as a correlating parameter, in the calculation of the total pressure drop for an incremental length of pipe.

The calculation procedure for the Hagedorn and Brown [1964] correlation is described below.

1. An incremental pipe length Δh, is chosen.

2. Starting with a point at which the pressure is known, the incremental pressure is assumed and the arithmetic average pressure is calculated.

3. Specific gravity of the oil and gas are calculated.

4. The mass associated with one barrel of stock tank liquid is calculated as.

$$m = 350\gamma_o\left(\frac{1}{1+WOR}\right)+350\gamma_w\left(\frac{WOR}{1+WOR}\right)+0.0764\gamma_G\,GLR \qquad (3.13)$$

where WOR = water-oil ratio
GLR = gas-liquid ratio
γ_x = specific gravity

5. Calculate the mass flow rate.

6. Calculate the density, viscosity and surface tension of the liquid and gas phases.

7. Calculate the liquid viscosity number, N_L

$$N_L = 0.15726\mu_L\left(\frac{1}{\rho_L\sigma_L^3}\right)^{1/4} \qquad (3.14)$$

Interpolating from Table 3.1, Graph 1, calculate CN_L

8. Calculate the liquid velocity number, N_{LV}

$$N_{LV} = 1.938v_{SL}\left(\frac{\rho_L}{\sigma_L}\right)^{1/4} \qquad (3.15)$$

9. Calculate the gas velocity number, N_{GV}

$$N_{GV} = 1.938v_{SG}\left(\frac{\rho_L}{\sigma_L}\right)^{1/4} \qquad (3.16)$$

9 Calculate the pipe diameter number, N_D

$$N_D = 120.872\,d\sqrt{\rho_L/\sigma_L} \qquad (3.17)$$

10. Calculate the correlating parameter, Φ_a as in Graph 2 of Table 3.1. Interpolating Graph 2 find the value of ψ.

where $\Phi_a = \dfrac{N_{LV}N_L^{0.38}}{N_D^{2.14}}$ (3.18)

11. Calculate the correlating parameter Φ_β, as in Graph 3 of Table 3.1, and then interpolate to find the value of H_L/ψ, and hence the value of the liquid holdup, H_L.

$$\text{where } \Phi_\beta = \left(\frac{N_{LV}}{N_{GV}^{0.575}}\right)\left(\frac{p}{p_{SC}}\right)^{0.1}\left(\frac{CN_L}{N_D}\right) \tag{3.19}$$

Table 3.1: Correlating Functions of Hagedorn and Brown [1964] from Carroll [1990]

GRAPH 1		GRAPH 2		GRAPH 3	
N_L	CN_L	Φ_α	ψ	Φ_β	H_L/ψ
0.002	0.0019	0.010	1.00	0.20	0.04
0.005	0.0022	0.020	1.10	0.50	0.09
0.010	0.0024	0.025	1.23	1.00	0.15
0.020	0.0028	0.030	1.40	2.00	0.18
0.030	0.0033	0.035	1.53	5.00	0.25
0.060	0.0047	0.040	1.60	10.00	0.34
0.100	0.0064	0.045	1.65	20.00	0.44
0.150	0.0080	0.050	1.68	50.00	0.65
0.200	0.0090	0.060	1.74	100.00	0.82
0.400	0.0115	0.070	1.78	200.00	0.92
		0.080	1.80	300.00	0.96
		0.090	1.83	1000.00	1.00

12. Calculate the Reynolds number N_{Re}, and the friction factor f.

$$N_{Re} = \frac{2.2 \times 10^{-2} q_o m}{\left(\mu_L^{H_L}\right)\left(\mu_L^{(1-H_L)}\right) d} \tag{3.20}$$

and $$\frac{1}{\sqrt{f}} = 1.74 - 2\log\left(\frac{2\varepsilon}{d} + \frac{18.7}{N_{Re}\sqrt{f}}\right) \tag{3.21}$$

where ε = roughness of the pipe

Equation 3.21 has to be solved using a trial and error procedure. Then using Equation 3.12 the value of Δh can be calculated.

The correlation implemented contains a modified form of the Hagedorn and Brown [1964] correlation. If the flow regime is found to be in bubble flow, the Griffith and Wallis [1961] correlation is used. The procedure followed is as follows.

1. Calculate the values of A and B.

$$A = 1.071 - \left[0.2218(v_{SL} + v_{SG})^2\right] / d \tag{3.22}$$

$$B = \frac{v_{SG}}{v_{SL} + v_{SG}} \tag{3.23}$$

If B is greater than or equal to A, then continue with Hagedorn and Brown correlation. If A is greater than B, the flow is in the bubble flow regime and the Griffith and Wallis [1961] correlation is used.

2. Griffith and Wallis [1961] used the following equation to calculate the gas holdup H_G

$$H_G = \frac{1}{2}\left[1 + \frac{q_L}{v_S A} - \sqrt{(1 + q_L/v_S A)^2 - \frac{4q_G}{v_S A}}\right] \tag{3.24}$$

where v_s = is the slip velocity, average value of 0.8 ft/sec.

With this value of gas holdup computed the rest of the computations are carried on in the same manner as in the Hagedorn and Brown correlation.

3.4 Aziz, Govier and Fogarasi Correlation

Aziz, Govier and Fogarasi [1972] proposed a method which was flow regime dependent. The flow regimes are defined using the following variables.

$$N_x = v_{SG}\left(\frac{\rho_G}{0.0764}\right)^{1/3}\left(\frac{72\rho_L}{62.4\sigma_L}\right)^{1/4} \tag{3.25}$$

$$N_y = v_{SL}\left(\frac{72\,\rho_L}{62.4\,\sigma_L}\right)^{1/4} \tag{3.26}$$

$$N_1 = 0.51(100N_y)^{0.172} \tag{3.27}$$

$$N_2 = 8.6 + 3.8N_y \tag{3.28}$$

$$N_3 = 70(100N_y)^{-0.152} \tag{3.29}$$

Bubble Flow - the case when

$$N_x < N_1$$

then the liquid holdup is calculated from

$$H_L = 1 - \frac{v_{SG}}{v_{bf}} \tag{3.30}$$

where v_{bf} = the bubble rise velocity in the flowing stream

$$v_{bf} = 1.2v_m + v_{bs} \tag{3.31}$$

and $$v_{bs} = 1.41\left[\frac{\sigma_L g(\rho_L - \rho_G)}{\rho_L^2}\right]^{1/4} \tag{3.32}$$

$$v_m = v_{SG} + v_{SL} \tag{3.33}$$

The hydrostatic head is then calculated by

$$\left(\frac{dp}{dL}\right)_{hs} = \frac{g}{g_c}\rho_m \tag{3.34}$$

The friction pressure drop is calculated as

$$\left(\frac{dp}{dL}\right)_f = \frac{f\rho_m v_m^2}{2g_c d} \tag{3.35}$$

In bubble flow the acceleration term is considered to be negligible. The summation of the friction pressure drop and the hydrostatic pressure drop gives the total pressure drop.

Slug Flow - The limits for this flow regime is defined as follows,

$$N_1 < N_x < N_2 \qquad for \quad N_y < 4$$

and $$N_1 < N_X < 26.5 \qquad for \quad N_y \geq 4$$

The liquid holdup is calculated as in Equation 3.30, and the bubble rise velocity as in Equation 3.31. In this case v_{bs} is defined as

$$v_{bs} = C\left[\frac{gd(\rho_L - \rho_G)}{\rho_L}\right]^{1/2} \tag{3.36}$$

and $$C = 0.345\left[1 - EXP(-0.029N_V)\right]\left[1 - EXP\left(\frac{3.37 - N_E}{m}\right)\right] \tag{3.37}$$

where $$N_E = \frac{gd^2(\rho_L - \rho_G)}{\sigma_L} \tag{3.38}$$

$$N_V = \frac{\left[d^3 g(\rho_L - \rho_G)\right]^{1/2}}{\mu_L} \tag{3.39}$$

In the slug flow regime the acceleration pressure drop is considered to be negligible. The total hydrostatic pressure drop is calculated as in Equation 3.34 and the friction pressure drop as

$$\left(\frac{dp}{dL}\right)_f = \frac{f\rho_L H_L v_m^2}{2g_c d} \tag{3.40}$$

Summation of the hydrostatic and friction pressure losses provides the total pressure loss.

Transition Flow - This flow regime exists when

$$N_2 < N_X < N_3 \qquad for \quad N_y < 4$$

When the flow regime fall in this transition zone, the total pressure drop is obtained by interpolating between the total pressure drops in the slug flow regime and the mist flow regime. The pressure drop can be expressed as

$$\left(\frac{dp}{dL}\right)_{Total} = A\left(\frac{dp}{dL}\right)_{Slug} + B\left(\frac{dp}{dL}\right)_{Mist} \tag{3.41}$$

where A and B represent weighting factors.

Mist Flow - This flow regime is encountered when

$$N_x > N_3 \qquad For \quad N_y < 4$$

and $$N_X > 26.5 \qquad For \quad N_y > 4$$

Aziz, Govier and Fogarasi [1972] use the procedure of Duns and Ros [1963]. Duns and Ros [1963] assumed that with high gas flow rates in this region, that the slip velocity would be zero. The mixture density is calculated from the no-slip velocities as

$$\rho_m = \rho_L \lambda_L + \rho_G \lambda_G \tag{3.42}$$

The hydrostatic head is calculated from Equation 3.34 with the density of the mixture from Equation 3.42.

The frictional pressure drop is calculated from

$$\left(\frac{dp}{dZ}\right)_f = \frac{f\rho_G v_{SG}^2}{2g_c d} \tag{3.43}$$

The friction factor is a function of the Reynolds number of the gas, which can be expressed as

$$N_{Re} = \frac{\rho_G v_{SG} d}{\mu_G} \tag{3.44}$$

Duns and Ros [1963] also accounted for the increase in roughness of the wall due to the presence of the liquid film. The ripples of liquid on the wall causes a drag on the gas. These processes is governed by the Weber number N_{We} and the liquid viscosity number N_μ.

$$N_{We} = \frac{\rho_G v_{SG}^2 \varepsilon}{\sigma_L} \tag{3.45}$$

$$N_{\mu} = \frac{\mu_L^2}{\rho_L \sigma_L \varepsilon} \tag{3.46}$$

The following equations provide the corrected values for the pipe roughness. For values of $N_{We} N_{\mu} \leq 0.05$

$$\frac{\varepsilon}{d} = \frac{0.0749 \sigma_L}{\rho_G v_{SG}^2 d} \tag{3.47}$$

For values of $N_{We} N_{\mu} > 0.05$

$$\frac{\varepsilon}{d} = \frac{0.3713 \sigma_L}{\rho_G v_{SG}^2 d} \left(N_{We} N_{\mu} \right)^{0.302} \tag{3.48}$$

Using the above relations the frictional pressure drop can be calculated.

The pressure drop due to acceleration can be included by defining an acceleration term E_K as

$$E_K = \frac{v_m v_{SG} \rho_{ns}}{g_c p} \tag{3.49}$$

The total pressure drop can be calculated as

$$\left(\frac{dp}{dL} \right)_{total} = \frac{\left(\frac{dp}{dL} \right)_{hydrostatic} + \left(\frac{dp}{dL} \right)_{friction}}{1 - E_K} \tag{3.50}$$

4. Separator Model

4.1 Introduction

Produced reservoir fluids are complex mixtures of hydrocarbons with different physical characteristics. The produced fluids separate into two phases - oil and gas. The separation of the oil from the gas at the surface is accomplished by conventional stage separation.

Stage separation is the process by which gaseous and liquid hydrocarbons are flashed (separated) into vapor and liquid streams by using two or many separators. As Figure 4.1 illustrates, the separation process can use as many separators as is desired. As the number of stages increases, the pressure of the separators is dropped slowly and the flashing process mimics differential liberation. Differential liberation is the "best" separation process as it leaves the maximum quantity of the valuable liquid phase.

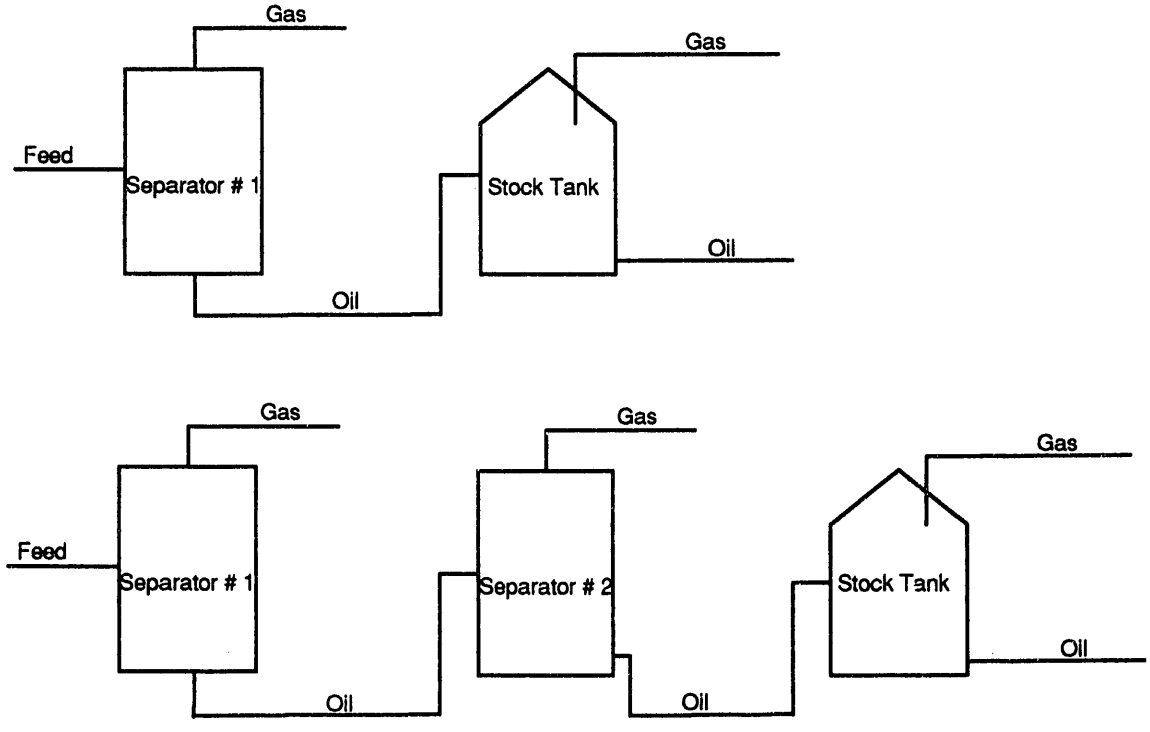

Figure 4.1: Schematic illustration of a two and three stage separation process (Ahmed [1989])

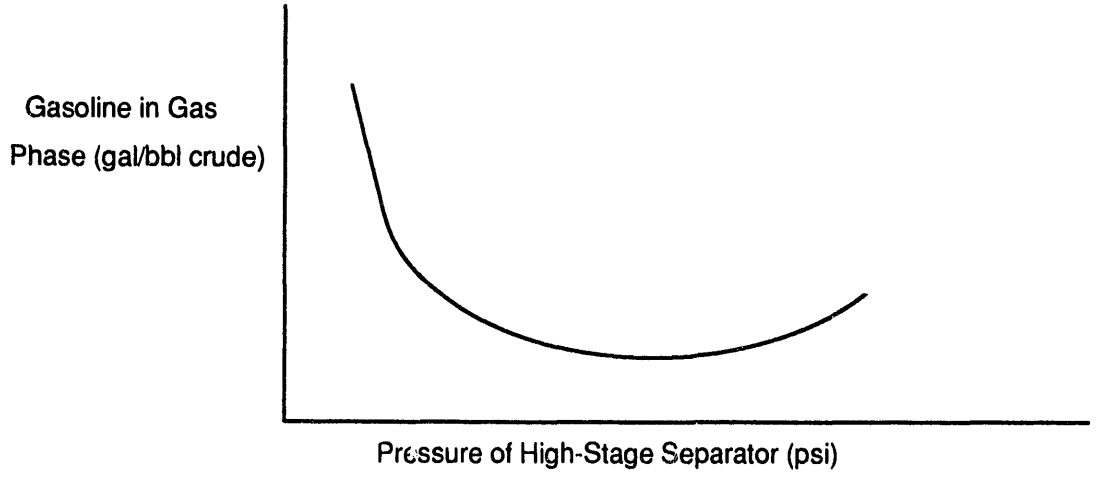

Figure 4.2: Gasoline Content of Gas Phase as a Function of High Stage Separator Pressure

Figure 4.2 shows the gasoline that is retained in the vapor phase as a function of separator pressure. The goal in stage separation is to flash the feed stream at that pressure which results in the least loss of gasoline and other valuable hydrocarbons to the gas phase (Chilingarian, Robertson and Kumar [1987]). The separation process is modeled by using a flash calculation. This chapter describes the flash process that was implemented in this study.

4.2 Flash Calculation

The oil and gas mixture that flows into a separator has a fixed overall composition. In the separator it is flashed at a fixed temperature and pressure. The flash calculation mimics this phase behavior and is used to determine the new composition of the oil and gas streams.

The following procedure from Orr [1991] is used to determine the phase compositions.

1. For a flash of a given overall composition, pressure and temperature, guess the resulting composition of the liquid stream x_i and the gas stream y_i

2. Calculate the equation of state parameters

3. Solve the equation of state and determine the molar volumes of the gas and liquid phases.

4. Calculate the partial fugacities of the liquid and gas phases.

5. Check if the partial fugacities of all the phases are equal. If the partial fugacities are equal, then the equilibrium composition of the output streams have been calculated.

6. If the fugacities are not equal, then improve the estimates of the composition of the phases and return to step 2.

A detailed description of each of the steps described above appears in Section 4.3 through to Section 4.8.

4.3 Initial Estimation of the Compositions

As a first guess, the Wilson and Deal [1962] equation is used to estimate the value of the equilibrium ratio k_i.

$$k_i = \frac{y_i}{x_i} = \frac{\exp\left[5.37(1+\omega_i)\left(1-\frac{1}{T_{ri}}\right)\right]}{P_{ri}} \tag{4.1}$$

where ω_i = accentric factor, available from the literature
T_{ri} = Reduced Temperature $(= T/T_{critical})$
p_{ri} = Reduced Pressure $(= p/p_{critical})$

A material balance on component i gives

$$z_i = Lx_i + (1-L)y_i \tag{4.2}$$

where L = liquid mole fraction

Using the definition for the equilibrium ratio from Equation 4.1, Equation 4.2 can be rewritten as

$$x_i = \frac{z_i}{L+(1-L)k_i} \tag{4.3}$$

and $$y_i = \frac{k_i z_i}{L+(1-L)k_i} \tag{4.4}$$

The solutions to Equations 4.3 and 4.4 are constrained by the following relation

$$\sum_i x_i - \sum_i y_i = 0 \tag{4.5}$$

Therefore, a solution to the following equation must be found

$$F(L) = \sum_i \frac{z_i(1-k_i)}{k_i + (1-k_i)L} \tag{4.6}$$

This equation can be solved to find the value of L using a Newton-Raphson iteration scheme.

$$L^{k+1} = L^k - \frac{F(L^k)}{\left.\frac{dF}{dL}\right|_{L^k}} \tag{4.7}$$

Convergence is assumed when both of the following conditions are met.

$$\left|L^{k+1} - L^k\right| < \varepsilon$$

and $$F(L^{k+1}) < \varepsilon$$

where ε = a small number, say 10^{-6}

4.4 Calculation of Equation of State (EOS) Parameters

The Soave-Redlich-Kwong (SRK) equation of state is used in this study. The SRK equation of state can be written as

$$p = \frac{RT}{V - b_i} - \frac{a_i \alpha_i}{V(V + b_i)} \tag{4.9}$$

This equation can be written in cubic form as

$$V^3 - \left[\frac{RT}{p}\right]V^2 + \left[\frac{(a\alpha)_m}{p} - \frac{b_m RT}{p} - b_m^2\right]V - \left[\frac{(a\alpha)_m b_m}{p}\right] = 0 \tag{4.10}$$

where V = Molar volume
R = Universal gas constant

and $$a_i = 0.42747\frac{R^2 T_c^2}{p_c} \tag{4.11}$$

$$b_i = \frac{0.08664\, RT_c}{p_c} \tag{4.12}$$

$$\alpha_i = \left[1 + \left(0.480 + 1.574\omega_i - 0.176\omega_i^2\right)\left(1 - T_{ri}^{0.5}\right)\right]^2 \tag{4.13}$$

$$(a\alpha)_m = \sum_i \sum_j \left[x_i x_j \left(a_i a_j \alpha_i \alpha_j\right)^{0.5} \left(K_{ij} - 1\right)\right] \tag{4.14}$$

$$b_m = \sum_i x_i b_i \tag{4.15}$$

K_{ij} is an empirically determined correction factor known as a binary interaction parameter, characterizing the binary formed by component i and j in the hydrocarbon mixture. The binary interaction parameters are used to model the intermolecular interaction through empirical adjustment.

4.5 Calculation of Liquid and Vapor Molar Volumes

The cubic form of the SRK equation of state (Equation 4.10) must be solved to find the molar volumes of the liquid and vapor. The roots can be calculated analytically or numerically using a Newton-Raphson iteration scheme. The analytical solution method is implemented in this study. Detailed explanation of the technique can be found in the notes of Orr [1991], and in the thesis by Carroll [1990].

4.6 Determining the Partial Fugacities

The partial fugacity $\hat{f}_i$, of a component can be calculated from the following relation

$$\hat{f}_i = p\, EXP\left[\int_0^p \left(\frac{Z-1}{p}\right) dp\right] \tag{4.16}$$

For the SRK equation of state the partial fugacity $\hat{f}_i$, of component i may be calculated from

$$\hat{f}_i = p \exp\left[\left(\frac{V}{V-b_m}\right) + \frac{b_i}{V-b_m} + \frac{2\sum x_i a_i a_j}{RTb_m} \ln\left(\frac{V+b_m}{V}\right) + \frac{(a\alpha)_m b_i}{RTb_m^2}\left(\ln\left(\frac{V+b}{V}\right) - \frac{b_m}{V+b_m}\right) - \ln\left(\frac{pV}{x_i RT}\right)\right] \tag{4.17}$$

4.7 Convergence Checks

The phase compositions and the molar volumes are found when the partial fugacities of each the components are the same, for all of the phases present. In this case, convergence is assumed when

$$\left| \frac{\hat{f}_i^V}{\hat{f}_i^L} - 1 \right| \leq \varepsilon \tag{4.18}$$

4.8 Modification of the Phase Compositions

The new values for the equilibrium ratios are found by a substitution scheme described below. Readers are referred to the work of Orr[1991] and that of Carroll [1990] for further detail.

$$k_i^{k+1} = \frac{\hat{f}_i^L}{\hat{f}_i^V} k_i^k \tag{4.19}$$

With the new values for the phase compositions the procedure is restarted (Section 4.3).

Nonlinear Optimization Algorithms

5.1 Introduction

Nonlinear optimization is concerned with finding the minimum or maximum of nonlinear functions. For example, when designing an aircraft an engineer may try to minimize the fuel consumption. This is known as the objective, and the mathematical statement which describes the behavior of the objective as a function of the chosen variables is known as the objective function. In this case, the fuel consumption is a function of many of the aircraft design variables such as wing span, size of the fuselage, size of engines used etc., which are known as the decision variables.

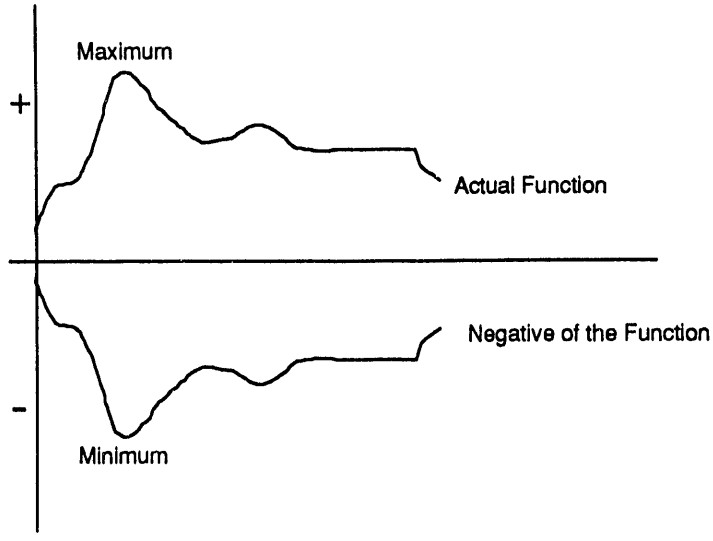

Figure 5.1 - Shows how the actual function and its negative can be used to find the minimum and the maximum of the function

As Figure 5.1 illustrates, maximization can be thought of as the minimization of the negative of the objective function. Nonlinear optimization algorithms are designed to find the minimum of objective functions. Therefore, in the rest of this discussion concentrates on the minimization of objective functions.

Nonlinear optimization algorithms can be separated in to two major classes. One, the algorithms based on gradients of the objective function such as those based Newton's method. Second, the algorithms based on function value comparisons such as the polytope algorithm and other searching techniques.

Readers are referred to Gill, Murray and Wright [1981] and to Scales [1985] for a comprehensive treatment of nonlinear optimization.

There are many techniques available to minimize an arbitrary function $F(\tilde{x})$ of n independent variables. Before describing these techniques, let me define some commonly used notation.

The objective function $F(x_1, x_2, x_3, x_4, \ldots\ldots x_n)$ will be denoted by $F(\tilde{x})$ where the variables are gathered together in a vector $\tilde{x}$

$$\tilde{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \qquad \text{and} \quad \tilde{p} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix} \tag{5.1}$$

All methods used to find the extrema of nonlinear functions are iterative. At the start of the k-th iteration, the current estimate of the minimum will be denoted by $\tilde{x}_k$. The k-th iteration then consists of computing the search vector $\tilde{p}_k$, from which the new estimate for the minimum $\tilde{x}_{k+1}$ is found according to the equation

$$\tilde{x}_{k+1} = \tilde{x}_k + \alpha_k \tilde{p}_k \tag{5.2}$$

where α_k is a scalar, and is obtained by a line search or is based on the theory of the method being used. It is the method of determining $\tilde{p}_k$ that largely distinguishes one method from another.

5.2 Method of Steepest Descent

Gradient methods involve searching along the vector $\tilde{p}_k$ from the current point $\tilde{x}_k$. The requirement is that α_k be chosen so that

$$F_{k+1} < F_k \qquad \text{where} \quad F_k = F(\tilde{x}_k) \tag{5.3}$$

It is not always possible to find a vector $\tilde{p}_k$ which satisfies Equation 5.3. The basic idea in gradient based algorithms is to find the descent direction. The descent direction is given by the vector $\tilde{p}_k$. With the descent direction known, a very small, positive value of α_k. will satisfy Equation 5.3.

Most of the techniques used to find the extrema of the functions use the first and second derivatives of the functions to compute $\tilde{p}_k$. The derivatives may be calculated analytically or approximated by using finite difference algorithms.

Using a Taylor series expansion F_{k+1}, can be written as

$$F_{k+1} = F(\tilde{x}_k + \alpha_k \tilde{p}_k) \approx F_k + \tfrac{1}{2}\alpha_k^2 \tilde{p}_k^T G_k \tilde{p}_k \tag{5.4}$$

In this case all the terms of order three or more have been ignored. The form indicated in Equation 5.4 is the quadratic approximation to the real function.

The term G_k is the Hessian Matrix of the function $F(\tilde{x})$ at the k-th iteration, and can be written as

$$G_k = \begin{bmatrix} \partial^2 F/\partial x_1^2 & \cdots & \cdots & \partial^2 F/\partial x_1 \partial x_n \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ \partial^2 F/\partial x_n \partial x_1 & \cdots & \cdots & \partial^2 F/\partial x_n^2 \end{bmatrix} \tag{5.5}$$

In order to satisfy Equation 5.3, $\tilde{p}_k$ must be chosen such that

$$\tilde{p}_k^T G_k \tilde{p}_k < 0 \tag{5.6}$$

The calculation of the first and second derivatives of the objective function $F(\tilde{x})$ may be carried out either analytically or numerically. The numerical, finite difference techniques used to approximate the values of the derivatives are based on Taylor series expansions. Readers are referred to Abramowitz and Stegun [1965] for a detailed description of the available numerical approximations.

Equation 5.4 is a second order approximation to the function $F(\tilde{x})$ at $\tilde{x}_k$. A first order approximation can be written as

$$F_{k+1} = F(\tilde{x}_k + \alpha_k \tilde{p}_k) \approx F_k + \alpha_k \tilde{g}_k^T \tilde{p}_k \tag{5.7}$$

If the condition in Equation 5.3 is to be satisfied

$$\tilde{g}_k^T \tilde{p}_k < 0 \tag{5.8}$$

$$\text{where } \tilde{g}_k = \begin{bmatrix} \partial F/\partial x_1 \\ \partial F/\partial x_2 \\ \vdots \\ \partial F/\partial x_n \end{bmatrix} \tag{5.9}$$

Equation 5.7 can be written in the form

$$F_{k+1} - F_k \approx \alpha_k \|\tilde{g}_k\| \|\tilde{p}_k\| \cos\theta \tag{5.10}$$

Now if all values in the right hand side of this equation are held constant, and only the angle θ is allowed to change, the equation will be most negative (< 0) when $\theta = \pi$. Thus, for a small value of α_k, the largest reduction in the value of the function will be in the direction

$$\tilde{p}_k = -\tilde{g}_k \tag{5.11}$$

This direction is known as *steepest descent direction,* and gives rise to the method of *steepest descent.*

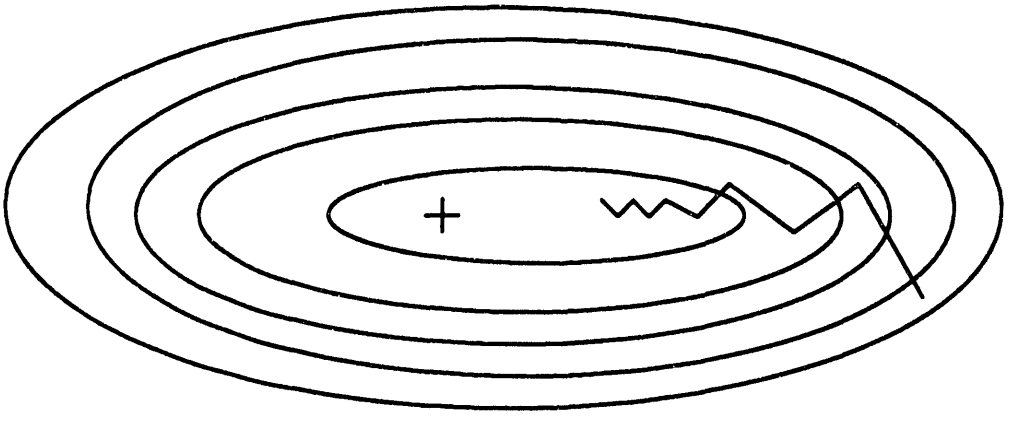

Figure 5.2: The steepest decent method showing the zig-zag progress.

In the case of this algorithm, the successive search directions are orthogonal, and as a result the iterations generate a sequence of points which follow a zig-zag path to the minimum. This results in very small steps being taken, which results in very slow convergence to the minimum.

The *Steepest decent* algorithm is very stable, and is theoretically simple. However, due to the slow rate of convergence, it is not used in optimization, but is the fundamental component of more elaborate techniques.

5.3 Newton's Method

Newton's method is designed to achieve quadratic termination with a positive definite Hessian matrix. Mathematically, *Newton's method* can be represented as follows. The minimum of a function occurs when

$$\nabla F = 0 \quad \text{and} \quad \nabla^2 F = 0 \tag{5.12}$$

The gradient of the function can be written as

$$\nabla F = \tilde{g}_{k+1} = \tilde{g}(\tilde{x}_k + \tilde{p}_k) = \tilde{g}_k + G_k \tilde{p}_k \tag{5.13}$$

This is the Taylor series expansion for the gradient of the quadratic function at $\tilde{x}_{k+1}$. The terms of order greater than two are neglected. If $\tilde{x}_{k+1}$ is the minimum of the function, then $\tilde{g}_{k+1}$ is equal to zero, and therefore

$$\tilde{p}_k = -G_k^{-1}\tilde{g}_k \tag{5.14}$$

For non-quadratic functions, $\tilde{x}_k + \tilde{p}_k$ will be in general an approximation to the minimum, and so an iterative process is required to find the minimum of the function. Note that in *Newton's method* α_k, the step length, is unity.

The key to finding the minimum of the function is the Hessian matrix G_k. If the function has a minimum, the Hessian matrix must be positive definite[1]. Problems arise with *Newton's method* when the calculated Hessian matrix is indefinite. This implies that the quadratic model for the actual function is neither bounded from above or below. There are a series of methods developed, known as *modified Newton methods* which overcome this problem. These methods are all based on constructing a new positive definite matrix $\overline{G}_k$, which is based on the Hessian matrix and is used to replace G_k in Equation 5.14.

The Hessian matrix can be decomposed, and written as the product of three matrices as shown in Equation 5.15. This particular decomposition technique is known as spectral decomposition.

$$G_k = U\Lambda U^T = \sum_{i=1}^{n} u_i \lambda_i u_i^T \tag{5.15}$$

where
- u_i = the eigenvectors of G_k
- λ_i = the eigenvalues of G_k
- U = the matrix composed of the eigenvectors of G_k
- Λ = the matrix composed of the eigenvalues of G_k on the diagonal, and zero's elsewhere.

---

[1] If all the eigenvalues of a symmetric matrix are strictly positive, then the matrix is said to be *positive definite*. Further, if A is a *positive definite* matrix, then for any non-zero vector x,

$$\tilde{x}^T A \tilde{x} > 0$$

There is a corresponding definition for a *negative definite* matrix.

One of the methods of ensuring positive definiteness is to add a small quantity, μ_k to each eigenvalue, ensuring that all the new eigenvalues are positive. This technique, and modification to it, were the work of Marquardt [1963], Goldfeld, Quandt, and Trotter [1966], and Greenstadt [1967]. This process can be represented in matrix notation as

$$\overline{G}_k = G_k + \mu_k I \tag{5.16}$$

The modified Hessian matrix is used in an iterative process, solving Equation 5.14 to find the minimum of the function under consideration.

The method of Gill and Murray [1974] uses the *Cholesky factorization* of G_k to modify *Newton's method.* The *Cholesky factorization* can be represented as

$$G_k = L_k D_k L_k^T \tag{5.17}$$

In this case D_k is a diagonal matrix and L_k is a lower triangular matrix with diagonal elements equal to one. The values of D_k (elements d_{ij}) and L_k (elements l_{ij}) can be calculated using the following expressions (g_{ij} are elements of G_k)

$$d_{jj} = g_{jj} - \sum_{q=1}^{j-1} d_{qq} l_{jq}^2 \tag{5.18}$$

$$l_{ij} = \left(g_{ij} - \sum_{q=1}^{j-1} d_{qq} l_{iq} l_{jq} \right) \Big/ d_{jj} \qquad for \quad i = j+1, n \tag{5.19}$$

Equation 5.14 can be written in terms of the *Cholesky factorization* as two systems of linear equations, which can easily be solved.

$$t_k = \quad -\overline{g}_k L_k^T \tag{5.20}$$

$$L_k^T p_k = D_k^{-1} t_k \tag{5.21}$$

In the Gill-Murray [1974] process, if any of the values of d_{ij} are less than a small positive constant δ, then a positive quantity r_{jj} is added to g_{ij}, and the *Cholesky factorization* is continued. The resulting matrices are then used in the minimization scheme as defined in Equations 5.20 and 5.21.

5.4 Quasi-Newton Methods

There are some nonlinear optimization algorithms which have the property of quadratic termination but do not use the second derivatives of the function. These methods sacrifice the speed of convergence, but save on the expense associated with calculating a new Hessian matrix for each iteration. During each iteration the Hessian

matrix is updated, and this new approximation to the Hessian is used to find the estimate to the minimum value.

The Newton iteration equation (Equation 5.14) can be written as

$$\tilde{p}_k = -H_k \tilde{g}_k \qquad \left(H_k \approx G_k^{-1}\right) \tag{5.22}$$

During each iteration H_k is updated by

$$H_{k+1} = H_k + Q_k^H \tag{5.23}$$

where Q_k^H is a updating matrix. Q_k^H is chosen such that the *quasi Newton* condition is satisfied. The *quasi Newton* condition can be stated as

$$H_{k+1}\left(\tilde{g}_{k+1} - \tilde{g}_k\right) = \rho_k\left(\tilde{x}_{k+1} - \tilde{x}_k\right) \tag{5.24}$$

where ρ_k is a scalar.

Combining Equation 5.23 and 5.24 results in the following equation

$$Q_k^H \Delta\tilde{g}_k = \rho_k \Delta\tilde{x}_k - H_k \Delta\tilde{g}_k \tag{5.25}$$

The solution to this equation can be written as

$$Q_k^H = \rho_k \frac{\Delta\tilde{x}_k \tilde{y}_k^T}{\tilde{y}_k^T \Delta\tilde{g}_k} - \frac{H_k \Delta\tilde{g}_k \tilde{z}_k^T}{\tilde{z}_k^T \Delta\tilde{g}_k} \tag{5.26}$$

In this case z_k and y_k are arbitrary vectors. If the following conditions are set

$$\rho_k = 1$$

$$\tilde{y}_k = \Delta\tilde{x}_k$$

and $\quad \tilde{z}_k = H_k \Delta\tilde{g}_k$

then the updating equation for the new Hessian, Equation 5.23 can be written as

$$H_{k+1} = H_k + \frac{\Delta\tilde{x}_k \Delta\tilde{x}_k^T}{\Delta\tilde{x}_k^T \Delta\tilde{g}_k} - \frac{H_k \Delta\tilde{g}_k \Delta\tilde{g}_k^T H_k}{\Delta\tilde{g}_k^T H_k \Delta\tilde{g}_k} \tag{5.27}$$

This is known as the Davidon-Fletcher-Powell (DFP) formula (Davidon [1959], Fletcher and Powell [1963]). This was the first form of the *quasi Newton* method used widely in function minimization.

Another formulation widely used today is the BFGS formula. This was first suggested by Broyden [1970], Fletcher [1970], Goldfarb [1970] and Shanno [1970]. It can be expressed as

$$H_{k+1} = \left[I - \frac{\Delta\tilde{x}_k \Delta\tilde{g}_k^T}{\Delta\tilde{x}_k^T \Delta\tilde{g}_k} \right] H_k \left[I - \frac{\Delta\tilde{x}_k \Delta\tilde{g}_k^T}{\Delta\tilde{x}_k^T \Delta\tilde{g}_k} \right]^T + \frac{\Delta\tilde{x}_k \Delta\tilde{x}_k^T}{\Delta\tilde{x}_k^T \Delta\tilde{g}_k} \tag{5.28}$$

where I is the identity matrix.

The convergence rate when using the *quasi Newton* algorithms is between linear and quadratic. The advantage of the method is the reduction in computation required to generate the new Hessian matrix for each iteration. It is expected that the reduction in the efficiency of convergence is offset by the more efficient computation of the Hessian matrix, and hence the overall efficiency of the method is greater than that of *Newton's method.*

5.5 Polytope Algorithm

The *polytope* method, also known in the literature as the simplex method, is based on function comparison techniques. It does not require the use of function gradients, and can be used on both smooth and non smooth functions.

The *polytope* algorithm (Gill, [1983]) is a fairly robust function comparison algorithm. For a problem of n decision variables, a polytope of $n+1$ points is created. The objective function is evaluated at each of the $n+1$ points. The point with the highest function value (known as the worst point) is chosen, and a new point is generated to replace this point. The method of choosing the new point is as follows.

Using the best n points (excluding the point with the highest function value), a centroid c is generated.

$$c = \frac{1}{n}\sum_{i=1}^{n} \tilde{x}_i \tag{5.29}$$

The centroid is then used to generate the reflected point.

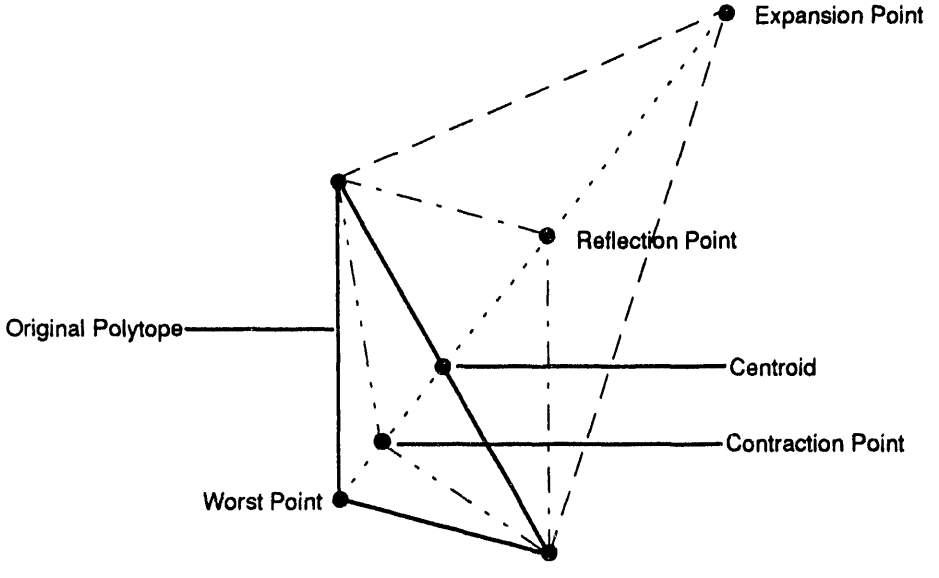

Figure 5.3: Two dimensional Polytope with the reflection, Expansion and Contraction Points. After Carroll [1990] and Gill, Murray and Wright [1981].

The reflected point is generated from the equation

$$x_r = c + \beta(c - x_{n+1})$$

where x_r = reflected point
x_{n+1} = Worst point
c = Centroid
β = reflection coefficient

The function is evaluated at x_r, and there are three cases to consider.

1. The function value falls within the set of existing n function values, (i.e. the new point is neither the best or the worst point). In this case x_r replaces x_{n+1} and the next iteration is begun.

2. The function value is the new best point. In this case, the assumption that the direction of reflection is "good" is made, and the polytope is expanded in this direction - a reflection coefficient of greater than one is used. If the new expanded point is "better" than the reflected point, the expanded point is accepted as the new best point, and the next iteration is started. If however the expanded point is not "good", the original reflected point is accepted and a new iteration is started.

3. If the reflected function value is greater than all the existing points, then the polytope is thought to be too large, and a contraction is made by setting the reflection coefficient to less than one. If the contraction succeeds, the new value is accepted and a new iteration is started. Otherwise another contraction is carried out.

After several cycles of reflection the existing polytope is discarded and a new polytope is constructed using the best two points. The other points of the polytope are spaced on a regular grid using the distance between the two points as the spacing between points.

One of the main difficulties with the *polytope* method is to decide when a minimum point has been reached, as opposed to when the polytope is undergoing temporary difficulty. Nelder and Mead [1965] suggested the following two criteria for convergence to a minimum point of a objective function. Convergence is assumed when either one of the two criteria is satisfied.

1. $$|F_1 - F_{n+1}| \le \varepsilon(1 + |F_1|)$$

2. $$\sum_{i=1}^{n+1}\left(F_i - \frac{\sum_{j=1}^{n+1} F_j}{n+1}\right)^2 \leq \varepsilon$$

The above description of the *polytope* algorithm (also known as the simplex algorithm) is based on the discussion in Gill, Murray and Wright [1981], Carroll [1990] and Press, Flannery, Teukolsky and Vetterling [1986].

6. Optimization of Production Systems

6.1 Introduction

The purpose of this study was to investigate the ability of the nonlinear optimization algorithms to find optimum values for decision variables that vary with time. For example to maximize the Present Value of oil and gas production from a single completion. The completion string -- tubing diameter -- is assumed to change with time. The optimization would produce the "best" tubing diameter for a given time interval, for example for each year, or two years, or five years of production.

The previous chapters have described in detail the theory used in developing each of the components of the production model used for this optimization study. This chapter will concentrate initially on how the nonlinear optimization algorithms were coupled to the model of the production system and the objective function, and next, describe the results obtained from several time dependent optimizations of a tubing string and a simple gas lift scheme.

6.2 Production Model

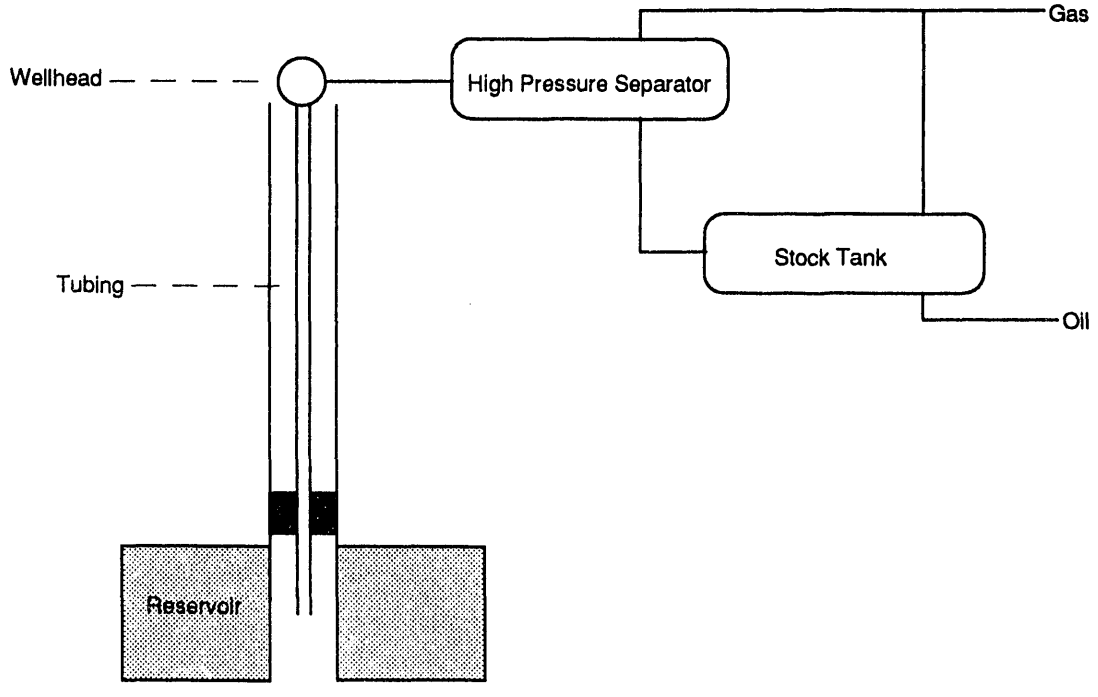

Figure 6.1: Schematic diagram of the Production Model

Figure 6.1 shows a schematic diagram of the production model used in this study. Note that in this case there is no horizontal flowline between the wellhead and the high pressure separator. The flowing wellhead pressure is the operating pressure of the high pressure separator. The decision variables chosen for studying are the tubing diameter and the high pressure separator pressure. Later in the study a gas lift system was implemented providing an additional decision variable, the gas injection rate.

6.3 Optimization Scheme and Algorithm

Figure 6.2 shows schematically how the nonlinear optimization algorithms are coupled with the objective function and production model. The iterative procedure for optimization is as follows.

1. The nonlinear optimization algorithm sends values for the decision variables to the production model.

2. Using the values for the decision variables the production model generates the production forecast.

3. Based on the production forecast the objective function generates the Present Value.

4. The nonlinear optimization algorithms use the Present Value to refine the estimate for the maximum, and sends the new estimates for the decision variables to the production model.

Iteration around the loop (in an anti-clockwise direction) as indicated in Figure 6.2 is continued until the maximum Present Value[2] is found, along with the associated values for the decision variables.

The IMSL mathematics library functions were used for optimization. Specifically DUMINF -- a quasi Newton algorithm that uses the BFGS formulation, and DUMPOL -- an implementation of the polytope algorithm. For a detailed explanation of these algorithms the reader is referred to Chapter 5 of this work.

The polytope algorithm always converged to the maximum point or to a point very close to the maximum point. The quasi Newton method did not always converge, specially when the time dependent variables were included in the optimization scheme. The "best" results were obtained by using a composite method, where the polytope algorithm was combined with the quasi Newton method. The technique employed was to

---

[2] In fact the algorithm finds the minimum of the negative of the objective function. This is equivalent to finding the maximum of the objective function. See Chapter 5 for further details.

use the polytope algorithm first to find the vicinity of the global maximum. Then the quasi Newton technique was used to locate the maximum point.

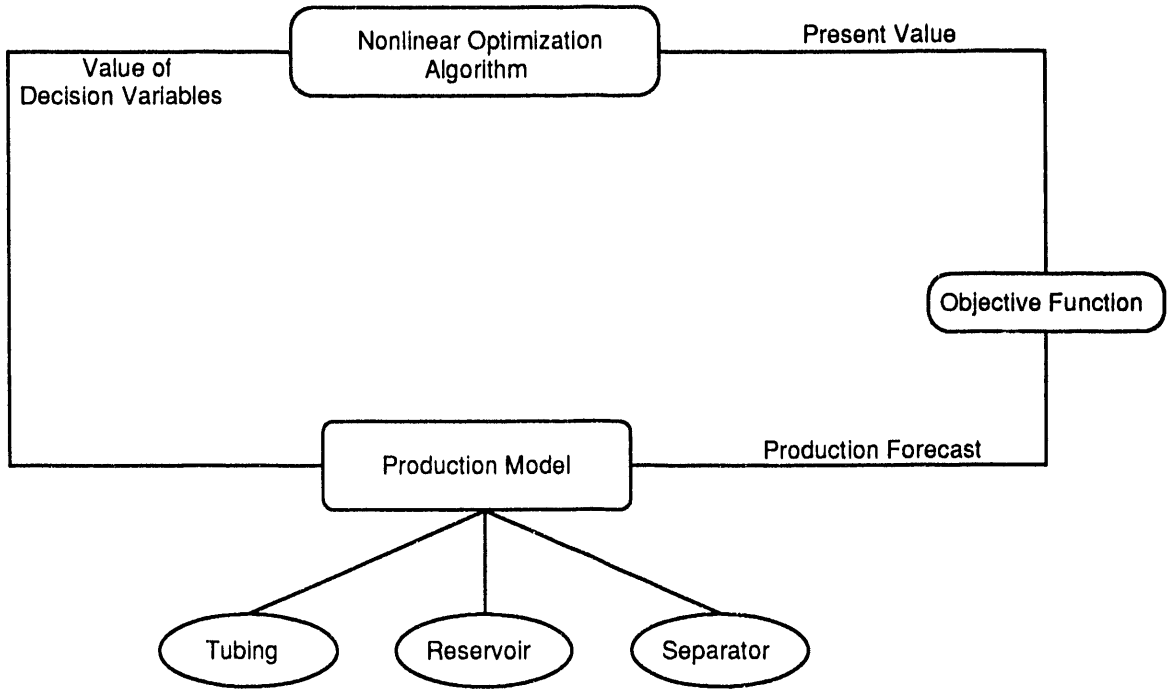

Figure 6.2: Schematic of the Iteration Process for Optimization

6.4 Objective Criteria

For the purpose of this study the objective criteria chosen was the Present Value of the oil and gas produced from the completion. The Present Value is a widely used method in project economics. The idea behind using Present Value is the concept of the "time value of money." The future income is discounted to reflect its worth in today's dollars.

The Present Value for a future income at time T, FI_T, can be represented mathematically as

$$PV_T = \frac{FI_T}{(1+R)^{T-\frac{1}{2}}} \qquad (6.1)$$

where R = the discount rate, and reflects the cost of capital to the company.

FI_T = Price of Oil * Oil Produced + Price of Gas * Gas Produced

It is assumed that the price of oil and gas will increase with time, and therefore the price is escalated at a inflation rate i. Equation 6.1 can be modified to

$$PV = \sum_{T=1}^{n} \left[\frac{FI_T}{\left(\frac{1+R}{1+i} \right)^{T-\frac{1}{2}}} \right] \tag{6.2}$$

The objective function can easily be modified to a net Present Value (NPV) form

$$NPV = \sum_{T=1}^{n} \left[\frac{FI_T - Cost_T}{\left(\frac{1+R}{1+i} \right)^{T-\frac{1}{2}}} \right] \tag{6.3}$$

where $Cost_T$ = Capital cost, tax payments, royalties, operating cost etc.

6.5 Tubing Diameter Optimization

The first sets of optimizations were carried out to obtain the optimum tubing diameters for a given well. This process finds the "best" tubing diameter for a producing time interval. The best in this case refers to the tubing size(s) that maximizes the Present Value. It mimics the real life case where a workover is performed to change out the tubing.

Figure 6.3 and 6.4 shows the Present Value surface for the case where a single tubing size and separator pressure are used for the entire life of the well. Figure 6.3 is based on the Aziz, Govier and Fogarasi [1972] and Figure 6.4 uses the Hagedorn and Brown correlation [1964]. As these figures indicate, the surface is rough (nonsmooth) and shows the presence of several local maxima.

The following time dependent tubing diameter optimization runs were made.

1. Tubing diameter changed after every 10 years of production.
2. Tubing diameter changed after every 5 years of production.
3. Tubing diameter changed every year of production.

The results from the optimizations are shown in numerical form in Tables 6.1 through to Table 6.4. The tubing sizes, production rates and pressures are plotted as a function of time for the Aziz, Govier and Fogarasi [1972] correlation in Figure 6.5 to Figure 6.7.

Table 6.1: Single Tubing Diameter Optimization Results

	Flow Correlation	
	Aziz, Govier and Fogarasi	Hagedorn and Brown
Separator Pressure (psi)	1113.77	1045.53
Tubing Diameter (inches)	9.27	13.44
Present Value ($ million)	305.76	311.44

Table 6.2: Optimization Results for Case When Tubing Diameter is Changed Every 10 Years.

	Flow Correlation	
	Aziz, Govier and Fogarasi	Hagedorn and Brown
Separator Pressure (psi)	1113.13	1029.71
Tubing Diameter (inches)		
(Year 0 - 10)	9.16	18.97
(Year 11 - 20)	11.17	11.36
Present Value ($ million)	306.64	313.02

Table 6.3: Optimization Results for Case When Tubing Diameter is Changed Every 5 Years.

	Flow Correlation	
	Aziz, Govier and Fogarasi	Hagedorn and Brown
Separator Pressure (psi)	1115.43	1029.4
Tubing Diameter (inches)		
(Year 0 - 5)	9.11	22.36
(Year 6 - 10)	9.26	21.17
(Year 11 - 15)	11.10	10.66
(Year 16 - 20)	12.78	9.78
Present Value ($ million)	306.80	314.19

Table 6.4: Optimization Results for Case When Tubing Diameter is Changed Every Year.

	Flow Correlation Aziz, Govier and Fogarasi
Separator Pressure (psi)	1089.8
Tubing Diameter (inches)	
(Year 1)	9.78
(Year 2)	9.55
(Year 3)	9.34
(Year 4)	9.25
(Year 5)	9.30
(Year 6)	9.47

(Year 7)	9.54
(Year 8)	10.03
(Year 9)	10.11
(Year 10)	10.80
(Year 11)	11.31
(Year 12)	11.73
(Year 13)	12.19
(Year 14)	12.57
(Year 15)	12.92
(Year 16)	13.03
(Year 17)	15.04
(Year 18)	11.77
(Year 19)	12.60
(Year 20)	13.38
Present Value ($ million)	354.07

6.6 Gas Lift Optimization

A simple gas lift model was developed to investigate the ability of the nonlinear optimization algorithms to optimize the gas injection rate with time (Brown [1977]).

The model assumed that the gas was injected at the bottom of the completion string (tubing) at a specified rate. The net effect of the gas injection is to decrease the apparent density of the fluid flowing in the tubing. As the wellhead pressure is held constant, the reduction in density, reduces the pressure drop between the wellhead and the perforations, decreasing the flowing bottomhole pressure, increasing the flowrate of oil and gas. Except for the injection of gas, the production model and the objective function used is identical to the model described in Section 6.2

The model also assumes that there is an endless supply of gas for injection, and as the objective function was formulated in terms of Present Value there is no cost associated with the injection of gas.

Initially, the tubing diameter, separator pressure, and the gas injection rate were used as the decision variables for optimization. This problem did not appear to have an optimum. As Figures 6.8, 6.9, and 6.10 show the Present Value of the production seems to increase slowly in the direction of increasing tubing diameter and increasing gas injection rate. This problem can be overcome in one of two ways, to apply a cost to the injection gas, or to fix the tubing diameter. To investigate the time dependence of the optimum gas injection rate, the tubing diameter was fixed at 5.90 inches (15.0 cm).

Figure 6.11 shows the Present Value surface for the gas injection scheme with the tubing diameter fixed at 5.90 inches (15 cm). Figures 6.12 and 6.13 show the same surface for a fixed tubing diameter of 3.93 inches (10 cm) and 7.87 inches (20 cm) respectively.

The results of the optimizations are presented in Table 6.5 through to Table 6.7. Figure 6.14 shows the optimum injection rates as a function of time, and Figure 6.15 the production rates for oil and gas. Figure 6.16 shows the accompanying pressure profiles.

Table 6.5: Single Gas Injection Rate Optimization Results

	Flow Correlation Aziz, Govier and Fogarasi
Separator Pressure (psi)	1217.20
Tubing Diameter (inches)	5.90 (fixed)
Injection Rate (mmscf/d)	9.78
Present Value ($ million)	341.35

Table 6.6: Optimization Results for Case When Gas Injection Rate is Changed Every 10 Years.

	Flow Correlation Aziz, Govier and Fogarasi
Separator Pressure (psi)	1215.89
Tubing Diameter (inches)	5.90 (fixed)
Injection Rate (mmscf/d)	
(Year 0 - 10)	10.78
(Year 11 - 20)	0.13
Present Value ($ million)	343.20

Table 6.7: Optimization Results for Case When Gas Injection Rate is Changed Every 5 Years.

	Flow Correlation Aziz, Govier and Fogarasi
Separator Pressure (psi)	1240.2
Tubing Diameter (inches)	5.90 (fixed)
Injection Rate (mmscf/d)	
(Year 0 - 5)	17.75
(Year 6 - 10)	16.82
(Year 11 - 15)	4.87
(Year 16 - 20)	0.13
Present Value ($ million)	344.19

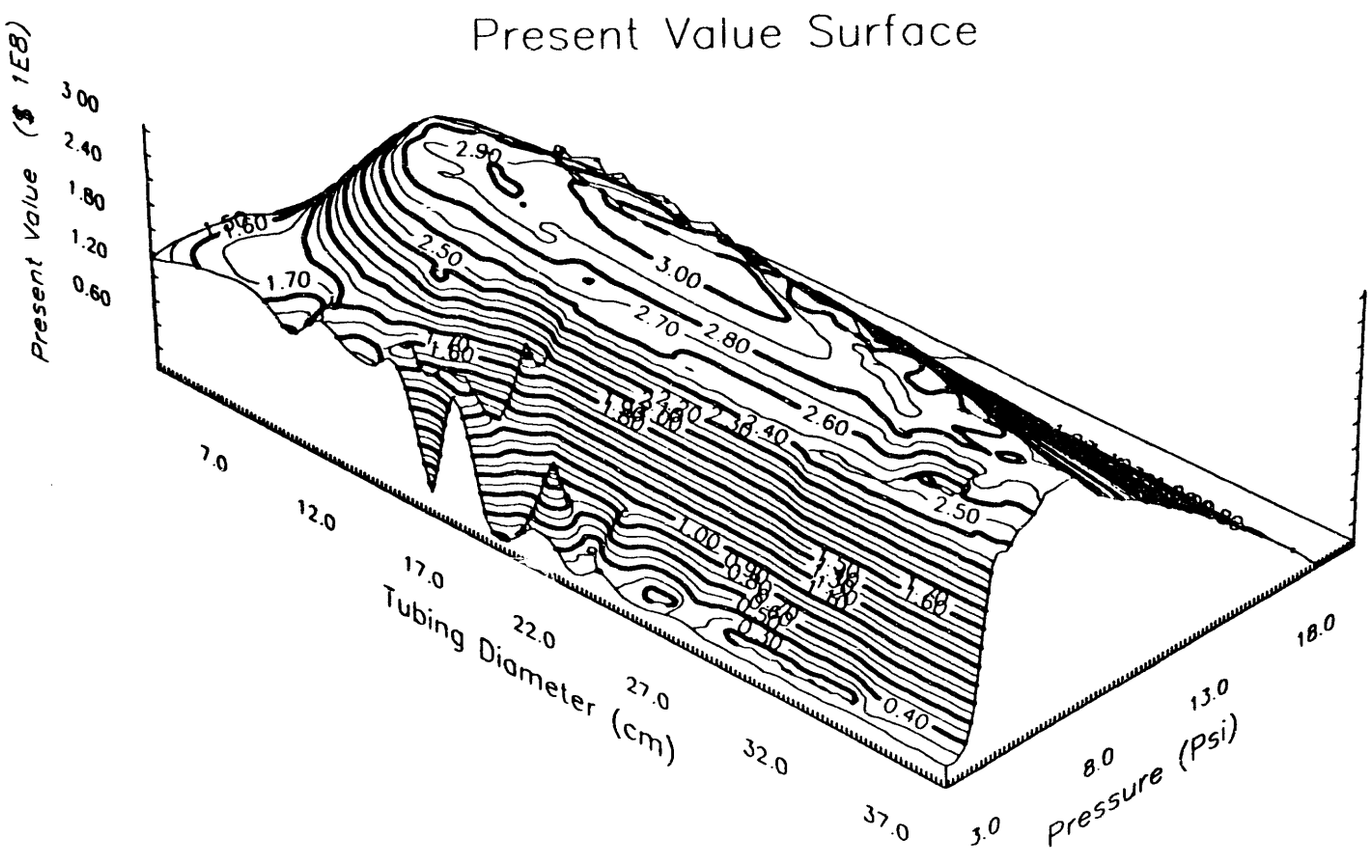

Figure 6.3: Present Value Surface for the Aziz, Govier and Fogarasi [1972] Correlation

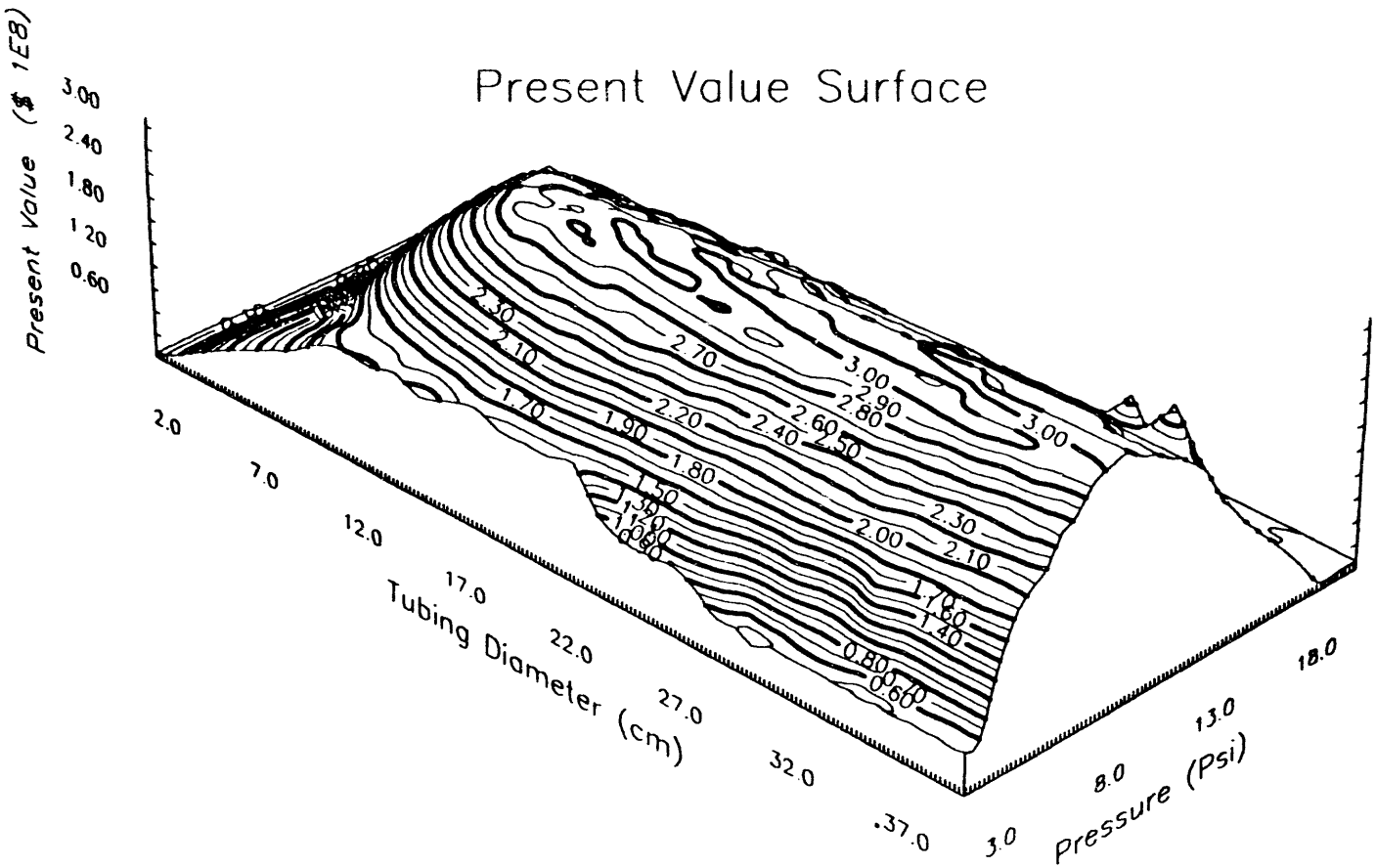

Figure 6.4: Present Value Surface for the Hagedorn and Brown [1964] Correlation

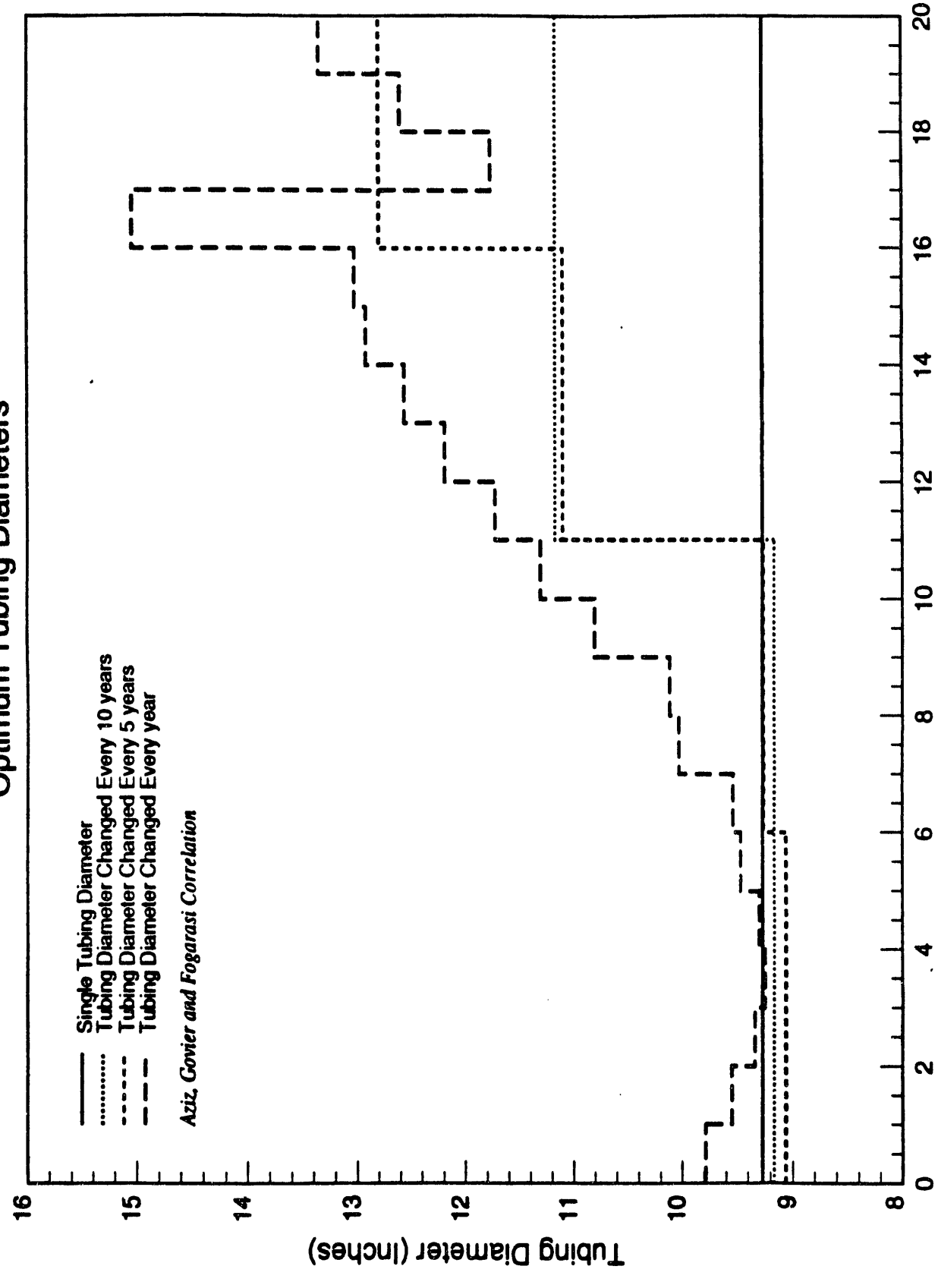

Figure 6.5 Optimum Tubing Diameters as a Function of Time

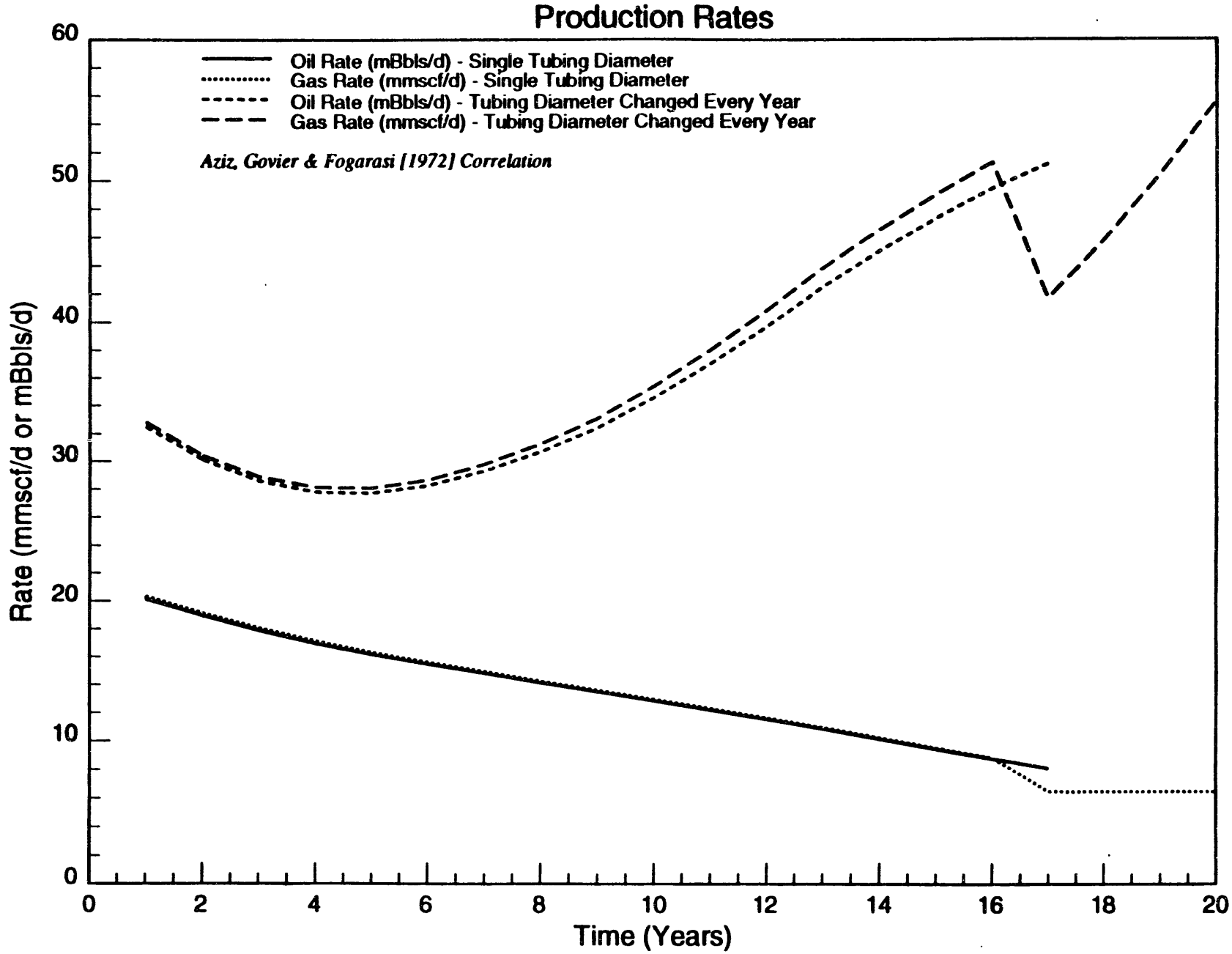

Figure 6.6 Production Rates of Oil & Gas for Tubing Optimizations

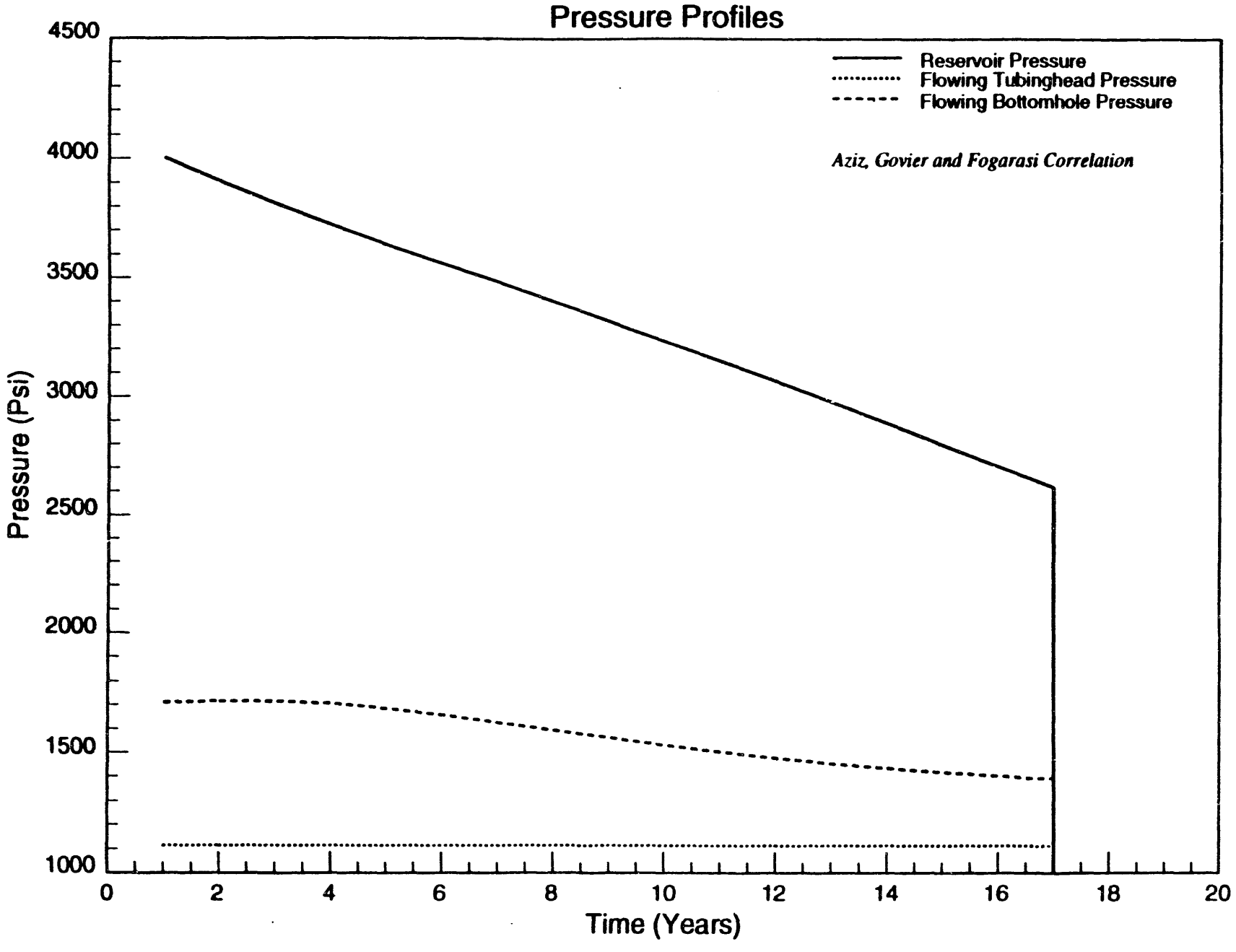

Figure 6.7 Pressure Profile For Single Tubing Diameter Optimization

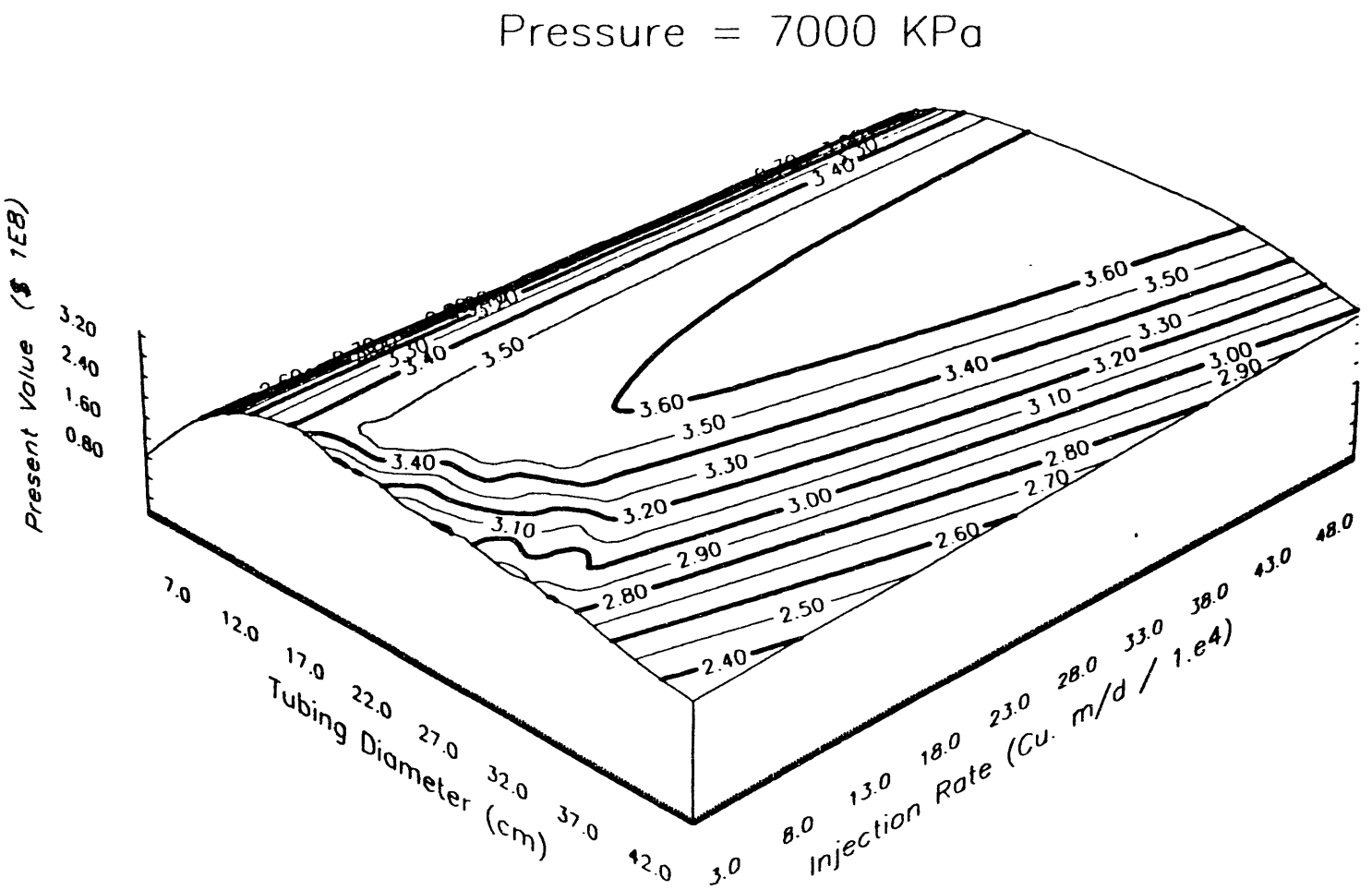

Figure 6.8 **Present Value Surface for Gas Injection with the Wellhead Pressure Fixed at 7000 KPa**

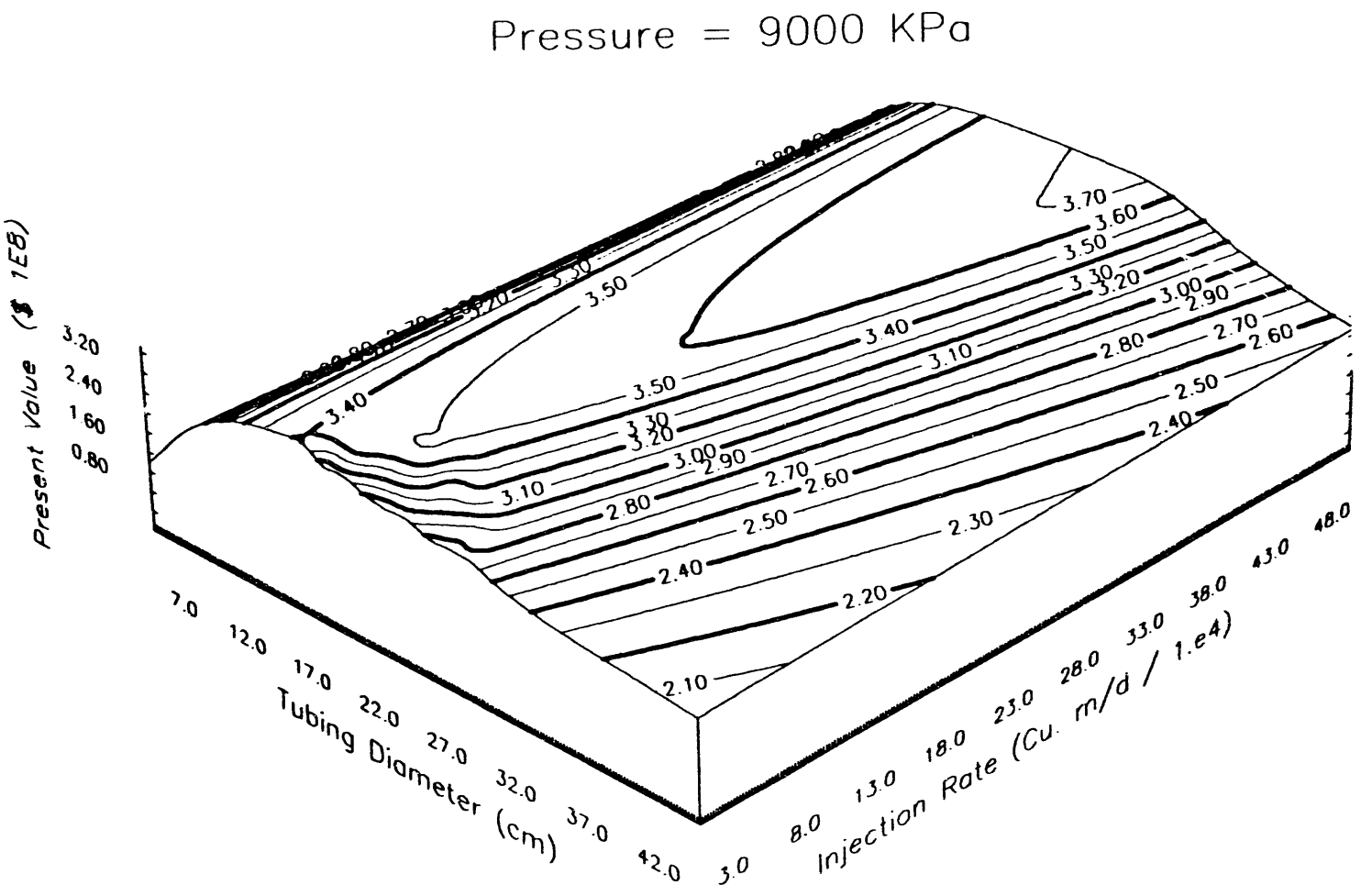

Figure 6.9 **Present Value Surface for Gas Injection with the Wellhead Pressure Fixed at 9000 KPa**

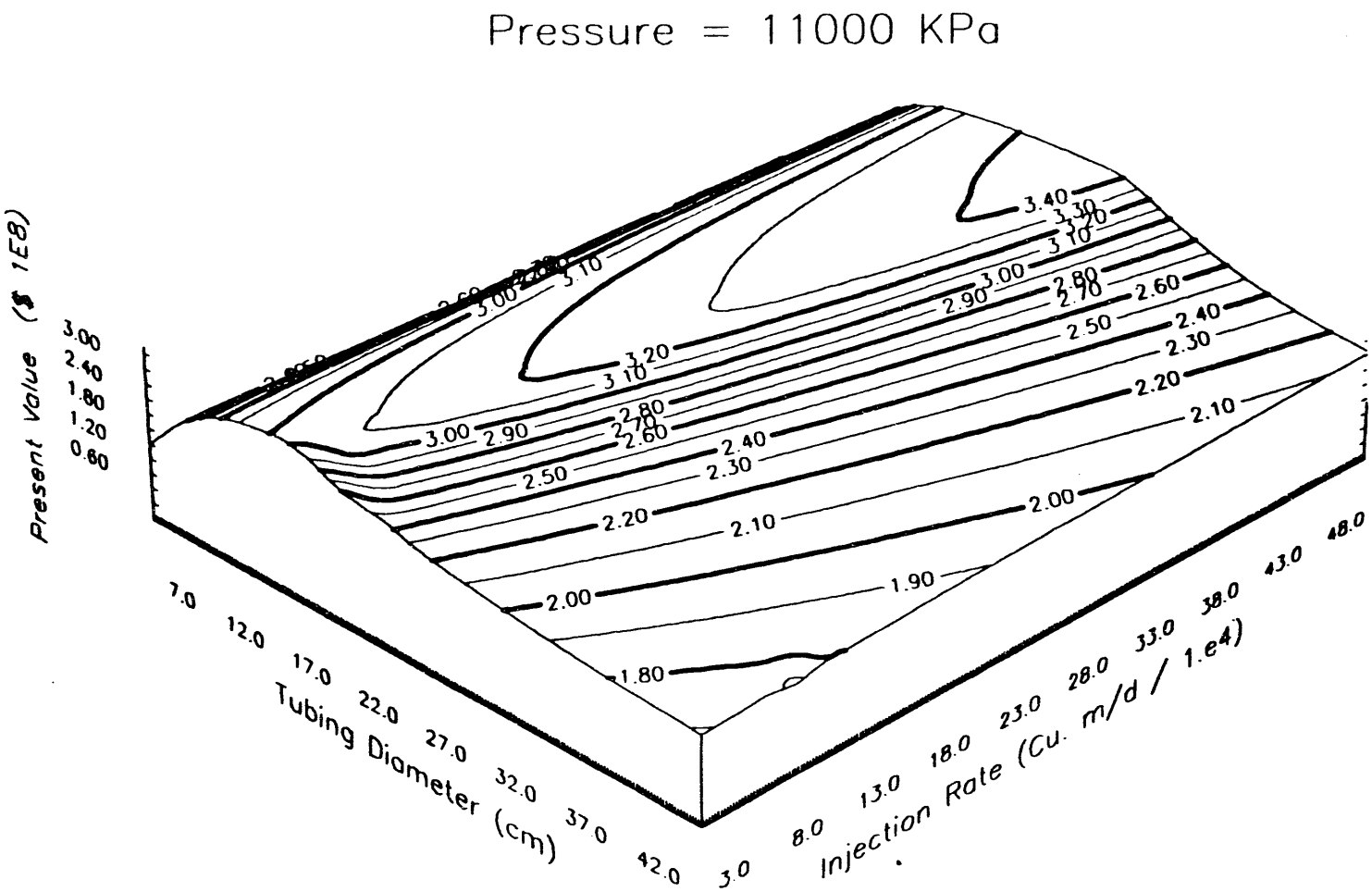

Figure 6.10 **Present Value Surface for Gas Injection with the Wellhead Pressure Fixed at 11000 KPa**

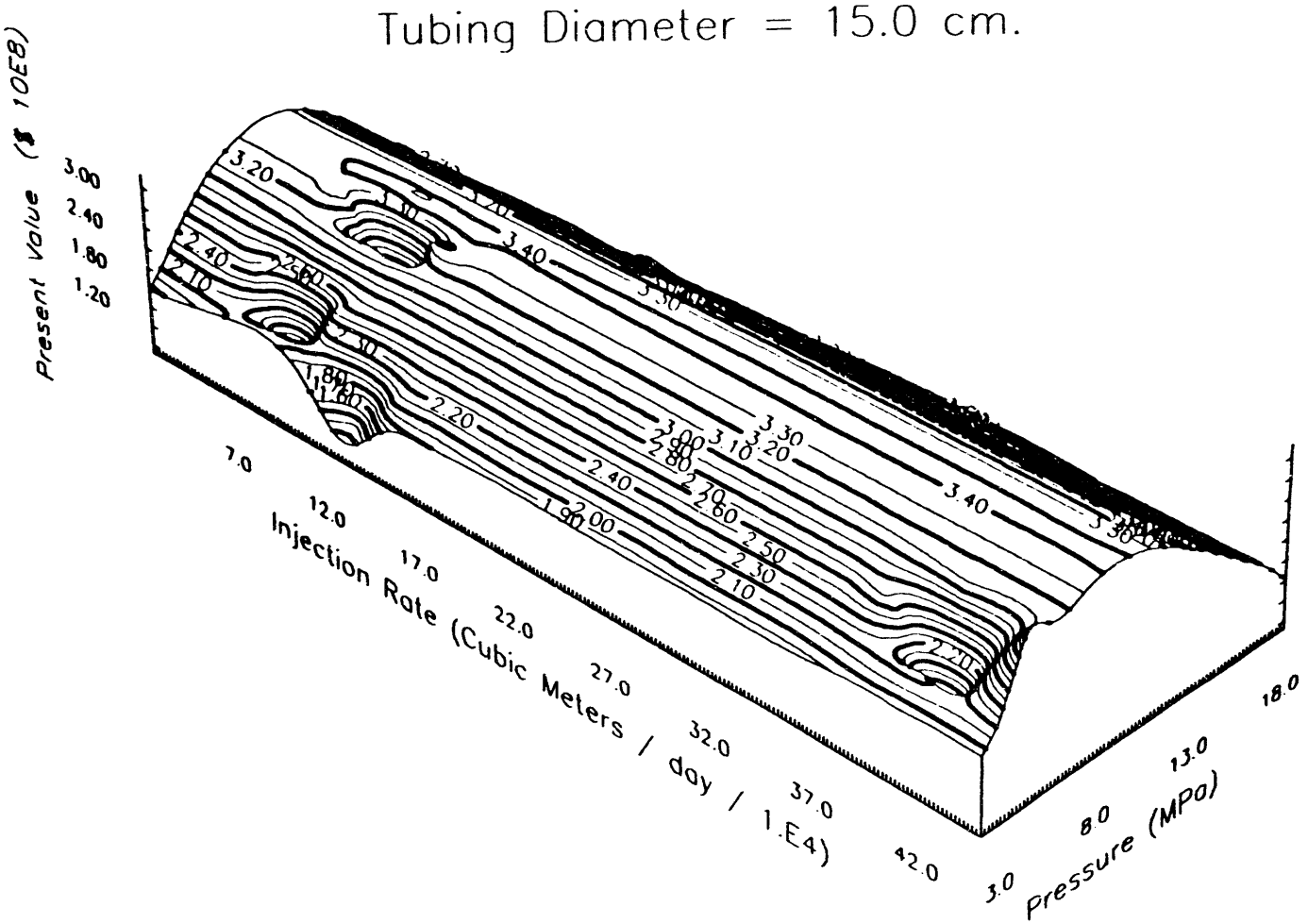

Figure 6.11 **Present Value Surface for Gas Injection with the Tubing Diameter Fixed at 5.9 inches.**

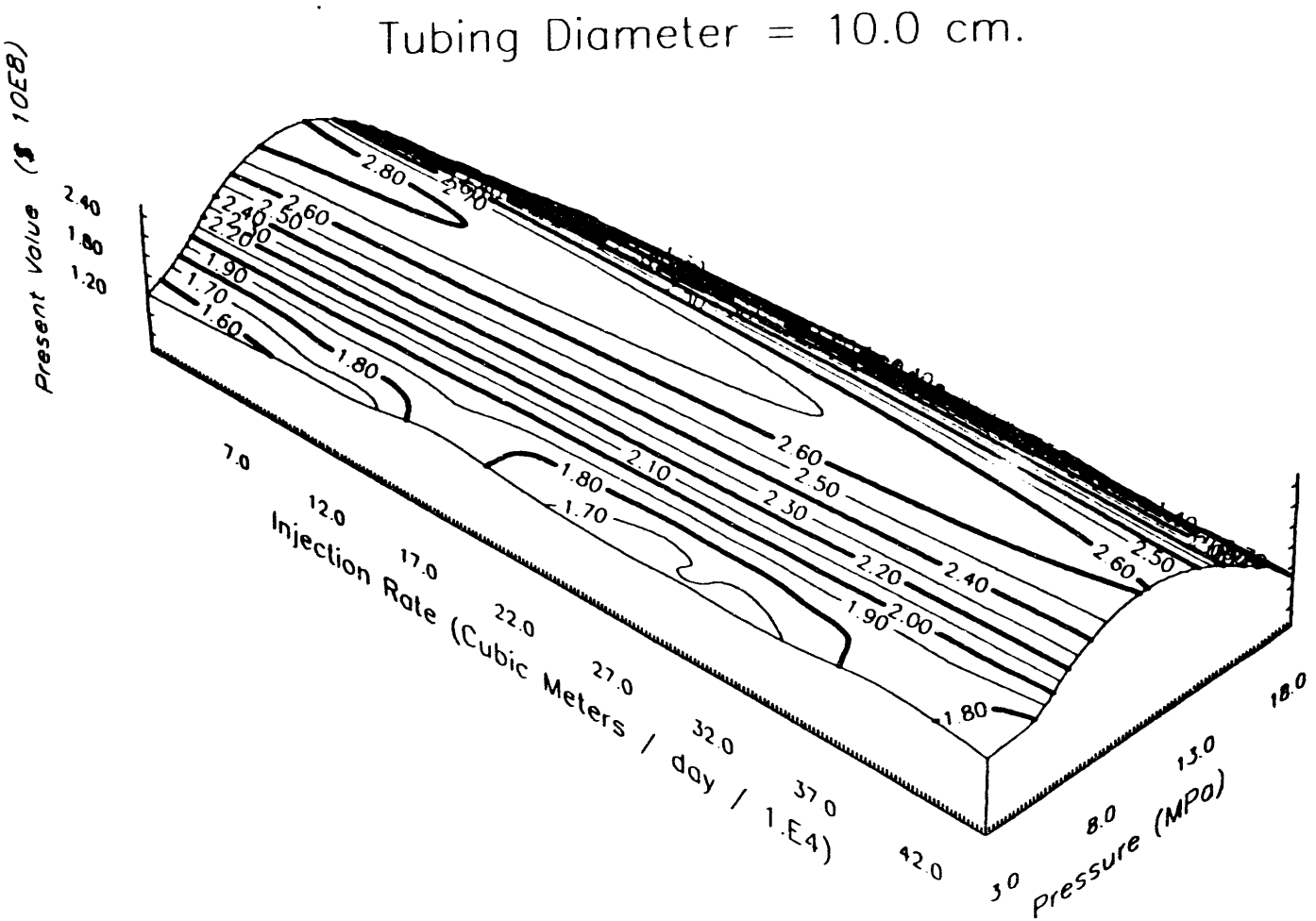

Figure 6.12 **Present Value Surface for Gas Injection with the Tubing Diameter Fixed at 3.9 inches**

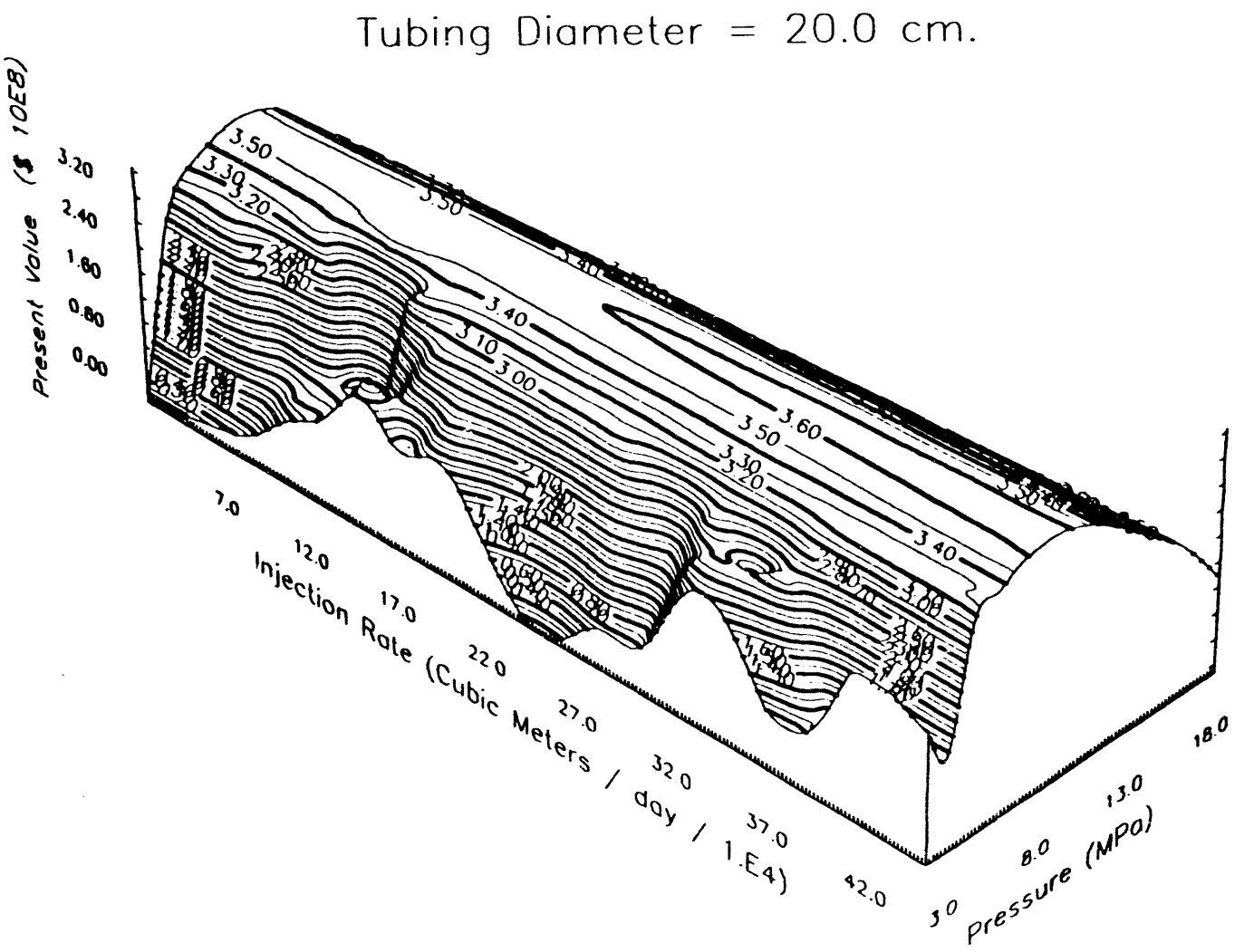

Figure 6.13 **Present Value Surface for Gas Injection with the Tubing Diameter Fixed at 7.87 inches**

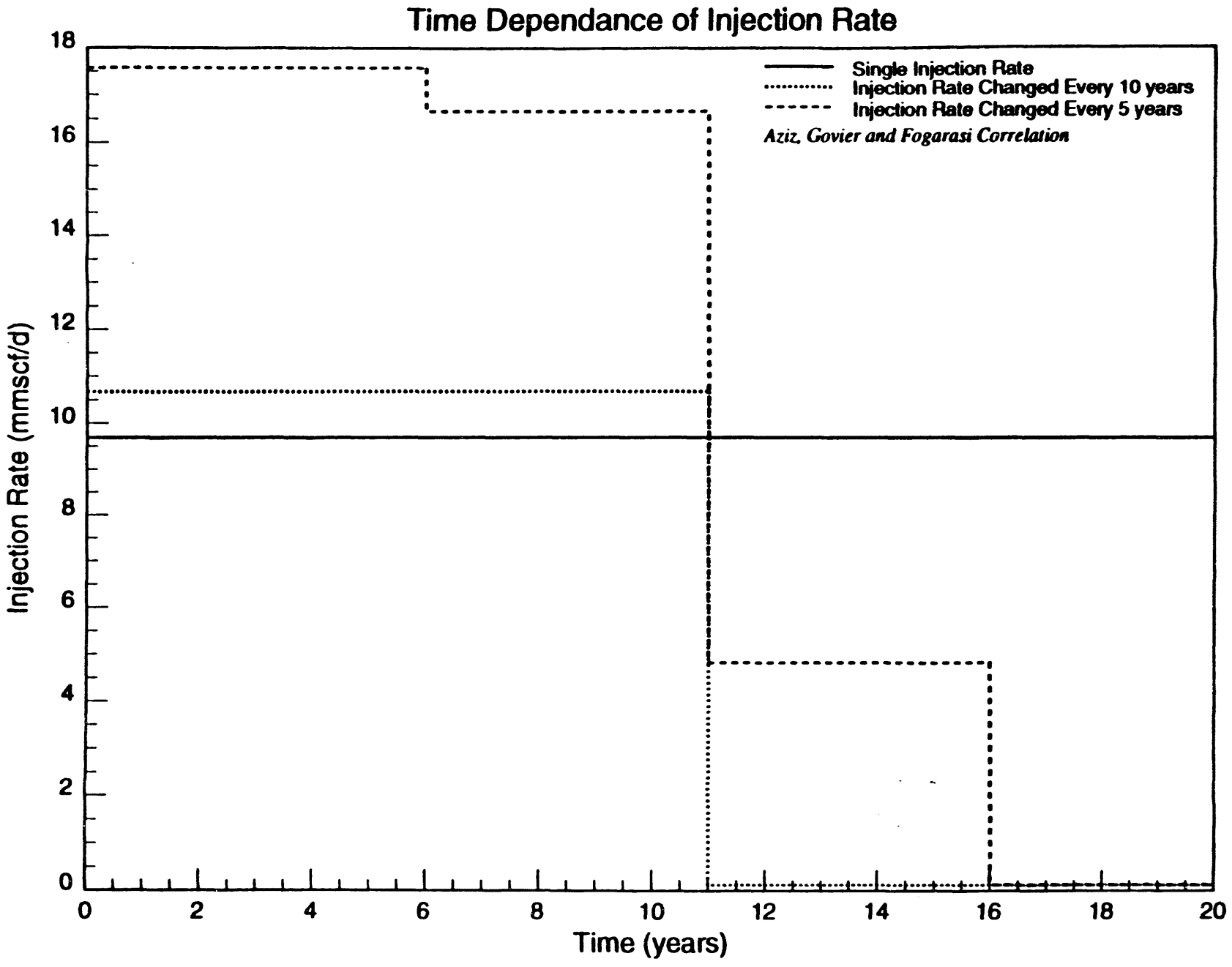

Figure 6.14 Optimum Gas Injection Rates as a Function of Time

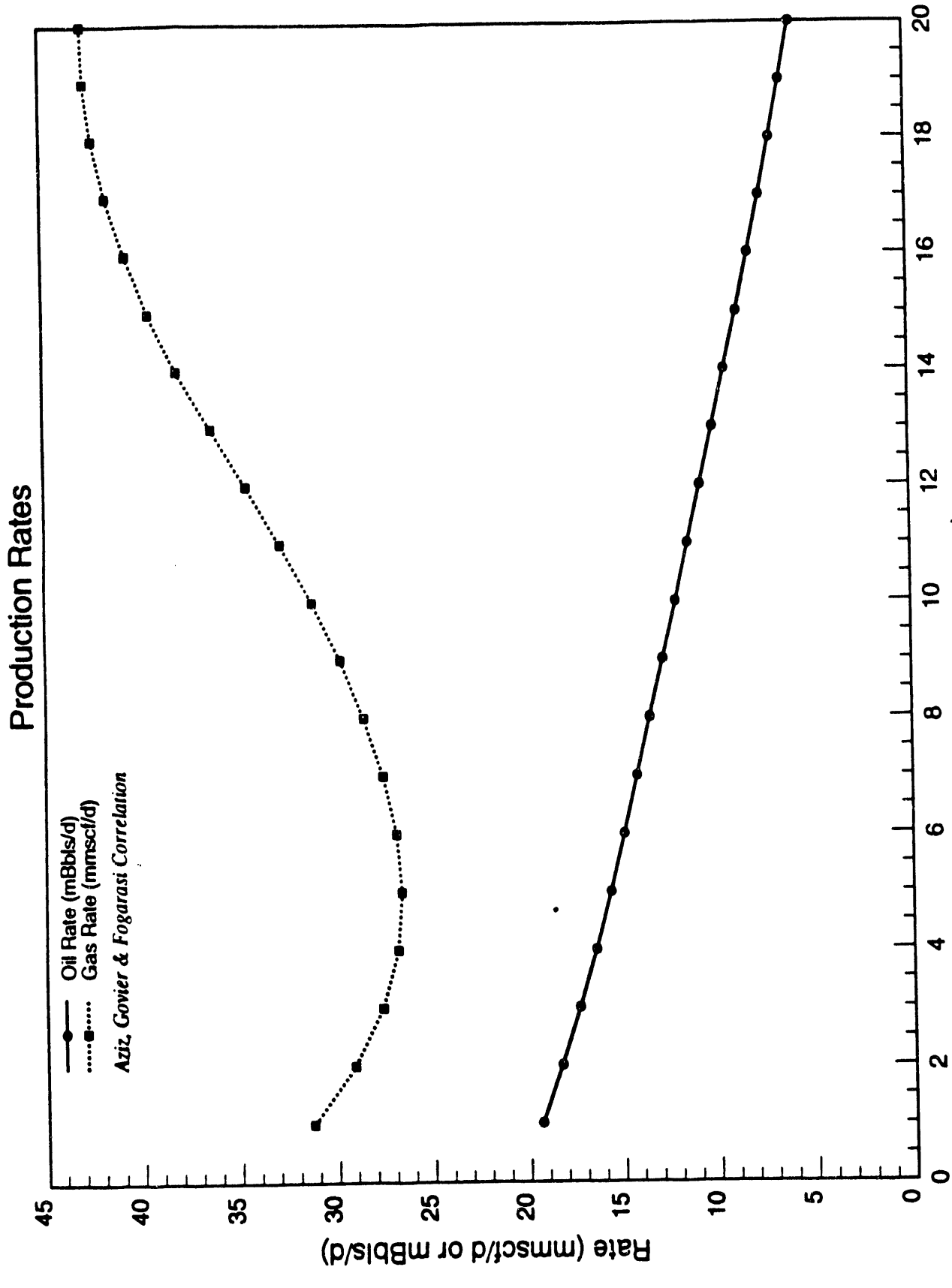

Figure 6.15 Oil and Gas Production Rates for Gas Injection Case

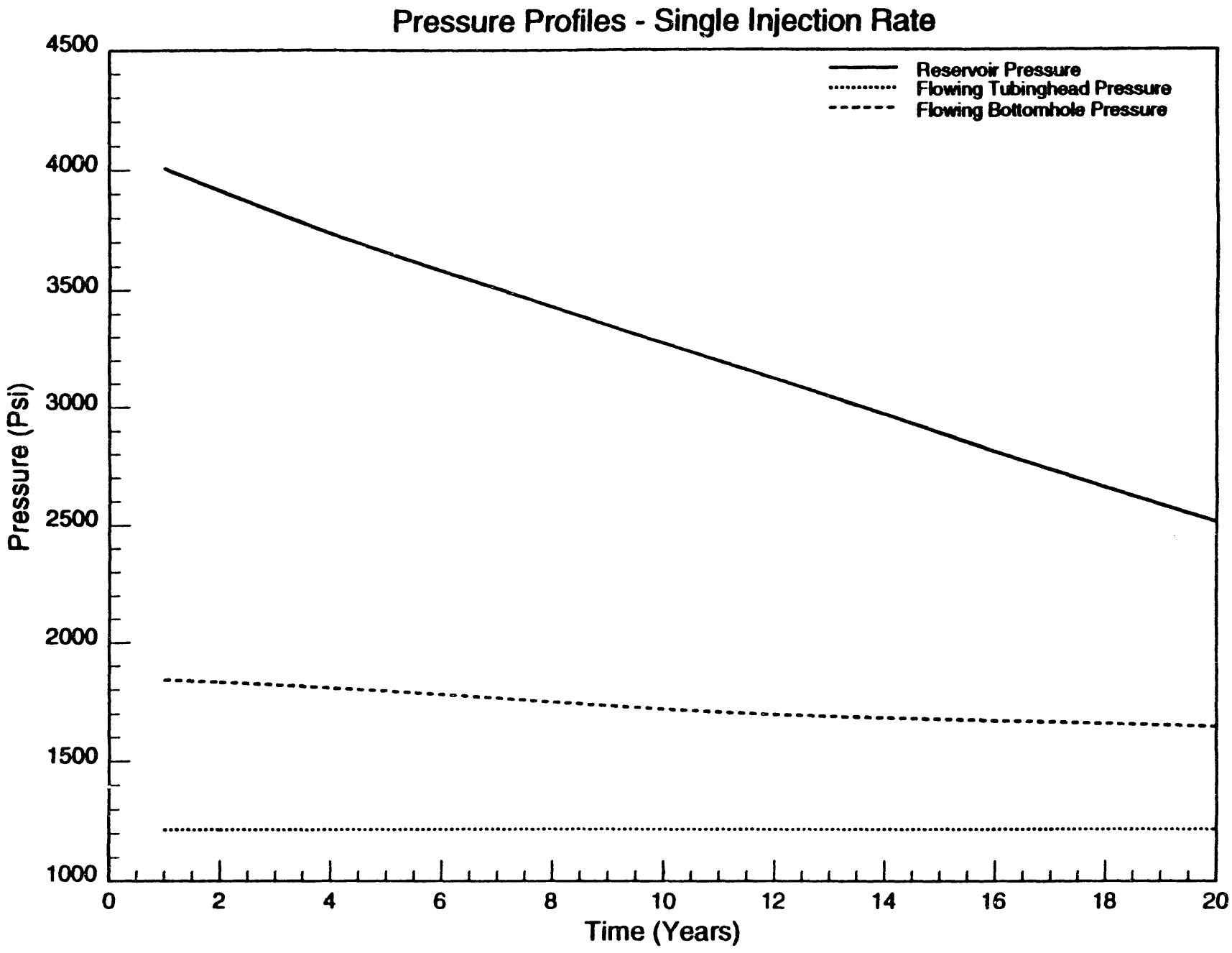

Figure 6.16 Pressure Profiles for Single Gas Injection Rate Optimization

7. Conclusions and Ideas for Future Projects

7.1 Conclusions

This study has demonstrated that nonlinear optimization techniques can be successfully applied to hydrocarbon productions systems.

Specifically this study shows that

- The nonlinear optimization techniques can be applied to production systems with decision variables that change with time.
- The present value surfaces are nonsmooth functions of the decision variables (tubing diameter, separator pressure, gas injection rate). The nonlinear optimization algorithms that do not use gradients of the objective functions like the polytope method are the more suitable algorithms.
- The best strategy for optimization of the production system was found to be to use the polytope algorithm to find the local area of the maximum point, and then to use a Newton based technique such as the BFGS algorithm (Broyden [1970], Fletcher [1970], Goldfarb [1970] and Shanno [1970]) to find the "exact" location of the maximum.
- The sensitivity of the decision variables to the objective function is a cause for concern, specially for problems with time dependent variables. For example, in this study the objective function -- the Present Value -- is weighted towards production in early years. The result is that production towards the end of the life of the reservoir has very little effect on the total present value. The objective surface is relatively flat, and therefore false convergence may be assumed by the algorithm away from the actual maximum point.

7.2 Future Work

Genetic algorithms and simulated annealing algorithms are being increasingly used in various engineering fields for optimization. Goldberg [1989], Holland [1975] and Katragadda [1991] have used genetic algorithms to optimize different problems. Readers are referred to Goldberg [1989] for a very good introduction to genetic algorithms. Katragadda [1991] used genetic algorithms to optimize a highly nonlinear problem -- the trajectory of a spacecraft on a mission from Earth to Mars.

Including the cost of production into the economic analysis by switching the objective function to use net present value (NPV), is highly desirable. For example when

considering the gas lift system, the use of NPV would provide a very different objective surface. The optimization would then be able to include the tubing diameter as one of the decision variable.

Another very interesting problem to which the nonlinear optimization algorithms can be applied to is a plunger lift system. Plunger lift is being used increasingly in industry as a method of dewatering gas wells, and as an artificial lift system for high gas - oil ratio wells that are unable to flow to the surface. The optimization of the plunger cycle time, as well as the tubing diameters, and other various decision variables would pose a challenging problem.

Bibliography

1. Abramowitz, M., and Stegun, I.A.: Handbook of Mathematical Functions, Dover Publications, New York (1965).

2. Ahmed, T.: Hydrocarbon Phase Behavior, Gulf Publishing, Houston, (1989).

3. Aziz, K., Govier, G.W., and Fogarasi, M.: "Pressure Drop in Wells Producing Oil and Gas," J. Canadian Petroleum Technology (July 1972) 38-48.

4. Beggs, H.D.: Production Optimization Using Nodal Analysis, OGCI Publications, Tulsa (1991).

5. Borthne, Gunnar.: A Simulation Model for Oil and Gas Condense Production Based on Material Balance and Inflow Performance Calculations, Norwegian Institute of Technology (1986).

6. Brown, K.E.: The Technology of Artificial Lift Methods, 1, 2a, 2b, 4, Penwell Publishing, Tulsa (1977).

7. Broyden, C.G.: "The Convergence of a Class of Double - Rank Minimization Algorithms I: General Considerations," J. Institute of Mathematical Applications, (1970), 12, 76 - 90.

8. Carroll, J.A.: Multivariate Production Systems Optimization, MS Thesis, Department of Petroleum Engineering, Stanford University, Stanford (1990).

9. Chilingarian, G.V., Robertson, J.O., and Kumar, S.: Surface Operations in Petroleum Production, 1, Elsevier, Amsterdam (1987).

10. Davidon, W.: "Variable Metric Method for Minimization," Argonne National Laboratory Report ANL - 5990, Argonne, Illinois.

11. Duns, H., and Ros, N.C.J.: "Vertical Flow of Gas and Liquid Mixtures in Wells," Proceedings of the 6th World Petroleum Congress (1963) 451 - 465.

12. Fletcher, R: "A New Approach to Variable Metric Algorithms," Computer Journal 13 (1970), 317 - 322.

13. Fletcher, R., and Powell, M.J.D.: "A Rapidly Convergent Descent Method for Minimization," Computer Journal, 6, (1963), 163 - 168.

14. Gill, P.E., and Murray, W.: "Newton Type Methods for Unconstrained and Linearly Constrained Optimization," J. Mathematical Progress, 7, (1974), 311 - 350.

15. Gill, P.E., Murray, W., Wright, M.H.: Practical Optimization, Academic Press, New York (1981).

16. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning, Addison - Wesley, Boston (1989).

17. Goldfarb, D.: "A Family of Variable Metric Updates Derived by Variational Means," J. Mathematical Computation, 24 (1970), 23 - 26.

18. Goldfeld, S.M., Quandt, R.E., and Trotter, H.F.: "Minimization by Quadratic Hill Climbing," Econometrica, 34, (1970), 541 - 551.

19. Govier, G.W., and Aziz, K.: The Flow of Complex Mixtures in Pipes, Van Nostrand Reinhold Co., New York (1972).

20. Greenstadt, J.L.: "On the Relative Efficiencies of Gradient Methods," Mathematics of Computation, 21 (1967) 360-67.

21. Griffith, P., and Wallis, G.B.: "Two Phase Slug Flow," J. Heat Transfer, ASME (Aug. 1961) 307-320.

22. Hagedorn, A.R., and Brown, K.E.: "Experimental Study of Pressure Gradients Occurring During Continuos Two Phase Flow in Small-Diameter Vertical Conduits," J. Petroleum Technology (April 1964) 475-484.

23. Holland, J.H.: Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor (1975).

24. IMSL Inc., IMSL MATH/LIBRARY Users Manual, IMSL, Houston, (1987).

25. Katragadda, L.K.: Genetic Algorithms and Nonlinear Programming for Optimal Low Thrust Spacecraft Trajectories, MS Thesis, Department of Aerospace Engineering and Engineering Mechanics, Iowa State University (1991).

26. Mian, M.A.: Petroleum Engineering Handbook for the Practicing Engineer, Penwell Books, Tulsa (1992).

27. Marquardt, D.W.: "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," J. SIAM (June 1963) 11, No. 2, 431-41.

28. Nelder, J.A., and Mead, R.: "A Simplex Method for Function Minimization," Computer Journal 7, 308-313.

29. Orr, F.M.: Notes for Petroleum Engineering 251, Stanford University, Stanford (1991).

30. Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T.: Numerical Recipes, Cambridge University Press, Cambridge (1989).

31. Scales, L.E.: Introduction to Nonlinear Optimization, Spring - Verlag., New York (1985).

32. Shanno, D.F.: "Conditioning of a Quasi - Newton Method for Function Minimization," J. Mathematical Computation, 24, (1970) 647 - 656.

33. Wilson, G.M., and Deal, C.H.: "Vapor Liquid Equilibria", Ind. Eng. Chem. Fundam., 1: 20 (1962).

APPENDIX

Source Code For the Well Model

Wellmodel.For


```
C**********************************************************************
C**** Program      Simplex
C**** Authors      Niranjan Ravindran, James Carroll and Gunnar Borthne
C**** Date         June, 1990
C**** Function     Driver for Well Model. Set for Time Dependant
C****              Optimization of Tubing Diameters and Gas Injection
C**********************************************************************

      Program Simplex

      Integer N, IDO, P
      Parameter (N=5)
      Integer MAXFCN, K, IPARAM(7)
      Real*8 FTOL, FVALUE, S, X(N), XGUESS(N)
      Real*8 RPARAM(7), XSCALE(N), CPU1, CPU2, Cpu
      Real*8 A(N), B(N), SS(N),F
      Character*100 ZIN, ZIN1, ZOUT1, ZOUT2, ZOUT3, ZOUT4
      External FCN,DUMPOL,DUMINF,DU4INF

C---- Start Execution


      Print *, 'Name of file for input data - GMS?'
      Read 5100, ZIN
      Print *, 'Name of file for input Start Point?'
      Read 5100, ZIN1
      Print *, 'Name of file for general output data?'
      Read 5100, ZOUT1
      Print *, 'Name of file for graph output data?'
      Read 5100, ZOUT2
      Print *, 'Name of file for triag output data?'
      Read 5100, ZOUT3
      Print *, 'Name of file for Plot output data?'
      Read 5100, ZOUT4

      Open (Unit=2,File=ZIN)
      Open (Unit=1,File=ZIN1)
      Open (Unit=3,File=ZOUT1)
      Open (Unit=4,File=ZOUT2)
      Open (Unit=5,File=ZOUT3)
      Open (Unit=8,File=ZOUT4)

C---- Unit 20, File Nspace is reserved for writing N-dimensional
C---- resutls.
C---- Unit 25, is reserved for reading "in" the top corner, and
C---- bottom corner of the N-space.

C---- Read the Initial guess, and the step size for the Polytope.

        do I=1,N
         read(1,*) XGUESS(I)
         X(I) = XGUESS(I)
        enddo
         read(1,*) S
         SO=S
      Call GMSINP
      Call GMSECHO

C---- Time Taken For Calculations.

        Call Getcpu(Cpu1)

C---- Convergence Tollarance.
        FTOL = 9.9999999999E-13

C---- Max No of Calls to GMSBODY
        MAXFCN = 1000000

C----   Goto 10

C----   Goto 20

C---- Call Polytope Algorithm, form IMSL
        Call DUMPOL(FCN,N,XGUESS,S,FTOL,MAXFCN,X,FVALUE)

        Call FCN(N,X,F)

        Write(6,*)'Optimum - Simplex1',(X(K),K=1,N),-F
        Write(5,*)'Optimum - Simplex1',(X(K),K=1,N),-F
        Write(8,*)'Optimum - Simplex1',(X(K),K=1,N),-F


C----   Goto 30

C---- Set up for Newton Algorithm.

10      do K=1,N
         XGUESS(K) = X(K)
         XSCALE(K) = 1.0d0
        Enddo

        FSCALE = 1.0

        Call DU4INF(IPARAM,RPARAM)


        IPARAM(3) = 1000.
        IPARAM(4) = 4000.
        IPARAM(5) = 4000.

        RPARAM(1) = RPARAM(4)
        RPARAM(2) = RPARAM(4)
        RPARAM(3) = RPARAM(4)
        RPARAM(5) = RPARAM(4)

        Write(5,*)(IPARAM(K),K=1,7)
        Write(5,*)(RPARAM(K),K=1,7)

C---- Call Modified Newton Algorithm
        Call DUMINF(FCN, N, XGUESS, XSCALE, FSCALE, IPARAM,
     *  RPARAM, X, FVALUE)

        Call FCN(N,X,F)

        Write(6,*)'Optimum Modnewt',(X(K),K=1,N),-F
        Write(5,*)'Optimum Modnewt',(X(K),K=1,N),-F
        Write(8,*)'Optimum Modnewt',(X(K),K=1,N),-F

         Goto 30

C------ Call DGGUES, the Point Generator.

        DATA IDO /0/

20      Open (Unit=25, File='Points')
        Open (Unit=20, File='Nspace')

        Do I=1,N
```

```
        Read(25,*)A(I),B(I)
       End do
        Read(25,*)P
       Close (Unit=25)

       Do I=1,P
        Call DGGUES(N, A, B, P, IDO, SS)
        Call FCN(N, SS, F)
        F = F / 1.0d08
        Write(20,*)SS(1)/1000.0, SS(2)/1000.0 ,-F
       Enddo
       Close (unit=20)
       Goto 40

C------ Calculate Time Taken
30     Call PLOTOUT(Cpu)

       Call Getcpu(Cpu2)
       Cpu = Cpu2   Cpu1

       Write(6,*)'Time',Cpu
       Write(5,*)'Time',Cpu
       Write(8,*)'Time',Cpu

5100   Format (A)

40     Close (Unit=2)
       Close (Unit=3)
       Close (Unit=4)
       Close (Unit=5)
       Close (Unit=8)

       Stop
       END

*****************************************************************************

       Subroutine FCN (N,X,F)
       Integer N
       Parameter (N3=5)
       Double Precision Pass(N3), Pass4
       Real*8 X(N),F
       Common /Pass/ Pass, Pass4

       Do I = 1, N
       Pass(I) = X(I)
       Pass(I) = Pass(I) / 1.00d5
       End do
       Pass(N3) = X(N3)

C---   Pass(1-4) tubing diameter1.
C---   Pass(3) Pressure
       do i=1,N
        If(X(i).le.0.0) then
            F=0.0
            Return
        Endif
       Enddo

C      Write(6,*)'FCN1',(Pass(I),I=1,N),-Pass4/1.0e8
       Call GMSBODY
       Write(6,*)'FCN2',(Pass(I),I=1,N),-Pass4/1.0e8
       Write(8,*)'FCN2',(Pass(I)/1000.,I=1,N),-Pass4/1.0e8
       F = Pass4

       Return

       End

*****************************************************************************
**** Subroutine  GMSECON
**** Author      James Carroll
**** Date        November, 1990
**** Function    This subroutine determines the objective criterion of
****            the well model.  The objective criterion is the
****            present value of the production stream.
*****************************************************************************
      Subroutine GMSECON(I)

*---- VARIABLES AND CONSTANTS

      Implicit Double Precision (A-Z)

      Integer N1, N2, IVNWEL, NWELLS, IERR, IEXE, IHC, IPRT, IUNIN,
     *    IWCTR, K, NWELL, NWT, NSTEP, I
      Character ZJOBID*60, ZC(8)*6
      Parameter (N1 = 100,N2 = 500)

      Dimension TPWMIN(N1), TRTEFM(N1), TRTEFT(N1), TSKN(N1), TTIM(N1),
     *    VGASP(0:N2), VGOR(0:N2), VOILP(0:N2), VPRSR(0:N2),
     *    VPRSWF(0:N2), VPRSWH(0:N2), VRTEG(0:N2), VRTEO(0:N2),
     *    VTIME(0:N2), IVNWEL(0:N2), NWELLS(N1), NPV1(0:N2)

*---- COMMON BLOCKS

      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     *    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /TRANS2/ C1, C2, C3, C4, C5, C6, C7, C8, C9
      Common /TRANS3/ ZC, ZJOBID
      Common /TRANS4/ VGASP, VGOR, VOILP, VPRSR, VPRSWF, VPRSWH, VRTEG,
     *    VRTEO, VTIME, IWCTR, NWELL, OILTI, GASTI, IVNWEL, K
      Common /FINANCE/ INFRATE, DISRATE, OILPRIC, GASPRIC
      Common /RESULTS/ NPV, NPV1

        IF(I.EQ.1)THEN
      NSTEP = IDINT(DNINT(XMXTIM/DELTIM))
      NPV = 0.
      DFEFF = (1+DISRATE) ** (DELTIM/365)
      DFACTOR = (1+DISRATE) ** (DELTIM/2/365)
      IFEFF = (1+INFRATE) ** (DELTIM/365)
      IFACTOR = (1+INFRATE) ** (DELTIM/2/365)
        ENDIF

*-----      Do 10 I = 1, NSTEP
        GASPRO = VRTEG(I) * IVNWEL(I) * DELTIM
        GASREV = GASPRO * GASPRIC * IFACTOR
        OILPRO = VRTEO(I) * IVNWEL(I) * DELTIM
        OILREV = OILPRO * OILPRIC * IFACTOR
        TOTREV = GASREV + OILREV
        NPV = NPV + TOTREV / DFACTOR
        NPV1(I) = NPV
        IFACTOR = IFACTOR * IFEFF
        DFACTOR = DFACTOR * DFEFF
*-----     10 Continue

      Return
      End
```

```
C*****************************************************************
C**** Subroutine  GMSINP
C**** Author      Gunnar Borthne (expanded by James Carroll)
C**** Date        April, 1986
C**** Function    This subroutine reads the input file of the well model.
C*****************************************************************
      Subroutine GMSINP

C---- DESCRIPTION OF INPUT DATA

C---- FORMAT    : TEXT STRING (1 LINE)
C     1. JOBID .: JOB IDENTIFICATION
C
C---- FORMAT    : 4 INTEGERS (1 LINE)
C     1. IHC    : HYDROCARBON TYPE
C                 = 0 : GAS CONDENSATE
C                 = 1 : OIL
C     2. IUNIN  : UNITS IDENTIFIER FOR INPUT DATA
C                 = 0 : METRIC UNITS
C                 = 1 : OIL FIELD UNITS
C     3. IPRT   : PRINT OPTION
C                 = 0 : TABLES OF RESULTS ONLY
C                 = 1 : + ECHO OF INPUT DATA
C                 = 2 : + ITERATION REPORT
C                 = 3 : + RESULTS PRINTED EACH TIMESTEP
C                 = 4 : + A MESSAGE FROM EACH ROUTINE
C     4. IEXE   : EXECUTION MODE
C                 = 0 : MATERIAL BALANCE ONLY
C                 = 1 : MATERIAL BALANCE AND IPR
C                 = 2 : MATERIAL BALANCE, IPR AND TUBING
C
C     ** NOTE : USE ONLY THE UNIT SYSTEM                          OIL
C               CHOSEN WITH IUNIN                    METRIC      FIELD
C                                                    UNITS       UNITS
C                                                    -------     --------
C---- FORMAT    : 6 DOUBLEPRECISION VARIABLES (1 LINE)
C     1. DELTIM : TIMESTEP LENGTH ..................:  YEARS       YEARS
C     2. XMXTIM : LENGTH OF SIMULATION .............:  YEARS       YEARS
C     3. HCPV   : HYDROCARBON PORE VOLUME ..........:    M3          BBL
C     4. PORI   : INITIAL POROSITY .................:     (FRACTION)
C     5. SATWI  : INITIAL WATER SATURATION .........:     (FRACTION)
C     6. CMPF   : FORMATION COMPRESSIBILITY ........:  1/KPA       1/PSI
C
C--- FORMAT    : 6 DOUBLEPRECISION VARIABLES (1 LINE)
C     1. PRM ...: PERMEABILITY .....................:    uM2          MD
C     2. THK ...: RESERVOIR THICKNESS ..............:      M          FT
C     3. RADW ..: WELLBORE RADIUS ..................:      M          FT
C     4. DSKN ..: NON-DARCY FLOW COEFFICIENT
C                 (RATE DEPENDENT SKIN TERM)
C                              ** IF IHC=0 (GAS) ...:  D/SM3       D/SCF
C                              ** IF IHC=1 (OIL) ...:  D/SM3       D/STB
C     5. DPINT .: PRESSURE INCREMENT IN SIMPSON-
C                 INTEGRATION, IPR ROUTINE .........:    KPA          PSI
C
C--- FORMAT    : 5 DOUBLEPRECISION VARIABLES (1 LINE)
C     1. API    : SPECIFIC GRAVITY OF OIL ..........:      DEGREES
C     2. GRVG   : SPECIFIC GRAVITY OF GAS .........:      FRACTION
C     3. TWF    : TEMPERATURE AT BOTTOMHOLE .......:      C          F
C     4. TWH    : TEMPERATURE AT WELLHEAD .........:      C          F
C     5. SIGMAO : OIL/GAS INTERFACIAL TENSION .....:      N/M   lbm/s2
C
C--- FORMAT    : 4 DOUBLEPRECISION VARIABLES (1 LINE)
C     1. CORR ..: CORRELATION NUMBER ...............:      INTEGER
C     2. DIAM   : DIAMETER OF TUBING ...............:      M          FT
C     3. LENGTH : LENGTH OF VERTICAL TUBING ........:      M          FT
C     4. AROUGH : RELATIVE ROUGHNESS OF TUBING .....:      M          FT
C
C---- FORMAT    : 4 DOUBLEPRECISION VARIABLES (1 LINE)
C     1. INFRATE: INFLATION RATE ...................:      FRACTION
C     2. DISRATE: DISCOUNT RATE ....................:      FRACTION
C     3. OILPRIC: PRICE OF OIL .....................:  $/M3        $/BBL
C     4. GASPRIC: PRICE OF GAS .....................:  $/kM3       $/MMSCF
C
C---- FORMAT    : 4 DOUBLEPRECISION VARIABLES (1 LINE)
C     1. N2     : GAS FRACTION OF NITROGEN .........:      FRACTION
C     2. CO2    : GAS FRACTION OF CARBON-DIOXIDE ...:      FRACTION
C     3. H2S    : GAS FRACTION OF HYDROGEN-SULFIDE .:      FRACTION
C     4. RSI    : INITIAL SOLUTION GAS-OIL RATIO ...:   M3/M3    SCF/BBL
C
C---- TABLE OF WELL CONTROL SPECIFICATIONS
C---- FORMAT    : 1 DOUBLEPRECISION + 1 INTEGER + 4 DOUBLEPRECISION
C                 ON EACH LINE
C                 (REPEAT LINES, END WITH -1)
C     1. TTIM   : TIME FOR WELL CONTROL ............:  YEARS       YEARS
C     2. NWELLS : NUMBER OF WELLS ..................:
C     3. TRETFM : FIELD MINIMUM PRODUCTION RATE
C                              ** IF IHC=0 (GAS) ...:  SM3/D       SCF/D
C                              ** IF IHC=1 (OIL) ...:  SM3/D       STB/D
C     4. TRTEFT : FIELD TARGET  PRODUCTION RATE
C                              ** IF IHC=0 (GAS) ...:  SM3/D       SCF/D
C                              ** IF IHC=1 (OIL) ...:  SM3/D       STB/D
C     5. TPWMIN : MINIMUM WELLHEAD PRESSURE ........:    KPA          PSIA
C     6. TSKN ..: TOTAL SKIN (EXCEPT RATE DEPENDENT) .:    DIMENSIONLESS
C     ** NOTE: - TTIM MUST INCREASE DOWN THE COLUMN.
C              - TO END THE TABLE, PUT:  -1  ON THE NEXT LINE.
C
C---- TABLE OF PVT DATA FOR THE OIL PHASE
C---- FORMAT    : 5 DOUBLEPRECISION VARIABLES ON EACH LINE
C                 (REPEAT LINES, END WITH -1)
C     1. TPRS   : PRESSURE FOR PVT DATA ............:      KPA          PSIA
C     2. TVISO  : OIL VISCOSITY ....................:     PA S           CP
C     3. TGORS  : SOLUTION GAS/OIL RATIO,
C                 IN OIL PHASE .....................:  SM3/SM3    SCF/BBL
C     4. TDENRO : DENSITY RATIO (GRAVITY RATIO),
C                 OIL-FROM-GAS / OIL-FROM-OIL ......:     DIMENSIONLESS
C     5. TFVFO  : OIL FORMATION VOLUME FACTOR ......:  (RES.VOL/STD.VOL)
C     ** NOTE: - TPRS MUST INCREASE DOWN THE COLUMN.
C              - TO END THE TABLE, PUT:  -1  ON THE NEXT LINE.
C
C---- TABLE OF PVT DATA FOR THE GAS PHASE
C---- FORMAT    : 4 DOUBLEPRECISION VARIABLES ON EACH LINE
C                 (REPEAT LINES, END WITH -1)
C                 ** NOTE : SAME PRESSURES AS ABOVE
C
C     1. TVISG  : GAS VISCOSITY ....................:      PA S           CP
C     2. TOGRS  : SOLUTION OIL/GAS RATIO,
C                 IN GAS PHASE .....................:  SM3/SM3 STB/MMSCF
C     3. TDENRG : DENSITY RATIO (GRAVITY RATIO),
C                 GAS-FROM-OIL / GAS-FROM-GAS ......:     DIMENSIONLESS
C     4. TFVFG  : GAS FORMATION VOLUME FACTOR ......:  (RES.VOL/STD.VOL)
C
C     ** NOTE: TO END THE TABLE, PUT:  -1  ON THE NEXT LINE
C
C---- TABLE OF RELATIVE PERMEABILITY VS. SATURATION
C---- FORMAT    : 3 DOUBLEPRECISION VARIABLES ON EACH LINE
C                 (REPEAT LINES, END WITH -1)
C     1. TSATG  : GAS SATURATION ...................:      (FRACTION)
C     2. TPRMRO : RELATIVE PERMEABILITY TO OIL .....:      (FRACTION)
```

```
*     3. TPRMRG : RELATIVE PERMEABILITY TO GAS .....!        (FRACTION)
*     ** NOTE: - TSATG MUST INCREASE DOWN THE COLUMN.
*              - TO END THE TABLE, PUT:  -1  ON THE NEXT LINE.
*
*---- VARIABLES AND CONSTANTS
*
      Implicit Double Precision (A-Z)

      Integer N1, N2, NWELLS, I, IERR, IEXE, IHC, IPRT, IUNIN, NWT,
     *    CORR, NPVT, NRP, IC, IT, NSTEP
      Character ZJOBID*60, ZC(8)*6
      Parameter (N1 = 100,N2 = 500)

      Dimension TPWMIN(N1), TRTEFM(N1), TRTEFT(N1), TSKN(N1), TTIM(N1),
     *    TDENRG(100), TDENRO(100), TFVFG(100), TFVFGX(100),
     *    TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100), TPRMRO(100),
     *    PRMLGO(100), TPRS(100), TSATG(100), TVISG(100), TVISO(100),
     *    NWELLS(N1), IC(8), IT(8)

*---- COMMON BLOCKS

      Common /CONTAM/ NITRO, CO2, H2S, NACL, RSI
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     *    CORR
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     *    VOLBW
      Common /MBAL2/ AO2, AG2, RO2, RG2, RGAV
      Common /MBAL3/ DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     *    SATG1, SATO1, VISG, VISO, XMBAL1
      Common /IPR1/ DPINT, DSKN, PRM, RADEQ, SKN, THK
      Common /ICOUNT/ IC, IT
      Common /FINANCE/ INFRATE, DISRATE, OILPRIC, GASPRIC
      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     *    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /TRANS2/ C1, C2, C3, C4, C5, C6, C7, C8, C9
      Common /TRANS3/ ZC, ZJOBID
      Common /SEP/ TSEP, PSEP

*---- CONVERSION FACTORS

      C1 = 0.100000D-02
      C2 = 0.404686D+04
      C3 = 0.304800D+00
      C4 = 0.689476D+01
      C5 = 0.986923D-03
      C6 = 0.158987D+00
      C7 = 0.561458D+01
      C8 = 0.283169D-01
      C9 = 0.555556D+00

*---- DATA STATEMENTS

      Data ZC /'RATE', 'WHPRS', 'RESPRS', 'MATBAL', 'IPR', 'FNPRS',
     *    'ISGN', 'INTPL'/

*---- START EXECUTION

      Print *, ' '
      Print *, '                        ',
     *    ' ----------------GMS ------------------------- '
      Print *, '                        ',
     *    '| Version 1.1                                  |'
      Print *, '                        ',
     *    '|   A general material balance and inflow      |'
      Print *, '                        ',
     *    '|   performance simulation model for oil and   |'
      Print *, '                        ',
     *    '|   gas condensate reservoirs.                 |'
      Print *, '                        ',
     *    '|     Gunnar Borthne, NTH, 1986                |'
      Print *, '                        ',
     *    '|                                              |'
      Print *, '                        ',
     *    '| Version 2.0                                  |'
      Print *, '                        ',
     *    '|   Expanded to include vertical multiphase    |'
      Print *, '                        ',
     *    '|   flow, flow through chokes, and nonlinear   |'
      Print *, '                        ',
     *    '|   optimization of decision variables.        |'
      Print *, '                        ',
     *    '|     James Carroll, Stanford University, 1990 |'
      Print *, '                        ',
     *    ' ---------------------------------------------- '
      Print *, ' '

*---- READ INPUT DATA FROM FILE,
*     CONVERT TO METRIC UNITS
*     TEST INPUT DATA IN METRIC UNITS

      IERR = 0
      Print *, 'Reading and testing input data...'
      Print *

*---- READ DATA LINE 1, JOB IDENTIFICATION

      Read (2,5000) ZJOBID

*---- READ AND TEST DATA LINE 2

      Call SKIP(2)
      Read (2,*) IHC, IUNIN, IPRT, IEXE
      Call ITEST(IHC,0,1,'IHC',IERR)
      Call ITEST(IUNIN,0,1,'IUNIN',IERR)
      Call ITEST(IPRT,0,4,'IPRT',IERR)
      Call ITEST(IEXE,0,2,'IEXE',IERR)

*---- READ AND TEST DATA LINE 3

      Call SKIP(2)
      Read (2,*) DELTIM, XMXTIM, HCPV, PORI, SATWI, CMPF
      If (IUNIN.EQ.1) Then
        HCPV = HCPV * C6
        CMPF = CMPF / C4
      End If
      Call TEST(DELTIM,0D+0,1D+0,'DELTIM',IERR)
      Call TEST(XMXTIM,0D+0,400D+0,'XMXTIM',IERR)
      Call TEST(HCPV,0D+0,1D+20,'HCPV',IERR)
      Call TEST(PORI,0D+0,1D+0,'PORI',IERR)
      Call TEST(SATWI,0D+0,1D+0,'SATWI',IERR)
      Call TEST(CMPF,0D+0,1D-1,'CMPF',IERR)
      DELTIM = DELTIM * 365.
      XMXTIM = XMXTIM * 365.

*---- READ AND TEST DATA LINE 4
```

```
      Call SKIP(2)
      Read (2,*) PRM, THK, RADW, DSKN, DPINT
      If (IUNIN.EQ.1) Then
        PRM = PRM * C5
        THK = THK * C3
        RADW = RADW * C3
        If (IHC.EQ.0) DSKN = DSKN / C8
        If (IHC.EQ.1) DSKN = DSKN / C6
        DPINT = DPINT * C4
      End If
      Call TEST(PRM,0D+0,10000D+0,'PRM',IERR)
      Call TEST(THK,0D+0,1000D+0,'THK',IERR)
      Call TEST(RADW,0D+0,3D+1,'RADW',IERR)
      Call TEST(DPINT,10D+0,10000D+0,'DPINT',IERR)

*---- READ DATA LINE 5, 6, 7, & 8

      Call SKIP(2)
      Read (2,*) API, GRVG, TWF, TWH, SIGMAO
      Call TEST(API,1D+0,1D+2,'API',IERR)
      Call TEST(GRVG,0.5D+0,1D+1,'GRVG',IERR)
      Call TEST(TWF,0D+0,0.25D+03,'TWF',IERR)
      Call TEST(TWH,0D+0,0.25D+03,'TWH',IERR)
      Call TEST(SIGMAO,0D+0,1D+4,'SIGMA',IERR)
      If (IUNIN.EQ.1) Then
        TWF = TWF * C9
        TWH = TWH * C9
        SIGMAO = SIGMAO * 14.60956181
      End If

      Call SKIP(2)
      Read (2,*) CORR, DIAM, LENGTH, AROUGH, TSEP, PSEP
      If (IUNIN.EQ.1) Then
        AROUGH = AROUGH * C3
        DIAM = DIAM * C3
        LENGTH = LENGTH * C3
        TSEP = TSEP * C9
        PSEP = PSEP * C4
      End If
      Call ITEST(CORR,1,3,'CORR',IERR)
      Call TEST(DIAM,0D+0,0.5D+0,'DIAM',IERR)
      Call TEST(LENGTH,0.1D+01,0.1D+05,'LENGTH',IERR)
      Call TEST(AROUGH,0D+0,1D+0,'AROUGH',IERR)
      Call TEST(TSEP,0D+0,1D+3,'TSEP',IERR)
      Call TEST(PSEP,0D+0,1D+5,'PSEP',IERR)

      Call SKIP(2)
      Read (2,*) INFRATE, DISRATE, OILPRIC, GASPRIC
      Call TEST(INFRATE,0D+0,1D+0,'INFRATE',IERR)
      Call TEST(DISRATE,0D+0,1D+0,'DISRATE',IERR)
      Call TEST(OILPRIC,0D+0,1D+3,'OILPRIC',IERR)
      Call TEST(GASPRIC,0D+0,1D+5,'GASPRIC',IERR)
      if (IUNIN.EQ.1) THEN
        OILPRIC = OILPRIC * C4
        GASPRIC = GASPRIC / (C8 * 1000000.)
      else
        GASPRIC = GASPRIC / 1D6
      endif

      Call SKIP(2)
      Read (2,*) NITRO, CO2, H2S, RSI
      Call TEST(NITRO,0D+0,1D+0,'NITRO',IERR)
      Call TEST(CO2,0D+0,1D+0,'CO2',IERR)
      Call TEST(H2S,0D+0,1D+0,'H2S',IERR)
      Call TEST(RSI,0D+0,1D+3,'RSI',IERR)
      If (IUNIN.EQ.1) RSI = RSI / 5.615

*---- READ AND TEST DATA TABLE 1

      I = 0
   10 Continue
      I = I + 1
      Call SKIP(2)
      Read (2,*) X0
      If (X0.GE.0.) Then
        Backspace (2)
        Read (2,*) TTIM(I), NWELLS(I), TRTEFM(I), TRTEFT(I), TPWMIN(I),
     *       TSKN(I)
*         Note: Input time in years, internal time in days.
        TTIM(I) = TTIM(I) * 365
        NWT = I
        X1 = C8
        If (IHC.EQ.1) X1 = C6
        If (IUNIN.EQ.1) Then
          TRTEFM(I) = TRTEFM(I) * X1
          TRTEFT(I) = TRTEFT(I) * X1
          TPWMIN(I) = TPWMIN(I) * C4
        End If
        Call TEST(TTIM(I)/365,0D+0,400D+0,'TTIM',IERR)
        If (I.GT.1) Call TESTGE(TTIM(I-1)/365,TTIM(I)/365,'TTIM',IERR)
        Call ITEST(NWELLS(I),1,500,'NWELLS',IERR)
        Call TEST(TRTEFM(I),0D+0,1D+7,'TRTEFM',IERR)
        Call TEST(TRTEFT(I),0D+0,2D+9,'TRTEFT',IERR)
        Call TEST(TPWMIN(I),0D+0,1D+5,'TPWMIN',IERR)
        Call TEST(TSKN(I),-1D+2,1D+2,'TSKN',IERR)
        Go To 10
      End If
      Call TEST(TTIM(1)/365,0D+0,0D+0,'TTIM(1)',IERR)
      Call ITEST(NWT,1,N1-1,'NWT >= N1',IERR)

*---- READ AND TEST DATA TABLE 2

      I = 0
   20 Continue
      I = I + 1
      Call SKIP(2)
      Read (2,*) X0
      If (X0.GE.0.) Then
        Backspace (2)
        Read (2,*) TPRS(I), TVISO(I), TGORS(I), TDENRO(I), TFVFO(I)
        NPVT = I
        If (IUNIN.EQ.1) Then
          TPRS(I) = TPRS(I) * C4
          TVISO(I) = TVISO(I) * C1
          TGORS(I) = TGORS(I) / C7
        End If
        Call TEST(TPRS(I),0D+0,5D+5,'TPRS',IERR)
        If (I.GT.1) Call TESTGE(TPRS(I-1),TPRS(I),'TPRS',IERR)
        Call TEST(TVISO(I),0D+0,.1D+0,'TVISO',IERR)
        Call TEST(TGORS(I),0D+0,5D+3,'TGORS',IERR)
        Call TEST(TDENRO(I),0D+0,10D+0,'TDENRO',IERR)
        Call TEST(TFVFO(I),0D+0,10D+0,'TFVFO',IERR)
        Go To 20
      End If

*---- READ AND TEST DATA TABLE 3

      I = 0
```

```
   30 Continue
      I = I + 1
      Call SKIP(2)
      Read (2,*) X0
      If (X0.GE.0.) Then
        Backspace (2)
        Read (2,*) TVISG(I), TOGRS(I), TDENRG(I), TFVFG(I)
        TFVFGX(I) = 1 / TFVFG(I)
        If (IUNIN.EQ.1) Then
          TVISG(I) = TVISG(I) * C1
          TOGRS(I) = TOGRS(I) * C7 / 1D+6
        End If
        Call TEST(TVISG(I),0D+0,.01D+0,'TVISG',IERR)
        Call TEST(TOGRS(I),0D+0,1D+0,'TOGRS',IERR)
        Call TEST(TDENRG(I),0D+0,10D+0,'TDENRG',IERR)
        Call TEST(TFVFG(I),0D+0,2D+0,'TFVFG',IERR)
        Go To 30
      End If
      Call ITEST(NPVT,I-1,I-1,'NPVT TABLES DIFFERENT LENGTH',IERR)
      Call ITEST(NPVT,2,N1,'NPVT',IERR)

*---- READ AND TEST DATA TABLE 4

      I = 0
   40 Continue
      I = I + 1
      Call SKIP(2)
      Read (2,*) X0
      If (X0.GE.0.) Then
        Backspace (2)
        Read (2,*) TSATG(I), TPRMRO(I), TPRMRG(I)
C---    Write(6,*)TSATG(I)
        NRP = I
        Call TEST(TSATG(I),0D+0,1D+0,'TSATG',IERR)
        If (I.GT.1) Call TESTGE(TSATG(I-1),TSATG(I),'TSATG',IERR)
        Call TEST(TPRMRO(I),0D+0,1D+0,'TPRMRO',IERR)
        Call TEST(TPRMRG(I),0D+0,1D+0,'TPRMRG',IERR)
        Go To 40
      End If
      Call ITEST(NRP,2,N1,'NRP',IERR)
      Call TEST(TSATG(1),0D+0,0D+0,'TSATG, FIRST VALUE',IERR)
      Call TEST(TSATG(NRP),1D+0-SATWI,1D+0,'TSATG, LAST VALUE',IERR)

      Close (2)
      TTIM(NWT+1) = XMXTIM

      Return
 5000 Format (A)
      End
*****************************************************************
**** Subroutine  GMSECHO
**** Author      Gunnar Borthne (expanded by James Carroll)
**** Date        April, 1986
**** Function    This subroutine echoes the input data of the well model.
*****************************************************************
      Subroutine GMSECHO

*     VARIABLES AND CONSTANTS

      Implicit Double Precision (A-Z)

      Integer N1, N2, NWELLS, I, IERR, IEXE, IHC, IPRT, IUNIN, NWT,
     *    CORR, NPVT, NRP, IC, IT, NSTEP
      Character ZJOBID*60, ZC(8)*6
      Parameter (N1 = 100,N2 = 500)

      Dimension TPWMIN(N1), TRTEFM(N1), TRTEFT(N1), TSKN(N1), TTIM(N1),
     *    TDENRG(100), TDENRO(100), TFVFG(100), TFVFGX(100),
     *    TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100), TPRMRO(100),
     *    PRMLGO(100), TPRS(100), TSATG(100), TVISG(100), TVISO(100),
     *    NWELLS(N1), IC(8), IT(8)

*---- COMMON BLOCKS

      Common /CONTAM/ NITRO, CO2, H2S, NACL, RSI
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     *    CORR
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     *    VOLBW
      Common /MBAL2/ AO2, AG2, RO2, RG2, RGAV
      Common /MBAL3/ DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     *    SATG1, SATO1, VISG, VISO, XMBAL1
      Common /IPR1/ DPINT, DSKN, PRM, RADEQ, SKN, THK
      Common /ICOUNT/ IC, IT
      Common /FINANCE/ INFRATE, DISRATE, OILPRIC, GASPRIC
      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     *    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /TRANS2/ C1, C2, C3, C4, C5, C6, C7, C8, C9
      Common /TRANS3/ ZC, ZJOBID

*---- WRITE INPUT DATA,
*     (BOTH METRIC UNITS AND OIL FIELD UNITS)

      Write (3,5000)
      If (IPRT.NE.0) Then
        Print *, 'Writing Input Data...'
        Print *
        Write (*,4)
        Write (4,4)
    4   format (4x,'Iter',7x,'Dtbg',12x,'Pbep',9x,'Obj Var',8x,'Side',
     *          7x,'Conv?')
        Write (3,5400) 'REVIEW OF INPUT DATA'
        Write (3,*) '--------------------'

*---- WRITE DATA LINE 1 AND 2

        Write (3,5500) ZJOBID, IHC, IUNIN
        Write (3,5600) IPRT, IEXE
        Write (3,5700) NWT, NPVT, NRP
        X1 = DELTIM / 365.

*---- WRITE DATA LINE 3

        X2 = XMXTIM / 365.
        X3 = HCPV / C6
        X4 = CMPF * C4
        Write (3,5800) X1, X2, HCPV, X3, PORI, SATWI, CMPF, X4

*---- WRITE DATA LINE 4

        X1 = PRM / C5
        X2 = THK / C3
        X3 = RADW / C3
        X4 = DSKN * C8
        X5 = DSKN * C6
        X6 = DPINT / C4
```

```
      Write (3,5900) PRM, X1, THK, X2, RADW, X3, DSKN, X4, X5, DPINT,
     *     X6

*---- WRITE DATA LINES 5-8

      Write (3,6000) ZJOBID
      Write (3,6100) 'TWF', TWF, 'TWH', TWH
      Write (3,6100) 'SIGMAO', SIGMAO, 'DIAM', DIAM
      Write (3,6100) 'LENGTH', LENGTH, 'AROUGH', AROUGH
      Write (3,6100) 'INFRATE', INFRATE, 'DISRATE', DISRATE
      Write (3,6100) 'OILPRIC', OILPRIC, 'GASPRIC', GASPRIC
      Write (3,6100) 'N2', NITRO, 'CO2', CO2
      Write (3,6100) 'H2S', H2S, 'RSI', RSI
      Write (3,6100) 'API', API, 'GRVG', GRVG
      If (CORR.EQ.1) Then
        Write (3,6200) 'CORR', CORR, 'Aziz, Govier, & Fogarasi'
      Else If (CORR.EQ.2) Then
        Write (3,6200) 'CORR', CORR, 'Hagedorn & Brown'
      Else If (CORR.EQ.3) Then
        Write (3,6200) 'CORR', CORR, 'Orkeszewski'
      End If

*---- WRITE DATA TABLE 2

      Write (3,6300) ZJOBID
      Do 10 I = 1, NPVT
        X1 = TPRS(I) / C4
        X2 = TVISO(I) / C1
        X3 = TGORS(I) * C7
        Write (3,6400) I, TPRS(I), X1, TVISO(I), X2, TGORS(I), X3,
     *      TDENRO(I), TFVFO(I)
   10   Continue
        Write (3,5200)

*---- WRITE DATA TABLE 3

      Write (3,6500) ZJOBID
      Do 20 I = 1, NPVT
        X1 = TPRS(I) / C4
        X2 = TVISG(I) / C1
        X3 = TGGRS(I) * 1D+6 / C7
        Write (3,6600) I, TPRS(I), X1, TVISG(I), X2, TGGRS(I), X3,
     *      TDENRG(I), TFVFG(I)
   20   Continue
        Write (3,5200)

*---- WRITE DATA TABLE 4

      Write (3,6700) ZJOBID
C---  Write(6,*)'NPP-table4'
      Do 30 I = 1, NRP
        Write (3,6800) I, TSATG(I), TPRMRO(I), TPRMRG(I)
C     Write(6,*)TSATG(I)
   30   Continue
        Write (3,5100)

*     WRITE DATA TABLE 1

      If (IHC.EQ.0) Then
        Write (3,6900) ZJOBID
        X0 = 1. / C8
      Else If (IHC.EQ.1) Then
        Write (3,7000) ZJOBID
        X0 = 1. / C6
      End If
      Do 40 I = 1, NWT
        X1 = TTIM(I) / 365.
        X2 = TTIM(I+1) / 365.
        X3 = TRTEFM(I) * X0
        X4 = TRTEFT(I) * X0
        X5 = TPWMIN(I) / C4
        Write (3,7100) I, TTIM(I), TTIM(I+1), X1, X2, NWELLS(I),
     *      TRTEFM(I), X3, TRTEFT(I), X4, TPWMIN(I), X5, TSKN(I)
   40   Continue
        Write (3,5300)
      End If

*---- IF ERRORS ARE DETECTED: WRITE MESSAGE AND STOP RUN

      If (IERR.GT.0) Then
        Write (3,*)
        Write (3,*) IERR, ' INPUT DATA ERROR(S) DETECTED'
        Print *, IERR, ' INPUT DATA ERROR(S) DETECTED'
        Stop
      End If
      Return

*---- FORMAT STATEMENTS

 5000 Format (1X,'*****************************************************
     *******/1X,'*****************************************************
     *******/1X,'**
     *     **'/1X,'**
     *     **'/1X,'**          ********      **        *      ********
     *     **'/1X,'**        **      **     **      **      **      **
     *     **'/1X,'**        **             **      **      **
     *     **'/1X,'**        **             **  *   *  *      ********
     *     **'/1X,'**        **   *****     **  * *   *             **
     *     **'/1X,'**        **        *    **   *   *      **      **
     *     **'/1X,'**         *********     **        *      ********
     *     **'/1X,'**
     *     **'/1X,'**
     *     **'/1X,'**** A GENERAL MATERIAL-BALANCE AND INFLOW-PERFORMANCE
     * ******'/1X,'** SIMULATION MODEL FOR OIL AND GAS-CONDENSATE RESERVO
     *IRS **'/1X,'********************** VERSION 1.1 ******************
     *******/1X,'***************** AUTHOR: GUNNAR BORTHNE *************
     *******')
 5100 Format (1X,39('-'))
 5200 Format (1X,100('-'))
 5300 Format (1X,126('-'))
 5400 Format (//,1X,A)
 5500 Format (///1X,'JOB IDENTIFICATION : ',A60//1X,
     *     'IHC     : HYDROCARBON TYPE ......................: ',I5/1X,
     *     '          = 0 : GAS CONDENSATE '/1X,'          = 1 : OIL '/1X,
     *     'IUNIN   : UNITS IDENTIFIER FOR INPUT DATA .......: ',I5/1X,
     *     '          = 0 : METRIC UNITS '/1X,
     *     '          = 1 : OIL FIELD UNITS ')
 5600 Format (1X,'IPRT    : PRINT OPTION ..........................: ',
     *     I5/1X,'          = 0 : TABLES OF RESULTS ONLY '/1X,
     *     '          = 1 : + ECHO OF INPUT DATA '/1X,
     *     '          = 2 : + ITERATION REPORT '/1X,
     *     '          = 3 : + RESULTS PRINTED EACH TIMESTEP'/1X,
     *     '          = 4 : + A MESSAGE FROM EACH ROUTINE '/1X,
     *     'IEXE    : EXECUTION MODE ........................ ',I5/1X,
     *     '          = 0 : MATERIAL BALANCE ONLY '/1X,
     *     '          = 1 : MATERIAL BALANCE AND IPR '/1X,
     *     '          = 2 : MATERIAL BALANCE, IPR AND TUBING ')
 5700 Format (1X,'NWT     : NUMBER OF WELL CONTROL SPECIFICATIONS ..: ',
```

```
     •    I5/1X,'NPVT    : NUMBER OF PVT DATA INPUT LINES .....: ',I5
     •    /1X,'NRP     : NUMBER OF REL. PERM. DATA INPUT LINES ..: ',I5)
5800 Format (1X,'DELTIM : TIMESTEP LENGTH (YEARS) ..............: ',
     •    G12.5/1X,
     •    'XMXTIM : LENGTH OF SIMULATION (YEARS) .....: ',G12.5/1
     •    X,'HCPV    : HYDROCARBON PORE VOLUME (M3) .....: ',G12.5/
     •    1X,'                                (BBL) .....: ',G12.5
     •    /1X,'PORI    : INITIAL POROSITY (FRACTION) .....: ',G12
     •    .5/1X,'SATWI   : INITIAL WATER SATURATION (FRACTION) ....: ',
     •    G12.5/1X,
     •    'CMPF    : FORMATION COMPRESSIBILITY (1/KPA) .....: ',G12.5/1
     •    X,'                                 (1/PSI) .....: ',G12.5)
5900 Format (//1X,
     •    'PRM     : PERMEABILITY (uM2) .....................: ',G12.5/1
     •    X,'                         (MD) .... ..............: ',G12.5/
     •    1X,'THK     : RESERVOIR THICKNESS (M) ..............: ',G12.5
     •    /1X,'                           (FT) ..............: ',G12
     •    .5/1X,'RADW    : WELLBORE RADIUS  (M) ..................: ',
     •    G12.5/1X,
     •    '                           (FT) ..................: ',G12.5/1
     •    X,'DSKN    : NON-DARCY FLOW COEFFICIENT (D/M3) .....: ',G12.5/
     •    1X,'                                (D/FT3) .....: ',G12.5
     •    /1X,'                                (D/BBL) .....: ',G12
     •    .5/1X,'DPINT   : PRESSURE INCREMENT IN SIMPSON- '/1X,
     •    '          INTEGRATION (KPA) ....................: ',F11.0/1
     •    X,'                        (PSI) ....................: ',F12.1)
6000 Format (//1X,'JOB IDENTIFICATION : ',A60//)
6100 Format (1X,2(A15,1X,G15.5,1X))
6200 Format (1X,A15,1X,I5,1X,A30)
6300 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,
     •    'PRESSURE-DEPENDENT PROPERTIES, OIL'/1X,100('-')/1X,
     •    '             PRESSURE                OIL VISCOSITY     ',
     •    '  SOLUTION GAS/OIL RATIO    SPECIFIC    OIL FVF  '/1X,
     •    '     ----------------------  ----------------------',
     •    '  ----------------------     GRAVITY  -----------'/1X,
     •    'NO.       KPA         PSIA         PA S          CP     ',
     •    '    SM3/SM3      SCF/BBL    RATIO,OIL  RES/STD VOL'/1X,
     •    '---  ----------  ----------  ----------  ----------',
     •    '  ----------  ----------  ----------  ----------')
6400 Format (1X,I3,F12.0,F12.1,6G12.5)
6500 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,
     •    'PRESSURE-DEPENDENT PROPERTIES, GAS'/1X,100('-')/1X,
     •    '             PRESSURE                GAS VICOSITY     ',
     •    '  SOLUTION OIL/GAS RATIO    SPECIFIC    GAS FVF  '/1X,
     •    '     ----------------------  ----------------------',
     •    '  ----------------------     GRAVITY  -----------'/1X,
     •    'NO.       KPA         PSIA         PA S          CP     ',
     •    '    SM3/SM3    BBL/MMSCF    RATIO,GAS  RES/STD VOL'/1X,
     •    '---  ----------  ----------  ----------  ----------',
     •    '  ----------  ----------  ----------  ----------')
6600 Format (1X,I3,F12.0,F12.1,6G12.5)
6700 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,
     •    'RELATIVE PERMEABILITIES AS FUNCTIONS OF GAS SATURATION'/1X,39
     •    ('-')/1X,'          GAS          OIL          GAS    '/1X,
     •    'NO.  SATURATION   REL.PERM     REL.PERM '/1X,
     •    '---  ----------  ----------  ----------')
6800 Format (1X,I3,3G12.5)
6900 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,'WELL CONTROL'/1X,126(
     •    '-')/1X,'                                        NUM-',
     •    '       FIELD MINIMUM            FIELD TARGET      ',
     •    '     MINIMUM WELLHEAD    ',' SKIN FACTOR'/1X,
     •    '                  TIME                    BER ',
     •    '    GAS PRODUCTION RATE     GAS PRODUCTION RATE ',
     •    '         PRESSURE        ',','              '/1X,
     •    '     ------------------------------  OF  ',
     •    '  ----------------------  ----------------------',
     •    '  ----------------------',' ----------'/1X,
     •    'NO.         D              YEARS     WELLS ',
     •    '      SM3/D       SCF/D       SM3/D       SCF/D  ',
     •    '       KPA         PSIA    ','   DIM.LESS '/1X,
     •    '---  ----------  ----------  ----------  -----',
     •    '  ----------  ----------  ----------  ----------',
     •    '  ----------  ----------',' ----------')
7000 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,'WELL CONTROL'/1X,126(
     •    '-')/1X,'                                        NUM-',
     •    '       FIELD MINIMUM            FIELD TARGET      ',
     •    '     MINIMUM WELLHEAD    ',' SKIN FACTOR'/1X,
     •    '                  TIME                    BER ',
     •    '    OIL PRODUCTION RATE     OIL PRODUCTION RATE ',
     •    '         PRESSURE        ',','              '/1X,
     •    '     ------------------------------  OF  ',
     •    '  ----------------------  ----------------------',
     •    '  ----------------------',' ----------'/1X,
     •    'NO.         D              YEARS     WELLS ',
     •    '      SM3/D       STB/D       SM3/D       STB/D  ',
     •    '       KPA         PSIA    ',','   DIM.LESS '/1X,
     •    '---  ----------  ----------  ----------  -----',
     •    '  ----------  ----------  ----------  ----------',
     •    '  ----------  ----------',' ----------')
7100 Format (1X,I3,F8.1,' -',F7.1,F7.2,' -',F6.2,I6,1X,4G12.5,F12.0,F12
     •    .1,F12.2)
      End
C*****************************************************************
C**** Subroutine  GMSBODY
C**** Author      Gunnar Borthne (modified by James Carroll)
C**** Date        April, 1986
C**** Function    This program is a material balance and inflow
C****   performance simulation model for oil and gas-condensate
C****   reservoirs.  The GMS formulations can utilize PVT data from the
C****   general PVT formulation which is based on flash separation of the
C****   oil  and  gas,  separately,  to  stock-tank conditions (STC). (For
C****   these data, flash has been done  from  each  pressure  step  in  a
C****   differential-liberation or constant-volume depletion process.  The
C****   gas/oil ratio in oil, oil/gas ratio in gas, and densities  at  STC
C****   are functions of the feed pressure to the flash process.)
C****        The inflow-performance procedure utilizes the pseudopressure
C****   concept with numerical integration of a pressure function.
C****   (Simpson's integration method is used.)
C****        A field with multiple wells can be simulated.  The field
C****   target and minimum production rates, minimum bottomhole or
C****   wellhead pressure, and the number of wells should be specified on
C****   the input as functions of time. All the wells are considered to be
C****   "equivalent" and produce at the same average reservoir conditions
C****   and have the same inflow-performance relationship.  Calculations
C****   are performed on a well-basis and multiplied by the number of
C****   wells to get field quantities.
C****
C****   For more information, see diploma theses by
C****        1) Gunnar Borthne, NTH, 1986;
C****        2) James Carroll, Stanford, 1990.
C*****************************************************************
      Subroutine GMSBODY

C---- VARIABLES AND CONSTANTS

      Implicit Double Precision (A-Z)

      Integer N1, N2, IVNWEL, NWELLS, I, IERR, IEXE, IHC, IPRT, IUNIN,
```

```
     *    IWCTR, J, K, NWELL, NWT, CORR, NPVT, NRP, IC, IT, NSTEP
      Character ZJOBID*60, ZC(8)*6
      Logical QWCTR, QSTOP, ICRIT
      Parameter (N1 = 100,N2 = 500,N3=5)

      Dimension TPWMIN(N1), TRTEFM(N1), TRTEFT(N1), TSKN(N1), TTIM(N1),
     *    VGASP(0:N2), VGOR(0:N2), VOILP(0:N2), VPRSR(0:N2),
     *    VPRSWF(0:N2), VPRSWH(0:N2), VRTEG(0:N2), VRTEO(0:N2), AN(8),
     *    VTIME(0:N2), TDENRG(100), TDENRO(100), TFVFG(100),
     *    TFVFGX(100), TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100),
     *    TPRMRO(100), PRMLGO(100), TPRS(100), TSATG(100), TVISG(100),
     *    TVISO(100), IVNWEL(0:N2), NWELLS(N1), IC(8), IT(8)
      Dimension Pass(N3), VRTEO1(0:N2), VRTEG1(0:N2), NPV1(0:N2)

*---- COMMON BLOCKS
      Common /PASS/ Pass, Pass4
      Common /CONTAM/ NITRO, CO2, H2S, NACL, RSI
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     *    CORR
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     *    VOLBW
      Common /MBAL2/ AO2, AG2, RO2, RG2, RGAV
      Common /MBAL3/ DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     *    SATG1, SATO1, VISG, VISO, XMBAL1
      Common /IPR1/ DPINT, DSKN, PRM, RADEQ, SKN, THK
      Common /ICOUNT/ IC, IT
      Common /FINANCE/ INFRATE, DISRATE, OILPRIC, GASPRIC
      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     *    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /TRANS2/ C1, C2, C3, C4, C5, C6, C7, C8, C9
      Common /TRANS3/ ZC, ZJOBID
      Common /TRANS4/ VGASP, VGOR, VOILP, VPRSR, VPRSWF, VPRSWH, VRTEG,
     *    VRTEO, VTIME, IWCTR, NWELL, OILTI, GASTI, IVNWEL, K
      Common /RESULTS/ NPV, NPV1
      Common /FLOW/ ICRIT
      Common /SEP/ TSEP, PSEP
      Common /BHRATE/ VRTEO1, VRTEG1

*---  INITALIZE ARRAYS

        Do I=0,N2
      VTIME(I) = 0.000d0
      VPRSR(I) = 0.000d0
      VPRSWF(I) = 0.000d0
      VPRSWH(I) = 0.0000d0
      VGOR(I) = 0.000d0
      VGASP(I) = 0.0000d0
      VOILP(I) = 0.000d0
      VRTEO(I) = 0.0d0
      VRTEG(I) = 0.0d0
      VRTEO1(I) = 0.0d0
      VRTEG1(I) = 0.0d0
      NPV1(I) = 0.0d0
        End do  .

*---- INITIALIZE VARIABLES

      DIAM=Pass(1)
      PSEP=Pass(N3)

        Do I=1,N1
        TPWMIN(I)=PSEP
        ENDDO

      ICRIT = .FALSE.
      K = 0
      IWCTR = 1
      QWCTR = .TRUE.
      QSTOP = .FALSE.
      PI = 3.141592654
      NWELLS(NWT+1) = 0
      PVTMIN = MIN(TPRS(1),TPRS(NPVT))
      PVTMAX = MAX(TPRS(1),TPRS(NPVT))
      PRSI = PVTMAX
      VOLB = HCPV / PORI / (1-SATWI)
      AREA = VOLB / THK
      RADE = SQRT(AREA/PI+RADW*RADW)

*---- INITIALIZE PRECALCULATED ARRAY OF LOG10 (REL.PERM. RATIO)

      Do 10 I = 1, NRP
        If (TPRMRO(I).LE.1D-15) Then
          PRMLGO(I) = 15.
        Else If (TPRMRG(I).EQ.0.) Then
          PRMLGO(I) = -15.
        Else
          PRMLGO(I) = LOG10(TPRMRG(I)/TPRMRO(I))
        End If
        If (PRMLGO(I).GT.15.) PRMLGO(I) = 15.
   10 Continue

*---- INITIALIZE VARIABLES TO BE USED BY MBAL
C---     Write(6,*)'Init 1102'
         Do I=1,NRP
C---     Write(6,*)TSATG(I)
         Enddo

      Call INTPL(TPRS,TGORS,PRSI,1,NPVT,GORSI,FRAC,J)
      FVFOI = TFVFO(J) + FRAC * (TFVFO(J+1)-TFVFO(J))
      VISOI = TVISO(J) + FRAC * (TVISO(J+1)-TVISO(J))
      OGRSI = TOGRS(J) + FRAC * (TOGRS(J+1)-TOGRS(J))
      VISGI = TVISG(J) + FRAC * (TVISG(J+1)-TVISG(J))
      DENROI = TDENRO(J) + FRAC * (TDENRO(J+1)-TDENRO(J))
      DENRGI = TDENRG(J) + FRAC * (TDENRG(J+1)-TDENRG(J))

*---- INTERPOLATE IN PRECALCULATED ARRAY OF RECIPROCAL OF FVFG

      FVFGI = 1 / (TFVFGX(J)+FRAC*(TFVFGX(J+1)-TFVFGX(J)))

      If (IHC.EQ.0) SATOI = 0.
      If (IHC.EQ.1) SATOI = 1 - SATWI
      SATGI = 1 - SATWI - SATOI
      Call INTPL(TSATG,PRMLGO,SATG1,1,NRP,Y,FRAC,J)
      PRMGOI = (1D+1) ** Y
      XMOBR = PRMGOI * VISOI / VISGI
      AO1 = PORI * (SATWI/FVFOI+SATGI*OGRSI*DENROI/FVFGI)
      AG1 = PORI * (SATGI/FVFGI+SATOI*GORSI*DENRGI/FVFOI)
      RO1 = (1.+OGRSI*DENROI*XMOBR*FVFOI/FVFGI)
      RG1 = GORSI * DENRGI + XMOBR * FVFOI / FVFGI

*---- CALCULATE INITIAL GAS AND OIL IN PLACE

      GASTI = 0.
      OILTI = 0.
      If (IHC.EQ.0) GASTI = HCPV / FVFGI
      If (IHC.EQ.1) OILTI = HCPV / FVFOI
```

```
*---- INITIALIZE REPORT ARRAYS

      VTIME(0) = 0.
      VTIME(1) = DELTIM
      VPRSR(0) = PRSI
      VPRSWF(0) = 0.
      VPRSWH(0) = 0.
      VGOR(0) = 0.
      VGASP(0) = 0.
      VOILP(0) = 0.

*---- INITIALIZE ITERATION COUNTERS

      Do 20 I = 1, 8
        IC(I) = 0
        IT(I) = 0
        AN(I) = 0.
   20 Continue

*---- WRITE RESULTS
*
*      Write (3,5400)
*      Write (3,*) 'CALCULATED RESULTS'
*      Write (3,*) '------------------'
*      If (IHC.EQ.0) Then
*        X1 = PRSI / C4
*        X2 = GASTI / C8
*        Write (3,5700) PRSI, X1, GASTI, X2
*      Else If (IHC.EQ.1) Then
*        X1 = PRSI / C4
*        X2 = OILTI / C6
*        Write (3,5800) PRSI, X1, OILTI, X2
*      End If
*      X1 = VOLB / C
*      X2 = AREA / C2
*      X3 = RADE / C3
*      Write (3,5900) SATOI, SATGI, VOLB, X1, AREA / 1000., X2, RADE, X3
*
*---- START TIMESTEP LOOP
*
*      Print *, 'Processing timestep :'
*      Print *
   30 Continue
      K = K + 1
*      Print 5300, K
C---     DIAM=Pass(K)

         If(K.LE.5) Then
         DIAM=Pass(1)
         Goto 32
         Endif

         If(K.LE.10) Then
         DIAM=Pass(2)
         Goto 32
         Endif

         If(K.LE.15) Then
         DIAM=Pass(3)
         Goto 32
         Endif

         If(K.LE.20) Then
         DIAM=Pass(4)
         Endif

32       If (K.LT.N2) Then

*---- WELL CONTROL

         If (QWCTR) Then
           NWELL = NWELLS(IWCTR)
           If (NWELL.EQ.0) Then
*                  (It's time to stop execution)
             Go To 50
           End If
           RTELO = TRTEFM(IWCTR) / NWELL
           RTEMAX = TRTEFT(IWCTR) / NWELL
           RTEW = RTEMAX
           RTECRIT = RTEMAX
           EPSRTE = RTEW * 0.0005
           EPSPRS = 0.5
           VOLBW = VOLB / NWELL
           RADE = SQRT(RADW**2+VOLBW/PI/THK)
           RADEQ = RADE / RADW
           PWHMIN = TPWMIN(IWCTR)
           SKN = TSKN(IWCTR)
           IWCTR = IWCTR + 1
           QWCTR = .FALSE.
         End If

*---- TIME CONTROL

         If (VTIME(K).GT.TTIM(IWCTR)) Then
           QWCTR = .TRUE.
           VTIME(K+1) = VTIME(K)
           VTIME(K) = TTIM(IWCTR)
         Else If (VTIME(K).EQ.TTIM(IWCTR)) Then
           QWCTR = .TRUE.
           VTIME(K+1) = VTIME(K) + DELTIM
         Else
           VTIME(K+1) = VTIME(K) + DELTIM
         End If
         DTIM = VTIME(K) - VTIME(K-1)

*---- Assume that the average reservoir pressure and the rate will not
*     increase from one timestep to another unless well control is
*     changed. Since XMBAL usually is not identically zero after a time-
*     step, PRSHI is set > VPRSR(K-1)  to avoid problems when the rate
*     is zero.  Shrink the rate interval which will be used for
*     calculations.

         PRSHI = MIN(VPRSR(K-1)*1.0001,PVTMAX)
   40    RTEHI = MIN(RTEW*1.0001,RTEMAX,RTECRIT)

*---- CALCULATE RATE, RESERVOIR PRESSURE, BOTTOMHOLE PRESSURE AND
*     WELLHEAD PRESSURE

         Call RATE(PRSHI,PVTMIN,PWHMIN,RTEHI,RTELO,EPSPRS,EPSRTE,K,IPRT,
     *       IEXE,IHC,PRS,PRSWF,PRSWH,RTEW,DGASP,DOILP,DGORP,QSTOP)
         If (ICRIT) Then
           RTECRIT = RTECRIT * 0.95
           ICRIT = .FALSE.
           Go To 40
         End If
         If (.NOT.QSTOP) Then
```

```
*---- UNPHYSICAL SATURATIONS ?

          If (SATO1.LT.0..OR.SATG1.LT.0.) Then
            Write (3,*) 'MAIN.  TIMESTEP:', K
            Write (3,*) '** ERROR **'
            Write (3,*) 'UNPHYSICAL SATURATION'
            Write (3,*) 'SATG1, SATO1', SATG1, SATO1
            Write (3,*)
            Stop
          End If

*---- UPDATE VARIABLES

          DGASPS = DGASP * NWELL
          DOILPS = DOILP * NWELL
*         (Volumes on field basis:)
          VGASP(K) = VGASP(K-1) + DGASPS
          VOILP(K) = VOILP(K-1) + DOILPS
*         (Rates on well basis:)
C---      Write(6,*)DGASP, DOILP, DTIM
          IF(DGASP.LE.1.0e-7)THEN
            DGASP=0.0
          ENDIF

          IF(DOILP.LE.1.0e-7)THEN
            DOILP=0.0
          ENDIF

            VRTEG(K) = DGASP / DTIM
            VRTEO(K) = DOILP / DTIM
            VRTEG1(K) = DGASP / DTIM
            VRTEO1(K) = DOILP / DTIM
            VGOR(K) = DGORP
            VPRSR(K) = PRS
            VPRSWF(K) = PRSWF
            VPRSWH(K) = PRSWH
            IVNWEL(K) = NWELL
            AO1 = AO2
            AG1 = AG2
            RO1 = RO2
            RG1 = RG2
          Call SEPARAT(K)
          Call GMSECON(K)
          Pass4 = -NPV

*---- WRITE INTERMEDIATE RESULTS IF REQUESTED

          If (IPRT.GE.3) Then
            Write (3,*)
            Write (3,5300)
            Write (3,5000) 'TIMESTEP NUMBER :', K, 'TIME (D):',
     *          VTIME(K)
            Write (3,5300)
            Write (3,*)
            Write (3,5100) 'VGASP(K)', VGASP(K), 'FVFO', FVFO
            Write (3,5100) 'VOILP(K)', VOILP(K), 'VISG', VISG
            Write (3,5100) 'PRS', PRS, 'OGRS', OGRS
            Write (3,5100) 'PRSWF', PRSWF, 'DENRG', DENRG
            Write (3,5100) 'PRSWH', PRSWH, 'FVFG', FVFG
            Write (3,5100) 'DGORP', DGORP, 'POR', POR
            Write (3,5100) 'VRTEG(K)', VRTEG(K), 'SATO1', SATO1
            Write (3,5100) 'VRTEO(K)', VRTEO(K), 'SATG1', SATG1
            Write (3,5100) 'VRTEO1(K)', VRTEO1(K), 'VRTEG1(K)', VRTEG1(K)
            Write (3,5100) 'PRSHI', PRSHI, 'XMBAL1', XMBAL1
            Write (3,5100) 'PVTMIN', PVTMIN, 'PRMGO', PRMGO
            Write (3,5100) 'PWHMIN', PWHMIN, 'VOLBW', VOLBW
            Write (3,5100) 'RTEHI', RTEHI, 'RADEQ', RADEQ
            Write (3,5100) 'RTELO', RTELO, 'RADE', RADE
            Write (3,5100) 'SKN', SKN, 'AO2', AO2
            Write (3,5100) 'VISO', VISO, 'AG2', AG2
            Write (3,5100) 'GORS', GORS, 'RO2', RO2
            Write (3,5100) 'DENRO', DENRO, 'RG2', RG2
            Write (3,5100) 'IVNWEL(K)', IVNWEL(K), 'NPV', NPV
            Write (3,*)
            Write (3,5400)
            Write (3,*)
          End If

*---- CHECK STOP CONDITIONS

          If (RTEW.GE.0.01.OR.RTEW.GE.RTEHI) Then
            If (PRS.LE.VPRSR(K-1)) Then

*---- END TIMESTEP LOOP

              Go To 30
            End If
          End If
        End If
      End If

*---- WRITE MESSAGES

   50 Continue
      If (K.GE.N2) Then
        Write (3,*) 'MAIN.  TIMESTEP:', K
        Write (3,*) 'ATTEMPT TO RUN TOO MANY TIMESTEPS.'
        Write (3,*) 'EXECUTION STOPS.'
        Write (3,*)
      End If
      If (RTEW.LT.0.01.AND.RTEW.LT.RTEHI) Then
        Write (3,*)
        Write (3,*) 'MAIN.  TIMESTEP:', K
        Write (3,*) 'THE RATE (RTEW) IS APPROXIMATELY ZERO.'
        Write (3,*) 'EXECUTION STOPS.'
        Write (3,*)
      End If
      If (PRS.GT.VPRSR(K-1)) Then
        Write (3,*) 'MAIN.  TIMESTEP:', K
        Write (3,*) 'PRS > VPRSR(K-1)'
        Write (3,*)
      End If

*---- FORMAT STATEMENTS

      Return
 5000 Format (1X,A,I5,5X,A,F16.4)
 5100 Format (1X,4(A16,E17.10))
 5200 Format (1X,4(A16,I17))
* 5300 Format ('+',I13)
* 5400 Format (' ')
 5300 Format (1X,75('-'))
 5400 Format (1X,75('='))
* 5700 Format (//1X,'PRSI    : INITIAL PRESSURE IS ASSUMED TO BE EQUAL    '
*     *    /1X,'          TO MAX. INPUT PVT-DATA PRESSURE (KPA) ..: ',F11
*     *    .0/1X,'                                          (PSIA) .....: ',
*     *    F12.1/1X,
*     *    'GASTI   : GAS VOLUME INITIALLY IN PLACE (SM3) ....: ',G12.5/1
```

```
*     *    X,'                                          (SCF) ....: ',G12.5)
* 5800 Format (//1X,'PRSI    : INITIAL PRESSURE IS ASSUMED TO BE EQUAL    '
*     *    /1X,'          TO MAX. INPUT PVT-DATA PRESSURE (KPA) ..: ',F11
*     *    .0/1X,'                                        (PSIA) .....: ',
*     *    F12.1/1X,
*     *    'OILTI  : OIL VOLUME INITIALLY IN PLACE (SM3) ....: ',G12.5/1
*     *    X,'                                          (STB) ....: ',G12.5)
* 5900 Format (//1X,
*     *    'SATOI  : INITIAL OIL SATURATION (FRACTION) ......: ',G12.5/1
*     *    X,'SATGI  : INITIAL GAS SATURATION (FRACTION) ......: ',G12.5/
*     *    1X,'VOLB   : BULK VOLUME OF RESERVOIR (M3) ..........: ',G12.5
*     *    /1X,'                                    (BBL) ..........: ',G12
*     *    .5/1X,'AREA   : TOTAL RESERVOIR AREA, FOR UNIFORM '/1X,
*     *    '                THICKNESS      (1E+3 M2) ..........: ',G12.5/1
*     *    X,'                                  (ACRES) ..........: ',G12.5/
*     *    1X,'RADE   : RESERVOIR RADIUS, FOR CIRCULAR SHAPE (M): ',G12.5
*     *    /1X,'                                          (FT): ',G12
*     *    .5)
      End
*************************************************************************
**** Subroutine  GMSOUT
**** Author      Gunnar Borthne (modified by James Carroll)
**** Date        April, 1986
**** Function    This subroutine outputs the results of the well model.
*************************************************************************
      Subroutine GMSOUT

*---- VARIABLES AND CONSTANTS

      Implicit Double Precision (A-Z)

      Integer N1, N2, IVNWEL, NWELLS, I, IERR, IEXE, IHC, IPRT, IUNIN,
     *    IWCTR, K, NSTEP, NWELL, NWT, CORR, NPVT, NRP, IC, IT
      Character ZJOBID*60, ZC(8)*6
      Parameter (N1 = 100,N2 = 500)

      Dimension TPWMIN(N1), TRTEFM(N1), TRTEFT(N1), TSKN(N1), TTIM(N1),
     *    VGASP(0:N2), VGOR(0:N2), VOILP(0:N2), VPRSR(0:N2),
     *    VPRSWF(0:N2), VPRSWH(0:N2), VRTEG(0:N2), VRTEO(0:N2), AN(8),
     *    VTIME(0:N2), TDENRG(100), TDENRO(100), TFVFG(100),
     *    TFVFGX(100), TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100),
     *    TPRMRO(100), PRMLGO(100), TPRS(100), TSATG(100), TVISG(100),
     *    TVISO(100), IVNWEL(0:N2), NWELLS(N1), IC(8), IT(8)

*---- COMMON BLOCKS

      Common /CONTAM/ NITRO, CO2, H2S, NACL, RSI
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     *    CORR
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     *    VOLBW
      Common /MBAL2/ AO2, AG2, RO2, RG2, RGAV
      Common /MBAL3/ DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     *    SATG1, SATO1, VISG, VISO, XMBAL1
      Common /IPR1/ DPINT, DSKN, PRM, RADEQ, SKN, THK
      Common /ICOUNT/ IC, IT
      Common /FINANCE/ INFRATE, DISRATE, OILPRIC, GASPRIC
      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     *    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /TRANS2/ C1, C2, C3, C4, C5, C6, C7, C8, C9
      Common /TRANS3/ ZC, ZJOBID
      Common /TRANS4/ VGASP, VGOR, VOILP, VPRSR, VPRSWF, VPRSWH, VRTEG,
     *    VRTEO, VTIME, IWCTR, NWELL, OILTI, GASTI, IVNWEL, K

*---- WRITE RESULTS

      Print *
      Print *, 'Writing results...'
      Print *
      NSTEP = K - 1

*---- WRITE TABLE OF CUMULATIVE PRODUCTION

      Write (3,5300) ZJOBID
      Do 10 I = 1, NSTEP
        X1 = VTIME(I) / 365.
        X2 = VGASP(I) / 1D+6
        X3 = VGASP(I) / C8 / 1D+6
        X4 = VOILP(I) / 1D+3
        X5 = VOILP(I) / C6 / 1D+3
        X6 = 0.
        If (X4.NE.0.) X6 = X2 / X4
        X7 = 0.
        If (X5.NE.0.) X7 = X3 / X5
        If (IHC.EQ.0) X8 = VGASP(I) / GASTI
        If (IHC.EQ.1) X8 = VOILP(I) / OILTI
        Write (3,5400) I, VTIME(I), X1, X2, X3, X4, X5, X6, X7, X8
   10 Continue
      Write (3,5000)

*---- WRITE TABLE OF PRESSURES AND PRODUCING GOR

      Write (3,5500) ZJOBID
      Do 20 I = 1, NSTEP
        X1 = VTIME(I) / 365.
        X2 = VPRSR(I) / C4
        X3 = VPRSWF(I) / C4
        X4 = VPRSWH(I) / C4
        X5 = VGOR(I) / 1D+3
        X6 = VGOR(I) * C7 / 1D+3
        Write (3,5600) I, VTIME(I), X1, VPRSR(I), X2, VPRSWF(I), X3,
     *      VPRSWH(I), X4, X5, X6
   20 Continue
      Write (3,5100)

*---- WRITE TABLE OF PRODUCTION RATES

      Write (3,5700) ZJOBID
      Write (3,5800)
      Do 30 I = 1, NSTEP
        X1 = VTIME(I-1) / 365.
        X2 = VTIME(I) / 365.
        X3 = VRTEG(I) / 1D+3
        X4 = VRTEG(I) / C8 / 1D+6
        X5 = X3 * IVNWEL(I)
        X6 = X4 * IVNWEL(I)
        X7 = VRTEO(I)
        X8 = VRTEO(I) / C6
        X9 = X7 * IVNWEL(I)
        X10 = X8 * IVNWEL(I)
        Write (3,5900) I, VTIME(I-1), VTIME(I), X1, X2, IVNWEL(I), X3,
     *      X4, X5, X6, X7, X8, X9, X10
   30 Continue
      Write (3,5200)

*---- WRITE TABLE OF REVENUES
```

```
      Write (3,6000) ZJOBID
      Write (3,6100)
      NPV = 0.
      DFEFF = (1+DISRATE) ** (DELTIM/365)
      DFACTOR = (1+DISRATE) ** (DELTIM/2/365)
      IFEFF = (1+INFRATE) ** (DELTIM/365)
      IFACTOR = (1+INFRATE) ** (DELTIM/2/365)
      Do 40 I = 1, NSTEP
        GASPRO = VRTEG(I) / C8 / 1D+6 * IVNWEL(I) * DELTIM
        GASREV = GASPRO * GASPRIC * IFACTOR
        OILPRO = VRTEO(I) / C6 * IVNWEL(I) * DELTIM
        OILREV = OILPRO * OILPRIC * IFACTOR
        TOTREV = GASREV + OILREV
        NPV = NPV + TOTREV / DFACTOR
        Write (3,6200) I, VTIME(I-1) / 365, VTIME(I) / 365, GASPRO,
     *      GASPRIC * IFACTOR, GASREV, OILPRO, OILPRIC * IFACTOR,
     *      OILREV, TOTREV, NPV
        IFACTOR = IFACTOR * IFEFF
        DFACTOR = DFACTOR * DFEFF
   40 Continue
      Write (3,5200)

*---- CALCULATE AND WRITE ITERATION REPORT

      If (IPRT.GE.2) Then
        Do 50 I = 1, 3
          X1 = IC(I+1) - IC(I) + IT(I)
          If (IT(I).NE.0) AN(I) = X1 / IT(I)
   50   Continue
        X1 = IC(5)
        If (X1.EQ.0.) X1 = 1.
        AN(5) = IC(6) / X1

        Write (3,6300)
        Do 60 I = 1, 8
          Write (3,6400) ZC(I), IC(I), IT(I), AN(I)
   60   Continue
        Write (3,6500)
      End If
      Return

*---- FORMAT STATEMENTS

 5000 Format (1X,108('-'))
 5100 Format (1X,120('-'))
 5200 Format (1X,132('-'))
 5300 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,'SIMULATION RESULTS'/1
     *    X,108('-')/1X,
     *    '                                FIELD CUMULATIVE   ',
     *    '     FIELD CUMULATIVE        FIELD CUMULATIVE   ',
     *    ' RECOVERY OF'/1X,
     *    '              TIME               GAS PRODUCTION     ',
     *    '       OIL PRODUCTION           GAS/OIL RATIO      ',
     *    ' PREF. PHASE'/1X,
     *    '      ------------------  ----------------------',
     *    '  ----------------------  ----------------------',
     *    '  ----------'/1X,
     *    'NO.       D       YEARS     1E+6 SM3       MMSCF  ',
     *    '  1E+3 SM3        MSTB    1E+3 SM3/SM3   MMSCF/MSTB',
     *    '    FRACTION '/1X,
     *    '---  ----------  -------  ----------  ----------',
     *    '  ----------  ----------  ----------  ----------',
     *    '  ----------')
 5400 Format (1X,I3,F12.1,F9.2,6G12.5,F12.5)
 5500 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,'SIMULATION RESULTS'/1
     *    X,120('-')/1X,
     *    '                                AVERAGE RESERVOIR  ',
     *    '         BOTTOMHOLE                WELLHEAD        ',
     *    '          PRODUCING       '/1X,
     *    '              TIME                 PRESSURE        ',
     *    '           PRESSURE                PRESSURE        ',
     *    '        GAS/OIL RATIO     '/1X,
     *    '      ------------------  ----------------------',
     *    '  ----------------------  ----------------------',
     *    '  ----------------------'/1X,
     *    'NO.       D       YEARS        KPA          PSIA   ',
     *    '       KPA          PSIA          KPA          PSIA   ',
     *    '1E+3 SM3/SM3  MMSCF/MSTB'/1X,
     *    '---  ----------  -------  ----------  ----------',
     *    '  ----------  ----------  ----------  ----------',
     *    '  ----------  ----------')
 5600 Format (1X,I3,F12.1,F9.2,3(F12.0,F12.1),2G12.5)
 5700 Format (/1X,'JOB IDENTIFICATION : ',A60//1X,'SIMULATION RESULTS'/1
     *    X,132('-')/1X,'                                                ',
     *    '              GAS PRODUCTION RATE                ',
     *    '                OIL PRODUCTION RATE               '/1X,
     *    '                                                NUM-',
     *    '  ----------------------------------------',
     *    '  ----------------------------------------'/1X,
     *    '                    TIME                  BER ',
     *    '          WELL                        FIELD        ',
     *    '           WELL                       FIELD        '/1X,
     *    '       ------------------------------ OF  ',
     *    '  ----------------------  ----------------------',
     *    '  ----------------------  ----------------------')
 5800 Format (1X,'NO.          D              YEARS     WELLS',
     *    '1E+3 SM3/D   MMSCF/D   1E+3 SM3/D   MMSCF/D  ',
     *    '    SM3/D       STB/D       SM3/D       STB/D   '/1X,
     *    '---  ----------  ----------  ----------  ----',
     *    '  ----------  ----------  ----------  ----------',
     *    '  ----------  ----------  ----------  ----------')
 5900 Format (1X,I3,F8.1,' -',F7.1,F7.2,' -',F6.2,I5,1X,2G11.5,3(G12.5,
     *    G11.5))
 6000 Format (//1X,'JOB IDENTIFICATION : ',A60//1X,'SIMULATION RESULTS'/
     *    1X,96('-')/1X,'                    ',
     *    '                  GAS                 ',
     *    '                 OIL                          '/1X,
     *    '                    ',' ---------------------------------',
     *    '  ------------------------------   TOTAL   '/1X,
     *    '                    ',' PRODUCTION   PRICE     REVENUE  ',
     *    '  PRODUCTION  PRICE    REVENUE     REVENUE  '/1X)
 6100 Format (1X,'NO.        YEARS    ',
     *    '   MMSCF/D    $/MMSCF        $      ',
     *    '    STB/D     $/STB          $           $'/1X,
     *    '---   ----------',' -----------  ---------  -----------',
     *    '  -----------  -------  -----------  -----------')
 6200 Format (1X,I3,F7.2,' -',F6.2,1X,F11.1,1X,F9.2,1X,F11.0,1X,F11.1,1
     *    X,F7.2,2(1X,F11.0),1X,F20.0)
 6300 Format (//1X,'                    ITERATION REPORT'/1X,
     *    '                  ----------------'//1X,
     *    '     NAME       IN-CALLS       SOLVE      IT/SOLVE '/1X,
     *    '  ----------  ----------  ----------  ----------')
 6400 Format (1X,A11,I10,4X,I8,2X,F10.1)
 6500 Format (//1X,'NAME      = NAME OF SUBROUTINE'/1X,
     *    'IN-CALLS = NUMBER OF CALL TO THIS SUBROUTINE'/1X,
     *    'SOLVE    = HOW MANY TIMES DID THE SUBROUTINE HAVE TO START'/1
     *    X,'           A SOLUTION PROCEDURE WITH ITERATIONS'/1X,
```

```
     *    'IT/SOLVE = (1) AVERAGE NUMBER OF ITERATIONS EACH TIME A'/1X,
     *    '           SOLUTION PROCEDURE WAS NECESSARY, OR (2) AVERAGE'/
     *    1X,'           NUMBER OF INTEGRATION STEPS PER INTEGRATION (IP
     *R)')
      End



*****************************************************************
**** TITLE .......:  FNPRS
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  IPR
**** OUT-CALLS ...:  INTPL
**** FUNCTION ....:  Calculate the pseudopressure integrand.
*****************************************************************
      Subroutine FNPRS(DGORP,P2,IHC,IPRT,F,XPRMGO)

*---  INPUT VARIABLES

      Double Precision DGORP, P2
      Integer IHC, IPRT

*---- OUTPUT VARIABLES

      Double Precision F, XPRMGO

*---- COMMON BLOCKS

      Double Precision TDENRG(100), TDENRO(100), TFVFG(100),
     *    TFVFGX(100), TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100),
     *    TPRMRO(100), PRMLGO(100), TPRS(100), TSATG(100), TVISG(100),
     *    TVISO(100)
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Integer NPVT, NRP
      Common /ICOUNT/ IC, IT
      Integer IC(8), IT(8)

*---- LOCAL VARIABLES

      Double Precision XSATG, XGORPF, GORS, FVFO, VISO, OGRS, FVFG,
     *    VISG, DENRO, DENRG, PRMRO, PRMRG, XPRML, FRAC
      Integer J

*---- START EXECUTION

      IC(6) = IC(6) + 1

*---- CALCULATE PRESSURE-DEPENDENT PROPERTIES

      Call INTPL(TPRS,TGORS,P2,1,NPVT,GORS,FRAC,J)
      FVFO = TFVFO(J) + FRAC * (TFVFO(J+1)-TFVFO(J))
      VISO = TVISO(J) + FRAC * (TVISO(J+1)-TVISO(J))
      OGRS = TOGRS(J) + FRAC * (TOGRS(J+1)-TOGRS(J))
      VISG = TVISG(J) + FRAC * (TVISG(J+1)-TVISG(J))
      DENRO = TDENRO(J) + FRAC * (TDENRO(J+1)-TDENRO(J))
      DENRG = TDENRG(J) + FRAC * (TDENRG(J+1)-TDENRG(J))
      FVFG = 1 / (TFVFGX(J)+FRAC*(TFVFGX(J+1)-TFVFGX(J)))

*---- CALCULATE RELATIVE PERMEABILITY RATIO

      XGORPF = (DGORP-GORS*DENRG) / (1-DGORP*OGRS*DENRO)
      XPRMGO = XGORPF * VISG * FVFG / VISO / FVFO
      If (XPRMGO.LT.0..AND.IPRT.GE.4) Then
        Write (3,*) 'FNPRS.'
        Write (3,*) 'MESSAGE:'
        Write (3,*) 'NEGATIVE RELATIVE PERMEABILITY RATIO (KRG/KRO)'
        Write (3,*) 'ENCOUNTERED DURING PSEUDOPRESSURE CALCULATIONS.'
        Write (3,*) '    KRG/KRO  = ', XPRMGO
        Write (3,*) '    PRESSURE = ', P2
        Write (3,*) 'THIS IS NOT AN ERROR IF THE PRESSURE IS DIFFERENT'
        Write (3,*) 'FROM THE BOTTOMHOLE PRESSURE SOUGHT BY "IPR"'
        Write (3,*) 'KRG/KRO IS SET EQUAL TO 1E-15, AND CALCULATIONS'
        Write (3,*) 'CONTINUE.  TRY SMALLER DELTIM OR DPINT.'
        Write (3,*)
      End If
      If (XPRMGO.LE.0.) Then
        XPRML = -15
      Else
        XPRML = LOG10(XPRMGO)
      End If

*---- CALCULATE GAS SATURATION

      Call INTPL(PRMLGO,TSATG,XPRML,1,NRP,XSATG,FRAC,J)

*---- CALCULATE RELATIVE PERMEABILITIES

      Call INTPL(TSATG,TPRMRO,XSATG,1,NRP,PRMRO,FRAC,J)
      PRMRG = TPRMRG(J) + FRAC * (TPRMRG(J+1)-TPRMRG(J))

*---- CALCULATE THE PRESSURE FUNCTION

      If (IHC.EQ.0) Then
        F = PRMRG / VISG / FVFG + PRMRO * GORS / VISO / FVFO
      Else If (IHC.EQ.1) Then
        F = PRMRO / VISO / FVFO + PRMRG * OGRS / VISG / FVFG
      End If

      Return
      End
*****************************************************************
**** TITLE .......:  INTPL
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  GMS, MATBAL, FNPRS
**** OUT-CALLS ...:  NONE
**** FUNCTION ....:  Linear X - Linear Y interpolation.
*****************************************************************
      Subroutine INTPL(TABX,TABY,X,I,MAX,Y,FRAC,J)

*---- INPUT VARIABLES

      Double Precision TABX(100), TABY(100), X
      Integer I, MAX, M

*---- OUTPUT VARIABLES

      Double Precision Y, FRAC
      Integer J

*---- COMMON BLOCK

      Common /ICOUNT/ IC, IT
      Integer IC(8), IT(8)
C---    Write(6,*)'TABX'
        do M=1,100
```

```
C---     Write(6,*)TABX(M),TABY(M)
         Enddo

*---- START EXECUTION

         IF(X.GE.1.0e-7) THEN
           GOTO 109
         ELSE
           X=1.0d-7
         ENDIF

109      IF(X.LE.TABX(1))THEN
         X=TABX(1)
         ENDIF

         IF(X.GE.TABX(MAX))THEN
         X=TABX(MAX)
         ENDIF

         IC(8) = IC(8) + 1
         J = I
10       Continue
         If (X.GE.TABX(J).AND.X.LE.TABX(J+1)) Then
*              (The appropriate interval is found)
         FRAC = (X-TABX(J)) / (TABX(J+1)-TABX(J))
         Y = TABY(J) + FRAC * (TABY(J+1)-TABY(J))
         Return
       End If
       J = J + 1
       If (J.LT.MAX) Go To 10
       Write (3,*) 'ERROR. INTPL'
       Write (3,*) 'TABX(1),TABX(MAX),X', TABX(1), TABX(MAX), X
       Write (3,*)
       Stop
       End
*************************************************************************
**** TITLE .......:  IPR
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  WHPRS
**** OUT-CALLS ...:  FNPRS
**** FUNCTION ....: The function F is integrated numerically from PRS to
****      PRSWF.  PRSWF is the unknown and is found by iterations.  If the
****      pressure is trying to move below PRSMIN during integration, then
****      PRSWF is set equal to PRSMIN and a smaller rate is calculated. The
****      input and output units are: pressure, kPa; gas-oil ratio,
****      std.vol/std.vol; rate, m3/D.
*************************************************************************
       Subroutine IPR(PRS,PRSMIN,RTEX,DGORP,EPSPRS,K,IPRT,IHC,PRSWF,RTEY,
      *    QSTOP)

*---- INPUT VARIABLES

       Double Precision PRS, PRSMIN, RTEX, DGORP, EPSPRS
       Integer K, IPRT, IHC

*---- OUTPUT VARIABLES

       Double Precision PRSWF, RTEY
       Logical QSTOP

*---- COMMON BLOCK

       Common /IPR1/ DPINT, DSKN, PRM, RADEQ, SKN, THK
       Double Precision DPINT, DSKN, PRM, RADEQ, SKN, THK
       Common /ICOUNT/ IC, IT
       Integer IC(8), IT(8)

*---- LOCAL VARIABLES AND CONVERSION FACTORS

       Double Precision AREA, C1, C2, C3, CN0, CN1, DP, DP2, DRVSUM, F1,
      *    F2, F4, P, P1, P2, PI, SUM1, SUM2, XPRMGO
       Integer I
       Logical QNORML
       Data C1, C2, C3 /1E-12, 1000., 86400/

*---- START EXECUTION

       IC(5) = IC(5) + 1
       If (IPRT.GE.4) Write (3,*) '         START IPR'
       QNORML = .TRUE.
       DP = DPINT
       DP2 = DP * 2.
       PI = 3.141592654
       CN0 = LOG(RADEQ) - 0.75 + SKN + DSKN * RTEX
       CN1 = CN0 / (2*PI*PRM*C1*THK) / C2
       AREA = RTEX * CN1 / C3
*      (AREA has units kPa/Pa s)
       P1 = PRS
       P2 = P1
       F1 = 0.
*      (Initialize F2:)
       Call FNPRS(DGORP,P2,IHC,IPRT,F2,XPRMGO)
       SUM1 = 0.
       SUM2 = 0.

*---- START INTEGRATION LOOP

       I = 0
    10 Continue
       I = I + 1
       F1 = F2
       P1 = P2
       P2 = P2 - DP2
       If (P2.LT.PRSMIN) Then
         P2 = PRSMIN
         DP2 = P1 - P2
         DP = DP2 / 2.
       End If
       Call FNPRS(DGORP,(P1-DP),IHC,IPRT,F4,XPRMGO)
       Call FNPRS(DGORP,P2,IHC,IPRT,F2,XPRMGO)
*           (SUM has units kPa/Pa s)
       SUM1 = SUM2
       SUM2 = SUM2 + (F1+4*F4+F2) * DP / 3.
       If (SUM2.LT.AREA) Then
         If (P2.EQ.PRSMIN) Then
           QNORML = .FALSE.
           Go To 30
         End If
         Go To 10
       End If

*---- START LOOP, FIND INTEGRATION LIMIT WITH A MODIFIED NEWTON-RAPHSON
*      ITERATION METHOD

       I = 0
    20 Continue
       I = I + 1
```

```
      If (ABS(P2-P1).GT.EPSPRS) Then

*---- EMERGENCY EXIT

        If (P1-P2.EQ.0..OR.SUM1-SUM2.EQ.0.) Then
          Write (3,*) 'IPR.     TIME STEP:', K
          Write (3,5100) ABS(P1-P2), ABS(SUM1-SUM2)
          Write (3,*) .
          P = P2
          Go To 30
        End If

        DRVSUM = (SUM2-SUM1) / (P2-P1)
        DP = (SUM2-AREA) / DRVSUM
        P = P2 - DP
        P = MIN(P,PRS)
        P = MAX(P,PRSMIN)
        P1 = P2
        F1 = F2
        P2 = P
        Call FNPRS(DGORP,P2,IHC,IPRT,F2,XPRMGO)
        SUM1 = SUM2
        SUM2 = SUM2 + (F1+F2) * (P1-P2) / 2.
        Go To 20
      End If

   30 Continue
      If (XPRMGO.LT.0.) Then
        Write (3,*) 'IPR.  TIMESTEP:', K
        Write (3,*) 'RELATIVE PERMEABILITY RATIO (KRG/KRO) IS'
        Write (3,*) 'NEGATIVE.  EXECUTION STOPS.'
        Write (3,*) 'TRY SMALLER DELTIM OR DPINT.'
        QSTOP = .TRUE.
      End If
      If (QNORML) Then
*---- PREPARE EXIT, NORMAL PROCEDURE
        PRSWF = P
        RTEY = RTEX
      Else
*---- PREPARE EXIT, LACK OF PVT DATA
        PRSWF = PRSMIN
        RTEY = SUM2 / CN1 * C3
        If (IPRT.GE.4) Then
          Write (3,*) 'IPR.  TIMESTEP:', K
          Write (3,*) 'The rate is reduced due to lack of PVT data'
          Write (3,*) 'Input  rate: ', RTEX
          Write (3,*) 'Output rate: ', RTEY
          Write (3,*)
        End If
      End If
      Return

*---- FORMAT STATEMENTS

 5000 Format (1X,A,I4,A,I4)
 5100 Format (1X,'*EMERGENCY EXIT*',/1X,
     *    'LENGTH OF CURRENT PRESSURE INTERVAL (KPA) .......:',G15.8,/1
     *    X,'DIFFERENCE IN SUM (KPA/PA S) ..................:',G15.8,
     *    //)
      End
************************************************************************
**** TITLE .......:  ISGN
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  RATE, WHPRS, RESPRS
**** OUT-CALLS ...:  NONE
**** FUNCTION ....:  Return sign of argument (-1, 0, +1)
************************************************************************
      Integer Function ISGN(X)
      Double Precision X

*---- COMMON BLOCK

      Common /ICOUNT/ IC, IT
      Integer IC(8), IT(8)

      IC(7) = IC(7) + 1
      If (X.GT.0.) Then
        ISGN = 1
      Else If (X.EQ.0.) Then
        ISGN = 0
      Else If (X.LT.0.) Then
        ISGN = -1
      End If

      Return
      End
************************************************************************
**** TITLE .......:  ITEST
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  GMS
**** OUT-CALLS ...:  NONE
**** FUNCTION ....:  DETECT ERROR IN INTEGER INPUT DATA, AND WRITE ERROR
****                 MESSAGE
************************************************************************
      Subroutine ITEST(IX,I1,I2,ZTXT,IERR)

*---- INPUT VARIABLES

      Integer IX, I1, I2
      Character ZTXT*(*)

*---- INPUT AND OUTPUT

      Integer IERR

      If (IX.LT.I1.OR.IX.GT.I2) Then
        IERR = IERR + 1
        Write (3,*)
        Write (3,*) '            *** ERROR ***'
        Write (3,*) 'MESSAGE ..............: ', ZTXT
        Write (3,*) 'VALUE ................: ', IX
        Write (3,*) 'PERMITTED INTERVAL ....: ', I1, I2
        Write (3,*)
      End If

      Return
      End
************************************************************************
**** TITLE .......:  MATBAL
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  RESPRS
**** OUT-CALLS ...:  INTPL
****
**** FUNCTION ....: Calculate material-balance error and related quantities
****      as functions of average reservoir pressure, production rate,
```

```
****     timestep length and other variables.
*****************************************************************
      Subroutine MATBAL(PRS,RTE,IPRT,IHC,DGASP,DOILP,XMBAL)

*---- INPUT VARIABLES

      Double Precision PRS, RTE
      Integer IPRT, IHC

*---- OUTPUT VARIABLES

      Double Precision DGASP, DOILP, XMBAL

*---- LOCAL VARIABLES

      Integer J
      Double Precision SATO, SATG, S, XMOBR, X1, X2, X3, ROAV, DOILPQ,
     *    DGASPQ, FRAC, Y

*---- COMMON BLOCKS  (MATBAL MODIFIES MBAL2, MBAL3)

      Double Precision TDENRG(100), TDENRO(100), TFVFG(100),
     *    TFVFGX(100), TFVFO(100), TGORS(100), TPRMRG(100),
     *    TPRMRO(100), PRMLGO(100), TPRS(100), TSATG(100), TVISG(100),
     *    TVISO(100)
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Integer NPVT, NRP
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     *    VOLBW
      Double Precision AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1,
     *    SATWI, VOLBW
      Common /MBAL2/ AO2, AG2, RO2, RG2, RGAV
      Double Precision AO2, AG2, RO2, RG2, RGAV
      Common /MBAL3/ DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     *    SATG1, SATO1, VISG, VISO, XMBAL1
      Double Precision DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     *    SATG1, SATO1, VISG, VISO, XMBAL1
      Common /ICOUNT/ IC, IT
      Integer IC(8), IT(8)

*---- START EXECUTION
C---    Write(6,*)'MATBL'
        do I=1,NRP
C---    Write(6,*)TSATG(I)
        ENDDO
      IC(4) = IC(4) + 1
      If (IPRT.GE.4) Write (3,*) '              START MBAL'

*---- CALCULATE PRESSURE-DEPENDENT PROPERTIES

      Call INTPL(TPRS,TGORS,PRS,1,NPVT,GORS,FRAC,J)
      FVFO = TFVFO(J) + FRAC * (TFVFO(J+1)-TFVFO(J))
      VISO = TVISO(J) + FRAC * (TVISO(J+1)-TVISO(J))
      OGRS = TOGRS(J) + FRAC * (TOGRS(J+1)-TOGRS(J))
      VISG = TVISG(J) + FRAC * (TVISG(J+1)-TVISG(J))
      DENRO = TDENRO(J) + FRAC * (TDENRO(J+1)-TDENRO(J))
      DENRG = TDENRG(J) + FRAC * (TDENRG(J+1)-TDENRG(J))
      FVFG = 1 / (TFVFGX(J)+FRAC*(TFVFGX(J+1)-TFVFGX(J)))
      POR = PORI * EXP(CMPF*(PRS-PRSI))

*---- CALCULATE SATURATIONS

      If (IHC.EQ.0) Then
*         (Calculate the oil saturation from)
*         (the gas material balance equation)
        DGASPQ = RTE * DTIM / VOLBW
        X1 = GORS * DENRG / FVFO
        X2 = POR * (1-SATWI) / FVFG
        X3 = POR * (X1-1/FVFG)
        SATO = (AG1-DGASPQ-X2) / X3
      Else If (IHC.EQ.1) Then
*         (Calculate the oil saturation from)
*         (the oil material balance equation)
        DOILPQ = RTE * DTIM / VOLBW
        X1 = OGRS * DENRO / FVFG
        X2 = POR * (1-SATWI) * X1
        X3 = POR * (1/FVFO-X1)
        SATO = (AO1-DOILPQ-X2) / X3
      End If
      SATG = 1 - SATWI - SATO

*---- CALCULATE RELATIVE PERMEABILITY RATIO AS A FUNCTION OF GAS
*     SATURATION.  USE S FOR INTERPOLATION
C---        Write(6,*)'T2078'
C---    Write(6,*)TSATG(2),TSATG(NPR-1)

      If (SATG.LT.TSATG(1)) Then
        S = TSATG(1)
      Else If (SATG.GT.TSATG(NRP)) Then
        S = TSATG(NRP)
      Else
        S = SATG
      End If
      Call INTPL(TSATG,PRMLGO,S,1,NRP,Y,FRAC,J)
      PRMGO = (1D+1) ** Y

*---- CALCULATE MOBILITY RATIO AND AO2,AG2,RO2,RG2 WHICH CONTAIN PARTS
*     OF THE OIL MATERIAL BALANCE EQUATIONS

      XMOBR = PRMGO * VISO / VISG
      AO2 = POR * (SATO/FVFO+SATG*OGRS*DENRO/FVFG)
      AG2 = POR * (SATG/FVFG+SATO*GORS*DENRG/FVFO)
      RO2 = (1.+OGRS*DENRO*XMOBR*FVFO/FVFG)
      RG2 = GORS * DENRG + XMOBR * FVFO / FVFG
      ROAV = (RO1+RO2) / 2.
      RGAV = (RG1+RG2) / 2.

*---- CALCULATE INCREMENTAL OIL AND GAS PRODUCTION

      If (IHC.EQ.0) Then
        DOILPQ = DGASPQ / RGAV * ROAV
      Else If (IHC.EQ.1) Then
        DGASPQ = DOILPQ / ROAV * RGAV
      End If
      DOILP = DOILPQ * VOLBW
      DGASP = DGASPQ * VOLBW

*---- CALCULATE MATERIAL BALANCE ERROR

      XMBAL1 = AO2 - AO1 + DOILPQ + AG2 - AG1 + DGASPQ
      If (RTE.EQ.0.) XMBAL1 = 0.
      XMBAL = XMBAL1

      SATO1 = SATO
      SATG1 = SATG

*---- FORMAT SPECIFICATIONS
```

```
      Return
 5000 Format (1X,4A16)
 5100 Format (1X,6I16)
 5200 Format (1X,4E16.8)
      End
*****************************************************************
**** TITLE .......:  RATE
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  GMS
**** OUT-CALLS ...:  WHPRS, ISGN
****
**** FUNCTION ....: Determine a well production rate (RTE) which matches
****      the rate and  wellhead pressure requirements.
**** 1.   The highest allowed rate (RTEHI) is tried first. If the calculated
****      wellhead pressure (PWHMIN) is greater than or equal to the spec-
****      ified  minimum wellhead pressure (PWHMIN), the task is finished. A
****      special situation may occur. If QPVT is true, the  rate  has  been
****      reduced  by  subroutine  RESPRS or IPR. It means that lack of PVT-
****      data is limiting the rate.  These  results  should  not  be  used.
****      Control is passed to the main program and execution is terminated.
**** 2.   If  PRSWH  < PWHMIN the rate is lowered until PRSWH = PWHMIN. This
****      is done first by a stepwise search to establish an interval with a
****      solution.  If  an  interval  is found, the rate is calculated by a
****      modified chord method. If an interval is  not  found,  control  is
****      passed to the main program and execution is terminated.
****
**** Summary:
**** 1.   PRSWH >= PWHMIN at the first calculation (PW1>=0)
****        a) QPVT is false.  No need for further calculations.
****           The pressures and rates are OK. Return and continue.
****        b) QPVT is true.   Lack of PVT-data. Rate can not be
****           increased. Return and terminate execution.
**** 2.   PRSWH < PWHMIN at the first calc. (PW1<0)
****      Search for a subinterval (R1,R2) on (RTELO,RTEHI)
****      with a solution to  PRSWH(RTE)=PWHMIN
****        a) An interval is found (PW1<0,PW2>=0)
****           Solve by modified chord method.  Return and continue.
****        b) An interval is not found  (R1=R2=RTELO)
****           Rate can not be reduced below RTELO. Return and terminate.
****
****
*****************************************************************
      Subroutine RATE(PRSHI,PRSLO,PWHMIN,RTEHI,RTELO,EPSPRS,EPSRTE,K,
     *     IPRT,IEXE,IHC,PRS,PRSWF,PRSWH,RTE,DGASP,DOILP,DGORP,QSTOP)

*---- INPUT VARIABLES

      Double Precision PRSHI, PRSLO, PWHMIN, RTEHI, RTELO, EPSPRS,
     *    EPSRTE
      Integer K, IPRT, IEXE, IHC

*---- OUTPUT VARIABLES

      Double Precision PRS, PRSWF, PRSWH, RTE, DGASP, DOILP, DGORP
      Logical QSTOP
      Logical ICRIT
      Common /FLOW/ ICRIT

*---- LOCAL VARIABLES

      Double Precision DRTE, DR, R1, R2, R, PW1, PW2, PWD, PW
      Integer I, J, LOOP, MAXITR, ISGN
      Logical QPVT

*---- COMMON BLOCK

      Common /ICOUNT/ IC, IT
      Integer IC(8), IT(8)

*--- START EXECUTION

      IC(1) = IC(1) + 1
      If (IPRT.GE.4) Write (3,*) 'START RATE'
      QSTOP = .FALSE.
      ICRIT = .FALSE.
      LOOP = 1
      MAXITR = 60
      DRTE = (RTEHI-RTELO) / LOOP
      DRTE = 1.001 * DRTE
      R1 = RTEHI

*---- INITIAL CALCULATION OF WELLHEAD PRESSURE, BOTTOMHOLE PRESSURE
*     RESERVOIR PRESSURE AND RATE

      Call WHPRS(PRSHI,PRSLO,R1,EPSPRS,EPSRTE,K,IPRT,IEXE,IHC,PRS,PRSWF,
     *    PRSWH,RTE,DGASP,DOILP,DGORP,QPVT,QSTOP)
      If (QSTOP) Return
      If (ICRIT) Return
      R1 = RTE
      PW1 = PRSWH - PWHMIN
      If (PW1.GE.0..AND..NOT.QPVT) Return
      If (.NOT.(PW1.GE.0..AND.QPVT)) Then
        If (RTE.LT.RTELO) Go To 40

*---- START LOOP, SEQUENTIAL SEARCH

        J = 0
        IT(1) = IT(1) + 1
   10   Continue
        J = J + 1
        If (J.GT.LOOP) Go To 50
        R = MAX(R1-DRTE,RTELO)

*---- CALCULATE WELLHEAD PRESSURE, BOTTOMHOLE PRESSURE RESERVOIR
*     PRESSURE AND RATE

        Call WHPRS(PRSHI,PRSLO,R,EPSPRS,EPSRTE,K,IPRT,IEXE,IHC,PRS,
     *      PRSWF,PRSWH,RTE,DGASP,DOILP,DGORP,QPVT,QSTOP)
        If (QSTOP) Return
        If (ICRIT) Return
        If (RTE.LT.RTELO) Go To 40
        R = RTE
        PW = PRSWH - PWHMIN

*---- TEST IF A SUBINTERVAL WITH A SOLUTION IS FOUND

        If (ISGN(PW1)*ISGN(PW).GT.0) Then
          R1 = R
          PW1 = PW
          If (R1.LE.RTELO) Go To 60
          Go To 10
        End If

*---- START LOOP, MODIFIED CHORD METHOD (PRSWH CONTROLS THE RATE)

        R2 = R
```

```
      PW2 = PW
      I = 0
   20 Continue
      I = I + 1
      If (I.GE.MAXITR) Go To 70
      If (ABS(PW).GE.EPSPRS.OR.ABS(R1-R2).GE.EPSRTE) Then
        PWD = (PW2-PW1) / (R2-R1)
        DR = PW2 / PWD
        R = R2 - DR

*---- EMERGENCY EXIT

        If ((R1-R)*(R-R2).LE.0) Then
          Write (3,*) 'RATE.    TIME STEP:', K
          Write (3,5200) ABS(R1-R2), ABS(DR), PW2
          Write (3,*)
          R = R2
          Go To 30
        End If

*---- CALCULATE WELLHEAD PRESSURE, BOTTOMHOLE PRESSURE RESERVOIR
*     PRESSURE AND RATE

        Call WHPRS(PRSHI,PRSLO,R,EPSPRS,EPSRTE,K,IPRT,IEXE,IHC,PRS,
     *       PRSWF,PRSWH,RTE,DGASP,DOILP,DGORP,QPVT,QSTOP)
        If (QSTOP) Return
        If (ICRIT) Return
        If (R.NE.RTE) Go To 80
        PW = PRSWH - PWHMIN
        If (ISGN(PW)*ISGN(PW2).GE.0) Then
          PW1 = PW1 / 2.
        Else
          R1 = R2
          PW1 = PW2
        End If
        R2 = R
        PW2 = PW
        Go To 20
      End If

   30 Continue
      RTE = R
      PRSWH = PW + PWHMIN
      Return
    End If

*---- CHECK STOP CONDITIONS,  WRITE COMMENTS

    If (PW1.GE.0..AND.QPVT) Then
      Write (3,*) 'RATE.  TIME STEP:', K
      Write (3,*) 'The rate has been reduced due to'
      Write (3,*) 'lack of PVT-data below PVTMIN'
      Write (3,*) 'The calculated wellhead pressure is higher'
      Write (3,*) 'than the specified minimum.'
      Write (3,*) 'This means that the rate (RTE) is controlled'
      Write (3,*) 'by the minimum PVT-data pressure and not by'
      Write (3,*) 'the minimum wellhead pressure (PWHMIN).'
      Write (3,*) 'Do not use the results from this time step.'
      Write (3,*) 'Return to the main program and terminate.'
      Write (3,*)
      QSTOP = .TRUE.
    End If
   40 Continue
    If (RTE.LT.RTELO) Then
      Write (3,*) 'RATE.  TIME STEP:', K
      Write (3,*) 'The rate (RTE) is now smaller than'
      Write (3,*) 'the minimum allowed rate (RTELO)'
      Write (3,*)
      QSTOP = .TRUE.
    End If
    If (QSTOP) Return
   50 Continue
    If (J.GT.LOOP) Then
      Write (3,*) 'RATE.  TIME STEP:', K
      Write (3,*) 'Sequential search. J=', J
      Write (3,*) 'Too many iterations'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
   60 Continue
    If (R1.EQ.RTELO) Then
      Write (3,*) 'RATE.  TIME STEP:', K
      Write (3,*) 'The target rate results in a too low wellhead'
      Write (3,*) 'pressure.  The rate has been reduced stepwise'
      Write (3,*) 'down to the specified minimum rate, but the'
      Write (3,*) 'wellhead pressure is still too low.'
      Write (3,*) 'Return to the main program and terminate.'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
   70 Continue
    If (I.GE.MAXITR) Then
      Write (3,*) 'RATE.  TIME STEP:', K
      Write (3,5100) 'Convergence not reached after', I, ' iterations'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
   80 Continue
    If (R.NE.RTE) Then
      Write (3,*) 'RATE. ERROR, R .NE. RTE'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If

*---- FORMAT STATEMENTS

 5000 Format (1X,/15('-')/1X,A,I4,A,I4)
 5100 Format (1X,A,I4,A,I4)
 5200 Format (1X,'*EMERGENCY EXIT*',/1X,
     *    'LENGTH OF CURRENT RATE INTERVAL (SM3/D) .....:',G15.8,/1
     *    X,'ESTIMATED ERROR (LENGTH OF NEXT RATE INTERVAL',/1X,
     *    'WHICH IS TOO SMALL FOR CONTINUED ITERATION) (SM3/D)',G15.8,//
     *    1X,'WELLHEAD PRESSURE FUNCTION (SHOULD BE ZERO) (KPA) :',G15
     *    .8,//)
    End
*******************************************************************
**** TITLE .......:  RESPRS
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  WHPRS
**** OUT-CALLS ...:  MATBAL, ISGN
****
**** FUNCTION ....:  Calculate the average reservoir pressure at the end of
****      the current timestep.  This is done by making the material-balance
```

```
****     error, which is calculated by the subroutine MATBAL, approach
****     zero.  Normally, pressure is the free variable.  If the maximum
****     pressure is equal to the minimum PVT data pressure, the rate has
****     to be reduced and is used the free variable for calls to MATBAL.
****
**** NOTE: The last call of subroutine MATBAL before the next timestep has
****     to be done with arguments equal to X and C  to obtain correct
****     calculation of AO1,AG1,RO1,RG1
*****************************************************************
      Subroutine RESPRS(PRSHI,PRSLO,RTEX,EPSPRS,EPSRTE,K,IPRT,IHC,PRS,
     *    RTEY,DGASP,DOILP,QSTOP)

*---- INPUT VARIABLES

      Double Precision PRSHI, PRSLO, RTEX, EPSPRS, EPSRTE
      Integer K, IPRT, IHC

*---- OUTPUT VARIABLES

      Double Precision PRS, RTEY, DGASP, DOILP
      Logical QSTOP

*---- LOCAL VARIABLES

      Double Precision DPRS, Y1, Y2, Y, YD, C, X1, X2, X, DX
      Character ZMODE*10
      Integer J, LOOP, I, MAXITR, ISGN
      Parameter (MAXITR = 30)

*---- COMMON BLOCK

      Common /ICOUNT/ IC, IT
      Integer IC(8), IT(8)

*---- START EXECUTION

      IC(3) = IC(3) + 1
      If (PRSHI.EQ.PRSLO) Then
        ZMODE = 'rate'
        Go To 20
      End If
      If (IPRT.GE.4) Write (3,*) '         START RESPRS'
      QSTOP = .FALSE.
      LOOP = 4
      X1 = PRSHI
      DPRS = (PRSHI-PRSLO) / LOOP * 1.001
      Call MATBAL(X1,RTEX,IPRT,IHC,DGASP,DOILP,Y1)

*---- START LOOP, SEQUENTIAL SEARCH

      IT(3) = IT(3) + 1
      J = 0
   10 Continue
      J = J + 1
      If (J.GT.LOOP) Go To 50
      X2 = MAX(X1-DPRS,PRSLO)
      Call MATBAL(X2,RTEX,IPRT,IHC,DGASP,DOILP,Y2)

*---- TEST IF A SUBINTERVAL WITH A SOLUTION IS FOUND

      If (ISGN(Y1)*ISGN(Y2).LE.0) Then
        ZMODE = 'pressure'
        Go To 20
      End If
      X1 = X2
      Y1 = Y2
      If (X1.LT.PRSLO) Go To 60

*---- TEST IF THE RATE HAS TO BE USED AS THE FREE VARIABLE

      If (X1.EQ.PRSLO) Then
        ZMODE = 'rate'
        Go To 20
      End If
      Go To 10

*---- APPLY THE MODIFIED CHORD METHOD TO FIND THE SOLUTION

   20 Continue
      If (ZMODE.EQ.'pressure') Then
        C = RTEX
        X = X2
        If (Y1.EQ.0.) Then
          X = X1
          Go To 40
        End If
        If (Y2.EQ.0.) Then
          X = X2
          Go To 40
        End If
      Else If (ZMODE.EQ.'rate') Then
        C = PRSLO
        X1 = RTEX
        X2 = 0.
        X = 0.
        Call MATBAL(C,X1,IPRT,IHC,DGASP,DOILP,Y1)
        Call MATBAL(C,X2,IPRT,IHC,DGASP,DOILP,Y2)
        If (Y1.EQ.0.) Then
          X = X1
          Go To 40
        End If
        If (Y2.EQ.0.) Then
          X = X2
          Go To 40
        End If
      Else
        Write (3,*) 'RESPRS. ERROR'
        Write (3,*)
        QSTOP = .TRUE.
        Return
      End If

*---- START LOOP, CHORD METHOD

      I = 0
   30 Continue
      I = I + 1
      If (I.GE.MAXITR) Go To 80
      If (X2-X1.EQ.0..OR.Y2-Y1.EQ.0.) Go To 70
      If (ABS(X1-X2).GE.EPSPRS.OR.ZMODE.NE.'pressure') Then
        If (ABS(X1-X2).GE.EPSRTE.OR.ZMODE.NE.'rate') Then
          YD = (Y2-Y1) / (X2-X1)
          DX = Y2 / YD
          X = X2 - DX

*---- EMERGENCY EXIT

          If ((X1-X)*(X-X2).LE.0) Then
```

```
          Write (3,*) 'RESPRS.  TIME STEP:', K
          If (ZMODE.EQ.'pressure') Then
            Write (3,5100) ABS(X1-X2), ABS(DX), Y2
          Else If (ZMODE.EQ.'rate') Then
            Write (3,5200) ABS(X1-X2), ABS(DX), Y2
          End If
          Write (3,*)
          X = X2
          Go To 40
        End If

*---- CALCULATE MATERIAL-BALANCE ERROR

        If (ZMODE.EQ.'pressure') Then
          Call MATBAL(X,C,IPRT,IHC,DGASP,DOILP,Y)
        Else If (ZMODE.EQ.'rate') Then
          Call MATBAL(C,X,IPRT,IHC,DGASP,DOILP,Y)
        End If
        If (ISGN(Y)*ISGN(Y2).GE.0) Then
          Y1 = Y1 / 2.
        Else
          X1 = X2
          Y1 = Y2
        End If
        X2 = X
        Y2 = Y
        Go To 30
      End If
    End If

*---- PREPARE TO EXIT

 40 Continue
    PRS = X
    RTEY = C
    If (ZMODE.EQ.'rate') Then
      PRS = C
      RTEY = X
    End If
    If (RTEY.LT.RTEX) Then
      Write (3,*) 'RESPRS.  TIME STEP:', K
      Write (3,*) 'THE RATE IS REDUCED DUE TO LACK OF PVT DATA'
      Write (3,*) 'INPUT  RATE: ', RTEX
      Write (3,*) 'OUTPUT RATE: ', RTEY
      Write (3,*)
    End If
    Return

*---- CHECK ERROR CONDITIONS

 50 Continue
    If (J.GT.LOOP) Then
      Write (3,*) 'RESPRS.  TIME STEP:', K
      Write (3,*) 'SEARCH ROUTINE, J=', J
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
 60 Continue
    If (X1.LT.PRSLO) Then
      Write (3,*) 'RESPRS.  TIME STEP:', K
      Write (3,*) 'ERROR,  X1 < PRSLO'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
 70 Continue
    If (X2-X1.EQ.0.) Then
      Write (3,*) 'RESPRS.  TIME STEP:', K
      Write (3,*) 'X2-X1 .EQ. 0.'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
    If (Y2-Y1.EQ.0.) Then
      Write (3,*) 'RESPRS.  TIME STEP:', K
      Write (3,*) 'Y2-Y1 .EQ. 0.'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If
 80 Continue
    If (I.GE.MAXITR) Then
      Write (3,*) 'RESPRS.  TIME STEP:', K
      Write (3,5000) 'CONVERGENCE NOT REACHED AFTER', I, ' ITERATIONS'
      Write (3,*)
      QSTOP = .TRUE.
      Return
    End If

*---- FORMAT STATEMENTS

 5000 Format (1X,A,I4,A,I4)
 5100 Format (1X,'*EMERGENCY EXIT*',/1X,
     *    'LENGTH OF CURRENT PRESSURE INTERVAL (KPA) .........:',G15.8,/1
     *    X,'ESTIMATED ERROR (LENGTH OF NEXT PRESSURE INTERVAL',/1X,
     *    'WHICH IS TOO SMALL FOR CONTINUED ITERATION) (KPA) :',G15.8,//
     *    1X,'MATERIAL BALANCE ERROR (DIMENSIONLESS) ...........:',G15
     *    .8,///)
 5200 Format (1X,'*EMERGENCY EXIT*',/1X,
     *    'LENGTH OF CURRENT RATE INTERVAL (SM3/D) ..........:',G15.8,/1
     *    X,'ESTIMATED ERROR (LENGTH OF NEXT RATE INTERVAL',/1X,
     *    'WHICH IS TOO SMALL FOR CONTINUED ITERATION) (SM3/D)',G15.8,//
     *    1X,'MATERIAL BALANCE ERROR (DIMENSIONLESS) ...........:',G15
     *    .8,///)
     End
************************************************************************
**** TITLE .......:  SKIP
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  GMS
**** OUT-CALLS ...:  NONE
**** FUNCTION ....:  Skip text lines in data files
************************************************************************
      Subroutine SKIP(IUNT)

*---- INPUT VARIABLES

      Integer IUNT

*---- LOCAL VARIABLES

      Integer I, N
      Character ZA*80, ZSPACE*80, ZB*160, Z*1
      Save ZSPACE
      Data ZSPACE /'                                '/

*---- START EXECUTION
```

```
   10 Continue
      Read (IUNT,5000,End=30) ZA
      I = 0
      ZB = ZA // ZSPACE
      N = INDEX(ZB,ZSPACE)
   20 Continue
      I = I + 1
      Z = ZB(I:I)
      If (Z.EQ.' '.AND.I.LT.N) Go To 20
      If (Z.GE.'0'.AND.Z.LE.'9'.OR.Z.EQ.'.'.OR.Z.EQ.'-'.OR.Z.EQ.'+')
     *    Then
        Backspace IUNT
        Go To 30
      End If
      Go To 10
   30 Continue

      Return
 5000 Format (2A)
      End
*****************************************************************
**** TITLE .......:  TEST
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  GMS
**** OUT-CALLS ...:  NONE
**** FUNCTION ....:  DETECT ERROR IN NUMERICAL INPUT DATA, AND WRITE ERROR
****                 MESSAGE
*****************************************************************
      Subroutine TEST(X,X1,X2,ZTXT,IERR)

*---- INPUT VARIABLES

      Double Precision X, X1, X2
      Character ZTXT*(*)

*---- INPUT AND OUTPUT

      Integer IERR

      If (X.LT.X1.OR.X.GT.X2) Then
        IERR = IERR + 1
        Write (3,*)
        Write (3,*) '            *** ERROR ***'
        Write (3,*) 'MESSAGE ..............: ', ZTXT
        Write (3,*) 'VALUE ................: ', X
        Write (3,*) 'PERMITTED INTERVAL ....: ', X1, X2
        Write (3,*)
      End If

      Return
      End
*****************************************************************
**** TITLE .......:  TESTGE
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  GMS
**** OUT-CALLS ...:  NONE
**** FUNCTION ....:  DETECT ERROR IN NUMERICAL INPUT DATA, AND WRITE ERROR
****                 MESSAGE
*****************************************************************
      Subroutine TESTGE(X1,X2,ZTXT,IERR)

*---- INPUT VARIABLES

      Double Precision X1, X2
      Character ZTXT*(*)

*---- INPUT AND OUTPUT

      Integer IERR

      If (X1.GE.X2) Then
        IERR = IERR + 1
        Write (3,*)
        Write (3,*) '            *** ERROR ***'
        Write (3,*) 'VARIABLE .............: ', ZTXT
        Write (3,*) 'VALUE ................: ', X1, X2
        Write (3,*) 'NOT INCREASING'
        Write (3,*)
      End If

      Return
      End
*****************************************************************
**** TITLE .......:  WHPRS
**** AUTHOR ......:  GUNNAR BORTHNE
**** DATE ........:  APRIL 1986
**** IN-CALLS ....:  RATE
**** OUT-CALLS ...:  RESPRS, IPR, TRAVERS, ISGN
**** FUNCTION ....: Calculate the wellhead pressure as a function of rate
****      and other variables. To do so, the average reservoir pressure,
****      bottomhole flowing pressure and pressure loss in tubing must be
****      calculated. While calculating the average reservoir pressure and
****      the well flowing pressure, the rate might get reduced due to lack
****      of PVT data below PVTMIN. QPVT is then set true. This should not
****      stop the execution since the rate might get further reduced
****      because of the specified minimum wellhead pressure. RTEHI1 is the
****      rate input variable and is generally different from RTEHI
*****************************************************************
      Subroutine WHPRS(PRSHI,PRSLO,RTEHI1,EPSPRS,EPSRTE,K,IPRT,IEXE,IHC,
     *    PRS,PRSWF,PRSWH,RTE,DGASP,DOILP,DGORP,QPVT,QSTOP)

*---- INPUT VARIABLES

      Implicit Double Precision (A-Z)
      Integer K, IPRT, IEXE, IHC, CORR, LOOP, I, J, ISGN, MAXITR,
     *   NPVT, NRP, IC(8), IT(8)
      Logical QPVT, QSTOP, ICRIT
      Dimension TDENRG(100), TDENRO(100), TFVFG(100), TVISO(100),
     *    TFVFGX(100), TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100),
     *    TPRMRO(100), PRMLGO(100), TPRS(100), TSATG(100), TVISG(100)

*---- COMMON BLOCK

      Common /CONTAM/ N2, CO2, H2S, NACL, RSI
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     *    VOLBW
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     *    CORR
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     *    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Common /ICOUNT/ IC, IT
      Common /FLOW/ ICRIT

*---- START EXECUTION
```

```
      IC(2) = IC(2) + 1
      LOOP = 5
      MAXITR = 40
      If (IPRT.GE.4) Write (3,*) '     START WHPRS'
      QPVT = .FALSE.
      QSTOP = .FALSE.
      ICRIT = .FALSE.
      R = RTEHI1

*---- CALCULATE RESERVOIR PRESSURE AND RATE

      Call RESPRS(PRSHI,PRSLO,R,EPSPRS,EPSRTE,K,IPRT,IHC,PRS,RTEA,DGASP,
     *     DOILP,QSTOP)
      DGORP = 1E+20
      If (DOILP.NE.0.) DGORP = DGASP / DOILP
      If (QSTOP) Return
      If (RTEA.LT.0.001) Then
        PRSWF = PRS
        PRSWH = PRS
        Go To 60
      End If

*---- CALCULATE BOTTOMHOLE PRESSURE AND RATE

      RTIN = RTEA

      If (IEXE.GE.1) Then
        Call IPR(PRS,PRSLO,RTIN,DGORP,EPSPRS,K,IPRT,IHC,PRSWF,RTEB,QSTOP
     *      )
      Else
        PRSWF = PRS
        RTEB = RTEA
      End If
      If (QSTOP) Return
      FN = RTEA - RTEB

      If (FN.EQ.0.) Then
        Go To 30
      Else If (FN.GT.0.) Then
        QPVT = .TRUE.
      Else
        Write (3,*) 'ERROR'
        Stop
      End If

*---- START LOOP, SEQUENTIAL SEARCH

      IT(2) = IT(2) + 1
      R1 = R
      FN1 = FN
      DRTE = R / LOOP * 1.001
      J = 0
   10 Continue
      J = J + 1
      If (J.GT.LOOP) Go To 60
      R = MAX(R1-DRTE,0D+0)

*---- CALCULATE RESERVOIR PRESSURE AND RATE

      Call RESPRS(PRSHI,PRSLO,R,EPSPRS,EPSRTE,K,IPRT,IHC,PRS,RTEA,DGASP,
     *     DOILP,QSTOP)
      DGORP = 1E+20
      If (DOILP.NE.0.) DGORP = DGASP / DOILP
      If (QSTOP) Return

*---- CALCULATE BOTTOMHOLE PRESSURE AND RATE

      RTIN = RTEHI1
      Call IPR(PRS,PRSLO,RTIN,DGORP,EPSPRS,K,IPRT,IHC,PRSWF,RTEB,QSTOP)
      If (QSTOP) Return
      FN = RTEA - RTEB

*---- TEST IF A SUBINTERVAL WITH SOLUTION IS FOUND

      If (ISGN(FN1)*ISGN(FN).GT.0) Then
        R1 = R
        FN1 = FN
        If (R1.LE.0.) Go To 40
        Go To 10
      End If

*---- START LOOP, MODIFIED CHORD METHOD

      FN2 = FN
      R2 = R
      I = 0
   20 Continue
      I = I + 1
      If (I.GE.MAXITR) Go To 50
      If (ABS(R1-R2).GE.EPSRTE) Then
        FND = (FN2-FN1) / (R2-R1)
        DR = FN2 / FND
        R = R2 - DR

*---- EMERGENCY EXIT

        If ((R1-R)*(R-R2).LE.0) Then
          Write (3,*) 'WHPRS.   TIME STEP:', K
          Write (3,5200) ABS(R1-R2), ABS(DR)
          Write (3,*)
          R = R2
          Go To 30
        End If

*---- CALCULATE RESERVOIR PRESSURE AND RATE

        Call RESPRS(PRSHI,PRSLO,R,EPSPRS,EPSRTE,K,IPRT,IHC,PRS,RTEA,
     *       DGASP,DOILP,QSTOP)
        DGORP = 1E+20
        If (DOILP.NE.0.) DGORP = DGASP / DOILP
        If (QSTOP) Return

*---- CALCULATE BOTTOMHOLE PRESSURE AND RATE

        RTIN = RTEHI1
        Call IPR(PRS,PRSLO,RTIN,DGORP,EPSPRS,K,IPRT,IHC,PRSWF,RTEB,
     *  QSTOP)
        If (QSTOP) Return
        FN = RTEA - RTEB
        If (ISGN(FN)*ISGN(FN2).GE.0) Then
          FN1 = FN1 / 2.
        Else
          R1 = R2
          FN1 = FN2
        End If
        R2 = R
        FN2 = FN
        Go To 20
```

```
      End If

   30 Continue
      RTE = R

*---- CALCULATE WELLHEAD PRESSURE

      If (IEXE.GE.2.AND.RTE.NE.0.) Then
        TWF = TWF * 1.8 + 460
        TWH = TWH * 1.8 + 460
        PRSWF = PRSWF * 0.14503774
        QO = RTE * 6.289811
        QG = RTE * DGORP * 35.314667
        DIAM = DIAM * 3.28084 * 12.
        LENGTH = LENGTH * 3.28084
        AROUGH = AROUGH * 3.28084
        Call TRAVERS(TWH,TWF,PRSWF,GRVG,API,QO,QG,DIAM,LENGTH,AROUGH,
     *       SIGMAO,CORR,PRSWH)
        TWF = (TWF-460) / 1.8
        TWH = (TWH-460) / 1.8
        PRSWF = PRSWF / 0.14503774
        DIAM = DIAM / 3.28084 / 12.
        LENGTH = LENGTH / 3.28084
        AROUGH = AROUGH / 3.28084
        PRSWH = PRSWH / 0.14503774
      Else
        PRSWH = PRSWF
      End If
      If (QSTOP) Return
      If (ICRIT) Return
      Return

*---- CHECK ERROR CONDITIONS

   40 Continue
      If (R1.LE.0.) Then
        Write (3,*) 'WHPRS. TIME STEP:', K
        Write (3,*) 'Interval with solution not found'
        Write (3,*)
        QSTOP = .TRUE.
        Return
      End If
   50 Continue
      If (I.GE.MAXITR) Then
        Write (3,*) 'WHPRS. TIME STEP:', K
        Write (3,5100) 'Convergence not reached after', I, ' iterations'
        Write (3,*)
        QSTOP = .TRUE.
        Return
      End If
   60 Continue
      If (J.GT.LOOP) Then
        Write (3,*) 'WHPRS. TIME STEP:', K
        Write (3,*) 'Sequential search. J=', J
        Write (3,*) 'Too many iterations'
        Write (3,*)
        QSTOP = .TRUE.
        Return
      End If
      If (ABS(RTEA).LE.0.001.OR.ABS(RTEB).LE.0.001) Then
        Write (3,*) 'WHPRS. TIME STEP:', K
        Write (3,*) 'The rate is zero (or nearly zero)'
        Write (3,*) 'Terminate execution'
        Write (3,*)
        QSTOP = .TRUE.
        Return
      End If

*---- FORMAT STATEMENTS

 5000 Format (1X,/1X,A,I4,A,I4)
 5100 Format (1X,A,I4,A,I4)
 5200 Format (1X,'*EMERGENCY EXIT*',/1X,
     *    'LENGTH OF CURRENT RATE INTERVAL (SM3/D) .....:',G15.8,/1
     *    X,'ESTIMATED ERROR (LENGTH OF NEXT RATE INTERVAL',/1X,
     *    'WHICH IS TOO SMALL FOR CONTINUED ITERATION) (SM3/D)',G15.8,//
     *    )
      End

*****************************************************************
***** Subroutine TRAVERS
***** Author      James Carroll
***** Date        November, 1990
***** In-calls
***** Out-calls
***** Function    This subroutine performs the pressure traverse
*****             calculations for a given multi-phase flow routine.
***** Reference   Two-Phase Flow in Pipes, J.A. Brill & H.D. Beggs, 1989
*****************************************************************
      Subroutine TRAVERS(TWH,TWF,PWF,GGRAV,API,QO,QG,DIAM,L,AROUGH,
     *    SIGMAO,CORR,PWH)

      Implicit Double Precision (A-Z)
      Integer ITER, CORR, IREG
      Logical ICRIT
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI
      Common /FLOW/ ICRIT

*---- Initialize parameters

      ICRIT = .FALSE.
      WGRAV = 1.07
      NACL = .001
      SIGMAW = 70.
      QW = 0.

*---- Make the necessary conversions

      QO = QO * 5.615
      QW = QW * 5.615
      DIAM = DIAM / 12

*---- Start pressure traverse

      PSTAR = PWF
      ZSTAR = 0.
      DZ = L / 10.
      DP = 100.

*---- Start main loop
*     Test if dZ will make Zstar greater than L

   10 ITER = 0
      If ((L-ZSTAR).LT.DZ) DZ = L - ZSTAR

*---- (Re) Evaluate Z and T

   20 Z = ZSTAR + DZ / 2.
```

```
      T = TWF - (TWF-TWH) * Z / L

*---- (Re) Evaluate P and solve for dPnew given dZ

   30 ITER = ITER + 1
      P = PSTAR - DP / 2.
      If (CORR.EQ.1) Then
        Call AZIZ(T,P,GGRAV,WGRAV,API,QO,QW,QG,DIAM,DZ,AROUGH,SIGMAO,
     *       SIGMAW,NX,NY,DPDZ,DPNEW,IREG,EK)
      Else If (CORR.EQ.2) Then
        Call HAGBRWN(T,P,GGRAV,WGRAV,API,QO,QW,QG,DIAM,DZ,AROUGH,SIGMAO,
     *       SIGMAW,NX,NY,DPDZ,DPNEW,IREG,EK)
      Else If (CORR.EQ.3) Then
        Call ORK(T,P,GGRAV,WGRAV,API,QO,QW,QG,DIAM,DZ,AROUGH,SIGMAO,
     *       SIGMAW,NX,NY,DPDZ,DPNEW,IREG,EK)
      Else
        Print *, 'Invalid correlation number in TRAVERS.FOR: ', CORR
        Stop
      End If

*---- Test for Critical Flow

      If (.NOT.ICRIT) Then

*---- Test for excessive iterations

        If (ITER.GT.10) Then
          DZ = .95 * DZ
          DP = .95 * DPNEW
          ITER = 0
          Go To 20
        End If

*---- Test if dP is greater than 100 psi

        If (DPNEW.GT.100.) Then
          DZ = 90 * DZ / DPNEW
          DP = 90
          Go To 20
        End If

*---- Test for convergence

        If (ABS(DPNEW-DP).GT.0.1) Then
          DP = DPNEW
          Go To 30
        End If

*---- Update Zstar and Pstar and repeat if necessary

        ZSTAR = ZSTAR + DZ
        DP = DPNEW
        PSTAR = PSTAR - DP
        If (ZSTAR.LT.L) Go To 10

        PWH = PSTAR
      End If
      QO = QO / 5.615
      QW = QW / 5.615
      DIAM = DIAM * 12

      Return
      End
*************************************************************************
```

```
***** Subroutine AZIZ
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Performs the Aziz, Govier, and Fogarasi multi-phase
*****           flow correlation.
***** Reference  The Flow of Complex Mixtures in Pipes, Govier & Aziz,
*****           1972.  Pressure Drop in Wells Producing Oil and Gas,
*****           Aziz, Govier, Fogarasi, 1972.
*************************************************************************
      Subroutine AZIZ(T,P,GGRAV,WGRAV,API,QO,QW,QG,DIAM,L,AROUGH,SIGMAO,
     *    SIGMAW,NX,NY,DPDZ,DP,IFLOW,EK)

*---- This program determines the pressure loss for a certain flow
*---- regime and a certain length of travel.

      Implicit Double Precision (A-Z)
      Integer IFLOW
      Logical ICRIT
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI
      Common /FLOW/ ICRIT

*---- Solve for basic variables.

      ICRIT = .FALSE.
      EK = 0.
      GC = 32.2
      G = GC
      RROUGH = AROUGH / DIAM
      AREA = 3.141592654 * DIAM * DIAM / 4
      QL = QO + QW
      FO = QO / QL
      FW = QW / QL
      SIGMAL = SIGMAO * FO + SIGMAW * FW

*---- Determine superficial velocities

      Call ZFACTOR(T,P,GGRAV,Z)
      Call SGOR(T,P,GGRAV,API,RS)
      Call BPP(T,GGRAV,API,PBP)
      Call FVF(T,P,GGRAV,API,RS,PBP,CO,BO,BW)
      VSG = ((QG-RS*QO/5.615)*14.696*T*Z) / (AREA*86400*P*520)
      If (VSG.LT.0) VSG = .01
      VSL = QL * (BO*FO+BW*FW) / (AREA*86400)
      VM = VSL + VSG

*---- Determine densities.

      Call DENS(T,P,GGRAV,WGRAV,API,DENSO,DENSW,DENSG)
      DENSL = DENSO * FO + DENSW * FW

*---- Determine viscosities.

      Call VISC(T,P,GGRAV,API,RS,PBP,UG,UO,UW)
      UL = UO * FO + UW * FW

*---- Determine the current flow conditions and flow regime boundaries.

      DUMMY = ((72.*DENSL)/(62.4*SIGMAL)) ** .25
      NY = VSL * DUMMY
      NX = VSG * DUMMY * (DENSG/.0764) ** .33333
      BOUND1 = .51 * (100.*NY) ** .172
      BOUND2 = 8.6 + 3.8 * NY
```

```
      BOUND3 = 70. * (100.*NY) ** (-.152)

*---- Determine the flow regime.
*       1 = Bubble
*       2 = Slug
*       3 = Transition
*       4 = Annular-mist

      If (NY.LT.4.) Then
        If (NX.LT.BOUND1) Then
          IFLOW = 1
        Else If (NX.LT.BOUND2) Then
          IFLOW = 2
        Else If (NX.LT.BOUND3) Then
          IFLOW = 3
        Else
          IFLOW = 4
        End If
      Else
        If (NX.LT.BOUND1) Then
          IFLOW = 1
        Else If (NX.LT.26.5) Then
          IFLOW = 2
        Else
          IFLOW = 4
        End If
      End If

*---- Transfer control to the appropriate correlation

      Go To (10,20,20,30)IFLOW

*---- BUBBLE FLOW
*     Determine the pressure gradient of elevation change.

   10 VBS = 1.41 * (SIGMAL*G*(DENSL-DENSG)/DENSL**2) ** .25
      VBF = 1.2 * VM + VBS
      HL = 1 - VSG / VBF
      DENSS = DENSL * HL + DENSG * (1-HL)
      DPDZEL = DENSS / 144

*---- Determine the pressure gradient due to friction.

      Call FRICFAC(DENSL,VM,DIAM,UL,RROUGH,FM)
      DPDZF = FM * DENSS * VM * VM / (2*GC*DIAM*144)

*---- Determine the total pressure gradient, ignoring acceleration.

      DPDZ = DPDZEL + DPDZF

      Go To 40

*---- SLUG FLOW
*     Determine the pressure gradient due to elevation.

   20 NE = G * DIAM * DIAM * (DENSL-DENSG) / SIGMAL
      NV = (DIAM*DIAM*DIAM*G*DENSL*(DENSL-DENSG)) ** .5 / UL
      If (NV.LE.18) Then
        M = 25.
      Else If (NV.LT.250) Then
        M = 69. * NV ** (-.35)
      Else
        M = 10.
      End If
      C = .345 * (1.-DEXP(-.029*NV)) * (1-DEXP((3.37-NE)/M))
      VBS = C * (G*DIAM*(DENSL-DENSG)/DENSL) ** .5
      VBF = 1.2 * VM + VBS
      HL = 1 - VSG / VBF
      DENSS = DENSL * HL + DENSG * (1-HL)
      DPDZEL = DENSS / 144

*---- Determine the pressure gradient due to friction.

      Call FRICFAC(DENSL,VM,DIAM,UL,RROUGH,FM)
      DPDZF = FM * DENSL * HL * VM * VM / (2*GC*DIAM*144)

*---- Determine the total pressure gradient, ignoring acceleration.

      DPDZ = DPDZEL + DPDZF

      If (IFLOW.EQ.2) Go To 40
      DPDZSLG = DPDZ

*---- MIST FLOW
*     Determine the pressure gradient of the elevation.

   30 DENSN = DENSL * VSL / VM + DENSG * VSG / VM
      DPDZEL = DENSN / 144

*---- Determine the pressure gradient due to friction.

      DUMMY = (VSG*UL/SIGMAL) ** 2 * DENSG / DENSL
      If (DUMMY.LE.0.005) Then
        MROUGH = .0749 * SIGMAL / (DENSG*VSG*VSG*DIAM)
      Else
        MROUGH = .3713 * SIGMAL * DUMMY ** .302 / (DENSG*VSG*VSG*DIAM)
      End If
      If (MROUGH.LT.RROUGH) MROUGH = RROUGH
      If (MROUGH.GT..5) MROUGH = .5
      If (MROUGH.LT..001) MROUGH = .001

      If (MROUGH.GT..05) Then
        FM = 4 * (1/((4*DLOG10(.27*MROUGH))**2)+.067*MROUGH**1.73)
      Else
        Call FRICFAC(DENSG,VSG,DIAM,UG,MROUGH,FM)
      End If
      DPDZF = FM * DENSG * VSG * VSG / (2*GC*DIAM*144)

*---- Determine the total pressure gradient, including acceleration
*     effects.

      EK = VM * VSG * DENSN / GC / P / 144
      DPDZ = (DPDZEL+DPDZF) / (1-EK)

*---- TRANSITION FLOW

      If (IFLOW.NE.4) Then
        DPDZMST = DPDZ
        WEIGHT = (BOUND3-NX) / (BOUND3-BOUND2)
        DPDZ = WEIGHT * DPDZSLG + (1-WEIGHT) * DPDZMST
      End If

*---  Test for Critical Flow

      If (DPDZ.GT..6) ICRIT = .TRUE.

*---- Determine the total pressure loss.
```

```
   40 DP = DPDZ * L

      Return
      End
*****************************************************************
***** Subroutine HAGBRWN
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Performs the Hagedorn & Brown Multi-phase flow
*****           correlation.
***** Reference  Two-Phase Flow in Pipes, Brill & Beggs, 1989. Original
*****           paper, Hagedorn & Brown, JPT, April, 1965. Gas
*****           Production Operations, H.D. Beggs, 1984.
*****************************************************************
      Subroutine HAGBRWN(T,P,GGRAV,WGRAV,API,QO,QW,QG,DIAM,L,AROUGH,
     *    SIGMAO,SIGMAW,NX,NY,DPDZ,DP,IREG,EK)

*---- This program determines the pressure loss in a vertical flow
*     string using the Hagedorn and Brown correlation.

      Implicit Double Precision (A-Z)
      Integer IREG, K
      Logical ICRIT
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI
      Common /FLOW/ ICRIT
      Dimension XHL(12), YHL(12), XCNL(10), YCNL(10), XPSI(12),
     *    YPSI(12), XHLL(12), XCNLL(10), YCNLL(10)

*---- Enter data arrays for liquid holdup correlation.

      Data XHL /.2, .5, 1., 2., 5., 10., 20., 50., 100., 200., 300.,
     *    1000./
      Data YHL /.04, .09, .15, .18, .25, .34, .44, .65, .82, .92, .96,
     *    1./
      Data XCNL /.002, .005, .01, .02, .03, .06, .1, .15, .2, .4/
      Data YCNL /.0019, .0022, .0024, .0028, .0033, .0047, .0064, .008,
     *    .009, .0115/
      Data XPSI /.01, .02, .025, .03, .035, .04, .045, .05, .06, .07,
     *    .08, .09/
      Data YPSI /1., 1.1, 1.23, 1.4, 1.53, 1.6, 1.65, 1.68, 1.74, 1.78,
     *    1.8, 1.83/

*---- Solve for basic variables.

      EK = 0.
      ICRIT = .FALSE.
      RROUGH = AROUGH / DIAM
      AREA = 3.141592654 * DIAM * DIAM / 4
      QL = QO + QW
      FO = QO / QL
      FW = QW / QL
      SIGMAL = SIGMAO * FO + SIGMAW * FW

*---- Determine superficial velocities

      Call ZFACTOR(T,P,GGRAV,Z)
      Call SGOR(T,P,GGRAV,API,RS)
      Call BPP(T,GGRAV,API,PBP)
      Call FVF(T,P,GGRAV,API,RS,PBP,CO,BO,BW)
      VSG = ((QG-RS*QO/5.615)*14.696*T*Z) / (P*520.*AREA*86400)
      If (VSG.LT.0) VSG = 0.01
      VSL = QL * (BO*FO+BW*FW) / (AREA*86400)
      VM = VSG + VSL
      HLNS = VSL / VM

*---- Determine densities.

      Call DENS(T,P,GGRAV,WGRAV,API,DENSO,DENSW,DENSG)
      DENSL = DENSO * FO + DENSW * FW

*---- Determine viscosities.

      Call VISC(T,P,GGRAV,API,RS,PBP,UG,UO,UW)
      UL = UO * FO + UW * FW

*---- Determine plotting parameters

      DUMMY = ((72.*DENSL)/(62.4*SIGMAL)) ** .25
      NY = VSL * DUMMY
      NX = VSG * DUMMY * (DENSG/.0764) ** .33333

*---- Determine if single phase flow exists.

      If (HLNS.GE.1) Then
        HL = 1.
        HLNS = 1.
        IREG = 1
        Go To 30
      Else If (HLNS.LE.0) Then
        HL = 0.
        HLNS = 0.
        IREG = 2
        Go To 30
      End If

*---- Check for bubble flow

      XLB = 1.071 - .2281 * VM * VM / DIAM
      If (XLB.LT..13) XLB = .13
      HGNS = VSG / (VSG+VSL)
      If (HGNS.LT.XLB) Then
        IREG = 3
        VS = .8
        HL = 1. - .5 * (1.+VM/VS-SQRT((1.+VM/VS)**2-4*VSG/VS))
        If (HL.LT.HLNS) HL = HLNS
        DENSS = DENSL * HL + DENSG * (1-HL)
        DPDZEL = DENSS / 144
        Call FRICFAC(DENSL,(VSL/HL),DIAM,UL,RROUGH,FF)
        DPDZF = FF * DENSL * (VSL/HL) ** 2 / (2.*32.2*DIAM*144)
        EK = 0.
        Go To 40
      End If

*---- Determine dimensionless numbers

      XNLV = 1.938 * VSL * (DENSL/SIGMAL) ** .25
      XNGV = 1.938 * VSG * (DENSL/SIGMAL) ** .25
      XND = 120.872 * DIAM * (DENSL/SIGMAL) ** .5
      XNL = 0.15726 * UL * (1/DENSL/SIGMAL**3) ** .25

*---- Prepare holdup correlations for interpolation.

      Do 10 K = 1, 10
        XCNLL(K) = DLOG(XCNL(K))
        YCNLL(K) = DLOG(YCNL(K))
   10 Continue
```

```
      Do 20 K = 1, 12
        XHLL(K) = DLOG(1.E-05*XHL(K))
   20 Continue

*---- Calculate liquid holdup

      IREG = 4
      XX = DLOG(XNL)
      CNL = DEXP(FLAGR(XCNLL,YCNLL,XX,2,10))
      XX = DLOG(XNLV*CNL/(XNGV**.575*XND)*(P/14.7)**.1)
      HL = FLAGR(XHLL,YHL,XX,2,12)
      XX = XNGV * XNL ** 0.38 / XND ** 2.14
      PSI = FLAGR(XPSI,YPSI,XX,2,12)
      If (PSI.LT.1.) PSI = 1.
      HL = HL * PSI
      If (HL.LT.0) HL = 0.
      If (HL.GT.1) HL = 1.

*---- Calculate slip and no-slip mixture densities

   30 DENSNS = DENSL * HLNS + DENSG * (1.-HLNS)
      DENSS = DENSL * HL + DENSG * (1.-HL)

*---- Calculate friction factor

      US = UL ** HL * UG ** (1-HL)
      Call FRICFAC(DENSNS,VM,DIAM,US,RROUGH,FF)

*---- Determine the total pressure loss.

      DPDZEL = DENSS / 144
      DPDZF = FF * DENSNS ** 2 * VM ** 2 / (2.*32.2*DIAM*DENSS*144)
      VSG = VM * (1-HLNS)
      EK = DENSS * VM * VSG / (32.2*P*144)
   40 DPDZ = (DPDZEL+DPDZF) / (1.-EK)
      DP = DPDZ * L

*---- Test for Critical Flow

      If (DPDZ.GT..6) ICRIT = .TRUE.

      Return
      End
*************************************************************************
***** Subroutine ORK
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Performs the Orkeszewski Multi-Phase Flow correlation.
***** Reference  Two-Phase Flow in Pipes, Brill & Beggs, 1989.
*************************************************************************
      Subroutine ORK(T,P,GGRAV,WGRAV,API,QO,QW,QG,DIAM,L,AROUGH,SIGMAO,
     *    SIGMAW,NLV,NGV,DPDZ,DP,IFLOW,EK)

*---- This program determines the pressure loss given by the
*     Orkeszewski multiphase flow correlation.

      Implicit Double Precision (A-Z)
      Integer IFLOW, I
      Logical ICRIT
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI
      Common /FLOW/ ICRIT
```

```
*---- Solve for basic variables.

      ICRIT = .FALSE.
      EK = 0.
      GC = 32.2
      G = GC
      RROUGH = AROUGH / DIAM
      AREA = 3.1415926554 * DIAM * DIAM / 4
      QL = QO + QW
      FO = QO / QL
      FW = QW / QL
      SIGMAL = SIGMAO * FO + SIGMAW * FW

*---- Determine superficial velocities

      Call ZFACTOR(T,P,GGRAV,Z)
      Call SGOR(T,P,GGRAV,API,RS)
      Call BPP(T,GGRAV,API,PBP)
      Call FVF(T,P,GGRAV,API,RS,PBP,CO,BO,BW)
      VSG = ((QG-RS*QO/5.615)*14.696*T*Z) / (AREA*86400*P*520)
      If (VSG.LT.0) VSG = .01
      VSL = QL * (BO*FO+BW*FW) / (AREA*86400)
      VM = VSL + VSG
      HGNS = VSG / VM
      HLNS = VSL / VM

*---- Determine densities.

      Call DENS(T,P,GGRAV,WGRAV,API,DENSO,DENSW,DENSG)
      DENSL = DENSO * FO + DENSW * FW

*---- Determine viscosities.

      Call VISC(T,P,GGRAV,API,RS,PBP,UG,UO,UW)
      UL = UO * FO + UW * FW

*---- Determine the current flow conditions and flow regime boundaries.

      DUMMY = 1.938 * (DENSL/SIGMAL) ** .25
      NLV = VSL * DUMMY
      NGV = VSG * DUMMY
      BOUND1 = 1.071 - .2218 * VM * VM / DIAM
      If (BOUND1.LT..13) BOUND1 = .13
      BOUND2 = 50 + 36 * NLV
      BOUND3 = 75 + 84 * NLV ** .75

*---- Determine the flow regime.
*     1 = Bubble
*     2 = Slug
*     3 = Transitional
*     4 = Mist

      If (HGNS.LT.BOUND1) Then
        IFLOW = 1
      Else If (NGV.LT.BOUND2) Then
        IFLOW = 2
      Else If (NGV.LT.BOUND3) Then
        IFLOW = 3
      Else
        IFLOW = 4
      End If

*---- Transfer control to appropriate correlation
```

```
      Go To (10,20,20,50)IFLOW

*---- BUBBLE FLOW

   10 VS = .8
      HL = 1. - .5 * (1.+VM/VS-SQRT((1.+VM/VS)**2-4*VSG/VS))
      If (HL.LT.HLNS) HL = HLNS
      DENSS = DENSL * HL + DENSG * (1-HL)
      DPDZEL = DENSS / 144
      Call FRICFAC(DENSL,(VSL/HL),DIAM,UL,RROUGH,FF)
      DPDZF = FF * DENSL * (VSL/HL) ** 2 / (2.*32.2*DIAM*144)
      EK = 0.
      DPDZ = (DPDZEL+DPDZF) / (1.-EK)
      Go To 60

*---- SLUG FLOW
*     Determine Vb

   20 VB = .5 * SQRT(G*DIAM)
      Do 30 I = 1, 5
        NREB = DENSL * VB * DIAM / UL
        NREL = DENSL * VM * DIAM / UL
        If (NREB.LE.3000) Then
          VB = (.546+8.74D-6*NREL) * SQRT(G*DIAM)
        Else If (NREB.GE.8000) Then
          VB = (.35+8.74D-6*NREL) * SQRT(G*DIAM)
        Else
          PSI = (.251+8.74D-6*NREL) * SQRT(G*DIAM)
          VB = .5 * (PSI+SQRT(PSI*PSI+13.59*UL/DENSL/SQRT(DIAM)))
        End If
   30 Continue
      If ((QW/(QO+QW)).GT..75) Then
        If (VM.LT.10) Then
          DEL = (0.013*DLOG10(UL)) / DIAM ** 1.38 - .681 + .232 *
     *         DLOG10(VM) - .428 * DLOG10(DIAM)
        Else
          DEL = (0.045*DLOG10(UL)) / DIAM ** .799 - .709 - .162 *
     *         DLOG10(VM) - .888 * DLOG10(DIAM)
        End If
      Else
        If (VM.LT.10) Then
          DEL = (0.0127*DLOG10(UL+1)) / DIAM ** 1.415 - .284 + .167 *
     *         DLOG10(VM) + .113 * DLOG10(DIAM)
        Else
          XX = -DLOG10(VM) * (.01*DLOG10(UL+1)/DIAM**1.571+.397+.63*
     *         DLOG10(DIAM))
          DEL = (0.0274*DLOG10(UL+1)) / DIAM ** 1.371 + .161 + .569 *
     *         DLOG10(DIAM) + XX
        End If
      End If
      XX = -.065 * VM
      If (VM.LE.10.AND.DEL.LT.XX) DEL = XX
   40 DENSS = (DENSL*(VSL+VB)+DENSG*VSG) / (VM+VB) + DENSL * DEL
      XX = (-VB/(VM+VB)) * (1-DENSS/DENSL)
      If (VM.GT.10.AND.DEL.LT.XX) Then
        DEL = XX
        Go To 40
      End If
      DPDZEL = DENSS / 144

*---- Determine friction gradient

      Call FRICFAC(DENSL,VM,DIAM,UL,RROUGH,FM)
      DPDZF = FM * DENSL * VM * VM * ((VSL+VB)/(VM+VB)+DEL) / (2*GC*DIAM
     *    *144)

*---- Determine the total pressure gradient, ignoring acceleration.

      DPDZ = DPDZEL + DPDZF

      If (IFLOW.EQ.2) Go To 60
      DPDZSLG = DPDZ

*---- MIST FLOW
*     Determine the pressure gradient of the elevation.

   50 DENSN = DENSL * VSL / VM + DENSG * VSG / VM
      DPDZEL = DENSN / 144

*---- Determine the pressure gradient due to friction.

      DUMMY = (VSG*UL/SIGMAL) ** 2 * DENSG / DENSL
      If (DUMMY.LE.0.005) Then
        MROUGH = .0749 * SIGMAL / (DENSG*VSG*VSG*DIAM)
      Else
        MROUGH = .3713 * SIGMAL * DUMMY ** .302 / (DENSG*VSG*VSG*DIAM)
      End If
      If (MROUGH.LT.RROUGH) MROUGH = RROUGH
      If (MROUGH.GT..5) MROUGH = .5
      If (MROUGH.LT..001) MROUGH = .001

      If (MROUGH.GT..05) Then
        FM = 4 * (1/((4*DLOG10(.27*MROUGH))**2)+.067*MROUGH**1.73)
      Else
        Call FRICFAC(DENSG,VSG,DIAM,UG,MROUGH,FM)
      End If
      DPDZF = FM * DENSG * VSG * VSG / (2*GC*DIAM*144)

*---- Determine the total pressure gradient, including acceleration
*     effects.

      EK = VM * VSG * DENSN / GC / P / 144.
      DPDZ = (DPDZEL+DPDZF) / (1-EK)
      If (IFLOW.NE.4) Then
        DPDZMST = DPDZ

*---- TRANSITION FLOW

        WEIGHT = (BOUND3-NGV) / (BOUND3-BOUND2)
        DPDZ = WEIGHT * DPDZSLG + (1-WEIGHT) * DPDZMST
      End If

*---- Test for Critical Flow

      If (DPDZ.GT..6) ICRIT = .TRUE.

*---- Determine the total pressure loss.

   60 DP = DPDZ * L

      Return
      End
*************************************************************************
***** Subroutine FLAGR
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
```

```
***** Function   Interpolation routine similar to FLAGR in Applied
*****            Numerical Method by Carnhan, Luther and Wilkes, John
*****            Wiley and Sons, Pg. 31.  FLAGR uses the Lagrange formula
*****            to evaluate the interpolating polynomial of degree ideg
*****            for argument XARG using the data values X(min)...X(max)
*****            and Y(min)...Y(max) where MIN=MAX-IDEG.  The X(i) values
*****            are not necessarily evenly spaced and can be in either
*****            increasing or decreasing order. X is the array of
*****            independent variable data point. Y is the array of
*****            dependent variable data points.  XARG is the argument
*****            for which an interpolated value is desired. IDEG is the
*****            degree of interpolated polynomial (1 is linear, 2 is
*****            quadratic, etc.) NPTS is the number of data points in X
*****            and Y.
***** Reference  Taken from Gas Production Operations, Beggs, 1984.
**********************************************************************
      Function FLAGR(X,Y,XARG,IDEG,NPTS)

      Implicit Double Precision (A-Z)
      Integer IDEG, NPTS, N, N1, L, MIN, MAX, I, J
      Dimension X(1), Y(1)

      N = IABS(NPTS)
      N1 = IDEG + 1
      L = 1
      If (X(2).LE.X(1)) L = 2

*---- Check to be sure that XARG is within range of X(i) values
*     for interpolation purposes.  If it is not, set FLAGR equal
*     to the appropriate terminal value (Y91) or Y(n)) and return.
*     Note that this precludes extrapolation of data.

      Go To (10,20), L
   10 If (XARG.LE.X(1)) Go To 30
      If (XARG.GE.X(N)) Go To 40
      Go To 50
   20 If (XARG.LT.X(1)) Then
        If (XARG.LE.X(N)) Go To 40
        Go To 50
      End If
   30 FLAGR = Y(1)
      Return
   40 FLAGR = Y(N)
      Return

*---- Determine value of Max

   50 Go To (60,80), L

*---- Data are in order of increasing values of X.

   60 Do 70 MAX = N1, N
        If (XARG.LT.X(MAX)) Go To 100
   70 Continue

*---- Data are in order of decreasing values of X.

   80 Do 90 MAX = N1, N
        If (XARG.GT.X(MAX)) Go To 100
   90 Continue

*---- Compute value of factor.

  100 MIN = MAX - IDEG
      FACTOR = 1.
      Do 110 I = MIN, MAX
        If (XARG.EQ.X(I)) Then
          FLAGR = Y(I)
          Return
        End If
        FACTOR = FACTOR * (XARG-X(I))
  110 Continue

*---- Evaluate interpolating polynomial

      YEST = 0.
      Do 130 I = MIN, MAX
        TERM = Y(I) * FACTOR / (XARG-X(I))
        Do 120 J = MIN, MAX
          If (I.NE.J) TERM = TERM / (X(I)-X(J))
  120   Continue
        YEST = YEST + TERM
  130 Continue
      FLAGR = YEST

      Return
      End
**********************************************************************
***** Subroutine BPP
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Estimates the bubble point pressure of a black oil
*****            mixture using the correlation of Vasquez and Beggs.
***** Reference  Thesis, M.E. Vasquez, University of Tulsa, 1976. HP41C
*****            Petroleum Fluids Pack Manual.
**********************************************************************
      Subroutine BPP(T,GGRAV,API,PBP)

      Implicit Double Precision (A-Z)
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI

      If (API.GT.30) Then
        A = .0178
        B = 1.187
        C = 23.931
      Else
        A = .0362
        B = 1.0937
        C = 25.724
      End If

*---- Estimate the bubble-point pressure.

      PBP = (RSI*5.615/(A*GGRAV*DEXP(C*API/T))) ** (1/B)

      Return
      End
**********************************************************************
***** Subroutine SGOR
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Estimates the solution gas oil ration of a black oil
*****            mixture.
***** Reference  Thesis, M.E. Vasquez, University of Tulsa, 1976. HP41C
```

```
*****            Petroleum Fluids Pack Manual.
**********************************************************************
      Subroutine SGOR(T,P,GGRAV,API,RS)

      Implicit Double Precision (A-Z)
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI

      If (API.GT.30) Then
        A = .0178
        B = 1.187
        C = 23.931
      Else
        A = .0362
        B = 1.0937
        C = 25.724
      End If

      RS = A * GGRAV * P ** B * DEXP(C*API/T)

      If (RS.GT.RSI*5.615) RS = RSI * 5.615

      Return
      End
**********************************************************************
***** Subroutine DENS
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Determines the density of oil, water, and gas.
***** Reference  Thesis, M.E. Vasquez, University of Tulsa, 1976. HP41C
*****            Petroleum Fluids Pack Manual.
**********************************************************************
      Subroutine DENS(T,P,GGRAV,WGRAV,API,DENSO,DENSW,DENSG)

      Implicit Double Precision (A-Z)
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI

*---- Determine PVT parameters

      Call ZFACTOR(T,P,GGRAV,Z)
      Call SGOR(T,P,GGRAV,API,RS)
      Call BPP(T,GGRAV,API,PBP)
      Call FVF(T,P,GGRAV,API,RS,PBP,CO,BO,BW)

*---- Compute the density of oil.

      OGRAV = 141.5 / (131.5+API)
      If (P.LT.PBP) Then
        DENSO = (350.4*OGRAV+.0764*GGRAV*RS) / (5.615*BO)
      Else
        DENSO = (350.4*OGRAV+.0764*GGRAV*RSI*5.615) / (5.615*BO)
        DENSO = DENSO * DEXP(CO*(P-PBP))
      End If

*---- Compute the density of water.

      DENSW = 62.4 * WGRAV / BW

*---- Compute the density of gas.

      DENSG = 2.7 * P * GGRAV / Z / T

      Return
      End
**********************************************************************
***** Subroutine FRICFAC
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Estimates the Darcy (or Moody) friction factor based on
*****            the method of Zigrang and Sylvester.
***** Reference  Explicit Approximations to the Solution of Colebrook's
*****            Friction Factor Equation, D.J. Zigrang and N.D.
*****            Sylvester, AIChe Journal (Vol 28, No 3), May, 1982.
**********************************************************************
      Subroutine FRICFAC(RHO,V,D,U,RROUGH,F)

      Implicit Double Precision (A-Z)

*---- Compute the Reynold's number.

      NRE = 1488 * RHO * V * D / U

*---- Compute the Darcy fricion factor.

      AA = RROUGH / 3.7
      BB = 5.02 / NRE
      TEMP = -2 * DLOG10(AA-BB*DLOG10(AA-BB*DLOG10(AA+13/NRE)))
      F = (1/TEMP) ** 2

      Return
      End
**********************************************************************
***** Subroutine FVF
***** Author     James Carroll
***** Date       November, 1990
***** In-calls
***** Out-calls
***** Function   Estimates the formation volume factors of oil and water
*****            using the correlation of Vazquez and Begga.
***** Reference  Thesis, M.E. Vasquez, University of Tulsa, 1976. HP41C
*****            Petroleum Fluids Pack Manual.
**********************************************************************
      Subroutine FVF(T,P,GGRAV,API,RS,PBP,CO,BO,BW)

      Implicit Double Precision (A-Z)
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI

*---- Define constants

      TF = T - 460
      If (API.GT.30) Then
        A = 1.1E-5
        B = 4.67E-4
        C = 1.337E-9
      Else
        A = 1.751E-5
        B = 4.677E-4
        C = -1.811E-8
      End If

*---- Determine Bo with respect to bubble point pressure.

      CO = (-1433+5*RSI*5.615+17.2*TF-1180*GGRAV+12.61*API) / (P*1E5)
      If (P.LE.PBP) Then
        BO = 1 + A * (TF-60) * (API/GGRAV) + (B+C*(TF-60)*(API/GGRAV)) *
```

```
*       RS
        Else
          BOBP = 1 + A * (TF-60) * (API/GGRAV) + (B+C*(TF-60)*(API/GGRAV))
*           * RSI * 5.615
          BO = BOBP * EXP(CO*(PBP-P))
        End If

*---- Estimate the FVF of gas-free water.

        A = .9947 + 5.8E-6 * TF + 1.02E-6 * TF ** 2
        B = -4.228E-6 + 1.8376E-8 * TF - 6.77E-11 * TF ** 2
        C = 1.3E-10 - 1.3855E-12 * TF + 4.285E-15 * TF ** 2
        BW = A + B * P + C * P * P

*---- Apply the salinity correction

        BW = BW * ((5.1E-8*P+(5.47E-6-1.95E-10*P)*(TF-60)+(-3.23E-8+
*       8.5E-13*P)*(TF-60)**2)*NACL+1)

        Return
        End
*************************************************************************
***** Subroutine VISC
***** Author      James Carroll
***** Date        November, 1990
***** In-calls
***** Out-calls
***** Function    This subprogram uses the Dempsey correlation to determine
*****             gas viscosity. This subroutine estimates the live oil
*****             viscosity, both above and below the bubble point pressure
*****             using the correlations of Beggs and Robinson and Vasquez
*****             and Beggs. Water viscosity is estimated from an emperical
*****             correlation.
***** Reference   Thesis, M.E. Vasquez, University of Tulsa, 1976. Estimating
*****             the Viscosity of Crude Oil Systems, H.D. Beggs and J.F.
*****             Robinson, JPT, Sept, 1985. HP41C Petroleum Fluids Pack
*****             Manual.
*************************************************************************
        Subroutine VISC(T,P,GGRAV,API,RS,PBP,UG,UO,UW)

        Implicit Double Precision (A-Z)
        Common /CONTAM/ N2, CO2, H2S, NACL, RSI

*---- Solve for critical temperature and pressure.

        PURE = 1. - H2S - CO2 - N2
        GRAVHC = (GGRAV-.9672*N2-1.5195*CO2-1.1765*H2S) / PURE

        TCHC = 187. + 330. * GRAVHC - 71.5 * GRAVHC * GRAVHC
        PCHC = 706. - 51.7 * GRAVHC - 11.1 * GRAVHC * GRAVHC

        TC = PURE * TCHC + 227.3 * N2 + 547.6 * CO2 + 672.4 * H2S
        PC = PURE * PCHC + 493. * N2 + 1071. * CO2 + 1306. * H2S

*---- Adjust Tc and Pc for gas contaminants.

        CWA = 120 * ((CO2+H2S)**0.9-(CO2+H2S)**1.6)
        CWA = CWA + 15 * (H2S**.5-H2S**4.)

        TCP = TC - CWA
        PCP = (PC*(T*(-CWA))) / (TC+H2S*(1-H2S)*CWA)

*---- Define constants

        A0 = -2.4621182
        A1 = 2.97054714
        A2 = -.286264054
        A3 = 8.05420522E-03
        A4 = 2.80860949
        A5 = -3.49803305
        A6 = .36037302
        A7 = -.010443241
        A8 = -.793385684
        A9 = 1.39643306
        A10 = -.149144925
        A11 = .00441015512
        A12 = 8.39387178E-02
        A13 = -.186408848
        A14 = .0203367881
        A15 = -.000609579263

*---- Determine pseudoreduced temperature and pressure

        TR = T / TCP
        PR = P / PCP

*---- Determine Dempsey correction factors

        UA = ((1.709E-5)-(2.062E-6)*GGRAV) * (T-460) + (.008188) - (.00615
*       ) * DLOG10(GGRAV)
        UB = N2 * ((.00848)*DLOG10(GGRAV)+.00959)
        UC = CO2 * ((.00908)*DLOG10(GGRAV)+.00624)
        UD = H2S * ((.00849)*DLOG10(GGRAV)+.00373)

        U1 = UA + UB + UC + UD

        X = A0 + A1 * PR + A2 * PR * PR + A3 * PR ** 3 + TR * (A4+A5*PR+A6
*       *PR*PR+A7*PR**3) + TR * TR * (A8+A9*PR+A10*PR*PR+A11*PR**3) +
*       TR ** 3 * (A12+A13*PR+A14*PR*PR+A15*PR**3)

        UG = U1 * DEXP(X) / TR

*---- Compute the oil viscosity
*       Estimate the dead oil viscosity.

        TF = T - 460
        C = 3.0324 - .02023 * API
        B = 10 ** C
        A = B * TF ** (-1.163)
        UOD = 10 ** A - 1.

*---- Estimate the live oil viscosity below the bubble point pressure.

        If (P.LE.PBP) Then
          B = 5.44 * (RS+150) ** (-.338)
          A = 10.715 * (RS+100) ** (-.515)
          UO = A * UOD ** B

*---- Estimate the live oil viscosity above the bubble point pressure.

        Else
          C = 2.6 * P ** 1.187 * DEXP(-8.98E-5*P-11.513)
          B = 5.44 * (RSI*5.615+150) ** (-.338)
          A = 10.715 * (RSI*5.615+100) ** (-.515)
          UOBP = A * UOD ** B
          UO = UOBP * (P/PBP) ** C
        End If
```

```
*---- Estimate the water viscosity

      UW = DEXP(1.003-1.479E-2*TF+1.982E-5*TF**2)

      Return
      End
*************************************************************************
***** Subroutine ZFACTOR
***** Author      James Carroll
***** Date        November, 1990
***** In-calls
***** Out-calls
***** Function    This subroutine calculates the z-factor using the
*****             method of Dranchuk, Purvis, and Robinson.  Critical
*****             temperature and pressure are estimated with Standing's
*****             correlation.  The Wichert-Aziz correction is made for
*****             gas contaminants.  The gas gravity is considered to be
*****             measured in-situ.
***** Reference   Comparisons Made for Computer Z-Factor Calculations,
*****             Oil & Gas Journal, Dec., 1976. HP41C Petroleum Fluid
*****             Pack Manual.
*************************************************************************
      Subroutine ZFACTOR(T,P,GGRAV,Z)


      Implicit Double Precision (A-Z)
      Common /CONTAM/ N2, CO2, H2S, NACL, RSI

*---- Determine the critical temperature and pressure.

      PURE = 1. - H2S - CO2 - N2
      GRAVHC = (GGRAV-.9672*N2-1.5195*CO2-1.1765*H2S) / PURE

      TCHC = 187. + 330. * GRAVHC - 71.5 * GRAVHC * GRAVHC
      PCHC = 706. - 51.7 * GRAVHC - 11.1 * GRAVHC * GRAVHC

      TC = PURE * TCHC + 227.3 * N2 + 547.6 * CO2 + 672.4 * H2S
      PC = PURE * PCHC + 493. * N2 + 1071. * CO2 + 1306. * H2S

*---- Adjust Tc and Pc for gas contaminants.

      CWA = 120 * ((CO2+H2S)**0.9-(CO2+H2S)**1.6)
      CWA = CWA + 15 * (H2S**.5-H2S**4.)

      TCP = TC - CWA
      PCP = (PC*(T*(-CWA))) / (TC+H2S*(1-H2S)*CWA)

*---- Compute reduced temperature and pressure.

      TR = T / TCP
      PR = P / PCP

*---- Determine constants for iteration.

      A = .06423
      B = .5353 * TR - .6123
      C = .3151 * TR - 1.0467 - .5783 / TR / TR
      D = TR
      E = .6816 / TR / TR
      F = .6845
      G = .27 * PR

*---- Perform Newton-Raphson iteration to obtain pseudoreduced density.

      RN = .27 * PR / TR
      R = 2 * RN

      Do 10, I = 1, 50
        If (ABS(RN-R).LT..001) Go To 20
        R = RN
        FR = A * R ** 6 + B * R ** 3 + C * R * R + D * R + E * R ** 3 *
     *       (1+F*R*R) * DEXP(-F*R*R) - G
        FRP = 6 * A * R ** 5 + 3 * B * R * R + 2 * C * R + D
        FRP = FRP + E * R * R * (3+F*R*R*(3-2*F*R*R)) * DEXP(-F*R*R)
        RN = R - FR / FRP
   10 Continue

   20 Z = .27 * PR / RN / TR

      Return
      End

*************************************************************************
***** Subroutine SEPARAT
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    Models the flash separation of oil and gas with a two-
*****             stage separation process. This subroutine serves mainly
*****             to link the separator model to the rest of the program.
*************************************************************************
      Subroutine SEPARAT(I)

      Implicit Double Precision (A-Z)
      Integer N1, N2, IVNWEL, IWCTR, K, NWELL, CORR, NWELLS, IERR, IEXE,
     *    IHC, IPRT, IUNIN, NWT, NSTEP, I
      Parameter (N1 = 100,N2 = 500)
      Dimension VGASP(0:N2), VGOR(0:N2), VOILP(0:N2), VPRSR(0:N2),
     *    VPRSWF(0:N2), VPRSWH(0:N2), VRTEG(0:N2), VRTEO(0:N2),
     *    VTIME(0:N2), IVNWEL(0:N2), TPWMIN(N1), TRTEFM(N1),
     *    TRTEFT(N1), TSKN(N1), TTIM(N1), NWELLS(N1)
      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     *    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /SEP/ TSEP, PSEP
      Common /TRANS4/ VGASP, VGOR, VOILP, VPRSR, VPRSWF, VPRSWH, VRTEG,
     *    VRTEO, VTIME, IWCTR, NWELL, OILTI, GASTI, IVNWEL, K
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     *    CORR

*---- Input is in Metric Units. Make necessary conversions to English
*     units.

      TSEP = TSEP * 1.8 + 460
      TWH = TWH * 1.8 + 460
      PSEP = PSEP * .14503774
      GRVW = 1.
      NSTEP = IDINT(DNINT(XMXTIM/DELTIM))
      Call DENS(TSEP,PSEP,GRVG,GRVW,API,DENSO,DENSW,DENSG)

c      Do 10 I = 1, NSTEP
        PWH = VPRSWH(I) * .14503774
        Call DENS(TWH,PWH,GRVG,GRVW,API,DENSO,DENSW,DENSG)
        QO = VRTEO(I) * 6.289811
        QG = VRTEG(I) * 35.314667

*---- If Oil and Gas rate is zero then dont flash!
```

```
        If((QO.LE.0.0).AND.(QG.LE.0.0)) then
        Goto 99
        Else
        Endif

        Call FLASHCAL(TSEP,PSEP,DENSO,DENSG,QO,QG)
99      VRTEO(I) = QO / 6.289811
        VRTEG(I) = QG / 35.314667
C   10 Continue

      TSEP = (TSEP-460) / 1.8
      TWH = (TWH-460) / 1.8
      PSEP = PSEP / .14503774

      Return
      End
*************************************************************************
***** Subroutine FLASHCAL
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    Controls the flash separation process.
***** Reference   Notes from PE 251, Fall 1988, Dr. Orr, Stanford. The
*****             Properties of Petroleum Fluids, McCain, 1973. Surface
*****             Production Operations, Arnold and Stewart, 1986.
*****             Classical Thermodynamics of Nonelectrolyte Solutions,
*****             Van Ness and Abbot, 1982.
*************************************************************************
*---- Variables:
*
*   aa      (a alpha)ij
*   ai      ai
*   a1      coefficient of z^2
*   a2      coefficient of z
*   a3      constant coefficient
*   bi      bi
*   bip     binary interaction parameter
*   comp    dummy composition
*   tol     convergence criteria
*   f       fugacity
*   flag    flag for convergence
*   fx      liquid fugacity
*   fy      vapor fugacity
*   k       k values (y/x)
*   name    name of component
*   nc      number of components
*   nx      liquid mole fraction
*   ny      vapor mole fraction
*   p       pressure in psia
*   pc      critical pressure in psia
*   r       universal gas constant
*   sa      sum of ai
*   sb      sum of bi
*   sumaa   sum of aa
*   t       temperature in fahrenheit
*   tc      critical temperature in rankine
*   vx      liquid molar volume
*   vy      vapor molar volume
*   w       pitzer's accentric factor
*   x       liquid composition
*   y       vapor composition
*   z       total mixture compostion
```

```
      Subroutine FLASHCAL(TSEP,PSEP,DENSO,DENSG,QO,QG)

      Implicit Double Precision (A-Z)
      Integer NC, FLAG, I
      Dimension Y1(9), X(9), Y(9), Z(9), AA(9,9), BI(9), COMP(9), F(9),
     *    FX(9), FY(9), K(9)
      Common /TP/ T, P, NC
      Common /XY/ X, Y, Z, NX, NY, ZX, ZY
      Common /MISC1/ AA, BI, COMP, F, SA, SB, V, VX, VY, R
      Common /FUG/ FX, FY, K, TOL, FLAG
      Common /Molar/ MLL, MGG
*---- Set initial conditions.

      PSC = 14.696
      TSC = 520.
      MG = 25.
      MO = 90.
      R = 10.732
      TOL = 1D-2

*---- Determine molar flow rate

      NT = QG * DENSG / MG + QO * DENSO * 5.615 / MO

        MLL = QO * DENSO * 5.615
        MGG = QG * DENSG

      Call INIT

*---- Perform flash for first-stage separation

      P = PSEP
      T = TSEP

      Call EQUILIB

      NY1 = NY
      NX1 = NX

      Do 10 I = 1, NC
        Z(I) = X(I)
        Y1(I) = Y(I)
   10 Continue

*---- Perform flash for second-stage separation of oil stream

      P = PSC
      T = TSC

      Call EQUILIB

      NY2 = NY
      NX2 = NX
      VY2 = ZY
      VX2 = ZX

*---- Perform flash for second-stage separation of gas stream

      P = PSC
      T = TSC

      Do 20 I = 1, NC
        Z(I) = Y1(I)
   20 Continue
```

```
      Call EQUILIB

      NY3 = NY
      NX3 = NX
      VY3 = ZY
      VX3 = ZX

*---- Determine volumetric flow rates

      NG = NY1 * NY3 * VY3 + NX1 * NY2 * VY2
      NO = NY1 * NX3 * VX3 + NX1 * NX2 * VX2
      QG = NT * NG / DENSG
      QO = NT * NO / 5.615 / DENSO

C       Write(6,*)'NG',NG,'NO',NO,'QG',QG,'QO',QO,'NT',NT

      Return
      End
*************************************************************************
***** Subroutine EQUILIB
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    Controls the equilibrium determination process based
*****            on flash calculations.
***** Reference   Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****            Classical Thermodynamics of Nonelectrolyte Solutions,
*****            Van Ness and Abbot, 1982.
*************************************************************************
      Subroutine EQUILIB
      Implicit Double Precision (A-Z)
      Integer NC, FLAG, I
      Dimension X(9), Y(9), Z(9), AA(9,9), BI(9), COMP(9), F(9), FX(9),
     *    FY(9), K(9)
      Common /TP/ T, P, NC
      Common /XY/ X, Y, Z, NX, NY, ZX, ZY
      Common /MISC1/ AA, BI, COMP, F, SA, SB, V, VX, VY, R
      Common /FUG/ FX, FY, K, TOL, FLAG

*---- Flag:  1 = normal operation
*            2 = all vapor
*            3 = all liquid
*            4 = flash didn't converge in 500 iterations
*            5 = new k's are within tolerance

      Call INITBIP

      GUESS = 0.05D+00
   10 NX = GUESS
      FLAG = 1
      COUNT = 0.

      Call WILSON

   20 If (FLAG.EQ.1) Then
        Call FLASH
        COUNT = COUNT + 1
        SUMBP = 0
        SUMDP = 0
        Do 30 I = 1, NC
          SUMBP = SUMBP + Z(I) * K(I)
          SUMDP = SUMDP + Z(I) / K(I)
   30   Continue
        If ((SUMDP.LE.1.).AND.(COUNT.GT.2.)) FLAG = 2
        If ((SUMBP.LE.1.).AND.(COUNT.GT.2.)) FLAG = 3
        If ((FLAG.EQ.3).AND.(GUESS.LT.1.)) Then
          FLAG = 4
        Else If ((FLAG.EQ.3).AND.(GUESS.GE.1.)) Then
          Do 40 I = 1, NC
            X(I) = Z(I)
            Y(I) = 0.
   40     Continue
          ZX = VX
          ZY = 0.
          NX = 1.
          NY = 0.
        Else If ((FLAG.EQ.2).AND.(GUESS.LT.1.)) Then
          FLAG = 4
        Else If ((FLAG.EQ.2).AND.(GUESS.GE.1.)) Then
          Do 50 I = 1, NC
            Y(I) = Z(I)
            X(I) = 0.
   50     Continue
          ZY = VY
          ZX = 0.
          NY = 1.
          NX = 0.
        Else
          Do 60 I = 1, NC
            If (Y(I).LE.0) Y(I) = 0.00001
            COMP(I) = Y(I)
   60     Continue
          Call MIXRULE
          Call CUBIC
          V = VY
          ZY = VY
          Call SRKFUG
          Do 70 I = 1, NC
            FY(I) = F(I)
   70     Continue
          Do 80 I = 1, NC
            If (X(I).LE.0) X(I) = 0.00001
            COMP(I) = X(I)
   80     Continue
          Call MIXRULE
          Call CUBIC
          V = VX
          ZX = VX
          Call SRKFUG
          Do 90 I = 1, NC
            FX(I) = F(I)
   90     Continue
          Call NEWK
        End If

        Go To 20
      End If

      If (FLAG.EQ.4.AND.GUESS.LT.1.) Then
        GUESS = GUESS + 0.05
        Go To 10
      Else If (FLAG.EQ.4.AND.GUESS.GE.1.) Then
        Write (6,5000)
        Return
      End If
```

```
      NY = 1 - NX

      Return
 5000 Format ('Failed to converge')
      End
*****************************************************************
***** Subroutine INIT
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    Initializes flash separation data.
*****************************************************************
      Subroutine INIT
      Implicit Double Precision (A-Z)
      Integer NC, I
      Character*16 NAME(9)
      Dimension BIP(9,9), GAMMA(9), TC(9), PC(9), W(9), X(9), Y(9),
     *    Z(9)
      Common /TP/ T, P, NC
      Common /TCPC/ BIP, GAMMA, TC, PC, W
      Common /CHAR/ NAME
      Common /XY/ X, Y, Z, NX, NY, ZX, ZY

      NC = 9

      Z(1) = 0.8013
      Z(2) = 0.0537
      Z(3) = 0.0409
      Z(4) = 0.0185
      Z(5) = 0.0100
      Z(6) = 0.0137
      Z(7) = 0.0085
      Z(8) = 0.0153
      Z(9) = 0.0334

      NAME(1) = 'METHANE'
      NAME(2) = 'ETHANE'
      NAME(3) = 'PROPANE'
      NAME(4) = 'n-BUTANE'
      NAME(5) = 'i-BUTANE'
      NAME(6) = 'n-PENTANE'
      NAME(7) = 'i-PENTANE'
      NAME(8) = 'HEXANE'
      NAME(9) = 'HEPTANE+'

      PC(1) = 667.80
      PC(2) = 707.80
      PC(3) = 616.30
      PC(4) = 550.70
      PC(5) = 529.10
      PC(6) = 488.60
      PC(7) = 490.40
      PC(8) = 436.90
      PC(9) = 421.90

      TC(1) = -116.63 + 460.
      TC(2) = 90.09 + 460.
      TC(3) = 206.01 + 460.
      TC(4) = 305.65 + 460.
      TC(5) = 274.98 + 460.
      TC(6) = 385.70 + 460.
      TC(7) = 369.10 + 460.
      TC(8) = 453.70 + 460.
      TC(9) = 616.73 + 460.

      W(1) = 0.0104
      W(2) = 0.0986
      W(3) = 0.1524
      W(4) = 0.2010
      W(5) = 0.1848
      W(6) = 0.2539
      W(7) = 0.2223
      W(8) = 0.3007
      W(9) = 0.3590

      GAMMA(1) = 0.3000
      GAMMA(2) = 0.3564
      GAMMA(3) = 0.5077
      GAMMA(4) = 0.5844
      GAMMA(5) = 0.5631
      GAMMA(6) = 0.6310
      GAMMA(7) = 0.6247
      GAMMA(8) = 0.6640
      GAMMA(9) = 0.7653

      Return
      End
*****************************************************************
***** Subroutine SRKFUG
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    This subroutine computes fugacity of components using
*****             the Soave Redlich Kwong equation of state.
***** Reference   Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****             Classical Thermodynamics of Nonelectrolyte Solutions,
*****             Van Ness and Abbot, 1982.
*****************************************************************
      Subroutine SRKFUG
      Implicit Double Precision (A-Z)
      Integer NC, I, J
      Dimension AA(9,9), BI(9), COMP(9), F(9)
      Common /TP/ T, P, NC
      Common /MISC1/ AA, BI, COMP, F, SA, SB, V, VX, VY, R

      Do 20 I = 1, NC

        SUMAA = 0

        Do 10 J = 1, NC
          SUMAA = SUMAA + COMP(J) * AA(I,J)
   10   Continue

        DUM1 = DLOG(V/(V-SB)) + BI(I) / (V-SB)
        DUM2 = (2.0*SUMAA/(R*T*SB)) * DLOG(V/(V+SB))
        DUM3 = ((SA*BI(I))/(R*T*SB*SB)) * (DLOG((V+SB)/V)-SB/(V+SB))
        DUM4 = DLOG((P*V)/(R*T*COMP(I)))

        F(I) = P * DEXP(DUM1+DUM2+DUM3-DUM4)

   20 Continue

      Return
      End
*****************************************************************
***** Subroutine INITBIP
```

```
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    This subroutine initializes the array of binary
*****             interaction parameters.  For SRK eos, hydrocarbon to
*****             hydrocarbon bip's are defined to be zero.
***** Reference   Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****             Classical Thermodynamics of Nonelectrolyte Solutions,
*****             Van Ness and Abbot, 1982.
************************************************************************

      Subroutine INITBIP
      Implicit Double Precision (A-Z)
      Integer NC, I, J
      Dimension BIP(9,9), GAMMA(9), TC(9), PC(9), W(9)
      Common /TP/ T, P, NC
      Common /TCPC/ BIP, GAMMA, TC, PC, W

      Do 20 I = 1, NC
        Do 10 J = 1, NC
          BIP(I,J) = 0
   10   Continue
   20 Continue

      Do 30 I = 2, NC
        BIP(1,I) = 0.1285713 * GAMMA(I) - 0.05857134
        BIP(I,1) = 0.1285713 * GAMMA(I) - 0.05857134
   30 Continue

      Return
      End
************************************************************************
***** Subroutine WILSON
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    Wilson equation is used to approximate the initial K
*****             values.
***** Reference   Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****             Classical Thermodynamics of Nonelectrolyte Solutions,
*****             Van Ness and Abbot, 1982.
************************************************************************

      Subroutine WILSON
      Implicit Double Precision (A-Z)
      Integer NC, FLAG, I
      Dimension FX(9), FY(9), K(9), BIP(9,9), GAMMA(9), TC(9), PC(9),
     *    W(9)
      Common /TP/ T, P, NC
      Common /FUG/ FX, FY, K, TOL, FLAG
      Common /TCPC/ BIP, GAMMA, TC, PC, W

      Do 10 I = 1, NC
        TR = T / TC(I)
        PR = P / PC(I)
        K(I) = DEXP(5.37*(1+W(I))*(1-1/TR)) / PR
   10 Continue

      Return
      End
************************************************************************
***** Subroutine FLASH
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function    Given the overall composition (z) and the k values(k),
*****             it will find the liquid (x) and vapor (y) mole fractions.
***** Reference   Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****             Classical Thermodynamics of Nonelectrolyte Solutions,
*****             Van Ness and Abbot, 1982.
************************************************************************

      Subroutine FLASH
      Implicit Double Precision (A-Z)
      Integer NC, FLAG, M, J, I, COUNT
      Dimension X(9), Y(9), Z(9), FX(9), FY(9), K(9)
      Common /TP/ T, P, NC
      Common /XY/ X, Y, Z, NX, NY, ZX, ZY
      Common /FUG/ FX, FY, K, TOL, FLAG

      COUNT = 0

      Do 30 J = 1, 500
        FL = 0
        DFDL = 0
        COUNT = COUNT + 1

        Do 10 I = 1, NC
          FL = FL + (Z(I)*(1-K(I))) / (K(I)+(1-K(I))*NX)
          DFDL = DFDL - (Z(I)*(1-K(I))**2) / ((K(I)+(1-K(I))*NX)**2)
   10   Continue

        NEWNX = NX - FL / DFDL

        If ((NEWNX.LE.0.).AND.(COUNT.GT.5)) Then
          NX = 0.
          FLAG = 2
          Return
        Else If ((NEWNX.GE.1.).AND.(COUNT.GT.5)) Then
          NX = 1.
          FLAG = 3
          Return
        End If

        If ((ABS(FL).LT.TOL).AND.(ABS(NEWNX-NX).LT.TOL)) Then
          NX = NEWNX
          Do 20 M = 1, NC
            X(M) = Z(M) / (K(M)+NX*(1-K(M)))
            Y(M) = K(M) * X(M)
   20     Continue
          Return
        End If

        NX = NEWNX

   30 Continue

      FLAG = 4

      Return
      End
************************************************************************
***** Subroutine MIXRULE
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
```

```
***** Out-calls
***** Function   Computes the mixing rules for the SRK eos.
***** Reference  Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****           Classical Thermodynamics of Nonelectrolyte Solutions,
*****           Van Ness and Abbot, 1982.
*************************************************************************

      Subroutine MIXRULE
      Implicit Double Precision (A-Z)
      Integer NC, I, J
      Dimension AA(9,9), BI(9), COMP(9), F(9), BIP(9,9), GAMMA(9),
     *    TC(9), PC(9), W(9), AI(9)
      Common /TP/ T, P, NC
      Common /MISC1/ AA, BI, COMP, F, SA, SB, V, VX, VY, R
      Common /TCPC/ BIP, GAMMA, TC, PC, W

      Do 10 J = 1, NC
        BI(J) = (0.08664*R*TC(J)) / PC(J)
        DUMMY = 0.48508 + 1.55171 * W(J) - 0.15613 * W(J) * W(J)
        ALPHA = (1.0+DUMMY*(1.0-DSQRT(T/TC(J)))) ** 2.0
        AI(J) = (0.42748*R*R*(TC(J)*TC(J))) / PC(J)
        AI(J) = AI(J) * ALPHA
   10 Continue

      SA = 0
      SB = 0

      Do 30 I = 1, NC
        SB = SB + COMP(I) * BI(I)
        Do 20 J = 1, NC
          AA(I,J) = (1.0-BIP(I,J)) * DSQRT(AI(I)*AI(J))
          SA = SA + COMP(I) * COMP(J) * AA(I,J)
   20   Continue
   30 Continue

      Return
      End
*************************************************************************
***** Subroutine CUBIC
***** Author     James Carroll
***** Date       December, 1990
***** In-calls
***** Out-calls
***** Function   Solves for the roots of a cubic equation.
*****              Z^3  +  A2 Z^2  +  A1 Z  +  A0  =  0
*****
*****              a2  coefficient of z^2
*****              a1  coefficient of z
*****              a0  constant coefficient
*****              r1  root 1
*****              r2  root 2
*****              r3  root 3
*****              nr  number of roots
*****              vx  liquid root
*****              vy  vapor root
*************************************************************************
      Subroutine CUBIC
      Implicit Double Precision (A-Z)
      Integer NC
      Dimension AA(9,9), BI(9), COMP(9), F(9)
      Common /TP/ T, P, NC
      Common /MISC1/ AA, BI, COMP, F, SA, SB, V, VX, VY, R

*     Compute coefficients

      A2 = -(R*T/P)
      A1 = (SA-SB*R*T-P*SB*SB) / P
      A0 = -(SA*SB) / P

      PI = 4 * DATAN(1.0D+00)
      B1 = (3*A1-A2**2) / 3
      B0 = (2*A2**3-9*A2*A1+27*A0) / 27.0
      D2 = (B1/3) ** 3 + (B0/2) ** 2

      If (D2.LT.0.0) Then

        If (B1.GT.0.0) Then
          PHI = DACOS(-(B0/2)*(-(ABS(-3/B1))**(3./2)))
        Else
          PHI = DACOS(-(B0/2)*((-3/B1)**(3./2)))
        End If

        DUM1 = -B1 / 3

        If (DUM1.LT.0.0) Then
          DUM1 = -((ABS(DUM1))**(1./2))
        Else
          DUM1 = DUM1 ** (1.0/2)
        End If

        R1 = (2*DUM1*DCOS(PHI/3)) - A2 / 3
        R2 = (2*DUM1*DCOS(PHI/3+2*PI/3)) - A2 / 3
        R3 = (2*DUM1*DCOS(PHI/3+4*PI/3)) - A2 / 3
        NR = 3

      Else If (D2.GT.0.0) Then
        DUM1 = DSQRT(D2) - B0 / 2
        DUM2 = -DSQRT(D2) - B0 / 2
        If (DUM1.LT.0.0) Then
          DUM1 = -(ABS(DUM1)) ** (1./3)
        Else
          DUM1 = DUM1 ** (1./3)
        End If

        If (DUM2.LT.0.0) Then
          DUM2 = -(ABS(DUM2)) ** (1./3)
        Else
          DUM2 = DUM2 ** (1./3)
        End If

        R1 = DUM1 + DUM2 - A2 / 3
        R2 = 0
        R3 = 0
        NR = 1

      Else
        DUM3 = -B0 / 2
        If (DUM3.LT.0.0) Then
          DUM3 = -(ABS(DUM3)) ** (1./3)
        Else
          DUM3 = DUM3 ** (1./3)
        End If

        R1 = 2 * DUM3 - A2 / 3
        R2 = -DUM3 - A2 / 3
        R3 = R1
        NR = 3
```

```
      End If

*---- Determine which is the correct root

      If (NR.EQ.3) Then
        VX = DMIN1(R1,R2,R3)
        VY = DMAX1(R1,R2,R3)
      Else
        VX = R1
        VY = R1
      End If

      Return
      End
*****************************************************************
***** Subroutine NEWK
***** Author      James Carroll
***** Date        December, 1990
***** In-calls
***** Out-calls
***** Function  A successive substitution scheme is used to compute new
*****           K values for a flash calculation.
***** Reference  Notes from PE 251, Fall 1988, Dr. Orr, Stanford.
*****           Classical Thermodynamics of Nonelectrolyte Solutions,
*****           Van Ness and Abbot, 1982.
*****************************************************************
      Subroutine NEWK
      Implicit Double Precision (A-Z)
      Integer NC, FLAG, I
      Dimension FX(9), FY(9), K(9)
      Common /TP/ T, P, NC
      Common /FUG/ FX, FY, K, TOL, FLAG

      Do 10 I = 1, NC
        FRATIO = FX(I) / FY(I)
        K(I) = FRATIO * K(I)
        TEST = ABS(FRATIO-1.)
        If (TEST.LE.TOL) FLAG = 5
   10 Continue

      Return
      End
*****************************************************************
      Subroutine Getcpu(Cpusec)
      real*4 tarray(2),etime
      real*8 Cpusec
      cpusec = etime(tarray)
      return
      end
*****************************************************************

      Subroutine PLOTOUT(Cpu)

*---- VARIABLES AND CONSTANTS

      Implicit Double Precision (A-Z)

      Integer N1, N2, IVNWEL, NWELLS, I, IERR, IEXE, IHC, IPRT, IUNIN,
     •    IWCTR, J  K, NWELL, NWT, CORR, NPVT, NRP, IC, IT, NSTEP
      Character ZJOBID*60, ZC(8)*6
      Logical QWCTR, QSTOP, ICRIT
      Parameter (N1 = 100,N2 = 500)

      Dimension TPWMIN(N1), TRTEFM(N1), TRTEFT(N1), TSKN(N1), TTIM(N1),
     •    VGASP(0:N2), VGOR(0:N2), VOILP(0:N2), VPRSR(0:N2),
     •    VPRSWF(0:N2), VPRSWH(0:N2), VRTEG(0:N2), VRTEO(0:N2), AN(8),
     •    VTIME(0:N2), TDENRG(100), TDENRO(100), TFVFG(100),
     •    TFVFGX(100), TFVFO(100), TGORS(100), TOGRS(100), TPRMRG(100),
     •    TPRMRO(100), PRMLGO(100), TPRS(100), TSATG(100), TVISG(100),
     •    TVISO(100), IVNWEL(0:N2), NWELLS(N1), IC(8), IT(8),
     •    VRTEO1(0:N2), VRTEG1(0:N2), NPV1(0:N2)

*---- COMMON BLOCKS

      Common /CONTAM/ NITRO, CO2, H2S, NACL, RSI
      Common /TBG/ API, GRVG, AROUGH, DIAM, LENGTH, TWF, TWH, SIGMAO,
     •    CORR
      Common /PROP/ TDENRG, TDENRO, TFVFG, TFVFGX, TFVFO, TGORS, TOGRS,
     •    TPRMRG, TPRMRO, PRMLGO, TPRS, TSATG, TVISG, TVISO, NPVT, NRP
      Common /MBAL1/ AG1, AO1, CMPF, DTIM, PORI, PRSI, RG1, RO1, SATWI,
     •    VOLBW
      Common /MBAL2/ AO2, AG2, RO2, RG2, RGAV
      Common /MBAL3/ DENRG, DENRO, FVFG, FVFO, GORS, OGRS, POR, PRMGO,
     •    SATG1, SATO1, VISG, VISO, XMBAL1
      Common /IPR1/ DPINT, DSKN, PRM, RADEQ, SKN, THK
      Common /ICOUNT/ IC, IT
      Common /FINANCE/ INFRATE, DISRATE, OILPRIC, GASPRIC
      Common /TRANS1/ TTIM, NWELLS, TRTEFM, TRTEFT, TPWMIN, TSKN, RADW,
     •    DELTIM, XMXTIM, HCPV, IHC, IUNIN, IPRT, IEXE, NWT, IERR, NSTEP
      Common /TRANS2/ C1, C2, C3, C4, C5, C6, C7, C8, C9
      Common /TRANS3/ ZC, ZJOBID
      Common /TRANS4/ VGASP, VGOR, VOILP, VPRSR, VPRSWF, VPRSWH, VRTEG,
     •    VRTEO, VTIME, IWCTR, NWELL, OILTI, GASTI, IVNWEL, K
      Common /RESULTS/ NPV, NPV1
      Common /FLOW/ ICRIT
      Common /SEP/ TSEP, PSEP
      Common /BHRATE/ VRTEO1, VRTEG1

      Open (Unit=9,File='Oil')
      Open (Unit=10,File='Gas')
      Open (Unit=11,File='ResP')
      Open (Unit=12,File='Bhp')
      Open (Unit=13,File='Ftp')
      Open (Unit=14,File='OilPS')
      Open (Unit=15,File='GasPS')
      Open (Unit=16,File='Cost')

        Write(9,*)NSTEP

        Do I=1,NSTEP
         Write(9,*)I, VRTEO1(I)*6.2898
        End do

        Write(14,*)NSTEP

        Do I=1,NSTEP
         Write(14,*)I,VRTEO(I)*6.2898
        End do

        Write(15,*)NSTEP

        Do I=1,NSTEP
         Write(15,*)I,VRTEG(I)*35.14
        End do

        Write(10,*)NSTEP

        Do I=1,NSTEP
```

```
      Write(10,*)I,VRTEG1(I)*35.14
     End do

     Write(11,*)NSTEP

     Do I=1,NSTEP
      Write(11,*)I,VPRSR(I)*(14.74/101.325)
     End do

     Write(12,*)NSTEP

     Do I=1,NSTEP
      Write(12,*)I,VPRSWF(I)*(14.74/101.325)
     End do

     Write(13,*)NSTEP

     Do I=1,NSTEP
      Write(13,*)I,VPRSWH(I)*(14.74/101.325)
     End do

     Write(16,*)NSTEP

     Do I=1,NSTEP
      Write(16,*)I,NPV1(I)
     End do

   Close (Unit=9)
   Close (Unit=10)
   Close (Unit=11)
   Close (Unit=12)
   Close (Unit=13)
   Close (Unit=14)
   Close (Unit=15)
   Close (Unit=16)

     Return
     End
```

END

DATE FILMED

6/1/93