Three-dimensional, three-phase black oil reservoir simulation using the IMPES method

Author:
Myint, Aung

Publication Date:
1988

DOI:
https://doi.org/10.26190/unsworks/5511

SR P T01
Form 1
WAIVER

THE UNIVERSITY OF NEW SOUTH WALES

DECLARATION RELATING TO DISPOSITION OF PROJECT REPORT/THESIS

This is to certify that I AUNG MYINT being a candidate for the degree of MASTER OF ENGINEERING am fully aware of the policy of the University relating to the retention and use of higher degree project reports and theses, namely that the University retains the copies submitted for examination and is free to allow them to be consulted or borrowed. Subject to the provisions of the Copyright Act, 1968, the University may issue a project report or thesis in whole or in part, in photostat or microfilm or other copying medium.

In the light of these provisions I grant the University Librarian permission to publish, or to authorize the publication of my project report/thesis, in whole or in part.

I also authorize the publication by University Microfilms of a 350 word abstract in *Dissertation Abstracts International* (applicable to doctorates only).

Signature.....

Witness.....

Date..... 23/12/88

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of a university or other institute of higher learning, except where due acknowledgement is made in the text of the thesis.

THREE-DIMENSIONAL, THREE-PHASE BLACK OIL RESERVOIR SIMULATION USING THE IMPES METHOD

by

Aung Myint, B.E. (Petroleum)

Supervisor: W.V. Pinczewski

A dissertation submitted to the Centre for Petroleum Engineering in partial fulfilment of the requirement for the Degree of Master of Engineering.

University of New South Wales

December, 1988.

DEDICATED TO MY PARENTS AND

MY WIFE, SAN SAN WAI

ACKNOWLEDGEMENTS

My thanks to Mark Stevenson, Ha Do, Malcom Somers for their friendship and assistance over the period of my research.

My thanks to Dr. H.A. Salisch for his encouragement and enthusiasm in helping to expand my research.

My thanks to Dr.J.R. Fanchi, Keplinger and Associates, Inc., Tulsa, Oklahoma, who sent me a copy of BOAST simulator program while completing my studies.

My sincere thanks to Professor Val Pinczewski for his patience and guidance, and his support of my endeavours.

My special thanks to Lynne for typing my thesis.

ABSTRACT

Reservoir simulation computer programs are an integral part of the modern petroleum engineering effort. Mathematical models are developed to describe the three-dimensional, multiphase flow in a black-oil reservoir, and converted to finite difference form. Then the subsequent linear algebraic systems are solved, and the results are evaluated. However, as anyone who has ever tried to develop a simulation program will testify, there is a big gulf between the basic theory and a working computer code. This paper has been devoted primarily to the programming aspects of the finite difference technique. Many of the program subroutines are suitable for use in other programs.

GOWSIM, GWSIM and WSIM are designed to be easy-to-use programs which are suited to simulation of primary depletion, pressure maintenance by water and/or gas injection, waterflooding, gas reservoir with active water drive, and single phase aquifer reservoir model. These simulators contain only an IMPES formulation with direct elimination (BAND) and line successive overrelaxation with additive corrections method (CLSOR). These simulators can function as an inexpensive tool for performing a variety of reservoir simulation problems.

LIST OF TABLES

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER 1: DERIVATION OF GENERAL FLOW EQUATIONS

Introduction

A numerical reservoir simulator is a computer program capable of solving the difference equations for flow in porous media. The difference equations are obtained by substituting Darcy's Law into the continuity equation for each phase. Darcy's Law is essentially the rate equation that describes the relationship between the flow rate and pressure gradient for single-phase flow in porous media; however, it has been extended to describe multiphase systems in reservoir engineering applications.

1.1 Darcy's Law

In 1856, Henry Darcy (1) proposed the concept of permeability that defines the ability of a rock to transmit fluid. Darcy's law states:

> "The rate of flow of a homogeneous fluid through a porous media is proportional to the pressure gradient and to the cross-sectional area normal to the direction of flow and inversely proportional to the viscosity of the fluid."

The differential form of the relationship is:

$$\vec{v}_p = -\frac{\overleftrightarrow{K}}{\mu_p}(\nabla p_p - \rho_p \nabla z) \qquad (1\text{-}1)$$

where

$\vec{v}_p$ - superficial phase velocity vector

$\overleftrightarrow{K}$ - absolute permeability tensor of the porous media

μ_p - phase viscosity

ρ_p - phase density

∇P_p - phase pressure gradient

z - elevation

The permeability tensor $(\overleftrightarrow{K})$ used in Eq.(1.1) must be determined experimentally. For most reservoir engineering evaluations, it is normally assumed that the reservoir co-ordinates are parallel to the principal axes of the permeability tensor, thus making permeability a diagonal tensor.

$$\overleftrightarrow{K} = \begin{bmatrix} K_x & & \\ & K_y & \\ & & K_z \end{bmatrix}$$

If $K_x = K_y = K_z$, the medium is said to be isotropic, otherwise it is anisotropic. A dimensional analysis of Eq.(1.1), yields

$$\frac{L}{T} = \frac{\overleftrightarrow{K}}{M/(LT)} \cdot \frac{M}{L^2T^2} = \frac{\overleftrightarrow{K}}{LT}$$

$$\overleftrightarrow{K} = L^2 \qquad (1.2)$$

Eq.(1.2) shows that the unit of permeability is $(\text{length})^2$.

1.2 Mathematical Model for Three-Dimensional, Three-Phase Flow

To construct a mathematical model for reservoir simulation, we have to formulate first the mathematical equations which govern flow of fluid within porous media, using the physical laws that apply. These governing equations form the basic building blocks of a numerical simulator and are developed by combining the following:

1. Conservation of mass.
2. Rate equation (Darcy's Law).
3. Equation of state.

Conservation of Mass

Consider the simultaneous flow of three immiscible phases (gas, oil and water) each of which contains N chemical components in a single reservoir element (Fig. 1.1). Three phases are flowing into and out of the element in the x, y and z directions.

Let C_{ip} be the mass fractions of the ith component in the gas, oil and water phases, where 'i' implies component and 'p' implies phase. Applying the Law of Conservation of Mass to the ith component, we have during an interval Δt:

Mass in	Mass out	Mass accumulation
of Component 'i'	− of component 'i'	= of component 'i' in reservoir element ...(1.3)

Mass In

$$\left(C_{ig}\rho_g v_{gx} + C_{io}\rho_o v_{ox} + C_{i\omega}\rho_\omega v_{\omega x}\right)_x \Delta y \Delta z \Delta t +$$

$$\left(C_{ig}\rho_g v_{gy} + C_{io}\rho_o v_{oy} + C_{i\omega}\rho_\omega v_{\omega y}\right)_y \Delta x \Delta z \Delta t +$$

$$\left(C_{ig}\rho_g v_{gz} + C_{io}\rho_o v_{oz} + C_{i\omega}\rho_\omega v_{\omega z}\right)_z \Delta x \Delta y \Delta t \quad . \quad . \quad (1\text{-}4)$$

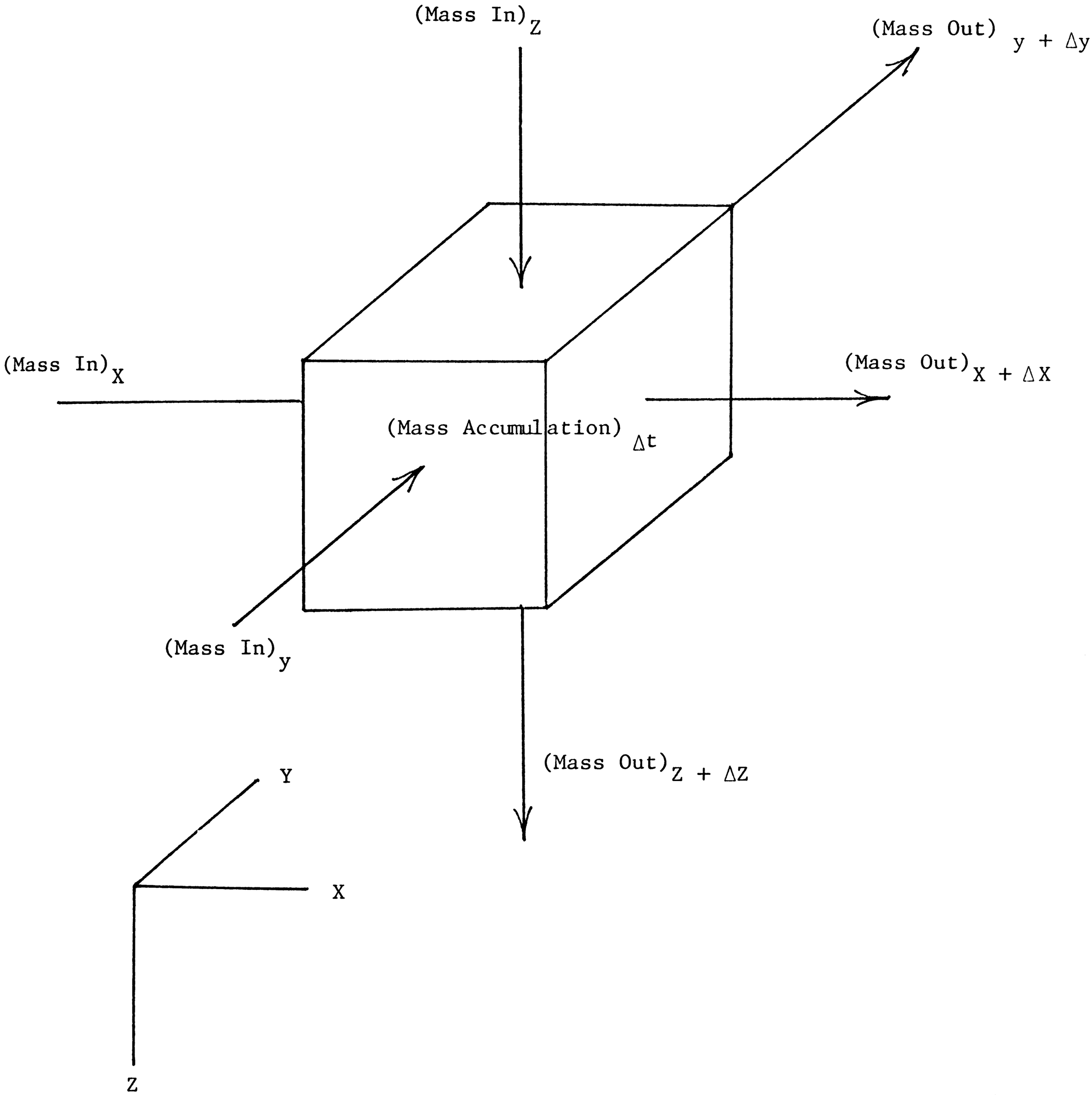

(Mass In)Z
(Mass Out) y + Δy
(Mass In)X
(Mass Out)X + ΔX
(Mass Accumulation) Δt
(Mass In)y
(Mass Out)Z + ΔZ
Y
X
Z

FIGURE (1.1)

where

C_{ig} - mass fraction of component 'i' in gas phase

C_{io} - mass fraction of component 'i' in oil phase

C_{iw} - mass fraction of component 'i' in water phase

ρ_g - gas phase density

ρ_o - oil phase density

ρ_w - water phase density

Δx - length of reservoir element

Δy - width of reservoir element

Δz - thickness of reservoir element

Δt - incremental time interval

v_{gx} - x-direction component of gas phase velocity

v_{gy} - y-direction component of gas phase velocity

v_{gz} - z-direction component of gas phase velocity

v_{ox} - x-direction component of oil phase velocity

v_{oy} - y-direction component of oil phase velocity

v_{wx} - x-direction component of water phase velocity

v_{wy} - y-direction component of water phase velocity

v_{wz} - z-direction component of water phase velocity

Mass Out

$$\left(C_{ig}\rho_g v_{gx} + C_{io}\rho_o v_{ox} + C_{i\omega}\rho_\omega v_{\omega x}\right)_{x+\Delta x} \Delta y \Delta z \Delta t +$$

$$\left(C_{ig}\rho_g v_{gy} + C_{io}\rho_o v_{oy} + C_{i\omega}\rho_\omega v_{\omega y}\right)_{y+\Delta y} \Delta x \Delta z \Delta t +$$

$$\left(C_{ig}\rho_g v_{gz} + C_{io}\rho_o v_{oz} + C_{i\omega}\rho_\omega v_{\omega z}\right)_{z+\Delta z} \Delta x \Delta y \Delta t +$$

$$q_i \Delta x \Delta y \Delta z \Delta t \quad . \quad . \quad . \quad . \quad . \quad . \quad (1\text{-}5)$$

In Eq.(1.5) we have added a source or sink (injection or production) term q_i, which equals the mass rate of production of the ith component per unit volume of reservoir element. A negative q_i implies injection.

Mass Accumulation

$$\left\{\left(\phi C_{ig}\rho_g S_g + \phi C_{io}\rho_o S_o + \phi C_{i\omega}\rho_\omega S_\omega\right)_{t+\Delta t} - \left(\phi C_{ig}\rho_g S_g + \phi C_{io}\rho_o S_o + \phi C_{i\omega}\rho_\omega S_\omega\right)_t\right\} \Delta x \Delta y \Delta z \quad . \quad . \quad . \quad (1\text{-}6)$$

First combining Eqs.(1.4), (1.5) and (1.6); then dividing it by $\Delta x\Delta y\Delta z\Delta t$ and rearranging gives:

$$-\frac{1}{\Delta x}\left\{\left(C_{ig}\rho_g v_{gx}+C_{io}\rho_o v_{ox}+C_{i\omega}\rho_\omega v_{\omega x}\right)_{x+\Delta x}-\right.$$

$$\left.\left(C_{ig}\rho_g v_{gx}+C_{io}\rho_o v_{ox}+C_{i\omega}\rho_\omega v_{\omega x}\right)_x\right\}-$$

$$\frac{1}{\Delta y}\left\{\left(C_{ig}\rho_g v_{gy}+C_{io}\rho_o v_{oy}+C_{i\omega}\rho_\omega v_{\omega y}\right)_{y+\Delta y}-\right.$$

$$\left.\left(C_{ig}\rho_g v_{gy}+C_{io}\rho_o v_{oy}+C_{i\omega}\rho_\omega v_{\omega y}\right)_y\right\}-$$

$$\frac{1}{\Delta z}\left\{\left(C_{ig}\rho_g v_{gz}+C_{io}\rho_o v_{oz}+C_{i\omega}\rho_\omega v_{\omega z}\right)_{z+\Delta z}-\right.$$

$$\left.\left(C_{ig}\rho_g v_{gz}+C_{io}\rho_o v_{oz}+C_{i\omega}\rho_\omega v_{\omega z}\right)_z\right\}-q_i=$$

$$\frac{1}{\Delta t}\left\{\phi\left(C_{ig}\rho_g S_g+C_{io}\rho_o S_o+C_{i\omega}\rho_\omega S_\omega\right)_{t+\Delta t}-\right.$$

$$\left.\phi\left(C_{ig}\rho_g S_g+C_{io}\rho_o S_o+C_{i\omega}\rho_\omega S_\omega\right)_t\right\} \quad . \quad . \quad . \quad . \quad . \quad (1\_7)$$

where

ϕ = porosity

S_g, S_o, S_w = gas, oil and water phase saturations respectively.

In the limit as Δx, Δy, Δz and Δt tend to zero, Eq.(1.7) becomes:

$$-\frac{\partial}{\partial x}\left(C_{ig}\rho_g v_{gx}+C_{io}\rho_o v_{ox}+C_{i\omega}\rho_\omega v_{\omega x}\right)-\frac{\partial}{\partial y}\left(C_{ig}\rho_g v_{gy}+C_{io}\rho_o v_{oy}+C_{i\omega}\rho_\omega v_{\omega y}\right)$$

$$-\frac{\partial}{\partial z}\left(C_{ig}\rho_g v_{gz}+C_{io}\rho_o v_{oz}+C_{i\omega}\rho_\omega v_{\omega z}\right)-q_i =$$

$$\frac{\partial}{\partial t}\left\{\phi\left(C_{ig}\rho_g S_g+C_{io}\rho_o S_o+C_{i\omega}\rho_\omega S_\omega\right)\right\} \quad . \quad . \quad . \quad . \quad (1\text{-}8)$$

Introducing the notation of divergence (2),

$$\mathrm{div}\left(C_{ig}\rho_g\vec{v}_g+C_{io}\rho_o\vec{v}_o+C_{i\omega}\rho_\omega\vec{v}_\omega\right)$$

$$=\nabla\cdot\left(C_{ig}\rho_g\vec{v}_g+C_{io}\rho_o\vec{v}_o+C_{i\omega}\rho_\omega\vec{v}_\omega\right)$$

$$=\left(\frac{\partial}{\partial x}\vec{i}+\frac{\partial}{\partial y}\vec{j}+\frac{\partial}{\partial z}\vec{k}\right)\cdot\left(C_{ig}\rho_g\vec{v}_g+C_{io}\rho_o\vec{v}_o+C_{i\omega}\rho_\omega\vec{v}_\omega\right)$$

$$=\left(\frac{\partial}{\partial x}\vec{i}+\frac{\partial}{\partial y}\vec{j}+\frac{\partial}{\partial z}\vec{k}\right)\cdot\left(C_{ig}\rho_g v_{gx}\vec{i}+C_{ig}\rho_g v_{gy}\vec{j}+C_{ig}\rho_g v_{gz}\vec{k}+\right.$$

$$C_{io}\rho_o v_{ox}\vec{i}+C_{io}\rho_o v_{oy}\vec{j}+C_{io}\rho_o v_{oz}\vec{k}+C_{i\omega}\rho_\omega v_{\omega x}\vec{i}+C_{i\omega}\rho_\omega v_{\omega y}\vec{j}+$$

$$\left.C_{i\omega}\rho_\omega v_{\omega z}\vec{k}\right)$$

$$=\frac{\partial}{\partial x}\left(C_{ig}\rho_g v_{gx}+C_{io}\rho_o v_{ox}+C_{i\omega}\rho_\omega v_{\omega x}\right)+$$

$$\frac{\partial}{\partial y}\left(C_{ig}\rho_g v_{gy}+C_{io}\rho_o v_{oy}+C_{i\omega}\rho_\omega v_{\omega y}\right)+$$

$$\frac{\partial}{\partial z}\left(C_{ig}\rho_g v_{gz}+C_{io}\rho_o v_{oz}+C_{i\omega}\rho_\omega v_{\omega z}\right) \quad . \quad . \quad . \quad (1\text{-}9)$$

Equating Eqs.(1.8) and (1.9), we obtain the "continuity equation".

$$-\nabla\cdot\left(C_{ig}\rho_g\vec{v}_g + C_{io}\rho_o\vec{v}_o + C_{i\omega}\rho_\omega\vec{v}_\omega\right) - q_i =$$

$$\frac{\partial}{\partial t}\left\{\phi\left(C_{ig}\rho_g S_g + C_{io}\rho_o S_o + C_{i\omega}\rho_\omega S_\omega\right)\right\} \quad \ldots \quad (1\text{-}10)$$

Rate Equation

Darcy's Law for multiphase flow is:

$$\vec{v}_p = -\frac{(K_e)_p}{\mu_p}\left(\nabla P_p - \rho_p \nabla z\right) \quad \ldots \quad (1\text{-}11)$$

where

$(K_e)_p$ - effective permeability of rock to fluid 'p' when more than one fluid is flowing.

Substitution of v_p in Eq.(1.10) gives:

$$\nabla.\left\{\frac{C_{ig}\rho_g(K_e)_g}{\mu_g}(\nabla P_g-\rho_g\nabla z)+\frac{C_{io}\rho_o(K_e)_o}{\mu_o}(\nabla P_o-\rho_o\nabla z)\right.$$

$$\left.+\frac{C_{i\omega}\rho_\omega(K_e)_\omega}{\mu_\omega}(\nabla P_\omega-\rho_\omega\nabla z)\right\}-q_i=$$

$$\frac{\partial}{\partial t}\left\{\phi(C_{ig}\rho_g S_g+C_{io}\rho_o S_o+C_{i\omega}\rho_\omega S_\omega)\right\} \quad . \quad . \quad . \quad (1\text{-}12)$$

1.3 Relative Permeability Phenomena

When more than one fluid flows through a porous media, the fluids interfere with each other and each fluid experiences effective permeability lower than that if it alone occupied the pore space. The relative permeability is defined as:

$$k_{rp}=\frac{(K_e)_p}{\overleftrightarrow{K}}\leqslant 1 \quad . \quad . \quad . \quad . \quad (1\text{-}13)$$

where

$\overleftrightarrow{K}$ = absolute phase permeability and is a function of fluid saturation.

Substituting $(K_e)_p$ in Eq. (1.12), we obtain

$$\nabla\cdot\left\{\frac{c_{ig}\rho_g\overleftrightarrow{K}k_{rg}}{\mu_g}(\nabla P_g-\rho_g\nabla z)+\frac{c_{io}\rho_o\overleftrightarrow{K}k_{ro}}{\mu_o}(\nabla P_o-\rho_o\nabla z)\right.$$

$$\left.+\frac{c_{iw}\rho_w\overleftrightarrow{K}k_{rw}}{\mu_w}(\nabla P_w-\rho_w\nabla z)\right\}-q_i=$$

$$\frac{\partial}{\partial t}\left\{\phi(c_{ig}\rho_g S_g+c_{io}\rho_o S_o+c_{iw}\rho_w S_w)\right\} \quad . \quad . \quad . \quad (1\text{-}14)$$

1.4 Black-Oil Reservoir Simulator

Most reservoir simulators are classified into three general groups according to the type of reservoir they are intended to simulate. They are gas reservoir simulators, black-oil reservoir simulators and compositional reservoir simulators. Gas reservoir simulators may be either single-phase or two-phase models depending on whether or not mobile water is present.

A black-oil reservoir simulator, on the other hand, is capable of simulating those systems where gas, oil and water are all present. Usually, mass transfer is accounted for between the gas and oil phases. This chapter provides detailed information concerning the development of a three-dimensional, three-phase black oil simulator. The formulation is then modified to model a two-phase gas reservoir, and also a simple single phase aquifer model.

For the black-oil model we assume that the oil phase consists of only two components, the dissolved gas and the residual or black oil that remains when the gas is liberated from solution. We make assumptions that no mass transfers occur between the oil and water phases. Gas is assumed to be soluble in both oil and water. A one-way phase transfer occurs between the gas and oil and also between the gas and water. This means the gas phase can be transported both in the oil and water phaes but the oil and water do not vaporize into the gas phase. These assumptions lead to the following conditions.

	Gas Phase	Oil Phase	Water Phase
Gas Component	$C_{gg} = 1$	$C_{go} = \frac{m_g}{m_g + m_o}$	$C_{g\omega} = \frac{m_g}{m_g + m_\omega}$
Oil Component	$C_{og} = 0$	$C_{oo} = \frac{m_o}{m_g + m_o}$	$C_{o\omega} = 0$
Water Component	$C_{\omega g} = 0$	$C_{\omega o} = 0$	$C_{\omega\omega} = \frac{m_\omega}{m_g + m_\omega}$

where (1-15)

m_g, m_o, m_w - masses of gas, oil and water phases respectively.

Formation Volume Factors

The Oil formation volume factor (B_o) is defined as:

$$B_o = \frac{V_o}{V_{osc}} \quad . \quad . \quad . \quad . \quad . \quad (1\text{-}16)$$

where

V_o - oil volume containing dissolved gas at reservoir conditions

V_{osc} - dead oil volume at standard conditions

Since $\rho_{osc} = \frac{m_o}{V_{osc}}$, $\rho_o = \frac{m_g + m_o}{V_o}$; then

$$B_o = \frac{m_g + m_o}{\rho_o} \cdot \frac{\rho_{osc}}{m_o}$$

$$\text{or} \quad \frac{m_o}{m_g + m_o} = C_{oo} = \frac{\rho_{osc}}{\rho_o B_o} \quad . \quad . \quad . \quad (1\text{-}17)$$

where

ρ_o = reservoir oil density

ρ_{osc} = dead oil density at standard conditions

Similarly we can show that

$$C_{\omega\omega} = \frac{\rho_{\omega sc}}{\rho_\omega B_\omega} \quad . \quad . \quad . \quad . \quad (1\text{-}18)$$

where

ρ_{wsc} = water density at standard conditions

ρ_w = water density at reservoir conditions

ρ_w = water formation volume factor

Solution Gas-Oil Ratio

Solution gas-oil ratio (R_{so}) is defined as:

$$R_{so} = \frac{V_{gsc}}{V_{osc}} \quad , \quad \text{or}$$

$$R_{so} = \frac{m_g/\rho_{gsc}}{m_o/\rho_{osc}} = \frac{m_g \rho_{osc}}{m_o \rho_{gsc}} \quad . \quad . \quad . \quad . \quad (1\text{-}19)$$

Now we calculate for

$$C_{go} = \frac{m_g}{m_g + m_o} = \frac{m_o}{m_g + m_o} \cdot \frac{m_g}{m_o}$$

$$= C_{oo} \frac{R_{so} \rho_{gsc}}{\rho_{osc}} \quad . \quad . \quad . \quad . \quad (1\text{-}20)$$

Subsitituting Eq.(1.17) for C_{oo} in Eq.(1.20), we get

$$C_{go} = \frac{\rho_{osc}}{\rho_o B_o} \cdot \frac{R_{so} \rho_{gsc}}{\rho_{osc}} = \frac{R_{so} \rho_{gsc}}{\rho_o B_o} \quad . \quad . \quad (1\text{-}21)$$

Similarly, we can show that

$$C_{g\omega} = \frac{R_{s\omega} \rho_{gsc}}{\rho_\omega B_\omega} \quad . \quad . \quad . \quad (1\text{-}22)$$

where

R_{sw} = solution gas-water ratio

Gas formation volume factor (B_g) is defined as:

$$B_g = \frac{V_g}{V_{gsc}} = \frac{m_g/\rho_g}{m_g/\rho_{gsc}} = \frac{\rho_{gsc}}{\rho_g} \quad \ldots \quad (1\text{-}23)$$

Now we have:

	Gas Phase	Oil Phase	Water Phase
Gas Component	$C_{gg} = 1$	$C_{go} = \frac{R_{so}\rho_{gsc}}{\rho_o B_o}$	$C_{g\omega} = \frac{R_{s\omega}\rho_{gsc}}{\rho_\omega B_\omega}$
Oil Component	$C_{og} = 0$	$C_{oo} = \frac{\rho_{osc}}{\rho_o B_o}$	$C_{o\omega} = 0$
Water Component	$C_{\omega g} = 0$	$C_{\omega o} = 0$	$C_{\omega\omega} = \frac{\rho_{\omega sc}}{\rho_\omega B_\omega}$

$$\ldots \quad (1\text{-}24)$$

From Eq.(1.14), we begin to derive oil equation first.

$$\nabla \cdot \left\{ \frac{\rho_{osc}\rho_o \overleftrightarrow{K} k_{ro}}{\rho_o B_o \mu_o} (\nabla P_o - \rho_o \nabla z) \right\} - q_o = \frac{\partial}{\partial t}\left(\frac{\phi \rho_{osc} \rho_o S_o}{\rho_o B_o} \right)$$

$$\ldots \quad (1\text{-}25)$$

Dividing Eq.(1.25) by ρ_{osc}, we get the <u>"oil equation"</u>.

$$\nabla.\left\{\frac{\overleftrightarrow{K}k_{ro}}{B_o\mu_o}(\nabla P_o - \varrho_o\nabla z)\right\} - \frac{q_o}{\varrho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) \quad . \quad . \qquad (1\text{-}26)$$

Similarly, we can have the <u>"water equation"</u> as:

$$\nabla.\left\{\frac{\overleftrightarrow{K}k_{r\omega}}{B_\omega\mu_\omega}(\nabla P_\omega - \varrho_\omega\nabla z)\right\} - \frac{q_\omega}{\varrho_{\omega sc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_\omega}{B_\omega}\right) \quad . \quad . \qquad (1\text{-}27)$$

Finally, the <u>"gas equation"</u> is:

$$\nabla.\left\{\frac{\varrho_{gsc}\overleftrightarrow{K}k_{rg}}{B_g\mu_g}(\nabla P_g - \varrho_g\nabla z) + \frac{R_{so}\varrho_{gsc}}{\varrho_o B_o}\frac{\varrho_o\overleftrightarrow{K}k_{ro}}{\mu_o}(\nabla P_o - \varrho_o\nabla z)\right.$$

$$\left. + \frac{R_{s\omega}\varrho_{gsc}}{\varrho_\omega B_\omega}\frac{\varrho_\omega\overleftrightarrow{K}k_{r\omega}}{\mu_\omega}(\nabla P_\omega - \varrho_\omega\nabla z)\right\} - q_g =$$

$$\frac{\partial}{\partial t}\left\{\phi\left(\frac{\varrho_{gsc}S_g}{B_g} + \frac{R_{so}\varrho_{gsc}}{\varrho_o B_o}\varrho_o S_o + \frac{R_{s\omega}\varrho_{gsc}}{\varrho_\omega B_\omega}\varrho_\omega S_\omega\right)\right\}$$

Dividing the above equation by ρ_{gsc}, we get

$$\nabla.\overleftrightarrow{K}\left\{\frac{k_{rg}}{B_g\mu_g}(\nabla P_g - \varrho_g\nabla z) + \frac{R_{so}k_{ro}}{B_o\mu_o}(\nabla P_o - \varrho_o\nabla z) + \frac{R_{sw}k_{rw}}{B_w\mu_w}(\nabla P_w - \varrho_w\nabla z)\right\} - \frac{q_g}{\varrho_{gsc}} =$$

$$\frac{\partial}{\partial t}\left\{\phi\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o} + \frac{R_{sw}S_w}{B_w}\right)\right\} \quad . \quad . \quad . \quad (1\text{-}28)$$

Phase Mobility

Phase mobility (λ_p) is defined as:

$$\lambda_p = \frac{k_{rp}}{\mu_p} \quad . \quad . \quad . \quad . \quad . \quad (1\text{-}29)$$

k_{rp} = relative permeability

μ_p = phase viscosity

Substituting λ_p in Eqs.(1.26), (1.27) and (1.28) gives

Oil Equation

$$\nabla.\overleftrightarrow{K}\left\{\left(\frac{\lambda_o}{B_o}\right)\left(\nabla P_o - \rho_o \nabla z\right)\right\} - \frac{q_o}{\rho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) \quad . \quad . \quad (1\text{-}30)$$

Water Equation

$$\nabla.\overleftrightarrow{K}\left\{\left(\frac{\lambda_w}{B_w}\right)\left(\nabla P_w - \rho_w \nabla z\right)\right\} - \frac{q_w}{\rho_{wsc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) \quad . \quad . \quad (1\text{-}31)$$

Gas Equation

$$\nabla.\overleftrightarrow{K}\left\{\frac{\lambda_g}{B_g}\left(\nabla P_g - \rho_g \nabla z\right) + \frac{R_{so}\lambda_o}{B_o}\left(\nabla P_o - \rho_o \nabla z\right) + \frac{R_{sw}\lambda_w}{B_w}\left(\nabla P_w - \rho_w \nabla z\right)\right\} - \frac{q_g}{\rho_{gsc}} =$$

$$\frac{\partial}{\partial t}\left\{\phi\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o} + \frac{R_{sw}S_w}{B_w}\right)\right\} \quad . \quad . \quad . \quad (1\text{-}32)$$

Capillary Pressures

Capillary pressure is defined as the difference in pressure between the wetting and non-wetting phases. Considering the wetting phase to be water and the non-wetting phase to be oil the oil-water capillary pressure may be written as:

$$P_{cow} = P_o - P_w \quad . \quad . \quad . \quad . \quad (1\_33)$$

and the gas-oil capillary pressure as

$$P_{cgo} = P_g - P_o \quad . \quad . \quad . \quad (1\_34)$$

Then $P_w = P_o - P_{cow}$ and

$$P_g = P_o + P_{cgo} \quad . \quad . \quad . \quad (1\_35)$$

Substitution of P_w and P_g in Eqs.(1.30), (1.31) and (1.32) gives

Oil Equation

$$\nabla.\overleftrightarrow{K}\left(\frac{\lambda_o}{B_o}\right)\left(\nabla P_o - \rho_o \nabla z\right) - \frac{q_o}{\rho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) \quad . \quad . \quad (1\text{-}36)$$

Let

$$CG_o = -\nabla.\overleftrightarrow{K}\left(\frac{\lambda_o}{B_o}\right)\rho_o \nabla z \quad . \quad . \quad . \quad (1\text{-}37)$$

Then Eq.(1.36) can be written in another form as:

$$\nabla.\overleftrightarrow{K}\left(\frac{\lambda_o}{B_o}\right)\nabla P_o + CG_o - \frac{q_o}{\rho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) \quad . \quad . \quad (1\text{-}38)$$

Water Equation

$$\nabla . \overleftrightarrow{K} \left(\frac{\lambda_\omega}{B_\omega} \right) \left(\nabla P_o - \nabla P_{cow} - \rho_\omega \nabla z \right) - \frac{q_\omega}{\rho_{\omega sc}}$$

$$= \frac{\partial}{\partial t} \left(\frac{\phi S_\omega}{B_\omega} \right) \qquad \ldots \ldots \qquad (1\text{-}39)$$

Let

$$CG_w = - \nabla . \overleftrightarrow{K} \left(\frac{\lambda_\omega}{B_\omega} \right) \left(\nabla P_{cow} + \rho_\omega \nabla z \right) \qquad \ldots \qquad (1\text{-}40)$$

The water equation becomes:

$$\nabla . \overleftrightarrow{K} \left(\frac{\lambda_\omega}{B_\omega} \right) \nabla P_o + CG_w - \frac{q_\omega}{\rho_{\omega sc}} = \frac{\partial}{\partial t} \left(\frac{\phi S_\omega}{B_\omega} \right) \qquad \ldots \qquad (1\text{-}41)$$

Gas Equation

$$\nabla.\overleftrightarrow{K}\left\{\frac{\lambda_g}{B_g}\left(\nabla P_o+\nabla P_{cgo}-\rho_g\nabla z\right)+\frac{R_{so}\lambda_o}{B_o}\left(\nabla P_o-\rho_o\nabla z\right)\right.$$

$$\left.+\frac{R_{sw}\lambda_w}{B_w}\left(\nabla P_o-\nabla P_{cow}-\rho_w\nabla z\right)\right\}-\frac{q_g}{\rho_{gsc}} =$$

$$\frac{\partial}{\partial t}\left\{\phi\left(\frac{S_g}{B_g}+\frac{R_{so}S_o}{B_o}+\frac{R_{sw}S_w}{B_w}\right)\right\} \quad . \quad . \quad . \quad (1\text{-}42)$$

$$\nabla.\overleftrightarrow{K}\left\{\left(\frac{\lambda_g}{B_g}+\frac{R_{so}\lambda_o}{B_o}+\frac{R_{sw}\lambda_w}{B_w}\right)\nabla P_o+\frac{\lambda_g}{B_g}\left(\nabla P_{cgo}-\rho_g\nabla z\right)\right.$$

$$\left.-\frac{R_{so}\lambda_o}{B_o}\rho_o\nabla z-\frac{R_{sw}\lambda_w}{B_w}\left(\nabla P_{cow}+\rho_w\nabla z\right)\right\}-\frac{q_g}{\rho_{gsc}}$$

$$=\frac{\partial}{\partial t}\left\{\phi\left(\frac{S_g}{B_g}+\frac{R_{so}S_o}{B_o}+\frac{R_{sw}S_w}{B_w}\right)\right\} \quad . \quad . \quad (1\text{-}43)$$

Let

$$CG_g = \nabla.\overleftrightarrow{K}\left\{\frac{\lambda_g}{B_g}\left(\nabla P_{cgo}-\rho_g\nabla z\right)-\frac{R_{so}\lambda_o}{B_o}\rho_o\nabla z\right.$$

$$\left.-\frac{R_{sw}\lambda_w}{B_w}\left(\nabla P_{cow}+\rho_w\nabla z\right)\right\} \quad . \quad . \quad . \quad (1\text{-}44)$$

Substituting CG_g in Eq.(1.43), we obtain:

$$\nabla.\overleftrightarrow{K}\left\{\left(\frac{\lambda_g}{B_g}+\frac{R_{so}\lambda_o}{B_o}+\frac{R_{sw}\lambda_w}{B_w}\right)\right\}\nabla P_o+CG_g$$

$$-\frac{q_g}{e_{gsc}}=\frac{\partial}{\partial t}\left\{\phi\left(\frac{S_g}{B_g}+\frac{R_{so}S_o}{B_o}+\frac{R_{sw}S_w}{B_w}\right)\right\} \quad . \quad . \quad . \quad . \quad (1\text{-}45)$$

1.5 Formulation of the IMPES Method

Having derived the governing equations for multiphase flow, we must now solve Eqs.(1.38), (1.41) and (1.45) to obtain the spatial distribution of pressure and saturation of each phase in the porous media. The two basic methods for solving multiphase equations are:

1. Simultaneous solution (SS) method.
2. Implicit pressure-explicit saturation (IMPES) method.

The essence of SS method will be discussed briefly because this study is directed at applying the IMPES method for solving the multiphase flow equations. In 1959, Douglas (3) first proposed a SS method that writes the saturation derivatives on the right hand sides of Eqs. (1.38), (1.41) and (1.45) in terms of the pressure derivatives and then solves the resulting equations simultaneously. The detailed discussion on SS method can be found in many reservoir simulation literatures (4, 5, 6, 7).

The IMPES method (8), on the other hand, combines the individual phase equations into a single, multiphase equation based on pressure, and then solves the pressure equation implicitly for the pressure distribution. After the pressure solution is obtained, the saturations are explicitly updated by substituting the results in the Eqs(1.38), (1.41). When S_o and S_w are known for new time level, the new capillary pressures are calculated, and then explicitly used at the next time step. The summary of IMPES method is shown in Figure (1.2).

In evaluating pressures from the pressure equation, a problem may arise: How can we determine the pressures if the solution of the equation depends on the calculation of phase mobilities which themselves depend on these pressures? We can get out of this dilemma by letting all pressure - and saturation - dependent terms lag behind the pressure terms. This is done by evaluating the mobilities and capillary pressures at the previous saturations and pressures. Implied in this approach is the assumption that the saturation and pressure values are not changing very rapidly. It is well justified in a large areal model study, for example, with well-distributed production and/ or injection rates.

Oil Equation

$$\nabla.\overleftrightarrow{K}\left(\frac{\lambda_o}{B_o}\right)\nabla P_o + CG_o - \frac{q_{lo}}{\rho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) \quad . \quad . \quad . \quad (1-48)$$

Water Equation

$$\nabla.\overleftrightarrow{K}\left(\frac{\lambda_w}{B_w}\right)\nabla P_o + CG_w - \frac{q_w}{\rho_{wsc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) \quad . \quad . \quad . \quad (1-49)$$

Gas Equation

$$\nabla.\overleftrightarrow{K}\left\{\left(\frac{\lambda_g}{B_g} + \frac{R_{so}\lambda_o}{B_o} + \frac{R_{sw}\lambda_w}{B_w}\right)\right\}\nabla P_o + CG_g$$

$$- \frac{q_g}{\rho_{gsc}} = \frac{\partial}{\partial t}\left\{\phi\left(\frac{S_g}{B_g} + \frac{R_{so}S_o}{B_o} + \frac{R_{sw}S_w}{B_w}\right)\right\} \quad . \quad . \quad . \quad (1-50)$$

Expanding the RHS of Eq.(1.48) and multiplying it by $(B_o - R_{so}B_g)$, we get

$$(B_o - R_{so}B_g)\left\{\frac{\phi}{B_o}\frac{\partial S_o}{\partial t} + \left(\frac{S_o}{B_o}\frac{\partial \phi}{\partial P_o} - \frac{S_o\phi}{B_o^2}\frac{\partial B_o}{\partial P_o}\right)\frac{\partial P_o}{\partial t}\right\}$$

$$= \phi\frac{\partial S_o}{\partial t} + \left(S_o\frac{\partial \phi}{\partial P_o} - \frac{S_o\phi}{B_o}\frac{\partial B_o}{\partial P_o}\right)\frac{\partial P_o}{\partial t} - \frac{R_{so}B_g\phi}{B_o}\frac{\partial S_o}{\partial t}$$

$$-\left(\frac{R_{so}B_gS_o}{B_o}\frac{\partial \phi}{\partial P_o} - \frac{R_{so}B_gS_o\phi}{B_o^2}\frac{\partial B_o}{\partial P_o}\right)\frac{\partial P_o}{\partial t} \quad \ldots \quad (1\_51)$$

Expanding the RHS of Eq.(1.49) and multiplying it by $(B_w - R_{sw}B_g)$, we get

$$(B_w - R_{sw}B_g)\left\{\frac{\phi}{B_w}\frac{\partial S_w}{\partial t} + \left(\frac{S_w}{B_w}\frac{\partial \phi}{\partial P_w} - \frac{S_w\phi}{B_w^2}\frac{\partial B_w}{\partial P_w}\right)\frac{\partial P_w}{\partial t}\right\}$$

$$= \phi\frac{\partial S_w}{\partial t} + \left(S_w\frac{\partial \phi}{\partial P_w} - \frac{S_w\phi}{B_w}\frac{\partial B_w}{\partial P_w}\right)\frac{\partial P_w}{\partial t} - \frac{R_{sw}B_g\phi}{B_w}\frac{\partial S_w}{\partial t}$$

$$-\left(\frac{R_{sw}B_gS_w}{B_w}\frac{\partial \phi}{\partial P_w} - \frac{R_{sw}B_gS_w\phi}{B_w^2}\frac{\partial B_w}{\partial P_w}\right)\frac{\partial P_w}{\partial t} \quad \ldots \quad (1\_52)$$

Expanding the RHS of Eq.(1.51) and multiplying it by B_g, we get

$$B_g\left\{\frac{\phi}{B_g}\frac{\partial S_g}{\partial t}+\left(\frac{S_g}{B_g}\frac{\partial\phi}{\partial P_g}-\frac{S_g\phi}{B_g^2}\frac{\partial B_g}{\partial P_g}\right)\frac{\partial P_g}{\partial t}+\left(\frac{R_{so}S_o}{B_o}\frac{\partial\phi}{\partial P_g}\right.\right.$$

$$+\frac{S_o\phi}{B_o}\frac{\partial R_{so}}{\partial P_g}-\frac{R_{so}S_o\phi}{B_o^2}\frac{\partial B_o}{\partial P_g}\Big)\frac{\partial P_g}{\partial t}+\frac{R_{sw}\phi}{B_w}\frac{\partial S_w}{\partial t}+$$

$$\left.\left(\frac{R_{sw}S_w}{B_w}\frac{\partial\phi}{\partial P_g}+\frac{S_w\phi}{B_w}\frac{\partial R_{sw}}{\partial P_g}-\frac{R_{sw}S_w\phi}{B_w^2}\frac{\partial B_w}{\partial P_g}\right)\frac{\partial P_g}{\partial t}\right\}$$

$$=\phi\frac{\partial S_g}{\partial t}+\left(S_g\frac{\partial\phi}{\partial P_g}-\frac{S_g\phi}{B_g}\frac{\partial B_g}{\partial P_g}\right)\frac{\partial P_g}{\partial t}+\frac{R_{so}B_g\phi}{B_o}\frac{\partial S_o}{\partial t}$$

$$+\frac{R_{so}B_gS_o}{B_o}\frac{\partial\phi}{\partial P_g}+\frac{B_gS_o\phi}{B_o}\frac{\partial R_{so}}{\partial P_g}-\frac{R_{so}B_gS_o\phi}{B_o^2}\frac{\partial B_o}{\partial P_g}\Big)\frac{\partial P_g}{\partial t}$$

$$+\frac{R_{sw}B_g\phi}{B_w}\frac{\partial S_w}{\partial t}+\left(\frac{R_{sw}B_gS_w}{B_w}\frac{\partial\phi}{\partial P_g}+\frac{B_gS_w\phi}{B_w}\frac{\partial R_{sw}}{\partial P_g}\right.$$

$$\left.-\frac{R_{sw}B_gS_w\phi}{B_w^2}\frac{\partial B_w}{\partial P_g}\right)\frac{\partial P_g}{\partial t}\qquad\ldots\qquad(1\_53)$$

Assuming that

$$\Delta_t P_o=\Delta_t P_w=\Delta_t P_g=\Delta_t P$$

Adding Eqs.(1.51), (1.52) and (1.53), we get

$$(B_o - R_{so}B_g)\left\{\nabla.\overleftrightarrow{K}\left(\frac{\lambda_o}{B_o}\right)\nabla P + CG_o - \frac{q_o}{\rho_{osc}}\right\} +$$

$$(B_w - R_{sw}B_g)\left\{\nabla.\overleftrightarrow{K}\left(\frac{\lambda_w}{B_w}\right)\nabla P + CG_w - \frac{q_w}{\rho_{wsc}}\right\} +$$

$$B_g\left\{\nabla.\overleftrightarrow{K}\left(\frac{\lambda_g}{B_g} + \frac{R_{so}\lambda_o}{B_o} + \frac{R_{sw}\lambda_w}{B_w}\right)\nabla P + CG_g\right.$$

$$\left. - \frac{q_g}{\rho_{gsc}}\right\} = \left[\phi\left(\frac{\partial S_o}{\partial t} + \frac{\partial S_w}{\partial t} + \frac{\partial S_g}{\partial t}\right) + \left\{(S_o + S_w + S_g)\frac{\partial \phi}{\partial P}\right.\right.$$

$$- \frac{S_o\phi}{B_o}\frac{\partial B_o}{\partial P} - \frac{S_g\phi}{B_g}\frac{\partial B_g}{\partial P} + \frac{B_gS_o\phi}{B_o}\frac{\partial R_{so}}{\partial P} +$$

$$\left.\left.\frac{B_gS_w\phi}{B_w}\frac{\partial R_{sw}}{\partial P}\right\}\right]\frac{\partial P}{\partial t} \qquad \ldots \qquad (1\text{-}54)$$

Since

$$S_o + S_w + S_g = 1 \quad ,$$

$$\frac{\partial S_o}{\partial t} + \frac{\partial S_w}{\partial t} + \frac{\partial S_g}{\partial t} = 0 \qquad \ldots \qquad (1\text{-}55)$$

Now the RHS of Eq.(1.54) becomes:

$$\left\{ \frac{\partial \phi}{\partial P} + S_o \phi \left(-\frac{1}{B_o} \frac{\partial B_o}{\partial P} + \frac{B_g}{B_o} \frac{\partial R_{so}}{\partial P} \right) + \right.$$

$$S_w \phi \left(-\frac{1}{B_w} \frac{\partial B_w}{\partial P} + \frac{B_g}{B_w} \frac{\partial R_{sw}}{\partial P} \right) +$$

$$\left. S_g \phi \left(-\frac{1}{B_g} \frac{\partial B_g}{\partial P} \right) \right\} \frac{\partial P}{\partial t} \quad . \quad . \quad . \quad . \quad (1\_56)$$

As the oil, water, gas and rock compressibilities are identified as,

$$c_o = -\frac{1}{B_o} \frac{\partial B_o}{\partial P_o} + \frac{B_g}{B_o} \frac{\partial R_{so}}{\partial P_o} \quad . \quad . \quad . \quad . \quad (1\_57)$$

$$c_w = -\frac{1}{B_w} \frac{\partial B_w}{\partial P_w} + \frac{B_g}{B_w} \frac{\partial R_{sw}}{\partial P_w} \quad . \quad . \quad . \quad . \quad (1\_58)$$

$$c_g = -\frac{1}{B_g} \frac{\partial B_g}{\partial P_g} \quad . \quad . \quad . \quad . \quad . \quad (1\_59)$$

$$c_r = \frac{1}{\phi} \frac{\partial \phi}{\partial P} \quad . \quad . \quad . \quad . \quad . \quad (1\_60)$$

Substituting c_o, c_w and c_g in Eq.(1.56), it becomes

$$\phi\left(c_r + c_o S_o + c_w S_w + c_g S_g\right)\frac{\partial P}{\partial t} \quad . \quad . \quad . \quad (1\_G1)$$

Now defining the total compressibility as

$$c_t = c_r + c_o S_o + c_w S_w + c_g S_g$$

we would get the "pressure equation", in which no explicit time derivatives of saturations are present.

1.6 Pressure Equation

$$(B_o - R_{so}B_g)\left\{\nabla\cdot\overleftrightarrow{K}\left(\frac{\lambda_o}{B_o}\right)\nabla P + CG_o - \frac{q_o}{\rho_{osc}}\right\} +$$

$$(B_w - R_{sw}B_g)\left\{\nabla\cdot\overleftrightarrow{K}\left(\frac{\lambda_w}{B_w}\right)\nabla P + CG_w - \frac{q_w}{\rho_{wsc}}\right\} +$$

$$B_g\left\{\nabla\cdot\overleftrightarrow{K}\left(\frac{\lambda_g}{B_g} + \frac{R_{so}\lambda_o}{B_o} + \frac{R_{sw}\lambda_w}{B_w}\right)\nabla P + CG_g - \frac{q_g}{\rho_{gsc}}\right\} = \phi c_t \frac{\partial P}{\partial t}$$

. . . (1_G2)

First numerically solving Eq.(1.62) for P^{n+1}, and then we would get S_o^{n+1}, S_w^{n+1} using the calculated P in Eqs.(1.48) and (1.49). S_g is again determined from the equality $S_o + S_w + S_g - 1$.

Once Eq.(1.63) is solved for P, then S_w^{n+1} can be explicitly determined from Eq.(1.69)

$$\nabla . \overleftrightarrow{K} \left(\frac{\lambda_w}{B_w}\right) \nabla P + CG_w - \frac{q_w}{\rho_{wsc}}$$

$$= \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) \quad . \quad . \quad . \quad . \quad . \quad . \quad (1\_64)$$

Using S_w^{n+1}, P_{cgw} is updated for new time level.

Aquifer Model

This model considers only the flow of a single phase, water. Since no oil and gas are present in the system, the oil and gas phase terms are deleted from the pressure Eq.(1.62) and we obtain the pressure equation for the aquifer model as follows:

$$B_w \left\{ \nabla . \overleftrightarrow{K} \left(\frac{\lambda_w}{B_w}\right) \nabla P + CG_w - \frac{q_w}{\rho_{wsc}} \right\}$$

$$= \phi c_t \frac{\partial P}{\partial t} \quad . \quad . \quad . \quad . \quad . \quad (1\_65)$$

Since only single phase is flowing in the system, relative permeability becomes unity. In Eq.(1.70), CG_w is defined as

$$CG_w = -\dot{\nabla}.\left(\frac{\overleftrightarrow{K}}{\mu_w B_w}\right)\rho_w \nabla z \quad . \quad . \quad . \quad . \quad (1\_66)$$

$$\lambda_w = \frac{k_{rw}}{\mu_w} = \frac{1}{\mu_w} \quad . \quad . \quad . \quad . \quad . \quad (1\_67)$$

$$c_t = c_r + c_w \quad . \quad . \quad . \quad . \quad . \quad (1\_68)$$

CHAPTER 2: FINITE DIFFERENCE APPROXIMATIONS

Introduction

By the application of the conservation of mass principle, we have now translated the three-dimensional, three-phase black oil model into a system of coupled, non-linear partial differential equations. Although many methods exist for obtaining analytical solutions of PDEs, they are primarily limited to linear equations with regular boundary conditions. For most practical situations, however, it is generally impossible to obtain exact or analytical solutions to the PDEs describing reservoirs because of their nonlinearities. Instead, finite difference approximation methods are used almost exclusively to treat such problems.

2.1 Finite Difference Method

Finite difference methods transform the nonlinear PDEs into sets of nonlinear algebraic equations which are solved for discrete points in the computational region. These algebraic equations are known as

the finite difference equations, and are derived by replacing the derivatives in the PDEs with algebraic approximations. To determine the discrete representation of derivatives, it is useful to review what a derivative means. Derivatives represent rates of change. A derivative contains information about the local variation of a function. Therefore, a difference approximation to a derviative can be achieved by using the Taylor series expansion of the function about a neighbouring point. Thus, expanding a function of four independent variables, say, P (x, y, z, t):

$$P(x+\Delta x, y, z, t) = P(x, y, z, t) + \Delta x \left(\frac{\partial P}{\partial x}\right)_{x,y,z,t} + \frac{(\Delta x)^2}{2!}\left(\frac{\partial^2 P}{\partial x^2}\right)_{x,y,z,t} + \frac{(\Delta x)^3}{3!}\left(\frac{\partial^3 P}{\partial x^3}\right)_{x^*,y,z,t} \quad . \quad . \quad . \quad (2\text{-}1)$$

where. $x \leqslant x^* \leqslant x + \Delta x$

Solving for $\frac{\partial P}{\partial x}$,

$$\left(\frac{\partial P}{\partial x}\right)_{x,y,z,t} = \frac{P(x+\Delta x, y, z, t) - P(x, y, z, t)}{\Delta x} - \frac{\Delta x}{2}\left(\frac{\partial^2 P}{\partial x^2}\right)_{x^*,y,z,t} \quad . \quad . \quad . \quad (2\text{-}2)$$

$$\left(\frac{\partial P}{\partial x}\right)_{x,y,z,t} = \frac{P^{t}_{i+1,j,k} - P^{t}_{ijk}}{\Delta x} + O(\Delta x) \quad . \quad . \quad (2\text{-}3)$$

where O(Δx) denotes terms containing first and higher powers of Δx, which is known as the truncation error. If this error is neglected from Eq.(2.3) what we would get now is called the finite difference approximation to the first-order derivative at the grid point (x, y, z or iΔx, jΔy, kΔz or in short i, j, k) at time level t. That is,

$$\left(\frac{\partial P}{\partial x}\right)^{t}_{ijk} \simeq \frac{P^{t}_{i+1,j,k} - P^{t}_{ijk}}{\Delta x} \quad . \quad . \quad . \quad . \quad (2\text{-}4)$$

Eq.(2.4) is referred to as a '<u>forward difference approximation</u>'.

Similarly we can expand P(x − Δx, y, z, t) about the point (x, y, z, t)as:

$$P(x-\Delta x,y,z,t) = P(x,y,z,t) - \Delta x\left(\frac{\partial P}{\partial x}\right)_{x,y,z,t} + \frac{(\Delta x)^2}{2!}\left(\frac{\partial^2 P}{\partial x^2}\right)_{x,y,z,t} - \frac{(\Delta x)^3}{3!}\left(\frac{\partial^3 P}{\partial x^3}\right)_{x^{**},y,z,t} \quad . \quad . \quad (2\text{-}5)$$

where

$$x - \Delta x \leqslant x^{**} \leqslant x$$

Solving for $\frac{\partial P}{\partial x}$, we obtain a '<u>backward difference approximation</u>'.

$$\left(\frac{\partial P}{\partial x}\right)^{t}_{ijk} \simeq \frac{P^{t}_{ijk} - P^{t}_{i-1,j,k}}{\Delta x} \qquad . \qquad . \qquad . \qquad (2\text{-}6)$$

In both the forward and backward difference approximation, the error is of the same order, that is , $O(\Delta x)$. On the other hand, the centered difference is of high order, since the error term is $O(\Delta x)^2$.

The '<u>centered difference approximation</u>' is derived from substracting Eq.(2.5) from Eq.(2.1) and rearranging it gives:

$$\left(\frac{\partial P}{\partial x}\right)^{t}_{ijk} \simeq \frac{P^{t}_{i+1,j,k} - P^{t}_{i-1,j,k}}{2(\Delta x)} \qquad . \qquad . \qquad . \qquad . \qquad (2\text{-}7)$$

In a similar manner we can obtain an approximation for the second derivative by adding Eqs.(2.1) and (2.5) and rearranging it gives:

$$\left(\frac{\partial^2 P}{\partial x^2}\right)^t_{ijk} \simeq \frac{P^t_{i-1,j,k} - 2P^t_{ijk} + P^t_{i+1,j,k}}{(\Delta x)^2} \quad \cdots \quad (2-8)$$

which is of second-order accuracy.

We have noted in the pressure equation and the phase-saturation equations the very frequent occurence of the second derivative of the form:

$$\nabla . \overleftrightarrow{K} \left(\frac{\lambda_p}{B_p}\right) \nabla P_p ,$$

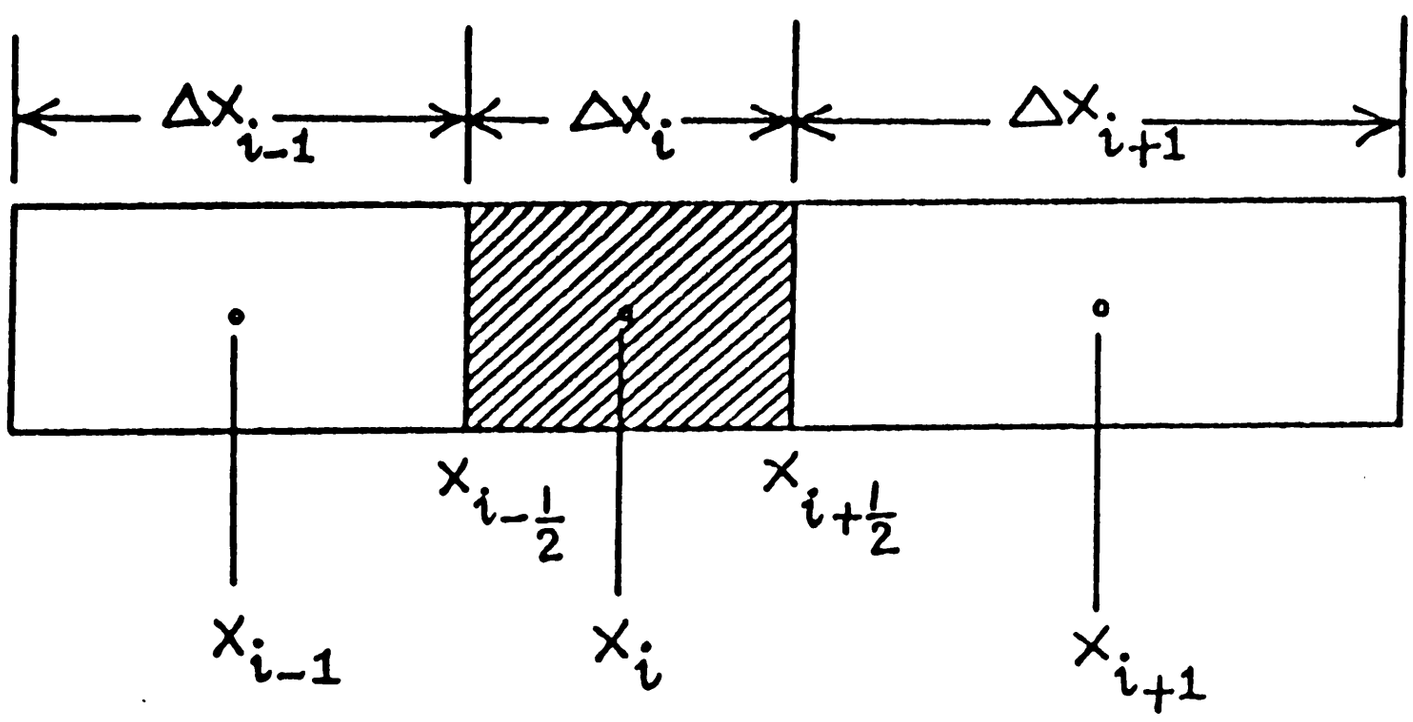
$$\frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P_p}{\partial x}\right] + \frac{\partial}{\partial y}\left[K_y\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P_p}{\partial y}\right]$$

$$+ \frac{\partial}{\partial z}\left[K_z\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P_p}{\partial z}\right] \quad \cdots \quad (2-9)$$

The terms in Eq.(2.9) are readily seen to be nonlinear because $K\left(\frac{\lambda_p}{B_p}\right)$ is function of pressure and position. Although analytical methods of solution for linear equations normally fail to cope with nonlinear differential equations, the finite difference method can be applied without modification to both linear and nonlinear problems. Now, we would try to approximate the function derivative, say, $\frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial x}\right]$ because it is the only approximation we need for our purpose. Using the block-centered grid system (Fig.2.1).

Let $\psi = K_x\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial x}$, then using a central difference approximation, we can write as:

$$\frac{\partial \psi}{\partial x} = \frac{\psi_{i+\frac{1}{2}} - \psi_{i-\frac{1}{2}}}{x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}}} = \frac{\psi_{i+\frac{1}{2}} - \psi_{i-\frac{1}{2}}}{\Delta x_i} \qquad (2-10)$$



(Fig.2.1).

$\psi_{i+\frac{1}{2}}$ and $\psi_{i-\frac{1}{2}}$ are determined as

$$\psi_{i+\frac{1}{2}} = \left(K_x \frac{\lambda_p}{B_p}\right)_{i+\frac{1}{2}} \frac{P_{i+1} - P_i}{x_{i+1} - x_i} \quad . \quad . \quad . \qquad (2-11)$$

$$\psi_{i-\frac{1}{2}} = \left(K_x \frac{\lambda_p}{B_p}\right)_{i-\frac{1}{2}} \frac{P_i - P_{i-1}}{x_i - x_{i-1}} \quad . \quad . \quad . \qquad (2-12)$$

Substituting Eqs.(2.11) and (2.12) in Eq.(2.10), we obtain:

$$\frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_\rho}{B_\rho}\right)\frac{\partial P}{\partial x}\right] = \frac{1}{\Delta x_i}\left(K_x\frac{\lambda_\rho}{B_\rho}\right)_{i+\frac{1}{2}}\frac{P_{i+1}-P_i}{x_{i+1}-x_i} - \left(K_x\frac{\lambda_\rho}{B_\rho}\right)_{i-\frac{1}{2}}\frac{P_i-P_{i-1}}{x_i-x_{i-1}}\Bigg] \tag{2-13}$$

$$= \frac{1}{\Delta x_i}\Bigg[\left(\frac{\lambda_\rho}{B_\rho}\right)_{i+\frac{1}{2}} K_{x,i+\frac{1}{2}} \frac{\Delta y_i \Delta z_i}{\Delta y_i \Delta z_i} \cdot \frac{P_{i+1}-P_i}{\frac{\Delta x_i + \Delta x_{i+1}}{2}} -$$

$$\left(\frac{\lambda_\rho}{B_\rho}\right)_{i-\frac{1}{2}} K_{x,i-\frac{1}{2}} \frac{\Delta y_i \Delta z_i}{\Delta y_i \Delta z_i} \frac{P_i - P_{i-1}}{\frac{\Delta x_i + \Delta x_{i-1}}{2}}\Bigg]$$

$$= \frac{1}{\Delta x_i \Delta y_i \Delta z_i}\Bigg[\left(\frac{\lambda_\rho}{B_\rho}\right)_{i+\frac{1}{2}} K_{x,i+\frac{1}{2}} (\Delta y_i \Delta z_i) \frac{2}{\Delta x_i + \Delta x_{i+1}} (P_{i+1} - P_i)$$

$$- \left(\frac{\lambda_\rho}{B_\rho}\right)_{i-\frac{1}{2}} K_{x,i-\frac{1}{2}} (\Delta y_i \Delta z_i) \frac{2}{\Delta x_i + \Delta x_{i-1}} (P_i - P_{i-1})\Bigg] \tag{2-14}$$

Assuming that the cross-sectional area remains constant from block to block, Eq.(2.14) can be written as:

$$(\Delta x_i \Delta y_i \Delta z_i)\frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_\rho}{B_\rho}\right)\frac{\partial P}{\partial x}\right] = \left(\frac{\lambda_\rho}{B_\rho}\right)_{i+\frac{1}{2}} (K_x A)_{i+\frac{1}{2}} \frac{2(P_{i+1}-P_i)}{\Delta x_i + \Delta x_{i+1}}$$

$$- \left(\frac{\lambda_\rho}{B_\rho}\right)_{i-\frac{1}{2}} (K_x A)_{i-\frac{1}{2}} \frac{2(P_i - P_{i-1})}{\Delta x_i + \Delta x_{i-1}} \tag{2-15}$$

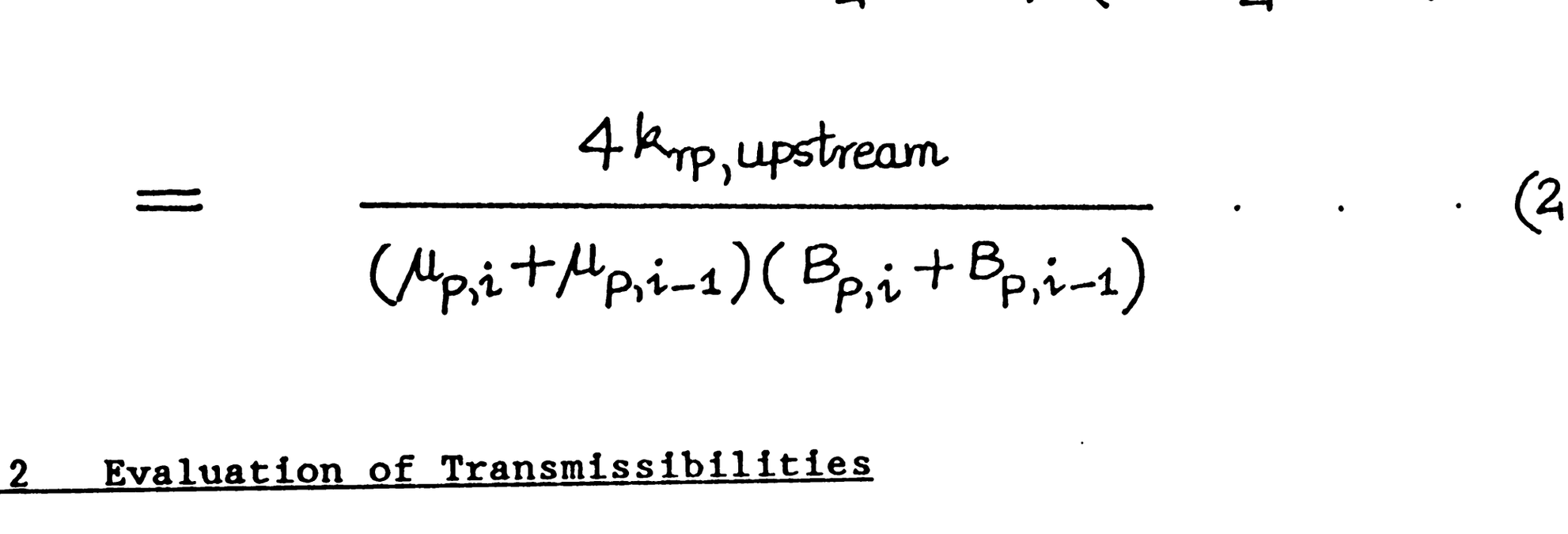
We still have the task to determine a suitable average for each of $\left(\frac{\lambda_p}{B_p}\right)$ and $\overline{KA}$. Industry experience (Crichlow(1977)for example) has shown that use of a phase mobility in the block which has the larger phase potential of the two neighbouring blocks yields more reliable results. Thus, we use the value of the phase saturation of the upsteam block at time level 'n' to determine an upstream phase relative permeability. This is then combined with the arithmetic mean values of the phase viscosities and phase formation volume factors.

$$\left(\frac{\lambda_p}{B_p}\right)_{i-\frac{1}{2}} = \left(\frac{k_{rp}}{\mu_p B_p}\right)_{i-\frac{1}{2}} = \frac{k_{rp,\,upstream}}{\left(\frac{\mu_{p,i}+\mu_{p,i-1}}{2}\right)\left(\frac{B_{p,i}+B_{p,i-1}}{2}\right)}$$

$$= \frac{4\,k_{rp,\,upstream}}{(\mu_{p,i}+\mu_{p,i-1})(B_{p,i}+B_{p,i-1})} \quad \ldots \quad (2\text{-}16)$$

2.2 Evaluation of Transmissibilities

Hence, we will derive a useful formula for a mean harmonic value of KA product. Supposing the transmissibility is piecewise constant with interface between (i) and (i-1) blocks (Fig.2.2).



(Fig.2.2).

Then the flow rate between block (i) and block (i-1) is

$$\overline{Q}_p = \left(\frac{\lambda_p}{B_p}\right)_{avg} \overline{KA} \; \frac{P_{i-1} - P_i}{\frac{\Delta x_{i-1} + \Delta x_i}{2}} \qquad (2\text{-}17)$$

where $\left(\frac{\lambda_p}{B_p}\right)_{avg}$ is calculated from Eq.(2.16). $\overline{KA}$ is an equivalent permeability-cross sectional area product for block (i-1) and block (i).

Applying Darcy's law to each shaded volume element with homogeneous physical properties:

$$\overline{Q}_p = \left(\frac{\lambda_p}{B_p}\right)_{avg} (K_x A)_{i-1} \frac{P_{i-1} - P_f}{\frac{\Delta x_{i-1}}{2}} \qquad (2\text{-}18)$$

and

$$\overline{Q}_p = \left(\frac{\lambda_p}{B_p}\right)_{avg} (K_x A)_i \frac{P_f - P_i}{\frac{\Delta x_i}{2}} \qquad (2\text{-}19)$$

where

P_f - the pressure at the interface.

Elimination of P_f from Eqs.(2.18) and (2.19) yields

$$P_{i-1} - P_i = \frac{1}{2}\bar{Q}_P\left(\frac{\beta_P}{\lambda_P}\right)_{avg}\left[\frac{\Delta x_{i-1}}{(K_xA)_{i-1}} + \frac{\Delta x_i}{(K_xA)_i}\right] \quad . \quad . \quad . \quad . \quad (2\text{-}20)$$

Substitution of Eq.(2.20) into Eq.(2.17) and solving it yields KA as

$$\overline{KA} = \frac{\Delta x_i + \Delta x_{i-1}}{\frac{\Delta x_i}{(K_xA)_i} + \frac{\Delta x_{i-1}}{(K_xA)_{i-1}}} \quad . \quad . \quad . \quad (2\text{-}21)$$

which is equivalent to $(K_xA)_{i-\frac{1}{2}}$

Similarly $(K_xA)_{i+\frac{1}{2}}$ can be determined using Eq.(2.21).

$$(K_xA)_{i+\frac{1}{2}} = \frac{\Delta x_i + \Delta x_{i+1}}{\frac{\Delta x_i}{(K_xA)_i} + \frac{\Delta x_{i+1}}{(K_xA)_{i+1}}} \quad \ldots \quad (2\text{-}22)$$

Substituting Eqs.(2.16), (2.21) and (2.22) for $\left(\frac{\lambda_p}{B_p}\right)$ and KA in Eq.(2.15), we obtain

$$(\Delta x_i \Delta y_i \Delta z_i)\frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial x}\right] = \frac{4\,k_{rp,upstream}}{(\mu_{p,i}+\mu_{p,i+1})(B_{p,i}+B_{p,i+1})}$$

$$\frac{2(K_xA)_i(K_xA)_{i+1}(P_{i+1}-P_i)}{\Delta x_i(K_xA)_{i+1}+\Delta x_{i+1}(K_xA)_i} - \frac{4\,k_{rp,upstream}}{(\mu_{p,i}+\mu_{p,i-1})(B_{p,i}+B_{p,i-1})}$$

$$\frac{2(K_xA)_i(K_xA)_{i-1}(P_i-P_{i-1})}{\Delta x_i(K_xA)_{i-1}+\Delta x_{i-1}(K_xA)_i} \quad \ldots \quad (2\text{-}23)$$

$$\Delta x_i \Delta y_i \Delta z_i = V_B \quad \ldots \quad (2\text{-}24)$$

$$A_{p,i+\frac{1}{2}} = \frac{4\,k_{rp,upstream}}{(\mu_{p,i}+\mu_{p,i+1})(B_{p,i}+B_{p,i+1})} \cdot$$

$$\frac{2(K_xA)_i\,(K_xA)_{i+1}}{\Delta x_i(K_xA)_{i+1}+\Delta x_{i+1}(K_xA)_i} \quad . \quad . \quad . \quad (2\text{-}25)$$

$$A_{p,i-\frac{1}{2}} = \frac{4\,k_{rp,upstream}}{(\mu_{p,i}+\mu_{p,i-1})(B_{p,i}+B_{p,i+1})} \cdot$$

$$\cdot \frac{2(K_xA)_i\,(K_xA)_{i-1}}{\Delta x_i(K_xA)_{i-1}+\Delta x_{i-1}(K_xA)_i} \quad . \quad . \quad . \quad (2\text{-}26)$$

$A_{p,i+\frac{1}{2}}$ and $A_{p,i-\frac{1}{2}}$ are called the '<u>finite difference transmissibilities</u>' between (i) and (i + 1), and between (i) and (i − 1) blocks. Substituting Eqs.(2.24), (2.25) and (2.26) into Eq:(2.23), we obtain

$$V_B \frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial x}\right] = A_{p,i+\frac{1}{2}}(P_{i+1}-P_i) +$$

$$A_{p,i-\frac{1}{2}}(P_{i-1}-P_i)$$

$$. \quad . \quad . \quad . \quad . \quad . \quad (2\text{-}27)$$

Using a linear difference operator to express Eq.(2.27) in a compact form,

$$V_B \cdot \frac{\partial}{\partial x}\left[K_x\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial x}\right] = \Delta_x A_x \Delta_x P$$

$$= A_{p,i+\frac{1}{2}}(P_{i+1} - P_i) + A_{p,i-\frac{1}{2}}(P_{i-1} - P_i) \quad . \quad . \quad . \quad . \quad . \quad (2\text{-}28)$$

Similarly, we can write for

$$V_B \cdot \frac{\partial}{\partial y}\left[K_y\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial y}\right] \quad \text{and} \quad V_B \cdot \frac{\partial}{\partial z}\left[K_z\left(\frac{\lambda_p}{B_p}\right)\frac{\partial P}{\partial z}\right]$$

using Eq.(2.9),

$$V_B \cdot \left[\nabla \cdot \overleftrightarrow{K}\left(\frac{\lambda_p}{B_p}\right)\nabla P\right] = \Delta_x A_x \Delta_x P + \Delta_y A_y \Delta_y P +$$

$$\Delta_z A_z \Delta_z P = \Delta A \Delta P \quad . \quad . \quad . \quad . \quad (2\text{-}29)$$

2.3 The IMPES Procedure to Solve the Pressure Equation

Using the above notation, we can write the pressure equation of Eq.(1.62) as:

$$\left(B_o^n - B_g^n R_{so}^n\right)_{ijk}\left(\Delta A_o^n \Delta P^{n+1} + GOWT - \frac{q_o V_B}{\rho_{osc}}\right)_{ijk}$$

$$+\left(B_w^n - B_g^n R_{sw}^n\right)_{ijk}\left(\Delta A_w^n \Delta P^{n+1} + GWWT - \frac{q_w V_B}{\rho_{wsc}}\right)_{ijk}$$

$$+\left(B_g^n\right)_{ijk}\left(\Delta A_g^n \Delta P^{n+1} + \Delta R_{so}^n A_o^n \Delta P^{n+1} + \Delta R_{sw}^n A_w^n \Delta P^{n+1}\right.$$

$$\left. + G\dot{G}WT - \frac{q_g V_B}{\rho_{gsc}}\right)_{ijk} = = \left(\frac{V_p^n c_t^n}{\Delta t}\right)_{ijk}\left(P_{ijk}^{n+1} - P_{ijk}^n\right)$$

. . . . . (2-30)

where

$$V_p^n = (\Delta x_i \Delta y_i \Delta z_i)\, \phi_{ijk}^n \qquad (2\text{-}31)$$

$$GOWT = V_B \cdot \left[-\nabla \cdot \overleftrightarrow{K} \left(\frac{\lambda_o}{B_o} \right) \nabla (\rho_o z) \right]$$

$$= -\Delta \overset{n}{A}_o \Delta (\rho_o z)^n \qquad (2\text{-}32)$$

$$GWWT = V_B \cdot \left[-\nabla \cdot \overleftrightarrow{K} \left(\frac{\lambda_w}{B_w} \right) \nabla (\rho_w z + P_{cow}) \right]$$

$$= -\Delta \overset{n}{A}_w \Delta (\rho_w z + P_{cow})^n \qquad (2\text{-}33)$$

$$GGWT = V_B \cdot \left[\nabla \cdot \overleftrightarrow{K} \left\{ \frac{\lambda_g}{B_g} \nabla (P_{cgo} - \rho_g z) - \frac{R_{so} \lambda_o}{B_o} \nabla (\rho_o z) \right.\right.$$

$$\left.\left. - \frac{R_{sw} \lambda_w}{B_w} \nabla (P_{cow} + \rho_w z) \right\} \right]$$

$$= \Delta \left[\overset{n}{A}_g \Delta (P_{cgo} - \rho_g z)^n - \overset{n}{R}_{so} \overset{n}{A}_o \Delta (\rho_o z)^n - \right.$$

$$\left. \overset{n}{R}_{sw} \overset{n}{A}_w \Delta (P_{cow} + \rho_w z) \right] \qquad (2\text{-}34)$$

Let

$$QOWG = \left(B_o^n - B_g^n R_{so}^n\right)_{ijk}\left(\frac{q_o V_B}{\varrho_{osc}} - GOWT\right)_{ijk}$$

$$+\left(B_w^n - B_g^n R_{sw}^n\right)_{ijk}\left(\frac{q_w V_B}{\varrho_{wsc}} - GWWT\right)_{ijk} +$$

$$\left(B_g^n\right)_{ijk}\left(\frac{q_g V_B}{\varrho_{gsc}} - GGWT\right)_{ijk} \quad . \quad . \quad . \quad . \quad (2\text{-}35)$$

Substituting Eq.(2.35) in Eq.(2.30), we obtain

$$\left(B_o^n - B_g^n R_{so}^n\right)_{ijk}\left(\Delta A_o^n \Delta P^{n+1}\right)_{ijk} + \left(B_w^n - B_g^n R_{sw}^n\right)_{ijk}$$

$$\left(\Delta A_w^n \Delta P^{n+1}\right)_{ijk} + \left(B_g^n\right)_{ijk}\left(\Delta A_g^n \Delta P^{n+1} + \Delta R_{so}^n A_o^n \Delta P^{n+1}\right.$$

$$\left. + \Delta R_{sw}^n A_w^n \Delta P^{n+1}\right)_{ijk} = QOWG + \left(\frac{V_p^n c_t^n}{\Delta t}\right)_{ijk}\left(P_{ijk}^{n+1} - P_{ijk}^n\right)$$

$$. \quad . \quad . \quad . \quad . \quad (2\text{-}36)$$

Let

$$B = QOWG - \left(\frac{V_p^n c_t^n}{\Delta t}\right)_{ijk} P_{ijk}^n \quad . \quad . \quad . \quad (2\text{-}37)$$

Substituting Eq.(2.37) in Eq.(2.36) and rearranging it gives

$$\left(B_o^n - B_g^n R_{so}^n\right)_{ijk}\left(\Delta A_o^n \Delta P^{n+1}\right)_{ijk} + \left(B_w^n - B_g^n R_{sw}^n\right)_{ijk}$$

$$\left(\Delta A_w^n \Delta P^{n+1}\right)_{ijk} + \left(B_g^n\right)_{ijk}\left(\Delta A_g^n \Delta P^{n+1} + \Delta R_{so}^n A_o^n \Delta P^{n+1}\right.$$

$$\left. + \Delta R_{sw}^n A_w^n \Delta P^{n+1}\right)_{ijk} - \left(\frac{V_p^n c_t^n}{\Delta t}\right)_{ijk} P_{ijk}^{n+1} = B$$

$$. \quad . \quad . \quad (2\text{-}38)$$

Now, we would expand the left-hand side terms of Eq.(2.38). To simplify the expansion, let us consider for x-direction expansion only.

$$\left(B_o - B_g R_{so}\right)_{ijk} \Delta_x A_{ox} \Delta_x P + \left(B_w - B_g R_{sw}\right)_{ijk} \Delta_x A_{wx} \Delta_x P$$

$$+ \left(B_g\right)_{ijk}\left(\Delta_x A_{gx} \Delta_x P + \Delta_x R_{so,x} A_{ox} \Delta_x P + \Delta_x R_{sw,x} A_{wx} \Delta_x P\right)$$

$$= \left(B_o - B_g R_{so}\right)_{ijk}\left[A_{o,i+\frac{1}{2}}\left(P_{i+1} - P_{ijk}\right) + A_{o,i-\frac{1}{2}}\left(P_{i-1} - P_{ijk}\right)\right]$$

$$+ \left(B_w - B_g R_{sw}\right)_{ijk}\left[A_{w,i+\frac{1}{2}}\left(P_{i+1} - P_{ijk}\right) + A_{w,i-\frac{1}{2}}\left(P_{i-1} - P_{ijk}\right)\right] +$$

$$(B_g)_{ijk}\left[A_{g,i+\frac{1}{2}}(P_{i+1}-P_{ijk})+A_{g,i-\frac{1}{2}}(P_{i-1}-P_{ijk})+\right.$$

$$R_{so,i+\frac{1}{2}}A_{o,i+\frac{1}{2}}(P_{i+1}-P_{ijk})+R_{so,i-\frac{1}{2}}A_{o,i-\frac{1}{2}}(P_{i-1}-P_i)$$

$$\left.+R_{sw,i+\frac{1}{2}}A_{w,i+\frac{1}{2}}(P_{i+1}-P_{ijk})+R_{sw,i-\frac{1}{2}}A_{w,i-\frac{1}{2}}(P_{i-1}-P_{ijk})\right] \quad (2\text{-}39)$$

Collecting the terms for P_{i+1} only from Eq.(2.39),

$$(B_o-B_gR_{so})_{ijk}A_{o,i+\frac{1}{2}}+(B_w-B_gR_{sw})_{ijk}A_{w,i+\frac{1}{2}}+$$

$$(B_g)_{ijk}A_{g,i+\frac{1}{2}}+(B_g)_{ijk}R_{so,i+\frac{1}{2}}A_{o,i+\frac{1}{2}}+(B_g)_{ijk}R_{sw,i+\frac{1}{2}}A_{w,i+\frac{1}{2}}$$

$$=\left[(B_o-B_gR_{so})_{ijk}+(B_g)_{ijk}R_{so,i+\frac{1}{2}}\right]A_{o,i+\frac{1}{2}}+$$

$$\left[(B_w-B_gR_{sw})_{ijk}+(B_g)_{ijk}R_{sw,i+\frac{1}{2}}\right]A_{w,i+\frac{1}{2}}+$$

$$(B_g)_{ijk}A_{g,i+\frac{1}{2}} =$$

$$\left[B_o-B_gR_{so,i}+B_g\left(\frac{R_{so,i+1}+R_{so,i}}{2}\right)\right]A_{o,i+\frac{1}{2}}+$$

$$\left[B_w-B_gR_{sw,i}+B_g\left(\frac{R_{sw,i+1}+R_{sw,i}}{2}\right)\right]A_{w,i+\frac{1}{2}}+$$

$$B_gA_{g,i+\frac{1}{2}} =$$

$$\left[B_o+\frac{B_g}{2}\left(R_{so,i+1}-R_{so,i}\right)\right]A_{o,i+\frac{1}{2}}+\left[B_w+\frac{B_g}{2}\left(R_{sw,i+1}\right.\right.$$

$$\left.\left.-R_{sw,i}\right)\right]A_{w,i+\frac{1}{2}}+B_gA_{g,i+\frac{1}{2}} \quad . \quad . \quad . \quad (2\text{-}40)$$

Referring to (Fig.2.3), we can imagine that the block (i + 1, j, k) lies to the east of block (i, j, k). Similarly the blocks (i, j + 1, k), (i − 1, j, k) and (i, j − 1, k) lie respectively to the north, west and south of the centre block (i, j, k). Blocks (i, j, k − 1) and (i, j, k + 1) lie top and bottom of the centre block (i, j, k) respectively. Thus, Eq.(2.40) can be expressed, in short, as AE_i.

$$AE_i = \left[B_o + \frac{B_g}{2}\left(R_{so,i+1} - R_{so,i}\right)\right] A_{o,i+\frac{1}{2}} +$$

$$\left[B_w + \frac{B_g}{2}\left(R_{sw,i+1} - R_{sw,i}\right)\right] A_{w,i+\frac{1}{2}} + B_g A_{g,i+\frac{1}{2}} \quad \cdots \quad (2\text{-}41)$$

Similarly we can write for AW_i, AN_j, AS_j, AT_k as follows:

$$AW_i = \left[B_o + \frac{B_g}{2}\left(R_{so,i-1} - R_{so,i}\right)\right] A_{o,i-\frac{1}{2}} +$$

$$\left[B_w + \frac{B_g}{2}\left(R_{sw,i-1} - R_{sw,i}\right)\right] A_{w,i-\frac{1}{2}} + B_g A_{g,i-\frac{1}{2}} \quad \cdots \quad (2\text{-}42)$$

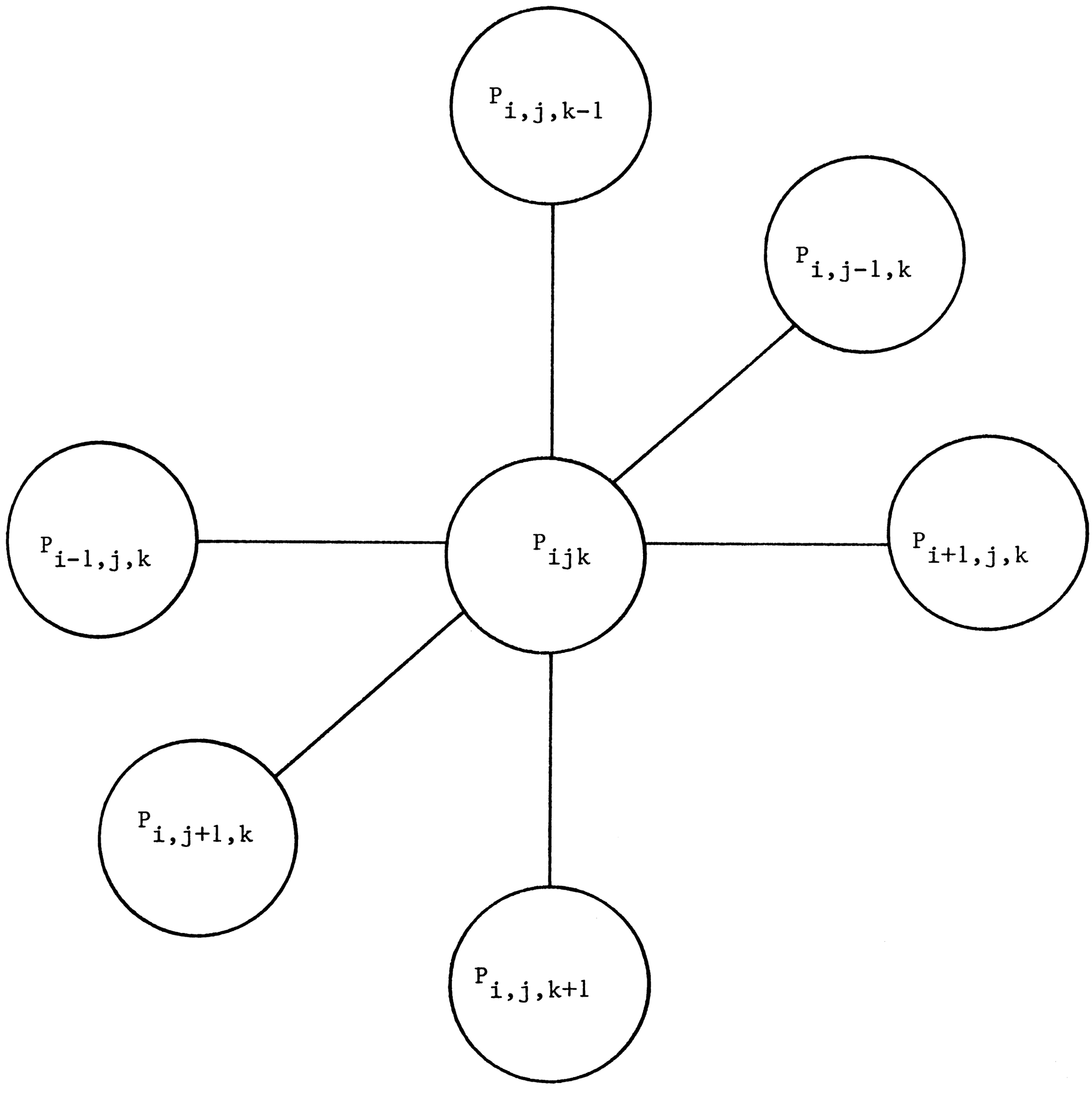


FIGURE (2.3)

$$AN_j = \left[B_o + \frac{B_g}{2}\left(R_{so,j+1} - R_{so,j}\right)\right] A_{o,j+\frac{1}{2}} +$$

$$\left[B_w + \frac{B_g}{2}\left(R_{sw,j+1} - R_{sw,j}\right)\right] A_{w,j+\frac{1}{2}} + B_g A_{g,j+\frac{1}{2}} \quad \cdot \quad (2\text{-}43)$$

$$AS_j = \left[B_o + \frac{B_g}{2}\left(R_{so,j-1} - R_{so,j}\right)\right] A_{o,j-\frac{1}{2}} +$$

$$\left[B_w + \frac{B_g}{2}\left(R_{sw,j-1} - R_{sw,j}\right)\right] A_{w,j-\frac{1}{2}} + B_g A_{g,j-\frac{1}{2}} \quad \cdot \quad (2\text{-}44)$$

$$AT_k = \left[B_o + \frac{B_g}{2}\left(R_{so,k-1} - R_{so,k}\right)\right] A_{o,k-\frac{1}{2}} +$$

$$\left[B_w + \frac{B_g}{2}\left(R_{sw,k-1} - R_{sw,k}\right)\right] A_{w,k-\frac{1}{2}} + B_g A_{g,k-\frac{1}{2}} \quad \cdot \quad (2\text{-}45)$$

$$AB_k = \left[B_o + \frac{B_g}{2}\left(R_{so,k+1} - R_{so,k}\right)\right] A_{o,k+\frac{1}{2}} +$$

$$\left[B_w + \frac{B_g}{2}\left(R_{sw,k+1} - R_{sw,k}\right)\right] A_{w,k+\frac{1}{2}} + B_g A_{g,k+\frac{1}{2}} \quad (2\text{-}46)$$

All of the quantities in Eqs.(2.41) to (2.46) are evaluated at the present (n) time level and many subscripts have been suppressed. Then, the pressure equation of Eq.(2.38) becomes:

$$AT_k P_{k-1}^{n+1} + AS_j P_{j-1}^{n+1} + AW_i P_{i-1}^{n+1} + AB_k P_{k+1}^{n+1} +$$

$$AN_j P_{j+1}^{n+1} + AE_i P_{i+1}^{n+1} - (AT_k + AS_j + AW_i + AB_k$$

$$+ AN_j + AE_i + \frac{V_p^n c_t^n}{\Delta t}) P_{ijk}^{n+1} = B \quad . \quad . \quad . \quad (2-47)$$

$$E = -(AT_k + AS_j + AW_i + AB_k + AN_j + AE_i$$

$$+ \frac{V_p^n c_t^n}{\Delta t}) \quad . \quad . \quad . \quad . \quad . \quad . \quad (2-48)$$

Finally, Eq.(2.47) becomes:

$$AT_k P_{k-1}^{n+1} + AS_j P_{j-1}^{n+1} + AW_i P_{i-1}^{n+1} + AB_k P_{k+1}^{n+1} +$$

$$AN_j P_{j+1}^{n+1} + AE_i P_{i+1}^{n+1} + E P_{ijk}^{n+1} = B \quad . \quad . \quad . \quad (2-49)$$

The total system of algebraic equations for the pressures at the new (n + 1) time level can be set for each block of the model grid using Eq.(2.49) because the coefficients of the (n + 1) time level pressures in Eq.(2.49) are all known. Different methods to solve these sets of simultaneous equations are presented in the next chapter. When the new pressures have been computed, they are used in oil and water phase saturation equations to find S_o^{n+1} and S_w^{n+1}.

2.4 Phase Saturation Equations

OIL

$$\left(\Delta A_o^n \Delta P^{n+1} + GOWT - \frac{q_o V_B}{\rho_{osc}} \right)_{ijk} =$$

$$\frac{1}{\Delta t}\left[\left(\frac{V_p S_o}{B_o}\right)^{n+1}_{ijk} - \left(\frac{V_p S_o}{B_o}\right)^{n}_{ijk}\right] \quad . \quad . \quad . \quad . \quad (2\text{-}50)$$

WATER

$$\left(\Delta A_w^n \Delta P^{n+1} + GWWT - \frac{q_w V_B}{\rho_{wsc}} \right)_{ijk} =$$

$$\frac{1}{\Delta t}\left[\left(\frac{V_p S_w}{B_w}\right)^{n+1}_{ijk} - \left(\frac{V_p S_w}{B_w}\right)^{n}_{ijk}\right] \quad . \quad . \quad . \quad . \quad (2\text{-}51)$$

$$\left(S_g \right)^{n+1}_{ijk} = 1 - \left(S_o \right)^{n+1}_{ijk} - \left(S_w \right)^{n+1}_{ijk} \quad . \quad . \quad (2\text{-}52)$$

Whenever the new pressures and saturation have been computed, these values are then considered the present values and the calculation is repeated till the end of simulation. In this way, the approximate numerical solution of the block oil simulator (three-phase, two-phase) and aquifer model may be obtained for an arbitarily long time. This procedure is known as the implicit pressure - explicit saturation (IMPES) method.

CHAPTER 3: SOLUTION METHODS FOR SOLVING SIMULTANEOUS EQUATIONS

Introduction

When Eq.(2.49) is applied at every grid point in the solution space of dimension II*JJ*KK number of blocks in the x, y, z - directions respectively (evaluating coefficient a_{ij} at time level n for the IMPES method), it gives rise to a linear system of equations of the form:

$$
\begin{aligned}
a^n_{1,1} P^{n+1}_1 + a^n_{1,2} P^{n+1}_2 + a^n_{1,3} P^{n+1}_3 + \cdots + a^n_{1,N} P^{n+1}_N &= B^n_1 \\
a^n_{2,1} P^{n+1}_1 + a^n_{2,2} P^{n+1}_2 + a^n_{2,3} P^{n+1}_3 + \cdots + a^n_{2,N} P^{n+1}_N &= B^n_2 \\
\cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots & \\
a^n_{N,1} P^{n+1}_1 + a^n_{N,2} P^{n+1}_2 + a^n_{N,3} P^{n+1}_3 + \cdots + a^n_{N,N} P^{n+1}_N &= B^n_N
\end{aligned}
\qquad (3\text{-}1)
$$

where

N = II*JJ*KK

n, n + 1 = old and new time levels respectively.

Using matrix notation to express Eq.(3.1) in a compact form, let

$$\underline{A} = \begin{bmatrix} a^n_{1,1} & a^n_{1,2} & a^n_{1,3} & \cdots & a^n_{1,N} \\ a^n_{2,1} & a^n_{2,2} & a^n_{2,3} & \cdots & a^n_{2,N} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a^n_{N,1} & a^n_{N,2} & a^n_{N,3} & \cdots & a^n_{N,N} \end{bmatrix} \quad \ldots \quad (3\text{-}2)$$

$$\underline{P} = \begin{bmatrix} P_1^{n+1} \\ P_2^{n+1} \\ \cdots \\ P_N^{n+1} \end{bmatrix} \qquad \underline{B} = \begin{bmatrix} B_1^n \\ B_2^n \\ \cdots \\ B_N^n \end{bmatrix} \quad \ldots \quad (3\text{-}3)$$

Then Eq.(3.1) can be written as:

$$\underline{A}\ \underline{P} = \underline{B} \quad \ldots \qquad (3.4)$$

where

$\underline{A}$ = coefficient matrix (evaluated at time level n for the IMPES when solving pressure equation (2.49))

$\underline{P}$ = solution vector

$\underline{B}$ = known column vector

Various methods exist for solving Eq.(3.4) but generally there are only two basic methods namely: direct method and iterative method, each of which has its particular advantages and disadvantages.

3.1 Direct Method

Direct methods are those in which the solution to Eq.(3.4) is obtained after completion of a fixed number of operations. One of the well known direct method is Gaussian elimination. The basic idea of this method is to reduce the system of n equations in n unknowns of Eq.(3.1) to a system of (n − 1) equations in (n − 1) unknowns first; then it is again reduced to the system of (n − 2) equations in (n − 2) unknowns. This process is repeated until finally one equation in one unknown is obtained. Thus, this one unknown is determined. The remaining unknowns are determined by back substitution. Actually, this process performs the necessary row eliminations in such a way that matrix $\underline{A}$ transforms to an upper triangular matrix, which is easy to solve for $\underline{P}$ by backward substitution. Some variations of Gaussian method include Gauss-Jordan reduction that avoids the step of back substitution. Another technique is LU decomposition which involves factorization of matrix $\underline{A}$ into lower and upper traingular matrices $\underline{L}$ and $\underline{U}$ so that $\underline{P}$ can be determined in the following sequence of steps:

(1) Factorization

$$\underline{L} = \left[\ell_{ij} \right] \quad \text{and} \quad \underline{U} = \left[u_{ij} \right]$$

are determined in this step, where

$$\ell_{ij} = 0 \quad \text{for} \quad i < j \quad \text{and} \quad u_{ij} = 0 \quad \text{for} \quad i > j$$

(2) Solve $\underline{L}\ \underline{Z} = \underline{B}$ for $\underline{Z}$ using the forward substitution.

(3) Finally solve $\underline{U}\ \underline{P} = \underline{Z}$ for $\underline{P}$ using the backward substitution.

There are many ways to factor matrix $\underline{A}$, but a common way to factor $\underline{A}$ is to choose the diagonal elements for $\underline{U}$ be 1's. This procedure is called Crout's method (9). A number of solution techniques employed in reservoir simulators, both direct and iterative, have their basis in the LU decomposition concept. The reason this method is popular in programs is that storage space may be economized. Thee is no need to store the zeros in either L or U, and the 1's on the diagonal of U can also be omitted since these values are the same and known. We can store the essential elements of U where zeros appear in the L array. More importantly, after any element, a_{ij}, of $\underline{A}$ is once used, it never again appears in the equations, thus $\underline{A}$ can be used to store an element of either L or U.

$$\begin{bmatrix} a^{n}_{1,1} & a^{n}_{1,2} & \cdots & a^{n}_{1,N} \\ a^{n}_{2,1} & a^{n}_{2,2} & \cdots & a^{n}_{2,N} \\ \cdot & \cdot & \cdot & \cdot \\ a^{n}_{N,1} & a^{n}_{N,2} & \cdots & a^{n}_{N,N} \end{bmatrix} \Longrightarrow \begin{bmatrix} \ell_{1,1} & u_{1,2} & \cdots & u_{1,N} \\ \ell_{2,1} & \ell_{2,2} & \cdots & u_{2,N} \\ \cdot & \cdot & \cdot & \cdot \\ \ell_{N,1} & \ell_{N,2} & \cdots & \ell_{N,N} \end{bmatrix}$$

The efficiency of direct solution methods, however, depends on the ordering of unknown points $\underline{P}$ for the model grid that would give rise to different appearance of band width 'W' for $\underline{A}$. Because of the fact that amount of computational work involved in factorising $\underline{A}$ is strongly dependent on the size of W, it is essential to label the unknown $\underline{P}$ so as to minimise W to the least.

We would now examine two types of ordering namely: natural ordering (BAND) and alternate diagonal ordering (D4). Fig.(3.1) shows a natural ordering of points for a three-dimensional (4 * 3 * 2) grid, which generates a banded matrix $\underline{A}$ after Eq.(2.49) is applied at every grid point in the model (Fig.3.2) prescribing no flow on the boundary. If there are sources or sinks in the model, matrix $\underline{A}$ will be a little altered.

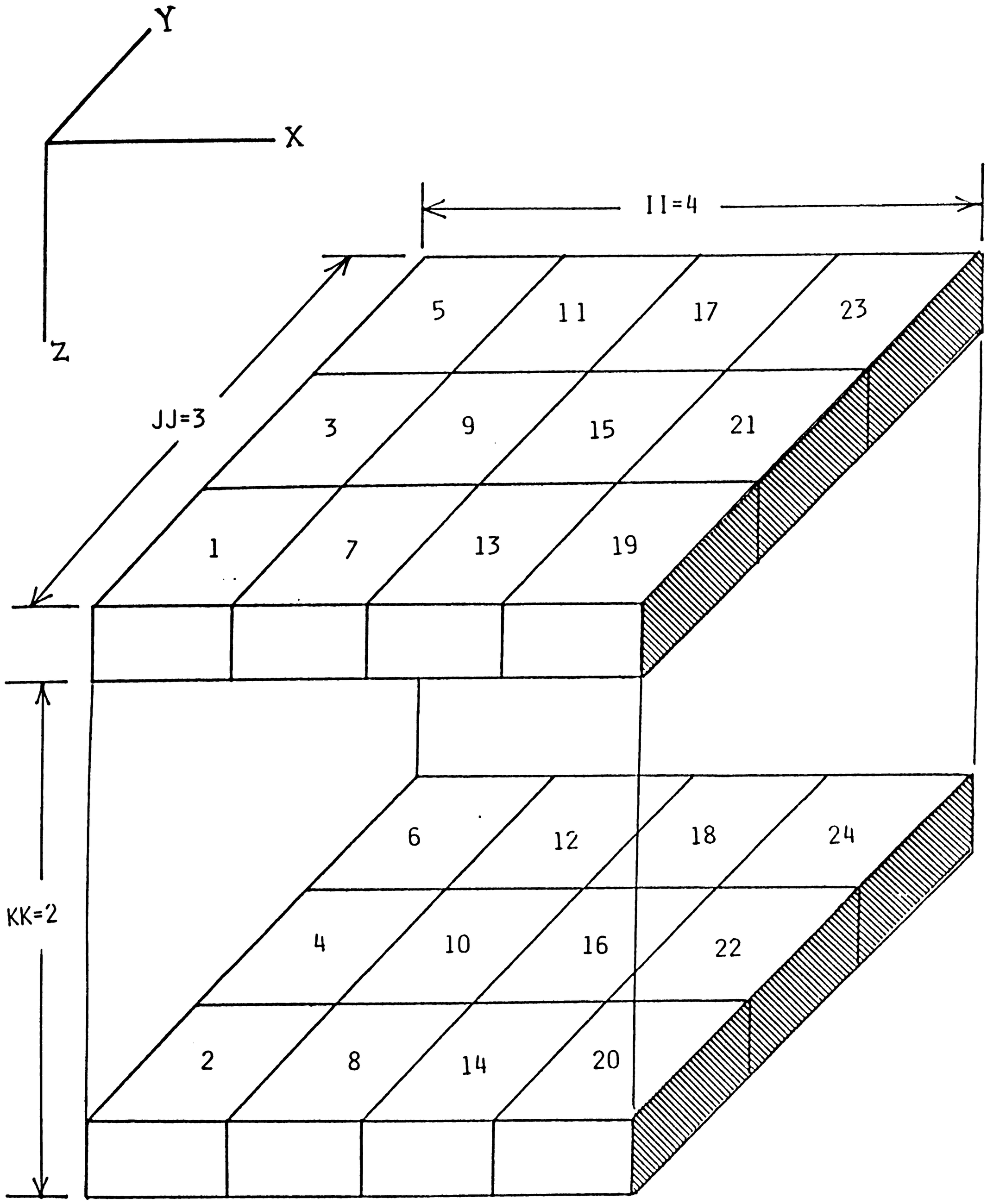

Y
X
Z
II=4
JJ=3
KK=2
5
11
17
23
3
9
15
21
1
7
13
19
6
12
18
24
4
10
16
22
2
8
14
20

FIGURE (3.1)

Upon inspection of matrix A, the maximum band width 'W' containing nonzero elements is found to be:

$$W_{max} = 2 * JJ * KK + 1$$

$$\text{i.e} \qquad W_{max} = 2*3*2 + 1 = 13 \qquad (3\text{-}5)$$

To reduce the size of W to the least, it is seen from Eq.(3.5) that the longest side of grid should be aligned in the x-direction which would maximise the efficiency of the BAND algorithm. The specific equations used in BAND are:

Factorization

$$\ell_{ij} = a_{ij} - \sum_{k=L_1(i)}^{j-1} \ell_{ik}\, u_{kj}$$

$$j = L_1, L_1+1, \ldots, i \qquad (3\text{-}6)$$

$$u_{ij} = \left(a_{ij} - \sum_{k=L_3(j)}^{i-1} \ell_{ik} u_{kj}\right) \Big/ \ell_{ii}$$

$$j = i+1, i+2, \ldots, L_2(i) \qquad (3\text{-}7)$$

where

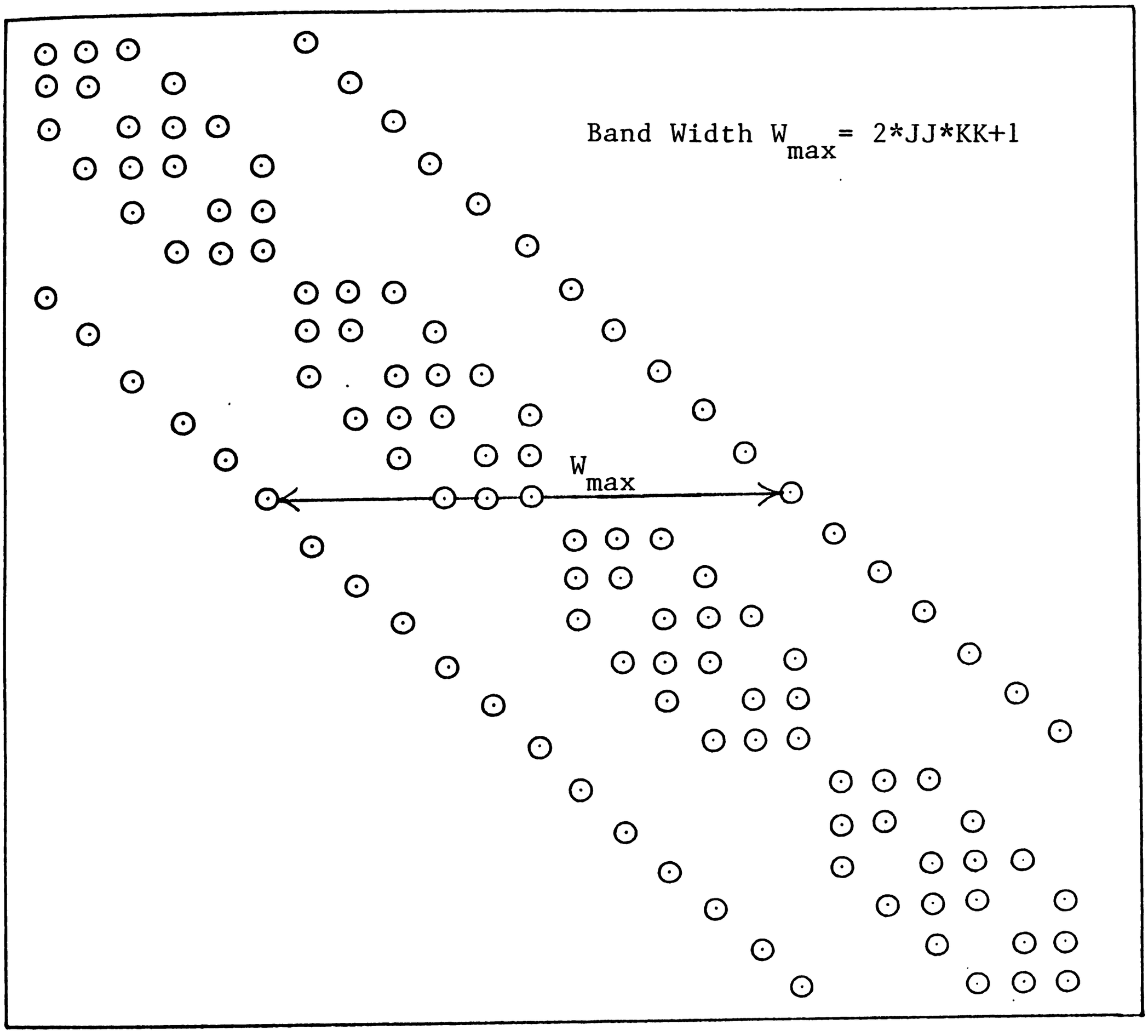


FIGURE (3.2)

$$\left.\begin{aligned} L_1(i) &= \max(i, i-(W-1)/2) \\ L_2(i) &= \max(N, i+(W-1)/2) \\ L_3(i) &= \max(1, j-(W-1)/2) \end{aligned}\right\} \quad \ldots \quad (3\text{-}8)$$

Forward Substitution

$$z_i = \left(B_i - \sum_{k=L_1(i)}^{i-1} \ell_{ik} z_k\right) / \ell_{ii}$$

$$i = 1,2,3, \ldots ,N \qquad (3\text{-}9)$$

Backward Substitution

$$P_i = z_i - \sum_{k=i+1}^{L_2(i)} u_{ik} P_k$$

$$i = N, N-1, \ldots ,2,1 \qquad (3\text{-}10)$$

Unlike natural ordering, the D4 ordering or alternate diagonal ordering (Fig.3.3), first employed by Price and Coats in the petroleum literature produces a substantial decrease in the subsequent matrix calculations. The concept of D4 ordering is such that any point P_{ijk} in a grid system is said to belong to a diagonal plane (m) if its indices satisfy the relation:

$$i + j + k = m \qquad \ldots(3.11)$$

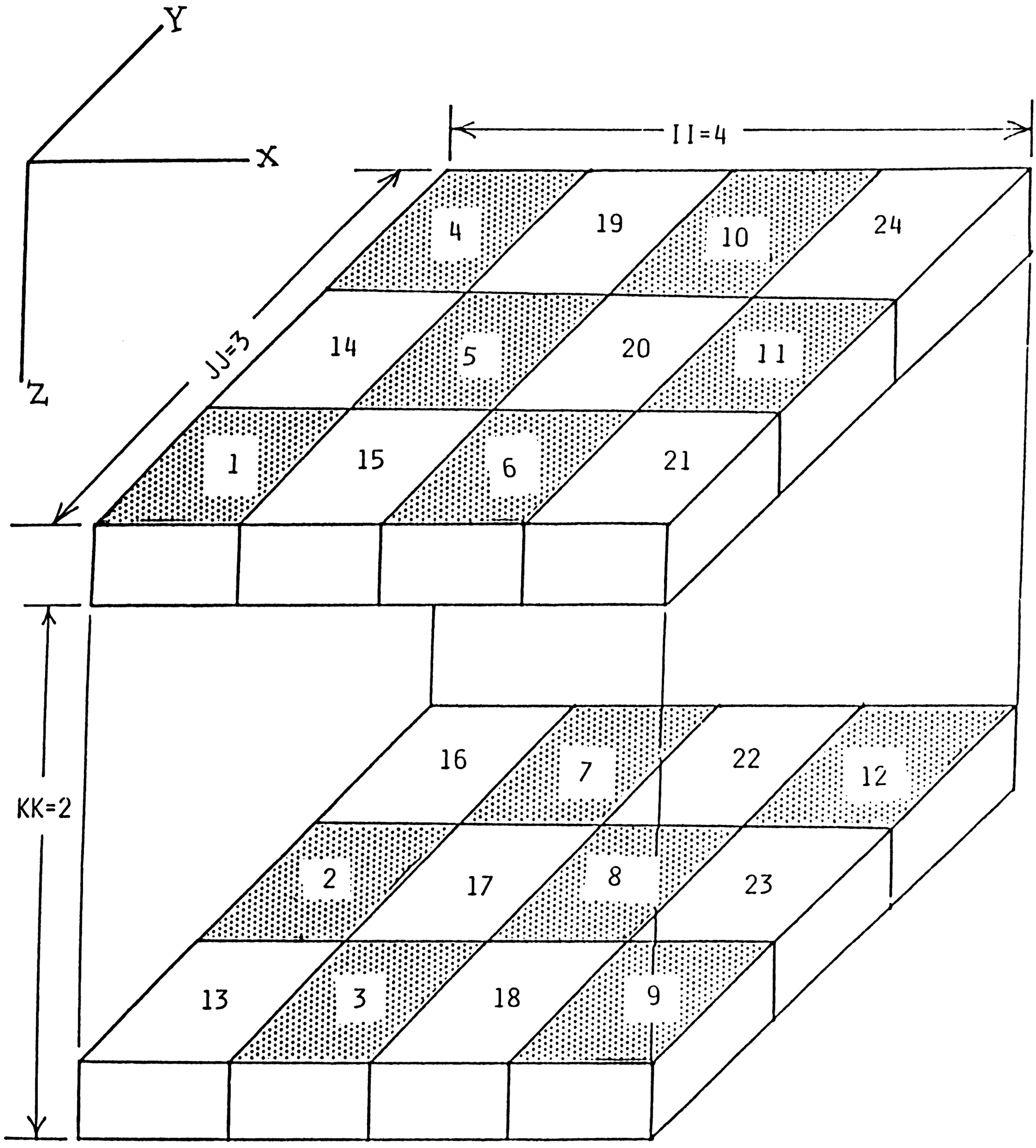

Y
X
Z
II=4
JJ=3
KK=2
4
19
10
24
14
5
20
11
1
15
6
21
16
7
22
12
2
17
8
23
13
3
18
9

FIGURE (3.3)

From this definition, it is evident that the smallest plane number is 3 while the largest number is M where M = II + JJ + KK

Now using D4 ordering to the 4*3*2 grid system, the new form of coefficient matrix A is developed as shown in Fig.(3.4). Upon inspection of Fig.(3.4), it is clearly seen that there are no nonzero terms below the main diagonal in the upper half portion of the matrix A. As a result, it reduces the total number of computations, especially when the matrix is large. However, D4 ordering might not be as efficient as natural ordering for small matrix A since some overhead computations are needed to set up the D4 ordering scheme.

In applying Gaussian elimination to matrix A, only left-hand quadrant needs to be zeroed out. In doing so, new nonzero elements are created in the lower right-hand quadrant. The upper half of matrix A would remain the same throughout the computation (Fig.3.5). To eliminate the left-hand quadrant below AA of Fig.(3.4), the program must be able to keep tracking where the lower diagonal block of nonzero entries is. This is accomplished by recognizing the following limits:

1. The lower diagonal block of nonzero entries starts at row (N + 1)/2 + 1 and ends at row N.

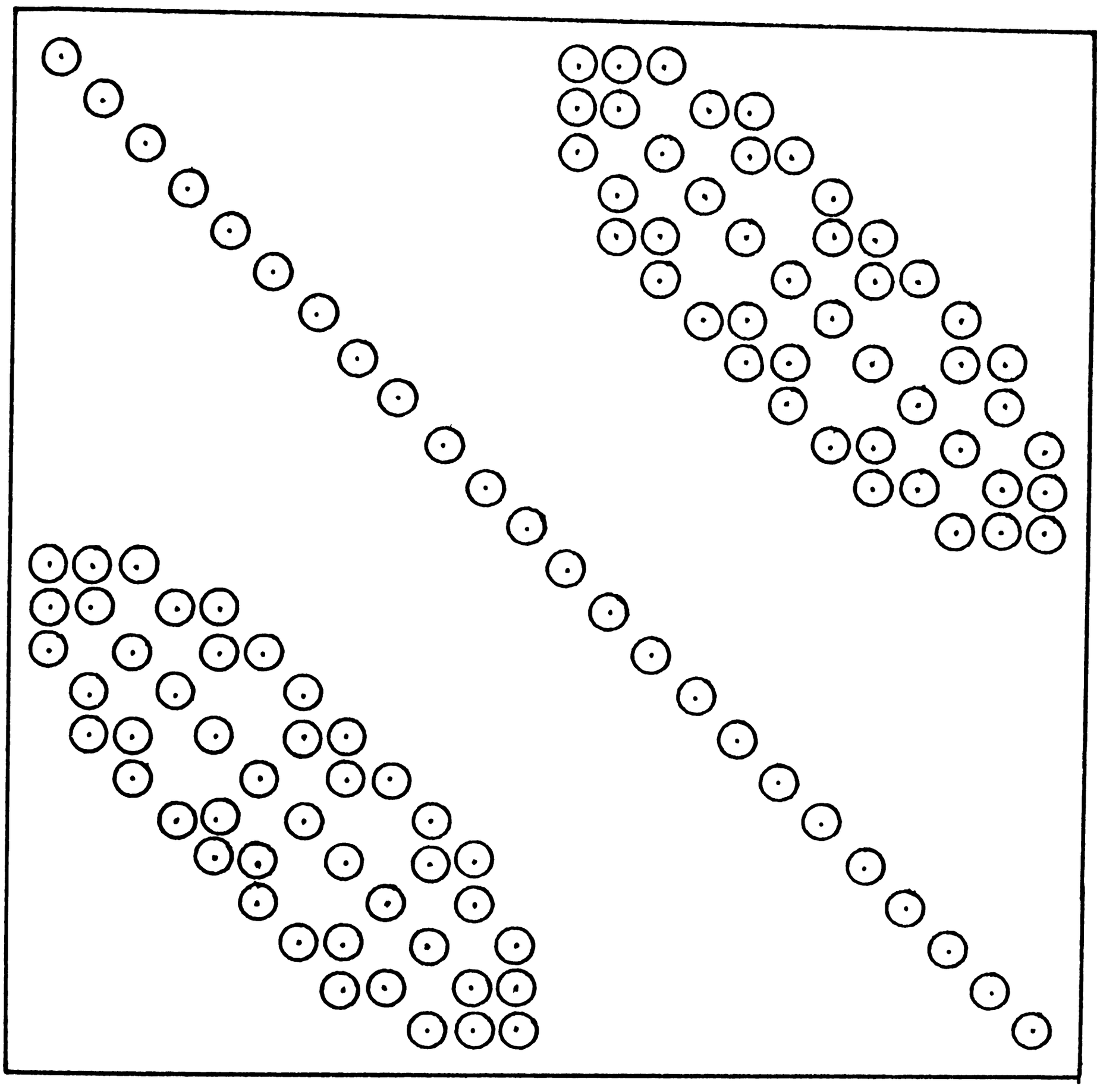


FIGURE (3.4)

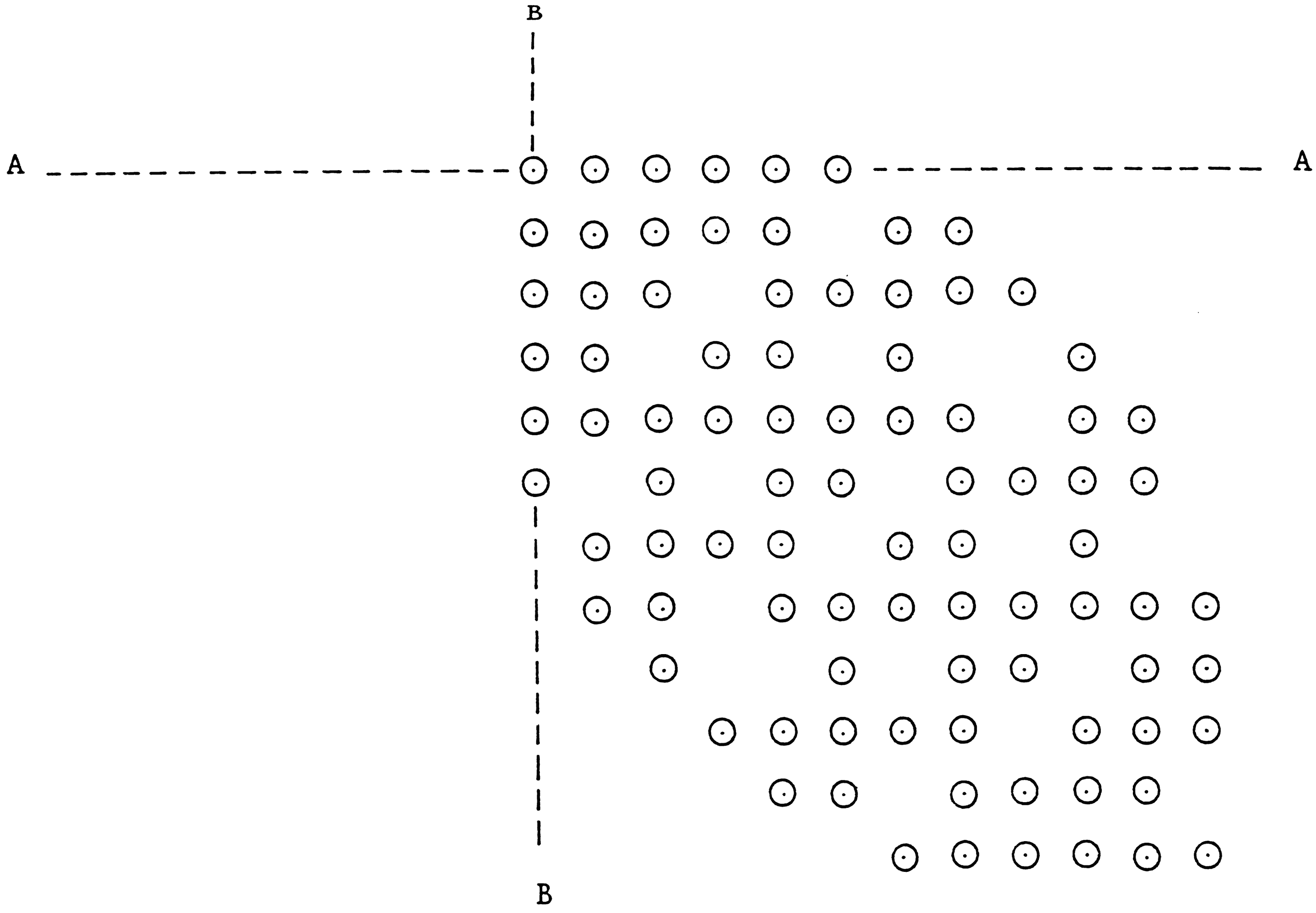


FIGURE (3.5)

2. The band width for this lower diagonal block is

   W_{lower} = JJ*KK + 1

3. There is a maximum of (JJ*KK+1)/2 + 1 nonzero entries in the first nonzero row of the lower diagonal block. For each subsequent row, the position of last nonzero entry in the row is increased by one.

After Gaussian elimination has been performed to zero-out the lower left-hand quadrant, we can solve for the unknown $\underline{P}$, starting from (N + 1)/2 + 1 to N, using a band width of 2*JJ*KK + 1 and BAND algorithm. Finally, backward substitution is used to solve for the remaining unknowns $\underline{P}$ above AA in Fig. (3.4); ie. P_1, P_2, P_3,....,$P_{(N-1)/2}$

3.2 Iterative Methods

Iterative methods are far most commonly used to solve a system of simultaneous equations in the reservoir simulators because they have distinct advantages over direct methods especially when N is very large and the system matrix is sparse. Iterative methods make use of an initial guess for the solution vector and iterates the system until some convergence criterion is reached. The iterative method, as opposed to the direct method of solving a set of simultaneous equations by elimination, is self-correcting if an error is made and has a considerable reduction of round-off error in the solution. More importantly, they do not require a large amount of computer storage.

Now considering the linear system given by Eq.(3.4), we would split the coefficient matrix $\underline{A}$ in such a way that

$$\underline{A} = \underline{D} - (\underline{D} - \underline{A}) \qquad \ldots(3.12)$$

where $\underline{D}$ is a diagonal matrix having the same diagonal entries as matrix $\underline{A}$.

Inserting Eq.(3.12) into Eq.(3.4) and rearranging yields

$$\underline{D}\ \underline{P} = (\underline{D} - \underline{A})\underline{P} + \underline{B} \qquad \ldots(3.13)$$

To begin our iterative procedure, we would label two time levels appropriately to the solution vector $\underline{P}$ so that we can substitute some initial guess into the right-hand side of Eq.(3.13) generating new approximations which are closer to the true values.

$$\underline{P}^{n+1} = \underline{D}^{-1}(\underline{D} - \underline{A})\underline{P}^{n} + \underline{D}^{-1}\underline{B} \qquad \ldots(3.14)$$

The new values are again substituted to Eq.(3.14) to generate a second approximation and the process is repeated until successive value of each of $\underline{P}$ are sufficiently alike. This iterative procedure is known as the point iterative Jacobi method. The algorithm for Jacobi iteration to compute each component of solution vector $\underline{P}$ can be outlined as follow:

$$P_i^{n+1} = \frac{B_i}{a_{ii}} - \sum_{\substack{j=1 \\ j\neq i}}^{N} \left(\frac{a_{ij}}{a_{ii}} P_j^n \right)$$

$$i = 1, 2, \ldots, N$$

$$n = 0, 1, 2, \ldots \qquad (3\text{-}15)$$

A sufficient condition for convergence is that

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{N} |a_{ij}| \qquad i = 1, 2, \ldots, N \qquad (3\text{-}16)$$

Jacobi method, however, does not make full use of the most recent estimates of $(P_i)^{n+1}$. In the above method, we calculated the second estimate of $(P_i)^{n+1}$ before we did the $(P_2)^{n+1}$, and new values of both $(P_1)^{n+1}$ and $(P_2)^{n+1}$ were available before we improved the value of $(P_3)^{n+1}$ In nearly all cases, the new values are better than the old, and should be used in preference to the proper values. When this is done, the method is known as the point iterative Gauss-Seidel method.

$$P_i^{n+1} = \frac{B_i}{a_{ii}} - \sum_{j=1}^{i-1} \left(\frac{a_{ij}}{a_{ii}} P_j^{n+1} \right) - \sum_{j=i+1}^{N} \left(\frac{a_{ij}}{a_{ii}} P_j^n \right) \qquad (3\text{-}17)$$

where

i = 1, 2, ..., N

n = 0, 1, 2,.....

Use of Eq.(3.17) can be applied (Peaceman (1977) for example) to a two-dimensional grid system for analysis. Eq. (3.4) can be written in another form as

$$d_{ij} P_{ij} - AW_{ij} P_{ij} - AE_{ij} P_{i+1,j} - AS_{ij} P_{i,j-1}$$
$$- AN_{ij} P_{i,j+1} = B_{ij} \quad . \quad . \quad . \quad (3-18)$$

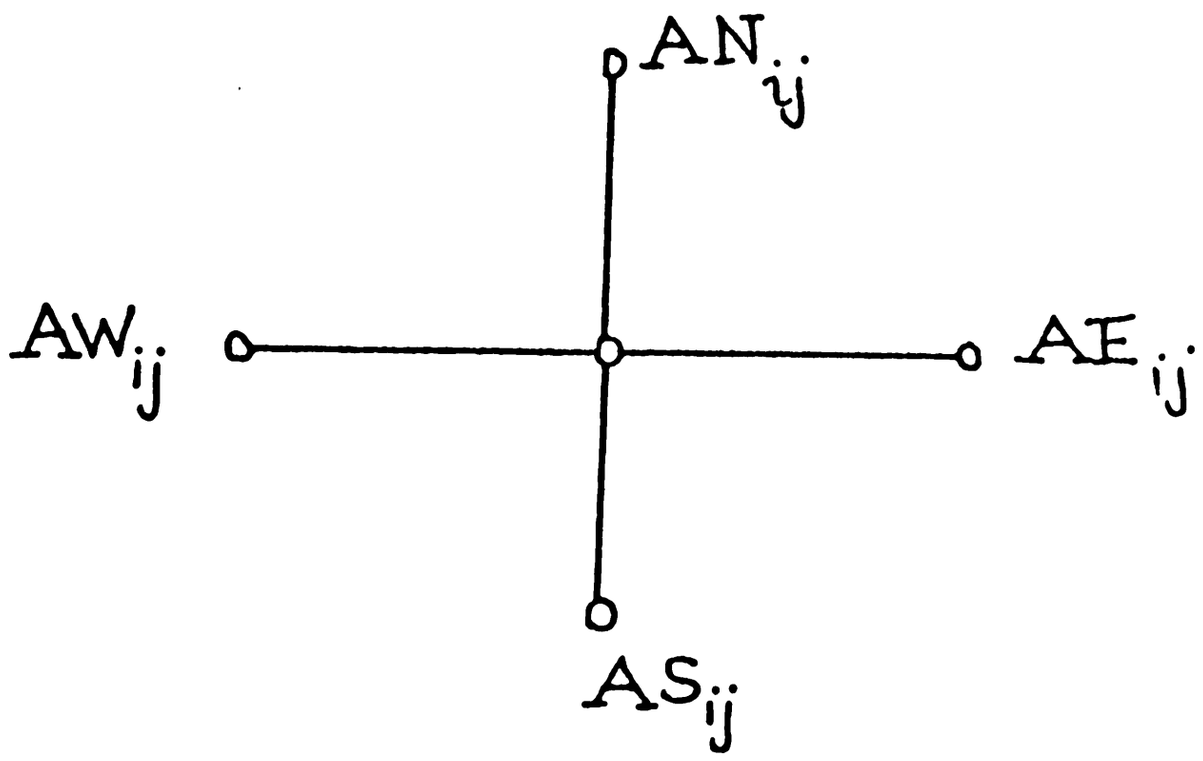


FIGURE (3.6)

Dividing Eq. (3.18) throughout by d_{ij} and rearranging gives

$$P_{ij} = (AW_{ij} P_{i-1,j} + AE_{ij} P_{i+1,j} + AS_{ij} P_{i,j-1}$$
$$+ AN_{ij} P_{i,j+1} + B_{ij}) / d_{ij} \quad . \quad . \quad . \quad (3-19)$$

Applying Gauss-Seidel procedure to Eq.(3.19), we can move from time level n to n+1. As in Fig.3.6, we shall consider the sequence wherein the points are taken in each row in order of increasing i, and then the rows are taken in order of increasing j. After traversing all the points, we start over again at the first point. A complete

iteration consists of one trip through all the points. It is schematically shown in Fig. (3.7). Referring 'k' as the number of counter for each complete iteration, $(P_{ij})^{k+1}$ (after the end of k iterations) can be found solving explicitly Eq.(3.19) as follow.

$$P_{ij}^{k+1} = (AW_{ij} P_{i-1,j}^{k+1} + AE_{ij} P_{i+1,j}^{k} + AS_{ij} P_{i,j-1}^{k+1} + AN_{ij} P_{i,j+1}^{k} + B_{ij})/d_{ij} \quad . \quad . \quad . \quad (3\text{-}20)$$

Another modification of this iterative method makes use of a weighting factor between old and new iterates. Applying a weighting factor w to Eq.(3.20), where $\omega > 1$, then

$$(P_{ij}^{k+1})_{SOR} = P_{ij}^{k} + \omega \left\{ (P_{ij}^{k+1})_{GS} - P_{ij}^{k} \right\} \quad . \quad . \quad (3\text{-}21)$$

where

$(P_{ij}^{k+1})_{GS}$ - elements of $\underline{P}$ after k iterations using Gauss-Seidel method.

$(P_{ij}^{k+1})_{SOR}$ - elements of $\underline{P}$ after k iterations using a weighting factor 'ω' to $(P_{ij}^{k+1})_{GS}$

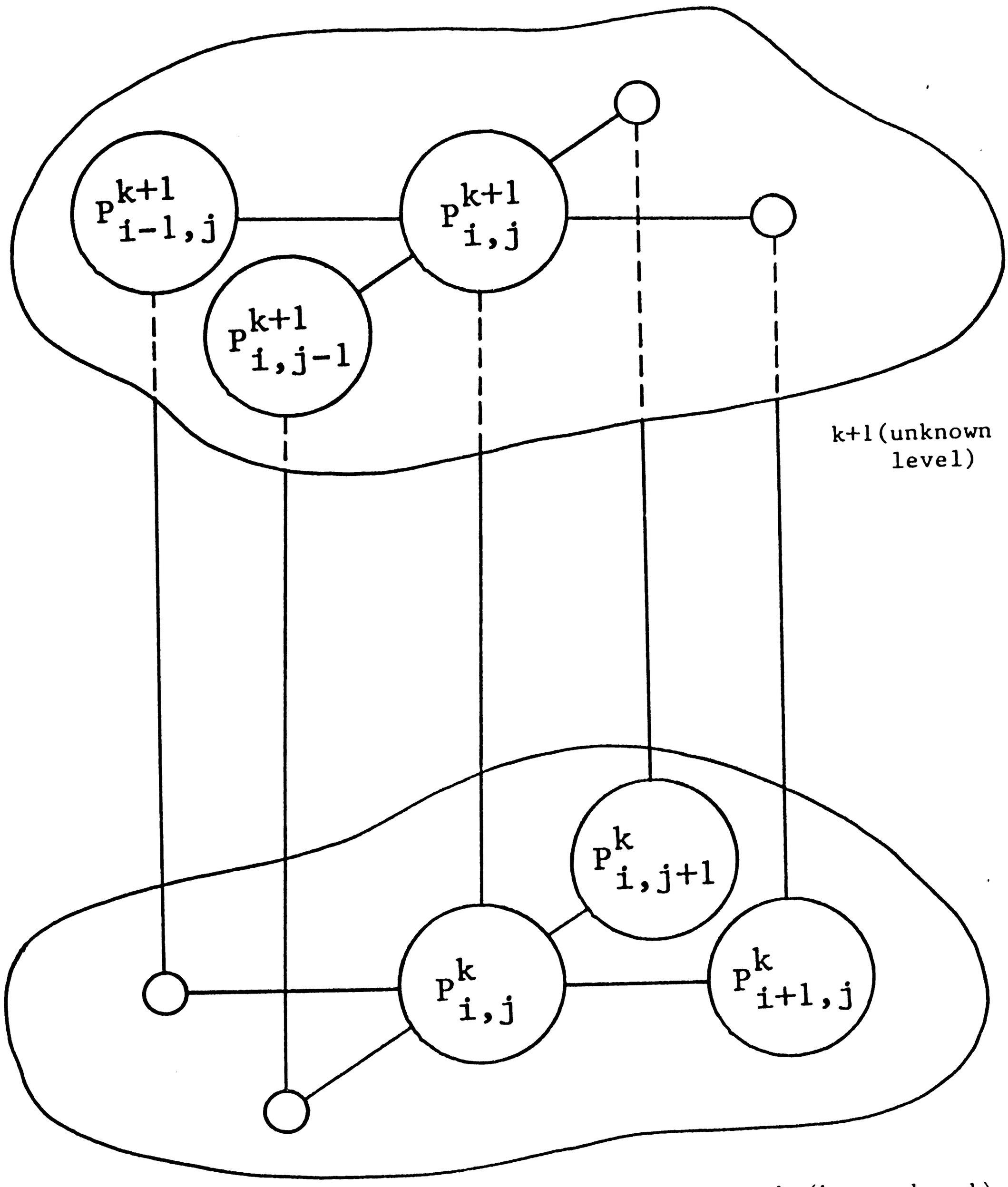

$P^{k+1}_{i-1,j}$
$P^{k+1}_{i,j}$
$P^{k+1}_{i,j-1}$
k+1 (unknown level)
$P^{k}_{i,j+1}$
$P^{k}_{i,j}$
$P^{k}_{i+1,j}$
k (known level)

FIGURE (3.7)

This procedure is known as point successive overrelaxation method. Eq.(3.21) can be represented in matrix from by $\underline{A}\ \underline{P} = \underline{B}$, where we split the coefficient matrix $\underline{A}$ into three matrices as

$$\underline{A} = \underline{L} + \underline{D} + \underline{U} \qquad \ldots(3.22)$$

in which

$\underline{L}$ = a lower triangular matrix containing the elements of $\underline{A}$ below the main diagonal, namely, $-AW_{ij}$ and $-AS_{ij}$

$\underline{D}$ = diagonal matrix whose elements are the diagonal elements of $\underline{A}$, namely, d_{ij}

$\underline{U}$ = upper traingular matrix containing the elements of $\underline{A}$ above the main diagonal, namely, $-AE_{ij}$ and $-AN_{ij}$

Substituting Eq. (3.22) into $\underline{A}\ \ \underline{P}\ = \underline{B}$, we obtain

$$(\underline{L} + \underline{D} + \underline{U})\underline{P} = \underline{B} \qquad \ldots(3.23)$$

In Jacobi method, we represent Eq.(3.23) in matrix form as follows

$$\underline{D}\ \underline{P}^{k+1} = -\ \underline{L}\ \underline{P}^{k} - \underline{U}\ \underline{P}^{k} + \underline{B}$$

or

$$\underline{P}^{k+1} = D^{-1}(-\ \underline{L}\ \underline{P}^{k} - \underline{U}\ \underline{P}^{k} + \underline{B}) \qquad \ldots(3.24)$$

Applying Gauss-Seidel method to Eq.(3.24), we obtain

$$(\underline{P}^{k+1})_{GS} = \underline{D}^{-1}(-\underline{L}\ \underline{P}^{k+1} - \underline{U}\ \underline{P}^{k} + \underline{B}) \qquad \ldots(3.25)$$

If we apply a weighting factor 'ω' to Eq. (3.25), we obtain the point SOR.

$$\underline{D}\,\underline{P}^{k+1} = \underline{D}\,\underline{P}^{k} + \omega\left[\underline{D}(\underline{P}^{k+1})_{GS} - \underline{D}\,\underline{P}^{k}\right]$$

$$= \underline{D}\,\underline{P}^{k} + \omega(-\underline{L}\underline{P}^{k+1} - \underline{U}\underline{P}^{k} + \underline{B} - \underline{D}\,\underline{P}^{k}) \qquad (3\text{-}26)$$

Rearranging and solving for $\underline{P}^{k+1}$, we obtain

$$\left(\underline{P}^{k+1}\right)_{SOR} = -\left(\omega\underline{L} + \underline{D}\right)^{-1}\left[(\omega-1)\underline{D} + \omega\underline{U}\right]\underline{P}^{k}$$

$$+\left(\omega\underline{L} + \underline{D}\right)^{-1}\underline{B} \qquad . \qquad . \qquad . \qquad (3\text{-}27)$$

This procedure is known as the point successive overrelaxation method. The methods treated so far have been point iterative methods, which could be solved explicitly. As an extension to SOR method, we can use LSOR method with which we advance in such a way that all points in a line be solved first implicitly, and then these values would be appropriately weighted using 'ω'. As we move from line to line, the updated values of the unknowns from the previous line could be used (Fig.3.8).

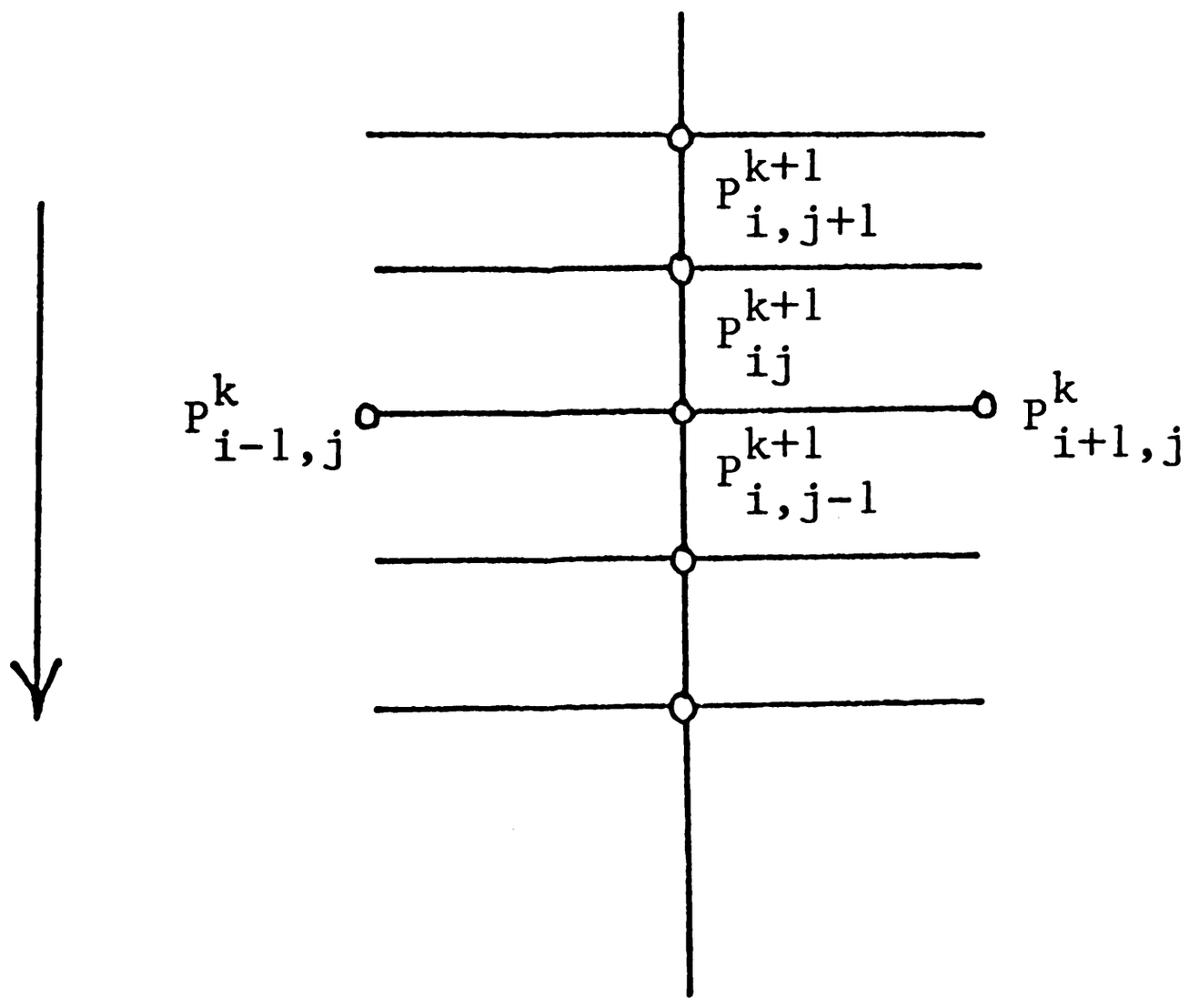


FIGURE (3.8)

$$P_{ij}^{k+1} = (AW_{ij} P_{i-1,j}^{k} + AE_{ij} P_{i+1,j}^{k} + AS_{ij} P_{i,j-1}^{k+1}$$

$$+ AN_{ij} P_{i,j+1}^{k+1} + B_{ij}) / d_{ij} \quad . \quad . \quad . \quad . \quad (3\text{-}28)$$

As soon as $(P_{ij})^{k\ 1}$ along the line are all calculated, they are overrelaxed using Eq.(3.21) before we advance to next line. This iterative method is known as line successive overrelaxation method (LSOR).

For closed boundary case, solution to Eq.(3.4) is difficult when the equations ae anisotropic; that is, when grid spacing is much longer in one direction than in the other. By anisotropic it means that two of the off diagonal coefficients in each equation are much larger than the other off-diagonal coefficients. Watts (10) describes an alternative method for use with anisotropic systems. It consists of a correction applied at each grid point in a line, coupled with LSOR method. This method is known as CLSOR, which is applied in our simulator.

Watt's method of additive corrections consists in adding to each P_{ij} in a column a constant, α_i, so that a 'corrected' value of $(P_{ij})^{k+1}$ is used as the starting point for the next iteration. Thus:

$$\left(P_{ij}^{k+1}\right)_{CLSOR} = \left(P_{ij}^{k+1}\right)_{LSOR} + \alpha_i \quad . \quad . \quad . \quad (3\text{-}29)$$

The α's are chosen so that the sum of the residuals for each column is reduced to zero. The basic idea of CLSOR can be explained briefly in the following sequence of steps:

1. $(P_{ij})_{GS}$ is determined first.

2. In second step, $(P_{ij})_{LSOR}$ is determined by

$$P_{ij}^{k+1} + \omega\left\{\left(P_{ij}^{k+1}\right)_{GS} - \left(P_{ij}^{k}\right)\right\}$$

3. Solve implicitly Eq.(3.30) for Watt's correction factor α_i:

$$\alpha_{i-1}\sum_{j=1}^{JJ} AW_{ij} + \alpha_i \sum_{j=1}^{JJ}\left(AS_{ij} + AN_{ij} - d_{ij}\right) +$$

$$\alpha_{i+1}\sum_{j=1}^{JJ} AE_{ij} = \sum_{j=1}^{JJ} R_{ij}^{k+1} \quad . \quad . \quad . \quad (3\text{-}30)$$

where

$$R_{ij}^{k+1} = d_{ij}(P_{ij}^{k+1})_{GS} - AW_{ij} P_{i-1,j}^{k+1} - AE_{ij} P_{i+1,j}^{k+1} -$$

$$AS_{ij} P_{i,j-1}^{k+1} - AN_{ij} P_{i,j+1}^{k+1} - B_{ij} \quad . \quad . \quad (3\_31)$$

4. Then $(P^{k+1}{}_{ij})_{LSOR}$ is corrected in a specified line by adding α_i.

$$(P_{ij}^{k+1})_{CLSOR} = (P_{ij}^{k+1})_{LSOR} + \alpha_i \qquad \ldots(3.32)$$

5. In such fashion, CLSOR can be applied to three dimensions, where its use shows to be fast in solving simulator equations.

CHAPTER 4: TREATMENT OF WELLS IN A SIMULATOR

Introduction

There are essentially two methods for treating individual wells in a simulator: by rate constraint, or by pressure constraint. However, very few papers have appeared in the literature about representing wells in numerical simulators (11, 12). We use these sources to construct the well model.

4.1 Rate Constraint Representation

Assume that we want to represent an oil production well in the simulator. We can write for oil flow rate

$$Q_o = \frac{q_o V_B}{\rho_{osc}} \qquad (4-1)$$

where

Q_o - volumetric oil flow rate at standard conditions

V_B - bulk value of the grid block containing the well

q_o - sink term

ρ_{osc} - oil density at standard condition

Employing the productivity index (PI) concept, Eq. (4.1) can also be written as

$$Q_o = PI\left(\frac{\lambda_o}{B_o}\right)(P_e - P_{wf}) \quad . \quad . \quad . \quad (4\text{-}2)$$

where

P_e = reservoir pressure, which is usually substituted by well grid block pressure 'P'

P_{wf} = flowing bottom-hole pressure which is less than P_e for production

Substituting 'P' for P_e in Eq.(4.2), we obtain

$$Q_o = PI\left(\frac{\lambda_o}{B_o}\right)(P - P_{wf}) \quad . \quad . \quad . \quad (4\text{-}3)$$

The variables in Eq.(4.3) are $Q_o = P$, while the remaining terms are considered to be known parameters. When Q_o or Q_t is specified, it is directly used in the pressure equation to obtain 'P'. There are three cases for representing rate constrainted wells:

1. Oil production rate (Q_o) specified
2. Total production rate (Q_t) specified
3. Injection rate specified

1. Oil Production Rate Specified

If the well of interest is completed in K layers and producing the total oil rate Q_o, the oil production rate from Kth layer is determined as

$$\left[Q_o\right]_k = Q_o \frac{\left[PI\left(\frac{\lambda_o}{B_o}\right)\right]_k}{\sum_{k=1}^{KK}\left[PI\left(\frac{\lambda_o}{B_o}\right)\right]_k} \quad \ldots \quad (4\text{-}4)$$

The water and gas production rates from the layer are related to $[Q_o]_k$ by the following equations:

$$\left[Q_w\right]_k = \left[Q_o\right]_k \left(\frac{\lambda_w B_o}{\lambda_o B_w}\right)_k \quad . \quad . \quad . \quad (4\text{-}5)$$

and

$$\left[Q_g\right] = \left[Q_o\right]_k \left[\left(\frac{\lambda_g B_o}{\lambda_o B_g}\right)_k + \left(R_{so}\right)_k\right] +$$

$$\left[Q_w\right]_k \left(R_{sw}\right)_k \quad . \quad . \quad . \quad (4\text{-}6)$$

Since PI's may be specified by layer in above equations, we are able to take into account permeability contrast in the model.

2. Total Production Rate Specified

When the total production rate Q_t other than Q_o is specified, we first compute the phase mobility ratios for all layers before determining the layered production rates. If α_{ot}, α_{wt}, α_{gt} denote the oil, water and gas mobilities respectively for all layers, we can determine them as follows:

$$\alpha_{ot} = \sum_{k=1}^{KK} \left(\frac{\lambda_o}{\lambda_o + \lambda_w + \lambda_g} \right)_k \quad \cdots \quad (4-7)$$

$$\alpha_{wt} = \sum_{k=1}^{KK} \left(\frac{\lambda_w}{\lambda_o + \lambda_w + \lambda_g} \right)_k \quad \cdots \quad (4-8)$$

$$\alpha_{gt} = \sum_{k=1}^{KK} \left(\frac{\lambda_g}{\lambda_o + \lambda_w + \lambda_g} \right)_k \quad \cdots \quad (4-9)$$

We now compute the total oil rate Q_o as

$$Q_o = \left(\frac{\alpha_{ot}}{\alpha_{ot} + \alpha_{wt} + \alpha_{gt}} \right) Q_t \quad \ldots \quad (4\text{-}10)$$

Having calculated for Q_o, we simply proceed as in Eq.(4.4) through Eq.(4.6) to obtain $[Q_o]_k$, $[Q_w]_k$ and $[Q_g]_k$.

3. Injection Rate Specified

If the well is a water or gas injector, we must first specify the total water or gas injection rates as Q_w or Q_g repsectively, and well injectivity indexes (WI) for each layer. The injection rate for each layer can be determined by:

$$\left[Q_w\right]_k = Q_w \frac{\left[WI(\lambda_o+\lambda_w+\lambda_g)\right]_k}{\sum_{k=1}^{KK}\left[WI(\lambda_o+\lambda_w+\lambda_g)\right]_k} \quad \cdot \quad \cdot \quad (4\text{-}11)$$

and

$$\left[Q_g\right]_k = Q_g \frac{\left[WI(\lambda_o+\lambda_w+\lambda_g)\right]_k}{\sum_{k=1}^{KK}\left[WI(\lambda_o+\lambda_w+\lambda_g)\right]_k} \quad \cdot \quad \cdot \quad (4\text{-}12)$$

It is important to note that allocation of injection fluids is based on total mobilities rather than injected fluid mobility. This is necessary for the following reason. If an injector is placed in a block where the relative permeability to the injection fluid is zero, then simulator using injection fluid mobility only would prohibit fluid injection, even though a real well would allow fluid injection. To avoid the unrealistic result of no fluid injection, the total mobility of the block is used.

Pressure Constraint Representation

When a well is pressure constrainted, Eq.(4.3) is used in place of Q_o in the pressure equation. After the pressure equation has been solved, the subsequent well block pressure is then substituted into Eq.(4.3) from which Q_o is determined. There are two cases for representing pressure constrainted wells.

1. Implicit pressure constraint
2. Explicit pressure constraint

4.2 Implicit Pressure Constraint

The source or sink terms in Eq.(4.3) may be written as

$$\frac{q_p V_B}{\varrho_{psc}} = \left[PID\left(\frac{\lambda_p}{B_p}\right)\right]^n \left(P^{n+1} - P_{wf}\right) \quad . \quad . \quad (4\text{-}13)$$

where

P = phase

PID = PI if the well is a producer and $P^{n+1} > P_{wf}$

PID = WI if the well is an injector and $P^{n+1} < P_{wf}$

Substituting Eq.(4.13) for the appropriate oil, water and gas phases into the pressure equation, we can implicitly solve for pressure. Then, the computed pressure P^{n+1} is replaced in Eq.(4.13) to yield rates. This is accomplished by the following sequence:

$$(E_{ijk})^{new} = (E_{ijk})^{old} - CPI \quad . \quad . \quad . \quad (4\text{-}14)$$

and

$$(B_{ijk})^{new} = (B_{ijk})^{old} - CPI * P_{wf} \quad . \quad . \quad (4\text{-}15)$$

where

$$CPI = 5.615(PID)_k \left[(B_o - B_g R_{so}) \frac{\lambda_o}{B_o} + (B_w - B_g R_{sw}) \frac{\lambda_w}{B_w} + B_g \left(\frac{\lambda_g}{B_g} \right) \right] \quad . \quad . \quad (4\text{-}16)$$

It is important to note that the new E and B terms are defined immediately before solving the linear system of pressure equations and after the E and B matrices defined by first in the pressure equation are computed.

4.3 Explicit Pressure Constraint

For oil production wells, the oil rate from the k*th* layer is given by

$$\left[Q_o\right]_k = \left[PID\left(\frac{\lambda_o}{B_o}\right)\right]_k^n \left(P^n - P_{wf}\right)_k \quad . \quad . \quad . \quad (4\text{-}17)$$

where

PID = PI and the explicit pressure P^n is used.

If $P^n < P_{wf}$, then the well is shut in. When $P^n > P_{wf}$, $[Q_o]_k$ is calculated and then substituted into Eqs. (4.5) and (4.6) to find $[Q_w]_k$ and $[Q_g]_k$ respectively.

For injection wells, the injection rate for a water or gas injection well is determined by

$$\left[Q_p\right]_k = \left[PID\left(\frac{\lambda_o + \lambda_w + \lambda_g}{B_p}\right)\right]^n \left(P^n - P_{wf}\right)_k \quad (4\text{-}18)$$

where PID = WI and P = phase. Fluid injection occurs when $P^n < P_{wf}$. If $P^n > P_{wf}$, the injection well is shut in. Total mobility is used for the injection well rate calculation.

4.4 Layer Flow Index (PID)

Layer flow index (PID) can be estimated by Peaceman (13,14) equation as:

$$\left[PID\right]_k = \left[\frac{0.00708\,Kh}{\ln\left(\frac{r}{r_w}\right) - \frac{1}{2} + S}\right]_k \qquad (4\text{-}19)$$

where

K = absolute permeability of layer k, md

h = layer thickness, ft

r_w = wellbore radius, ft

r = equivalent radius of grid block containing the well

s = skin factor, dimensionless

In reservoir simulation, the horizontal dimensions of any grid block containing a well are always much larger than the wellbore radius of that well. It long has been recognized that the pressure calculated for a well block will be greatly different from the flowing bottom-hole pressure of the modelled well. Peaceman has developed the relation between the equivalent radius of a well block to the dimensions of block itself for the cases when Δx and Δy are equal and not equal.

For a square grid (ie. $\Delta x = \Delta y$),

$$r \simeq 0.2\,\Delta x \qquad (4\text{-}20)$$

This equation is valid for both steady state and transient conditions. Substituting the value of r into Eq.(4.19), we obtain

$$\left[PID\right]_k = \left[\frac{0.00708\,Kh}{\ln\left(\frac{0.2\Delta x}{r_w}\right) - \frac{1}{2} + S}\right]_k \qquad (4\text{-}21)$$

Coupling '½' with the logarithmic term,

$$(PID)_k = \left\{\frac{0.00708\,Kh}{\ln\left(\frac{0.121\,\Delta x}{r_w}\right) + S}\right\}_k \qquad (4\text{-}22)$$

For nonsquare grid (ie. $\Delta x \neq \Delta y$),

$$r \simeq 0.14 \sqrt{(\Delta x)^2 + (\Delta y)^2} \quad . \quad . \quad . \quad (4\text{-}23)$$

Substituting Eq.(4.23) into Eq.(4.19), we obtain

$$\left[PID\right]_k = \left[\frac{0.00708\, Kh}{\ln\left(\frac{0.14\sqrt{(\Delta x)^2+(\Delta y)^2}}{r_w}\right) - \frac{1}{2} + S}\right]$$

or

$$\left[PID\right] = \left[\frac{0.00708\, Kh}{\ln\left(\frac{0.0849\sqrt{(\Delta x)^2+(\Delta y)^2}}{r_w}\right) + S}\right] \quad . \quad . \quad . \quad (4\text{-}24)$$

CHAPTER 5: COMPUTER CODES

Introduction

This chapter describes computer codes for three kinds of three-dimensional, multiphase reservoir simulators namely:

1. GOWSIM (Three-dimensional, three-phase black oil simulator)
2. GWSIM (Three-dimensional, gas-water reservoir simulator)
3. WSIM (Three-dimensional, single phase aquifer simulator)

All the programs simulate isothermal, darcy flow in three dimensions. They assume reservoir fluids can be described by fluid phases of constant composition with physical properties that depend on pressure only. Technically, all the programs are the finite-difference, implicit pressure-explicit saturation (IMPES) numerical simulators using direct (BAND) and iterative line successive overrelaxation with additive corrections (CLSOR) solution methods for solving systems of algebraic equations. The well model in the simulator allows specification of rate or pressure constraints on well performance, and the user is free to add or recomplete wells during simulation.

5.1 General Program Overview

The data input section is divided into two parts:

1. Initialization data section
2. Recurrent data section

The initialization data includes:

(i) the reservoir model grid dimensions and geometry

(ii) the porosity distribution within the reservoir

(iii) the permeability distribution within the reservoir

(iv) fluid PVT data

(v) the rock relative permeability and capillary pressure data

(vi) initial pressure and saturation distributions within the reservoir

(vii) specification of solution method and

(viii) various run control parameter.

The initialization data are read only once at the beginning of simulation. The way they must be read is explained in the next section and the nomenclatures of the subroutines in the simulator.

The recurrent data includes:

(i) the location and initial specifications of wells in the model

(ii) time step control information for advancing simulation through time

(iii) a schedule of individual well rate and/or pressure performance

(iv) changes in well completions and operations over time and

(v) controls on the type and frequency of printout information provided by the simulator.

Throughout the input data file, title cards or input data descriptive lines are read before each major and many minor sections of input data. These cards are used to serve as delineators to make the input data file easier to read and edit. All data values can be read under free format input that allows for easier reading, but results are printed under fixed format. The common block file allows the user all re-dimensioning to be done simply by changing the subsequent parameters in the common block file as desired with no need to recompile the main program. However, care must be taken to check if the parameters in the common block file match with the input data before starting the simulation run.

If a full grid of input values (II*JJ*KK) must be read for a particular model, the following input order must be followed. Each layer will be laid out as shown below, and Z-direction values will increase going down.

I = 1 I = 2 I = 3

J = 1

J = 2

.

.

5.2 General Flow Chart of GOWSIM

Before we present the computer algorithms for simulators, the following flow chart may explain in general how the pressure equation is developed and used to determine the pressures and saturations within the reservoir. It implies for simulation of three-dimensional, three-phase black oil reservoir using the IMPES method. When all the oil terms in the pressure equation are deleted appropriately, what we would get is called the gas-water reservoir simulator. Similarly we can derive a single-phase, water reservoir simulator from the gas-water simulator.

Conservation of Mass

$$-\nabla.(\vec{J}_p) - q_p = \frac{\partial C_p}{\partial t}$$

Fluid Fluxes

$$(\vec{J}_o) = \frac{\rho_{osc}}{B_o}\vec{v}_o \qquad (\vec{J}_w) = \frac{\rho_{wsc}}{B_w}\vec{v}_w$$

$$(\vec{J}_g) = \frac{\rho_{gsc}}{B_g}\vec{v}_g + R_{so}\frac{\rho_{gsc}}{B_o}\vec{v}_o + R_{sw}\frac{\rho_{gsc}}{B_w}\vec{v}_w$$

Phase Concentrations

$$C_o = \frac{\phi \rho_{osc} S_o}{B_o}$$

$$C_w = \frac{\phi \rho_{wsc} S_w}{B_w}$$

$$C_g = \phi \rho_{gsc}\left(\frac{S_g}{B_g} + R_{so}\frac{S_o}{B_o} + R_{sw}\frac{S_w}{B_w}\right)$$

Coupling the Above Equations

$$-\nabla\cdot\left(\frac{\vec{v}_o}{B_o}\right)-\frac{q_o}{\rho_{osc}}=\frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right)$$

$$-\nabla\cdot\left(\frac{\vec{v}_w}{B_w}\right)-\frac{q_w}{\rho_{wsc}}=\frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right)$$

$$-\nabla\cdot\left(\frac{\vec{v}_g}{B_g}+\frac{R_{so}\vec{v}_o}{B_o}+\frac{R_{sw}\vec{v}_w}{B_w}\right)-\frac{q_g}{\rho_{gsc}}=$$

$$\frac{\partial}{\partial t}\left[\phi\left(\frac{S_g}{B_g}+R_{so}\frac{S_o}{B_o}+R_{sw}\frac{S_w}{B_w}\right)\right]$$

Rate Equation (Darcy's Law)

$$\vec{v}_p=-\overleftrightarrow{K}\cdot\lambda_p\nabla\left(P_p-\frac{\rho_p z}{144}\right)$$

$$\lambda_p=\frac{k_{rp}}{\mu_p}$$

Phase Densities (Equation of State)

$$\rho_o = \frac{1}{B_o}\left(\rho_{osc} + R_{so}\rho_{gsc}\right)$$

$$\rho_w = \frac{1}{B_w}\left(\rho_{wsc} + R_{sw}\rho_{gsc}\right)$$

$$\rho_g = \frac{\rho_{gsc}}{B_g}$$

Capillary Pressures

$$P_{cow} = P_o - P_w$$

$$P_{cgo} = P_g - P_o$$

Oil Phase Equation

$$\nabla . \overleftrightarrow{K} . \left(\frac{\lambda_o}{B_o}\right) \nabla \left(P_o - \frac{\rho_o z}{144}\right) - \frac{q_o}{\rho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) = L_o$$

Let $CG_o = -\nabla . \overleftrightarrow{K} . \left(\frac{\lambda_o}{B_o}\right) \nabla \left(\frac{\rho_o z}{144}\right)$; then

$$\nabla . \overleftrightarrow{K} . \left(\frac{\lambda_o}{B_o}\right) \nabla P_o + CG_o - \frac{q_o}{\rho_{osc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_o}{B_o}\right) = L_o$$

Water Phase Equation

$$\nabla . \overleftrightarrow{K} . \left(\frac{\lambda_w}{B_w}\right) \nabla P_o + CG_w - \frac{q_w}{\rho_{wsc}} = \frac{\partial}{\partial t}\left(\frac{\phi S_w}{B_w}\right) = L_w$$

$$CG_w = -\nabla . \overleftrightarrow{K} . \left(\frac{\lambda_w}{B_w}\right) \nabla \left(P_{cow} + \frac{\rho_w z}{144}\right)$$

Gas Phase Equation

$$\nabla.\overleftrightarrow{K}.\left(\frac{\lambda_g}{B_g}+R_{so}\frac{\lambda_o}{B_o}+R_{sw}\frac{\lambda_w}{B_w}\right)+CG_g-\frac{q_g}{\rho_{gsc}}=$$

$$\frac{\partial}{\partial t}\left[\phi\left(\frac{S_g}{B_g}+R_{so}\frac{S_o}{B_o}+R_{sw}\frac{S_w}{B_w}\right)\right]=L_g$$

$$CG_g=\nabla.\overleftrightarrow{K}.\ \frac{\lambda_g}{B_g}\nabla\left(P_{cgo}-\frac{\rho_g z}{144}\right)-R_{so}\frac{\lambda_o}{B_o}\nabla\left(\frac{\rho_o z}{144}\right)$$

$$-R_{sw}\frac{\lambda_w}{B_w}\nabla\left(P_{cow}-\frac{\rho_w z}{144}\right)$$

Coupling Above 3-Phase Equations

$$\left(B_o-R_{so}B_g\right)L_o+\left(B_w-R_{sw}B_g\right)L_w+B_gL_g$$

$$=\ \phi c_t\frac{\partial P_o}{\partial t}$$

$$c_t=c_r+c_oS_o+c_wS_w+c_gS_g$$

$$S_o+S_w+S_g=1$$

Solve Pressure Equation

Implicitly

$$(B_o - R_{so}B_g)\left[\nabla.\overleftrightarrow{K}.\left(\frac{\lambda_o}{B_o}\right)\nabla P_o + CG_o - \frac{q_o}{\rho_{osc}}\right] +$$

$$(B_w - R_{sw}B_g)\left[\nabla.\overleftrightarrow{K}.\left(\frac{\lambda_w}{B_w}\right)\nabla P_o + CG_w - \frac{q_w}{\rho_{wsc}}\right] +$$

$$B_g\left[\nabla.\overleftrightarrow{K}.\left(\frac{\lambda_g}{B_g} + R_{so}\frac{\lambda_o}{B_o} + R_{sw}\frac{\lambda_w}{B_w}\right)\nabla P_o + CG_g - \frac{q_g}{\rho_{gsc}}\right] = \phi c_t \frac{\partial P_o}{\partial t}$$

Solve Explicitly for

Saturations

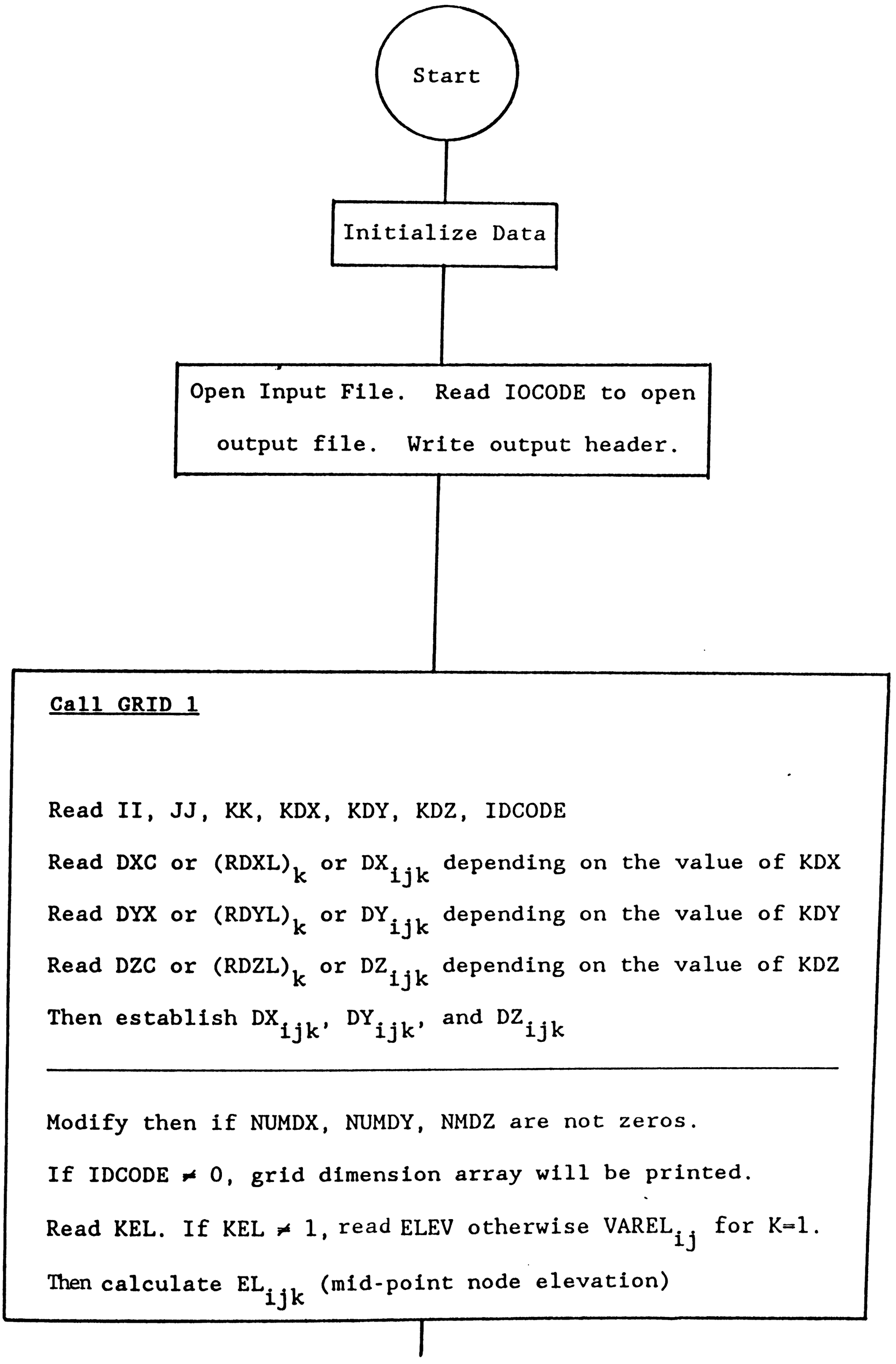
$$\frac{1}{\Delta t}\left[\left(\frac{\phi S_o}{B_o}\right)^{n+1} - \left(\frac{\phi S_o}{B_o}\right)^{n}\right] =$$

$$\nabla.\overleftrightarrow{K}.\left(\frac{\lambda_o}{B_o}\right)\nabla\left(P_o - \frac{\rho_o z}{144}\right) - \frac{q_o}{\rho_{osc}}$$

Similarly, calculate $(S_w)^{n+1}$ Then

$$(S_g)^{n+1} = 1 - (S_o + S_w)^{n+1}$$

5.3 Flow Chart for GWSIM Simulator



Call PARM1

Read KPH, KKX, KKY, KKZ, NUMP, NUMKX, NUMKY, NUMKZ, IPCODE

Read PHIC or $(RPHL)_k$ or VP_{ijk} depending on the value of KPH

Read KXC or $(RKXL)_k$ or KX_{ijk} depending on the vlaue of KKX

Read KYC or $(RKYL)_k$ or KY_{ijk} depending on the value of KKY

Read KZC or $(RKZL)_k$ or KZ_{ijk} depending on the value of KKZ

Then establish VP_{ijk}, KX_{ijk}, KY_{ijk} and KZ_{ijk}

Modify them if NUMP, NUMKX, NUMKY, NUMKZ are not zeros.

If IPCODE ≠ 0, modified porosity and permeability distributions will be printed.

Call TRAN1

Calculate TX_{ijk}, TY_{ijk}, TZ_{ijk}

$$TX_{ijk} = \frac{0.012656 * Al_{ijk} * Al_{i-1,jk}}{DX_{ijk} * Al_{i-1,jk} + DX_{i-1,jk} * Al_{ijk}}$$

$$Al_{ijk} = KX_{ijk} * DY_{ijk} * DZ_{ijk}$$

Similarly TY_{ijk}, TZ_{ijk} are calculated.

Modify TX, TY, TZ if NUMTX, NUMTY, NUMTZ are not zeros.

If ITCODE ≠ 0, modified transmissibility distribution is printed.

Call TABLE

READ SAT_i, $KRWT_i$. $KRGT_i$, $PCGWT_i$ for I = 1, NTE

Note: SAT(1) = −0.1 and SAT(N) = 1.1

Read PMAXT, PWT_i, $MUWT_i$, BWT_i, $RSWT_i$ for i = 1, NTE

Note: PWT (Last entry) = PMAXT

Read PGT_i, $MUGT_i$, BGT_i, CRT_i for i = 1, NTE

Note: PGT (Last entry) = PMAXT

Read RHOSCW, RHOSW

Calculate $BWPT_i = (\partial B_w/\partial p)_i$, $RWSPT_i = (\partial R_{sw}/\partial p)_i$, $BGPT_i = (\partial B_g/\partial p)_i$

Call UINIT 1

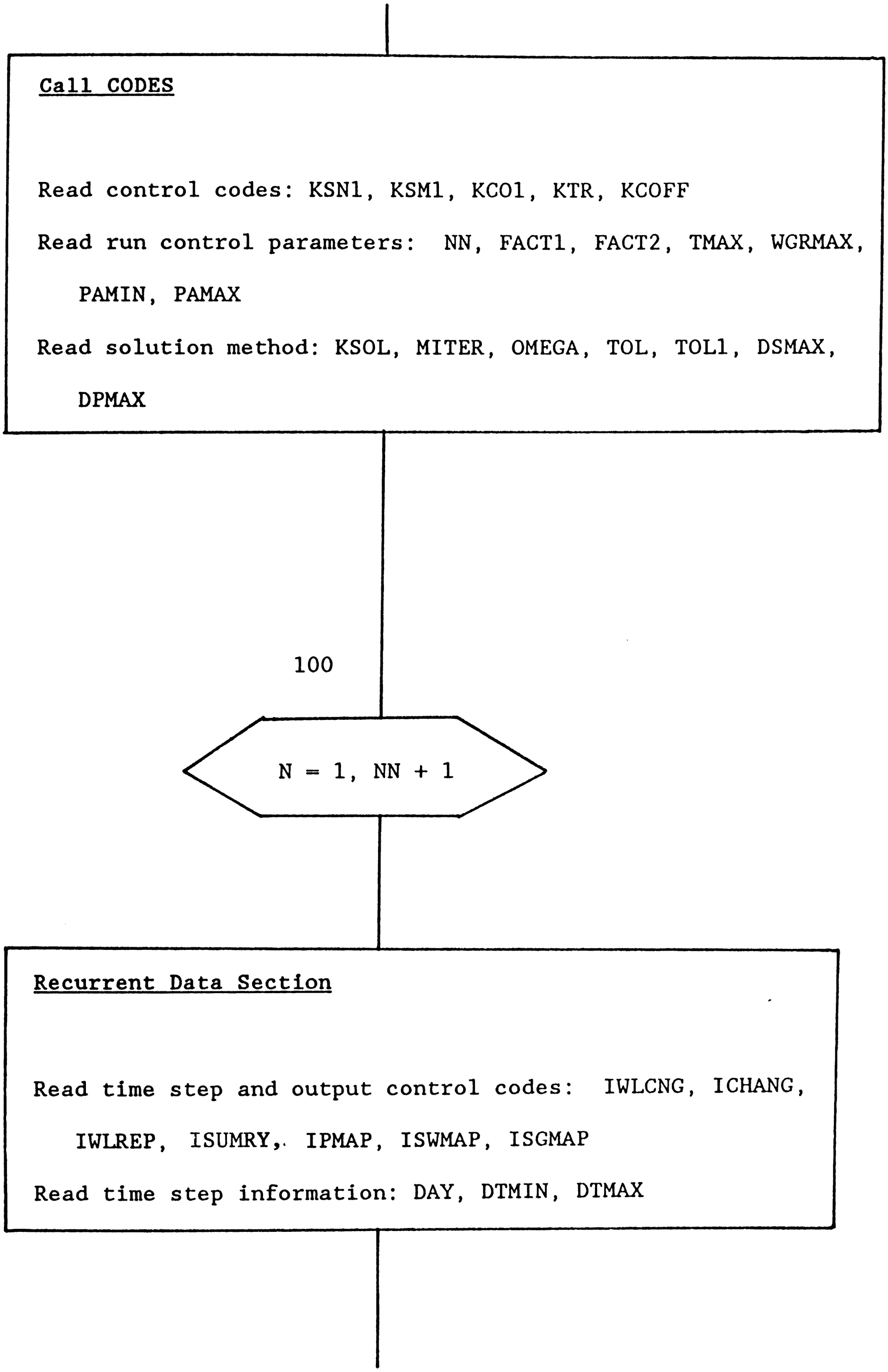
Read KPI, KSI

Read PGWC, GWC if KPI = 0, and calulate PN_{ijk}.

If KPI = 1, read PN_{ijk}.

Read SWC if KSI = 0, and establish SW_{ijk} and SG_{ijk}.

If KSI = 1, read SW_{ijk} and calculate SG_{ijk} (ie $1 - SW_{ijk}$).


Call CODES
Read control codes: KSN1, KSM1, KCO1, KTR, KCOFF
Read run control parameters: NN, FACT1, FACT2, TMAX, WGRMAX,
PAMIN, PAMAX
Read solution method: KSOL, MITER, OMEGA, TOL, TOL1, DSMAX,
DPMAX
100
N = 1, NN + 1
Recurrent Data Section
Read time step and output control codes: IWLCNG, ICHANG,
IWLREP, ISUMRY, IPMAP, ISWMAP, ISGMAP
Read time step information: DAY, DTMIN, DTMAX

Call NODES

Read $WELLID_j$, $IQN1_j$, $IQN2_j$, $IQN3_j$, $LAYER_j$, KIP_j, QNW_j, QVG_j,

QVT_j for j = 1, NVQN

Read $(WRAD)_k$, $(SKIN)_k$, PWF_{jk} Note: If $(SKIN)_k \geq 500$, $PID_{jk} = 0$

Then calculate PID_{jk} for j = 1, NVQN, and k = $IQN3_j$, LAY where

$LAY = IQN3_j + LAYER_j - 1$

Calculate:

(i) $Resvol = \Sigma\ V_{ijk} * DX_{ijk} * DY_{ijk} * DZ_{ijk}$

(ii) CW and CG; then $CT = CR + CW * SW_{ijk} + CG * SG_{ijk}$

(iii) $(OWIP)_k$, $(ODGIP)_k$, $(OFGIP)_k$

(iv) $TOWIP = \sum_{k=1}^{kk} (OWIP)_k$, $TODGIP = \sum_{k=1}^{kk} (ODGIP)_k$

$TOGFIP = \sum_{k=1}^{kk} (OFGIP)_k$

Call PRTPS

Print summary report.

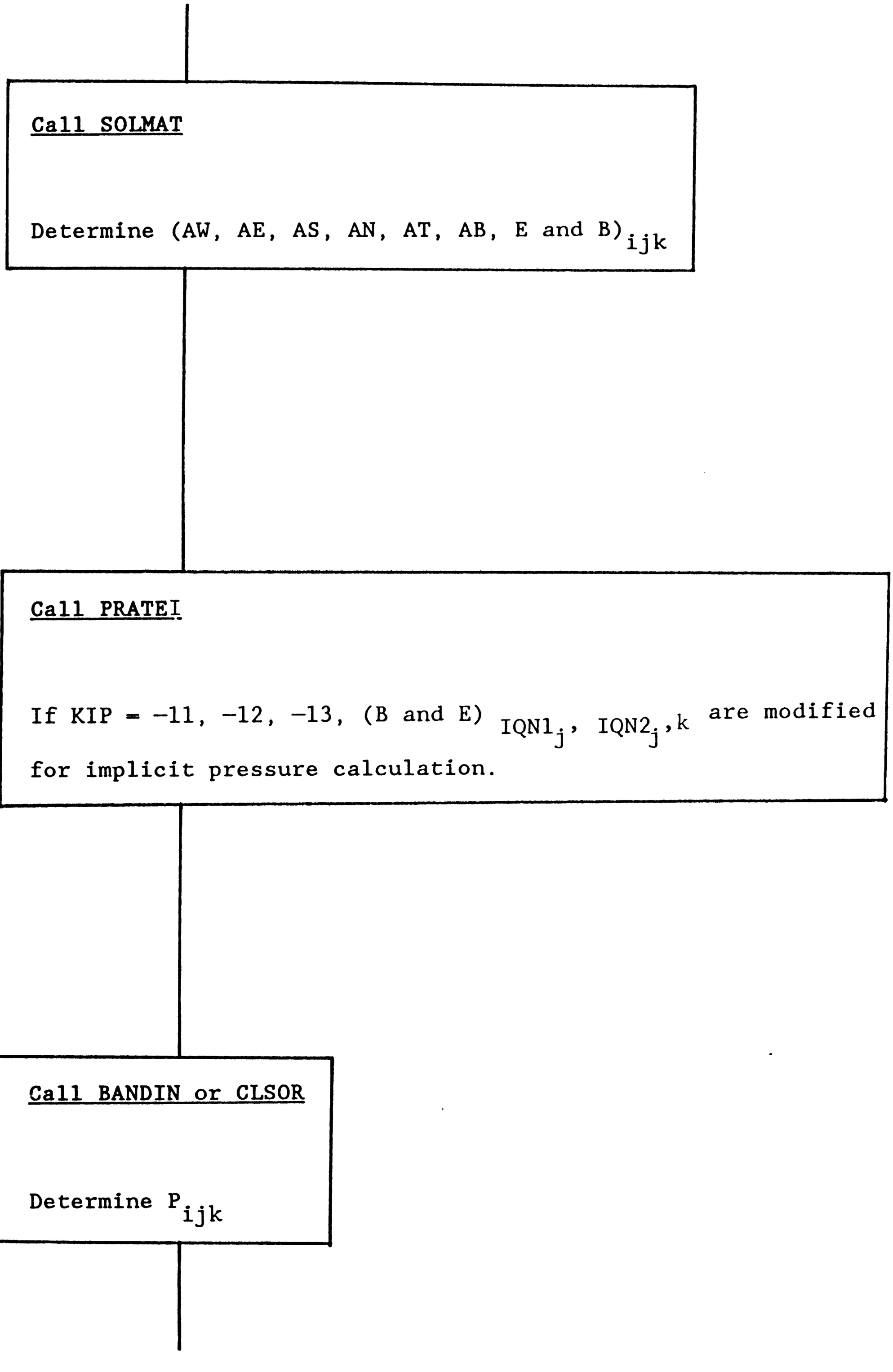
Print pressure and saturation distribution maps if IPMAP, ISWMAP and ISGMAP are not zeros.

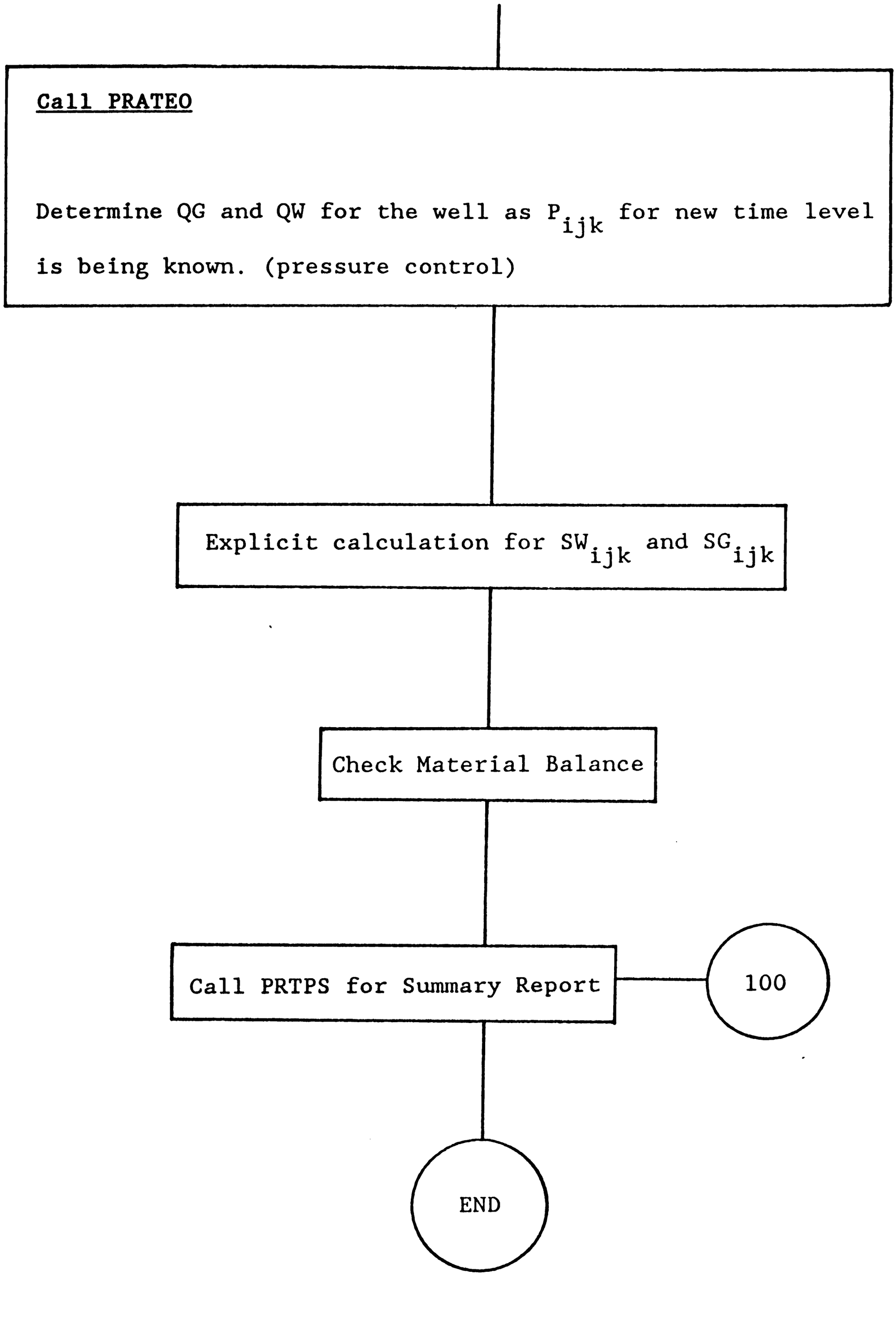
Call QRATE

Calculate first GMG_{jk}, and GMW_{jk} for j = 1, NVQN and k = IQN3, LAY.

If KIP = 1, 2, 3 (rate constraint), calculate $(QW, QG)_{IQN1_j, IQN2_j, k}$ and then calculate PWFC.

If KIP = −1, −2, −3 (PI and pressure control - explicit pressure calculation), calculate $(QW, QG)_{IQN1_j, IQN2_j, k}$


Call SOLMAT
Determine (AW, AE, AS, AN, AT, AB, E and B)ijk
Call PRATEI
If KIP = −11, −12, −13, (B and E) IQN1j, IQN2j,k are modified for implicit pressure calculation.
Call BANDIN or CLSOR
Determine Pijk


Call PRATEO
Determine QG and QW for the well as P_{ijk} for new time level is being known. (pressure control)
Explicit calculation for SW_{ijk} and SG_{ijk}
Check Material Balance
Call PRTPS for Summary Report
100
END

5.4 Codes for Type of Input to be Used

The following codes concerning data input conventions apply to all of the input data.

Grid Dimensions and Geometry

Kdx, Kdy, Kdz: codes for controlling input of x, y and z-direction grid dimensions respectively.

Code	Meaning
Kdx = −1	The x-direction grid dimensions are the same for all blocks in the grid. Only one value (Dxc) must be read.
Kdx = 0	The x-direction dimensions are read for each grid block in the first row of layer one (K=1). These same x-direction dimensions are assigned to all other rows and all other layers in the model grid. II values of Rdxl must be read.
Kdx = +1	The x-direction dimensions are read for every grid block in layer one. These same x-direction dimensions are assigned to all other layers in the model grid. II*JJ values of $Dx_{ij'1}$ must be read.
Kdy = −1	The y-direction grid dimensions are the same for all blocks in the grid. Only one value of Dyc must be read.
Kdy = 0	The y-direction dimenions are read for each grid block in the first column of layer one. These same y-direction

dimensions are assigned to all other columns and all other layers in the model grid. JJ values of Rdyl must be read.

Kdy = +1 The y-direction dimenions are read for every grid block in layer one. These same y-direction dimensions are assigned to all other layers in the model grid. II*JJ values of $Dy_{ij'1}$ must be read.

Kdz = −1 The z-direction grid dimensions (block thicknesses) are the same for all blocks in the grid. Only one value of Dzc must be read.

Kdz = 0 A constant value of thickness (Rdzl) is read for each layer in the grid. KK values of Rdzl must be read.

Kdz = +1 The z-direction grid dimensions (block thicknesses) are read for every block in the model grid. II*JJ*KK values of Dz_{ijk} must be read.

Idcode : print code

Idcode = 0 means 'do not print the modified grid dimensions'

Idcode = 1 means 'print the modified grid dimensions'

Kel : input code for depth value

Kel = 0 means 'a single constant value of Elev is read for the depth to the top of all grid blocks in layer one.'

Kel = 1 means 'a separate depth value must be read for each grid block in layer one. II*JJ value of $Varel_{ij}$ must be read'.

Porosity and Permeability Distributions

Kph — code for controlling porosity data input

Kkx, KKy, Kkz — codes for controlling the x, y and z-direction permeability data input respectively.

Code	Meaning
- 1	A single constant value is read and assigned to all blocks in the model grid. Only one value is read.
0	A constant value is read for each of the KK layers in the grid. Individual layers may have a different, but constant, value. KK values must be read.
+ 1	A separate value is read for each block in the grid. II*JJ*KK values must be read.

Ipcode — print code

Ipcode = 0 means 'do not print modified porosity and permeability distributions'

Ipcode = 1 means 'print modified porosity and permeability distributions'

Itcode — print code

Itcode = 0 means 'do not print the modified transmissibility distributions'

Itcode = 1 means 'print the modified transimissibility distributions'

Pressure and Saturation Initialization

There are two options for pressure and saturation initialization.

The initial pressure distribution can be calculated by the program for equilibrium conditions given the location of the gas/oil contact and oil/water contact (for GOWSIM) and the gas/water contact (for GWSIM) and the pressure at contact.

The initial pressure distribution can be read on a block-by-block basis.

Saturations can either be read as constant values over the entire grid or the entire saturation distributions are read on a block-by-block basis.

Kpi,Ksi pressure and saturation initialization codes respectively

Code	Meaning
Kpi = 0	Use equilibrium pressure initialization. Input required will be pressures at the oil/water contact and gas/oil contact (for GOWSIM) and pressure at the gas/water contact (for GWSIM) and depths to each contact.
Kpi = 1	Use non-equilibrium pressure initialization. Pressures for each block must be read on a block-by-block basis. II*JJ*JJ values of pressure must be read.
Ksi = 0	Initial phase saturations are constant over the entire model grid.
Ksi = 1	Phase saturations must be read for each grid block on a block-by-block basis. That is, 2*II*JJ*KK values of saturations must be read for GOWSIM and II*JJ*KK vlaues of saturation for GWSIM.

Debug and Diagnostics Codes

Ksnl	CLSOR parameter debug output control code
Ksml	solution matrix debug output control code
Kcol	compressibility and formation volume factor debut output control code
Ktr	transmissibility debug output control code
Kcoff	density and saturation debug output control code

The zero values of above codes mean 'do not print diagnostics output'

The 1's for above codes mean 'print diagnostics output'.

Solution Method Specifications

Ksol — solution method code

Ksol = 1 means 'direct solution - BAND algorithm'

Ksol = 2 means 'CLSOR - iterative line successive overrelaxation with additive corrections method'

Time Step and Output Control

Iwlcng — code to tell program whether or not the well information cards should be read this time step.
Iwlcng = 1 means 'read well information this time step'
Iwlcng = 0 means 'do not read well information this time step'

Iwlrep — output code to control printing of the well report

Isumry — output code to control printing of the time step summary report

Ipmap — output code to control printing of the map of grid block pressures

Isomap — output code to control printing of the grid block oil saturations

Iswmap — output code to control printing of the grid block water saturations

Isgmap — output code to control printing of the grid block gas saturations

Ipbmap — output code to control printing of the grid block saturation pressures

Code	Meaning
Kip = 1	Production (oil) well - rate specified
Kip = 2	Water well - rate specified
Kip = 3	Gas well - rate specified
Kip = −1	Production (oil) well - PI and FBHP control (explicit pressure calculation)
Kip = −2	Water well - PI and FBHP control (explicit pressure calculation)
Kip = −3	Gas well - PI and FBHP control (explicit pressure calculation)
Kip = −11	Production (oil) well - PI and FBHP control (implicit pressure calculation)
Kip = −12	Water well - PI and FBHP control (implicit pressure calculation)
Kip = −13	Gas well - PI and FBHP control (implicit pressure calculation)

Gas wells (Kip = 3, −3, −13) will produce or inject single phase gas only.

Water wells (Kip = 2, −2, −12) will produce or inject single phase water only.

Oil wells (Kip = 1, −1, −11) will produce three phases in proportions determined by reservoir conditions and well constraints. Negative rates indicate fluid injection while positive rates indicate fluid production.

There is an important comment on PI. Once a well has been completed in a certain layer, that layer must continue to be specified on all succeeding well information cards, even if the layer or the well is shut in. To shut in a well, set that layer skin equal to or greater than 500. To shut in a well, set all of its layer skin's equal to or greater than 500.

Value of Output Code	Meaning
0	Do not ptint the report/array
1	Print the report/array at every time step during this interval

Well Information

Kip — code for specifying both well type and whether the well's production (injection) performance is determined by specifying rates or by specifying flowing bottom-hole pressure and also whether an explicit or implicit pressure calculation is to be done.

CHAPTER 6: INPUT DATA FILES AND RESULTS

Introduction

This chapter provides a variety of problems which have been solved using the GOWSIM, GWSIM and WSIM simulators. These problems cover a range of different model grid configurations and reservoir conditions:

1. One-dimensional, linear Buckley-Leverett waterflood displacement model.
2. Two-dimensional, areal grid model showing primary depletion of an undersaturated reservoir.
3. Two-dimensional, x-z cross-sectional model showing waterflooding of a layered system with multizone completions.
4. Two-dimensional, x-z cross-sectional model showing production from oil patch and gas patch reservoirs, with fluid contact levels at different depths, underlained by aquifer.
5. Linear, three-dimensional, two-phase model showing a single gas well producing from gas-water reservoir.
6. Two-dimensional, cross-section model of gas-water reservoir with two different fluid contact levels.
7. Three-dimensional water reservoir model.

In most of these problems, the GOWSIM solution is compared with either an analytical solution (in the case of the Buckley-Leverett) or with the solution of one or more commerically available black oil simulators' result on the same problem. It is quite noticeable that the general behaviour of reservoir prformance prediction is similar for all of the simulators, and thus it can provide a reasonable forecast of the general performance to be expected from a reservoir under user-specified operating condtions and constraints. Results from a reservoir simulation study must be viewed as an approximation to the expected performance and not as an exact performance prediction. The application of GWSIM and WSIM simulators on some reservoir problems are presented also.

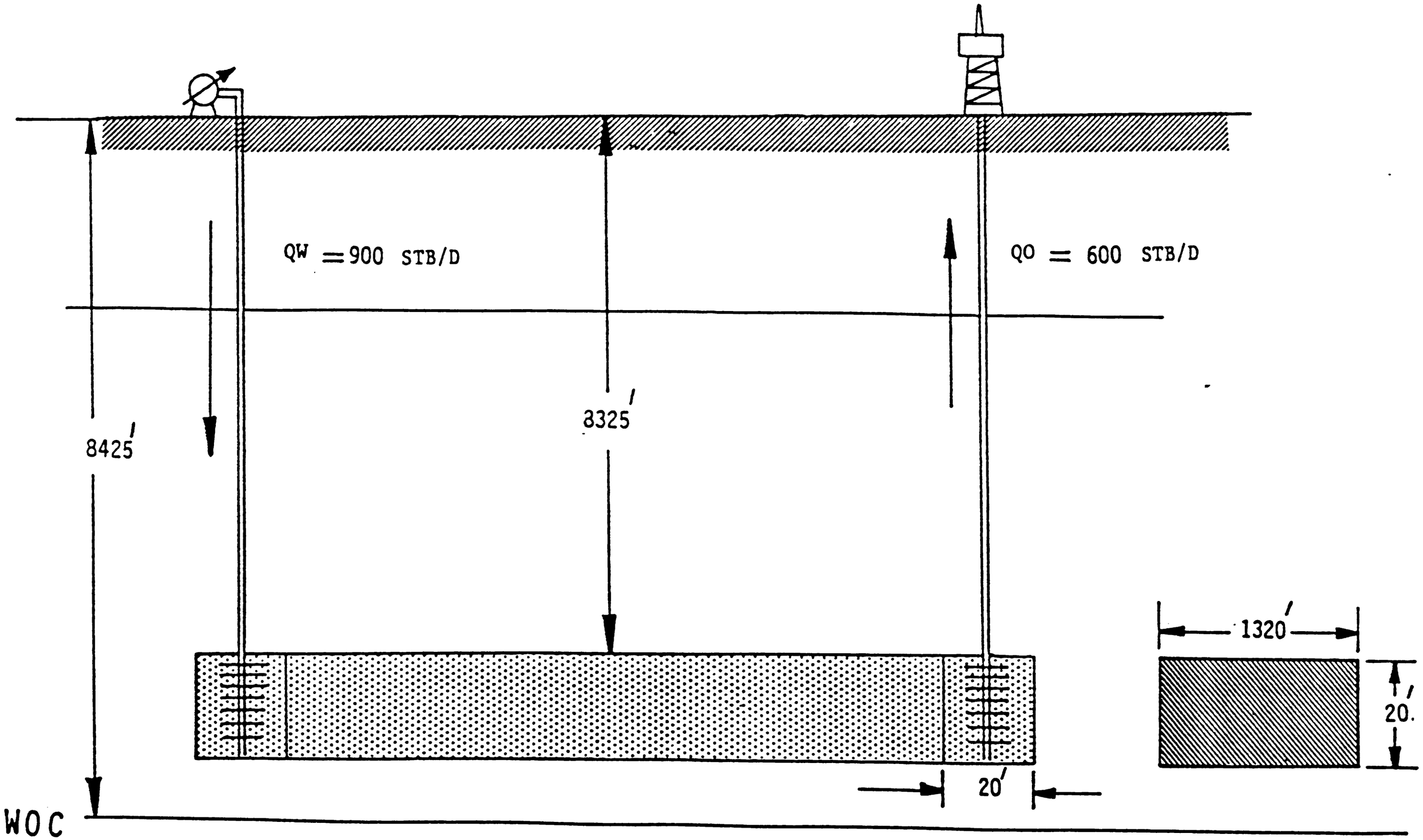
Problem 1: One-Dimensional, Linear Buckley-Leverett Waterflood Model

Among the very few analytical solutions available that can be used to check black oil simulators, one such solution is the Buckley-Leverett line-drive waterflood calculation. GOWSIM performs this calculation to provide a test of saturation results. The structure of problem 1 is illustrated in Fig. (6.1). Data for the Buckley-Leverett waterflood are presented as GOWSIM input data on free formatted coding form in Table (6.1).

The model is a one-dimensional, linear, homogeneous reservoir with an oil well at one end grid block producing 600 STBD (B_o = 1.5), which is balanced by water injection under rate constraint of 900 STBD (B_w = 1.0) into the opposite end grid block -

Frontal results from the GOWSIM calculation are shown in Table (6.2) after an elapsed time of 360 days. These results are plotted against the analytical solution in Fig.(6.2), which shows good agreement except at the piston-like displacement front. This dispersing of the saturation front is a phenomenom inherent in all finite difference simulators and is known as numerical dispersion. This effect can be reduced by adjusting the time step in simulator.

NOTE: (The data file name for GOWSIM is BOAST.DAT and the output file name is BOAST.RES and the common block file name is COMMB3.FOR. Before running the program, it is important to check if the input data match with the parameters in the common block file).



One-Dimensional Linear Model Showing
Buckley-Leverett Calculation

FIGURE (6.1)

TABLE 6.1 INPUT DATA FOR PROBLEM 1

```
      BLACK OIL SIMULATOR WATERFLOOD: BUCKLEY-LEVERETT.
40,1,1
GRID BLOCK LENGTHS
-1,-1,-1
20.
1320.
20.
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
CAPROCK BASE DEPTHS
0
8325.
POROSITY AND PERMEABILITY
-1,-1,-1,-1
.25
200.
200.
20.
POROSITY AND PERMEABILITY MODIFICATION CARDS
2,0,0,0,1
1,1,1,0.125
40,1,1,0.125
TRANSMISSIBILITY MODIFICATIONS
0,0,0,0
  SAT     KRO       KRW        KRG          PCOW         PCGO
-.1,0.0,0.0,0.0,0.0,0.0
.1,0.0,0.0,0.0,0.0,0.0
.2,.00147,0.0,.075,0.0,0.0
.3,.00228,.0122,.19,0.0,0.0
.4,.037,.0244,.41,0.0,0.0
.5,.0571,.0336,.72,0.0,0.0
.6,.134,.0672,.87,0.0,0.0
.7,.207,.1344,.94,0.0,0.0
.8,.604,.2688,.9667,0.0,0.0
.9,1.,.4704,.9933,0.0,0.0
1.1,1.,.5,1.,0.0,0.0
   PBO      VSLOPE       BSLOPE         RSLOPE          PMAX
4014.7,0.0,-.000001,0.0,9014.7,0
    P          MUO          BO          RSO
14.7,2.,1.5,1.
4014.7,2.,1.5,1.
9014.7,2.,1.5,1.
    P          MUW          BW          RSW
14.7,1.,1.,0.0
4014.7,1.,1.,0.0
9014.7,1.,1.,0.0
    P          MUG          BG          CR
14.7,.008,.9358,.000003
264.7,.0096,.067902,.000003
514.7,.0112,.035228,.000003
1014.7,.014,.017951,.000003
2014.7,.0189,.009063,.000003
2514.7,.0208,.007266,.000003
3014.7,.0228,.006064,.000003
4014.7,.0268,.004554,.000003
5014.7,.0309,.003644,.000003
9014.7,.047,.002167,.000003
    RHOSCO        RHOSCW        RHOSCG
46.244,62.238,.0647
EQUILIBRIUM PRESSURE INITIALIZATION/CONSTANT SATURATIONS
```

TABLE 6.1 Continued

```
0,0
4806.6,0.0,8425.,8300.
.8,.2,0.0
KSN1    KSM1     KC01       KTR     KCOFF
0,0,0,0,0
NMAX  FACT1  FACT2   TMAX   WORMAX   GORMAX    PAMIN    PAMAX
500,1.2,.5,365.,20.,500000.,14.7,10000.
KSOL   MITER   OMEGA   TOL   TOL1     DSMAX     DPMAX
2,100,1.7,.1,0.0,.05,100.
RECURRENT DATA
1,1,1,1,1,1,1,1,0
3.,3.,3.,0.0,0.0,0.0
RATES
2
'PROD',40,1,1,1,1,600.,0.0,0.0,0.0
.25,23.955,0.0
'INJ',1,1,1,1,2,0.0,-900.0,0.0,0.0
.25,-1.533,0.0
0,8,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
3.,3.,3.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
3.,3.,3.,0.0,0.0,0.0
```

TABLE 6.2

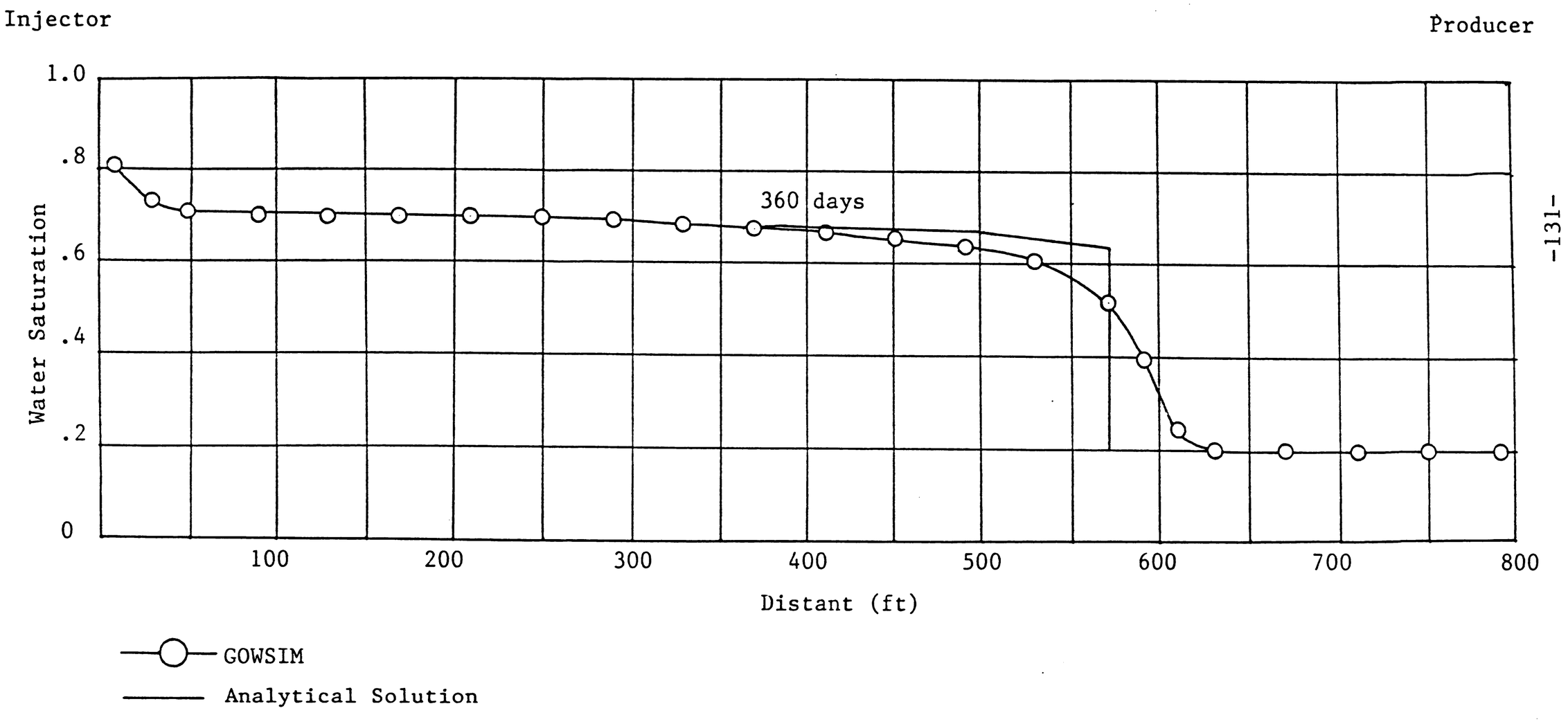
```
**********************************************************************
*                                                                    *
*              SUMMARY REPORT: UNSW BOAST (VERSION 2.0)              *
*                                                                    *
**********************************************************************
```

```
ELAPSED TIME (DAYS)       =   360.00    TIME STEP NUMBER      =  120       TIME STEP (DAYS)      =     3.00

CURRENT AVG RES PRESSURE =   4747.6    PREVIOUS AVG RES PRESSURE =  4748.1    PRESSURE DPMAX( 29,  1,  1) =      4.2
OIL DSMAX ( 30,  1,  1)     = -0.04205    GAS DSMAX (  1,  1,  1)     = -0.00003    WATER DSMAX( 30,  1,  1)    =  0.04205

OIL MATERIAL BALANCE (%)     =-0.000584    GAS MATERIAL BALANCE (%)      =0.7627E-01    WATER MATERIAL BALANCE (%)    =0.1192E-04

OIL PRODUCTION RATE (STB/D)   =0.6000E+03    CUM. OIL PRODUCTION (STB)    =0.2160E+06
GAS PRODUCTION RATE (MSCF/D)=0.6000E+00    CUM. GAS PRODUCTION (MSCF)   =0.2160E+03
WATER PRODUCTION RATE (STB/D)=0.8398E-01    CUM. WATER PRODUCTION (STB) =0.2062E+02

GAS INJECTION RATE (MSCF/D) =0.0000E+00    CUM. GAS INJECTION (MSCF)    =0.0000E+00
WATER INJECTION RATE (STB/D)=-.9000E+03    CUM. WATER INJECTION (STB)   =-.3240E+06

PRODUCING WOR (STB/STB)      =0.1400E-03    CUM. WOR (STB/STB)           =0.9547E-04
PRODUCING GOR (SCF/STB)      =0.1000E+01    CUM. GOR (SCF/STB)           =0.1000E+01
```



```
            ****RESERVOIR PRESSURE DISTRIBUTION****

K= 1

     5187. 5177. 5160. 5140. 5118. 5096. 5074. 5051. 5029. 5007. 4984. 4962. 4940. 4917. 4894. 4871. 4848. 4824. 4800. 4776.
     4751. 4725. 4699. 4672. 4644. 4615. 4584. 4549. 4509. 4461. 4432. 4419. 4409. 4399. 4389. 4379. 4369. 4359. 4349. 4339.

            *******OIL SATURATION********

K= 1

     0.192 0.266 0.289 0.297 0.299 0.300 0.300 0.300 0.300 0.300 0.301 0.302 0.303 0.305 0.307 0.311 0.314 0.319 0.323 0.328
     0.334 0.339 0.345 0.352 0.361 0.372 0.393 0.432 0.480 0.605 0.753 0.797 0.800 0.800 0.800 0.800 0.800 0.800 0.800 0.800

            *******WATER SATURATION********

K= 1

     0.808 0.734 0.711 0.703 0.701 0.700 0.700 0.700 0.700 0.700 0.699 0.698 0.697 0.695 0.693 0.689 0.686 0.681 0.677 0.672
     0.666 0.661 0.655 0.648 0.639 0.628 0.607 0.568 0.520 0.395 0.247 0.203 0.200 0.200 0.200 0.200 0.200 0.200 0.200 0.200


**************************************************  END OF REPORT  **************************************************
```

FIGURE (6.2)


Injector
Producer
1.0
.8
.6
.4
.2
0
Water Saturation
360 days
100
200
300
400
500
600
700
800
Distant (ft)
GOWSIM
Analytical Solution

Problem 2: Two-Dimensional Areal Model Showing Primary Depletion of an Undersaturated Reservoir

In this problem, a single well is producing under a rate constraint of 300 BOPD from the centre of a two-dimensional areal reservoir model (II = 9, JJ = 9, KK = 1) with production coming from fluid expansion drive (Fig.6.3). The input data file for this problem is shown in Table (6.3).

The results for this problem illustrate primarily the simulator pressure solution because saturations do not change for the undersaturated reservoir (Table 6.4). These results are presented in Fig. 6.4 along with the solution to this same problem using the BOAST simulator with LSOR method.

FIGURE (6.3)

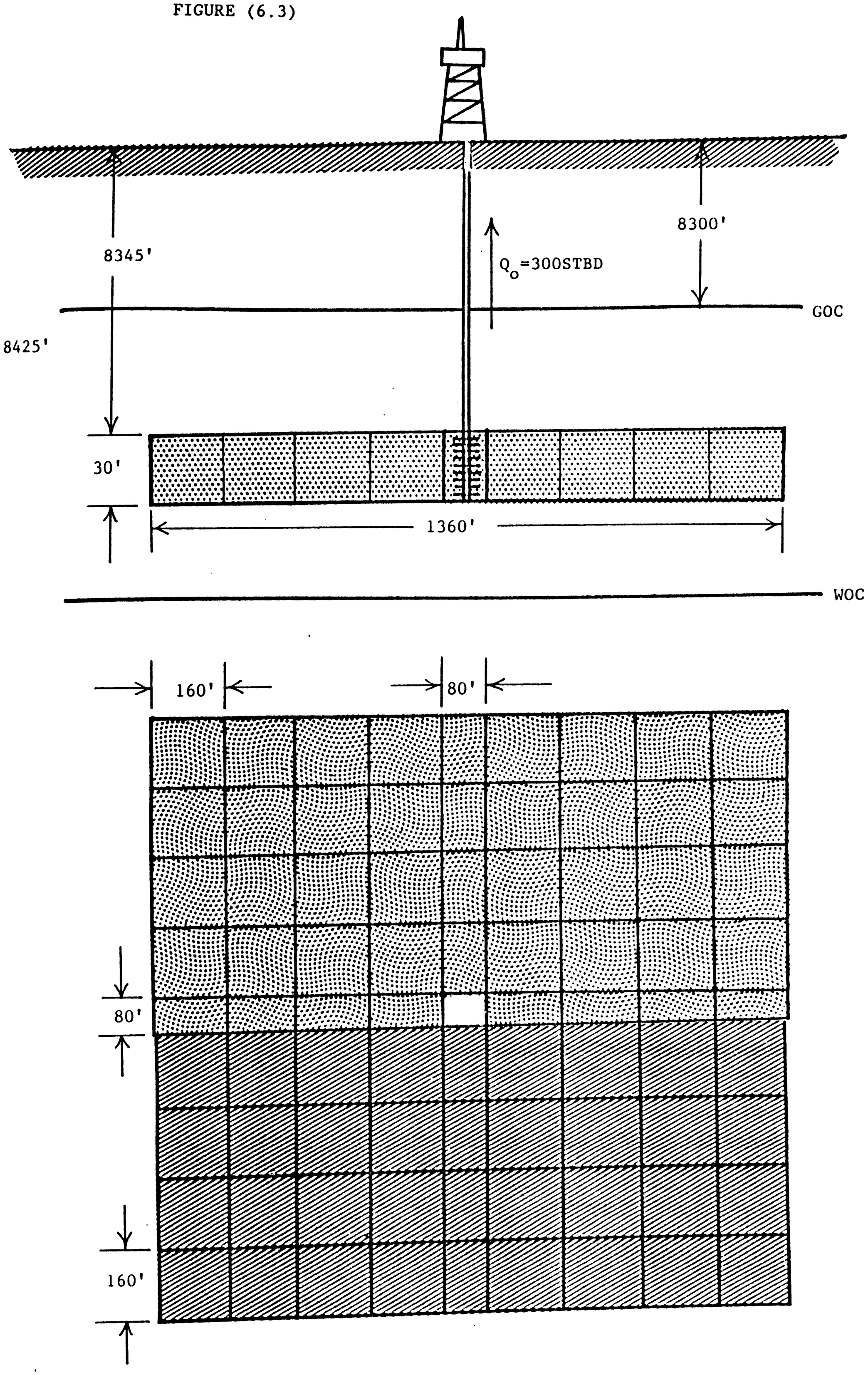


TABLE 6.3 INPUT DATA FOR PROBLEM 2

```
     2D, PRIMARY DEPLETION OF AN UNDERSATURATED RESERVOIR
7,7,1
GRID BLOCK LENGTHS
0,0,-1
160.,160.,160.,160.,80.,160.,160.,160.,160.
160.,160.,160.,160.,80.,160.,160.,160.,160.
30.
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
DEPTH TO TOP OF SAND
0
8345.
POROSITY AND PERMEABILITY DISTRIBUTIONS
1,-1,-1,-1
.28,.28,.28,.28,.28,.28,.28,.28,.28
.28,.28,.28,.28,.28,.28,.28,.28,.28
.28,.28,.28,.28,.28,.28,.28,.28,.28
.28,.28,.28,.28,.28,.28,.28,.28,.28
.28,.28,.28,.28,.28,.28,.28,.28,.28
.,.,.,.,.,.,.,.,.
.,.,.,.,.,.,.,.,.
.,.,.,.,.,.,.,.,.
.,.,.,.,.,.,.,.,.
100.
100.
100.
POROSITY AND PERMEABILITY MODIFICATION CARDS
0,0,0,0,0
TRANSMISSIBILITY MODIFICATIONS
0,0,0,0
SAT     KRO     KRW     KRG      PCOW     PCGO
-.1,0.0,0.0,0.0,0.0,0.0
.02,0.0,0.0,0.0,0.0,0.0
.12,0.0,0.0,.02,0.0,0.0
.2,0.0,.02,.06,0.0,0.0
.3,0.0,.04,.2,0.0,0.0
.4,.03,.07,.46,0.0,0.0
.5,.09,.12,.7,0.0,0.0
.6,.17,.18,.87,0.0,0.0
.7,.3,.27,.91,0.0,0.0
.8,.5,.51,.94,0.0,0.0
.88,.75,.71,.97,0.0,0.0
1.,1.,1.,1.,0.0,0.0
1.1,1.,1.,1.,0.0,0.0
PBO     VSLOPE     BSLOPE      RSLOPE      PMAX      IREPRS
4014.7,.000046,-.0000232,0.0,9014.7,0
P       MUO       BO       RSO
14.7,1.04,1.062,1.0
264.7,.975,1.15,90.5
514.7,.91,1.207,180.
1014.7,.83,1.295,371.
2014.7,.695,1.435,636.
2514.7,.641,1.5,775.
3014.7,.591,1.565,930.
4014.7,.51,1.695,1270.
5014.7,.449,1.827,1618.
9014.7,.203,2.357,2984.
P         MUW       BW       RSW
14.7,.5,1.019,0.0
1014.7,.501,1.016,0.0
```

TABLE 6.3 Continued

```
2014.7,.502,1.013,0.0
4014.7,.505,1.007,0.0
6014.7,.51,1.001,0.0
9014.7,.52,.992,0.0
P       MUG       BG       CR
14.7,.008,.9358,.000003
264.7,.0096,.067902,.000003
514.7,.0112,.035228,.000003
1014.7,.014,.017951,.000003
2014.7,.0189,.009063,.000003
2514.7,.0208,.007266,.000003
3014.7,.0228,.006061,.000003
4014.7,.0268,.004554,.000003
5014.7,.0309,.003644,.000003
9014.7,.047,.002167,.000003
RHOSCO      RHOSCW      RHOSCG
46.244,62.238,.0647
EQUILIBRIUM PRESSURE INITIALIZATION/CONSTANT SAT: INITIALIZATION
0,0
5400.,0.0,8425.,8300.
.88,.12,0.0
KSNL    KSML    KCOL     KTR    KCOFF
0,0,0,0,0
NMAX  FACT1  FACT2  TMAX  WORMAX   GORMAX   PAMIN  PAMAX
2000,1.25,.5,1460.,20.,500000.,150.,20000.
KSOL  MITER  OMEGA   TOL  TOL1   DSMAX   DPMAX
2,250,1.7,.1,0.0,.05,100.
RECURRENT DATA
1,1,1,1,1,1,1,1,1
1.,1.,1.,0.0,0.0,0.0
WELL P-1 PRODUCING AT 300 BOPD
1
'P-1',5,5,1,1,1,300.,0.0,0.0,0.0
.25,23.955,0
0,8,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
1.,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,0,1,1
1.,1.,1.,0.0,0.0,0.0
```

TABLE 6.4

TWO-DIMENSIONAL AREAL MODEL

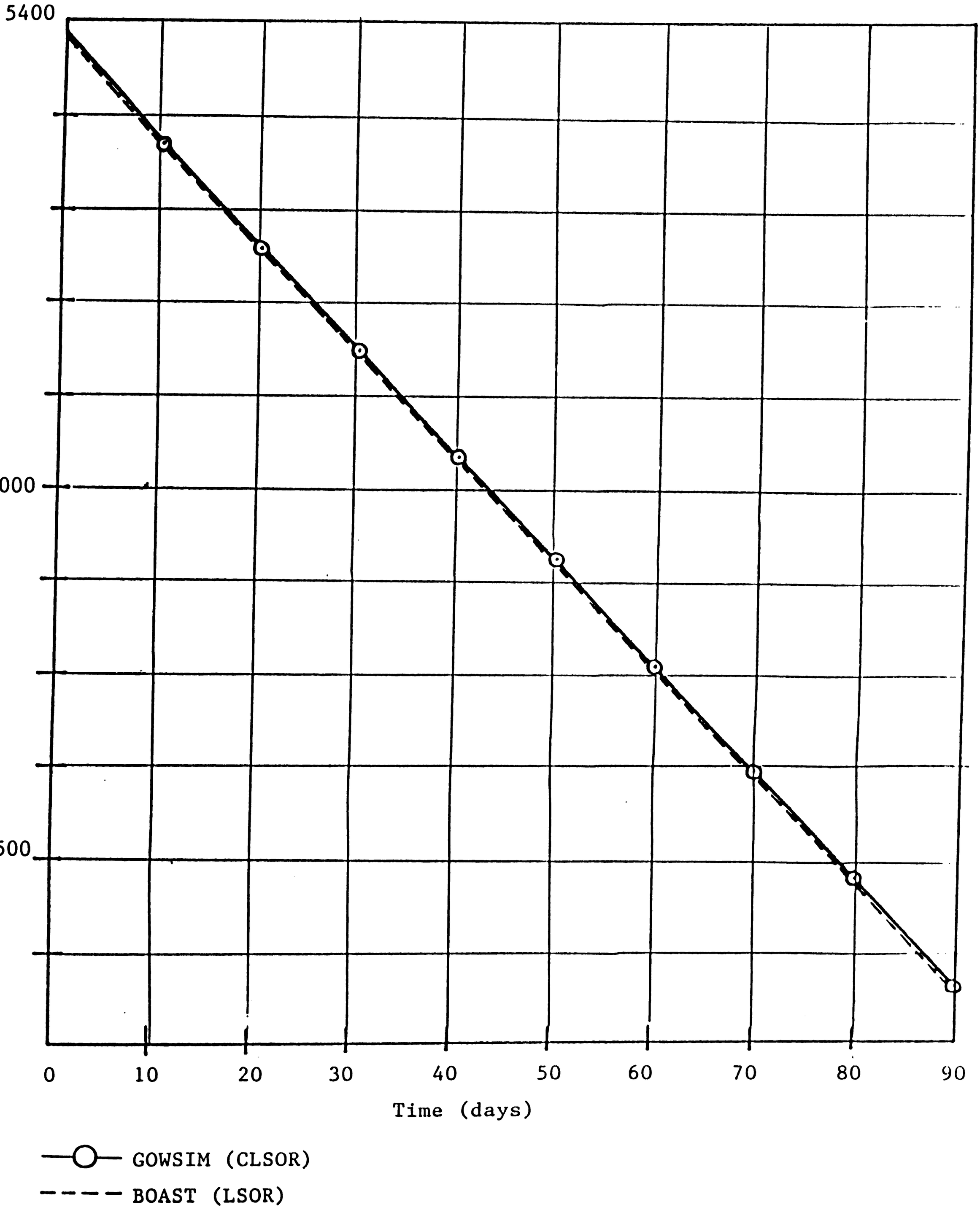
PRIMARY DEPLETION OF AN UNDERSATURATED RESERVOIR

NO.	TIME (DAYS)	P_{res}(psia)	ΣQ_{oil} (stb)
1	1.0	5372.3	300.0
2	10.0	5271.7	3000.0
3	20.0	5159.6	6000.0
4	30.0	5047.2	9000.0
5	40.0	4934.3	12000.0
6	50.0	4821.1	15000.0
7	60.0	4707.5	18000.0
8	70.0	4593.6	21000.0
9	80.0	4479.3	24000.0
10	90.0	4364.6	27000.0

FIGURE 6.4

TWO DIMENDIONSAL AREAL MODEL

PRIMARY DEPLETION OF AN UNDERSATURATED RESERVOIR



Problem 3: Cross-Section Model Showing Line-Drive Waterflooding of an Undersaturated Reservoir

A linear, cross-sectional grid model with II = 20, JJ = 1, KK = 5 contains a well at each end of grid block. Production is from a well (I = 20, J = 1) completed in layer 2, 3 and 5 while layer 4 is shut in. The well is under PI and flowing FBHP control at pressure of 4015 psia. Layer 4 is shut in effectively by setting skin's value to a very large number so that PI equals to zero. At the same time, a rate specified water injection well (Q_w = 900 STBD) is completed in layers 4 and 5 at the end (I = 1, J = 1) opposite to the production well location. The complete nature of the problem is illustrated in Fig. (6.5).

An essentially constant oil production rate is maintained in the pressure constrained production well until water-breakthrough occurs. (Fig. 6.6). Breakthrough occurs first in the lowermost layer 5, followed by layer 3 and then the uppermost layer 2. Oil production falls off rapidly after breakthrough. The simulated calculations using BOAST (LSOR) against GOWSIM (CLSOR) are plotted in Fig. (6.6). This shows good agreement except water-breakthrough occurs some days slower for GOWSIM.

Input data for this problem is presented in Table 6.5.

Note: Porosity is halved in the end grid blocks to account for the block-centered finite-difference forumulation. A vertical to horizontal permeability ratio of 0.1 is used so that too much vertical fluid migration is restricted.

FIGURE (6.5) Line Drive Waterflood

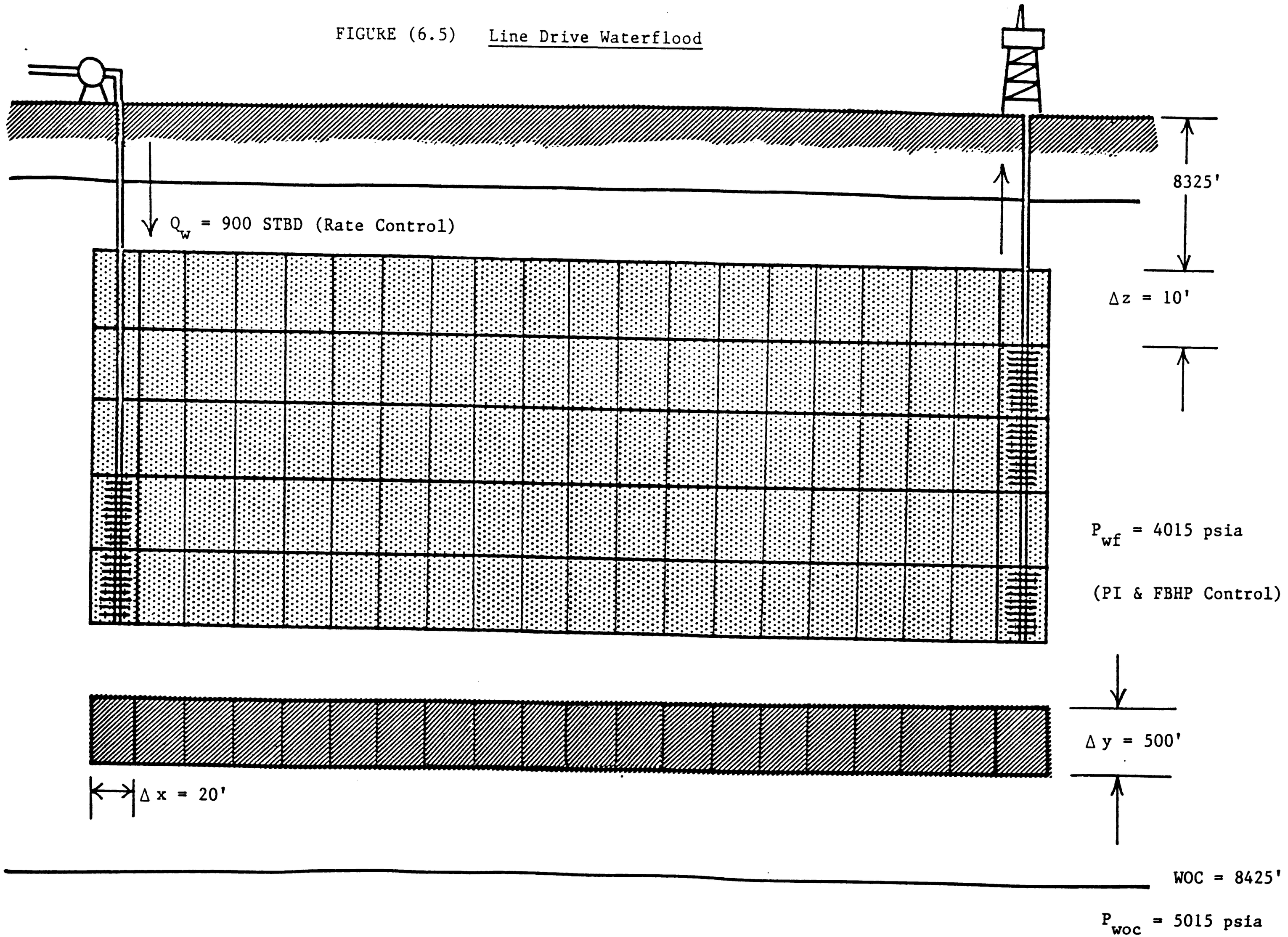


TABLE 6.5 INPUT DATA FOR PROBLEM 3

```
      LINE DRIVE WATERFLOOD: CROSS-SECTION RUN
20,1,5
GRID BLOCK LENGTHS
-1,-1,-1
20.
500.
10.
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
CAPROCK BASE DEPTHS
0
8325.
POROSITY AND PERMEABILITY
-1,-1,-1,-1
.25
200.
200.
20.
POROSITY AND PERMEABILITY MODIFICATION CARDS
10,0,0,0,1
1,1,1,.125
1,1,2,.125
1,1,3,.125
1,1,4,.125
1,1,5,.125
20,1,1,.125
20,1,2,.125
20,1,3,.125
20,1,4,.125
20,1,5,.125
TRANSMISSIBILITY MODIFICATIONS
0,0,0,0
  SAT     KRO      KRW       KRG         PCOW        PCGO
-.1,0.0,0.0,0.0,0.0,0.0
.02,0.0,0.0,0.0,0.0,0.0
.1,0.0,0.0,.025,0.0,0.0
.2,.00147,0.0,0.075,0.0,0.0
.3,.00228,.0122,.19,0.0,0.0
.4,.037,.0244,.41,0.0,0.0
.5,.0571,.0336,.72,0.0,0.0
.6,.134,.0672,.87,0.0,0.0
.7,.207,.1344,.94,0.0,0.0
.8,.604,.2688,.9666667,0.0,0.0
.9,1.,.4704,.9933333,0.0,0.0
1.1,1.,.5,1.,0.0,0.0
   PBO     VSLOPE      BSLOPE        RSLOPE        PMAX
4014.7,.000046,-.0000232,0.0,9014.7,0
    P        MUO          BO          RSO
14.7,1.04,1.062,1.
1014.7,.83,1.295,371.
2014.7,.695,1.435,636.
3014.7,.594,1.565,930.
4014.7,.51,1.695,1270.
5014.7,.449,1.827,1618.
9014.7,.203,2.357,2984.
    P          MUW          BW          RSW
14.7,.5,1.019,0.0
1014.7,.501,1.016,0.0
2014.7,.502,1.013,0.0
4014.7,.505,1.007,0.0
6014.7,.51,1.001,0.0
9014.7,.52,.992,0.0
    P          MUG          BG          CR
14.7,.008,.9358,.000003
264.7,.0096,.067902,.000003
514.7,.0112,.035228,.000003
1014.7,.014,.017951,.000003
2014.7,.0189,.009063,.000003
2514.7,.0208,.007266,.000003
3014.7,.0228,.006064,.000003
4014.7,.0268,.004554,.000003
5014.7,.0309,.003644,.000003
9014.7,.047,.002167,.000003
    RHOSCO        RHOSCW        RHOSCG
46.244,62.238,.0647
EQUILIBRIUM PRESSURE INITIALIZATION/CONSTANT SATURATIONS
0,0
5015.,0.0,8425.,8300.
.8,.2,0.0
KSN1     KSM1     KCO1      KTR     KCOFF
0,0,0,0,0
```

TABLE 6.5 Continued

```
NMAX  FACT1  FACT2   TMAX   WORMAX   GORMAX   PAMIN   PAMAX
500,1.2,.5,365.,20.,500000.,150.,10000.
KSOL   MITER   OMEGA   TOL   TOL1   DSMAX   DPMAX
2,100,1.7,.1,0.0,.05,100.
RECURRENT DATA
1,1,1,1,1,1,1,0,0
1.,.1,10.,0.0,0.0,0.0
RATES
2
'PROD1',20,1,2,4,-1,600.,0.0,0.0,0.0
.25,24.44,4015.
.25,15.,4015.
.25,50000.0,4015.
.25,52.76,4015.
'INJ-1',1,1,4,2,2,0.0,-900.,0.0,0.0
.25,52.76,0.0
.25,24.44,0.0
0,8,0,0,0,0,0,0,0
1.,.1,10.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
1.,.1,10.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
2.,.1,10.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
2.,.1,10.,0.0,0.0,0.0
0,9,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,14,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
0,4,0,0,0,0,0,0,0
6.,.1,30.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
6.,.1,30.,0.0,0.0,0.0
```

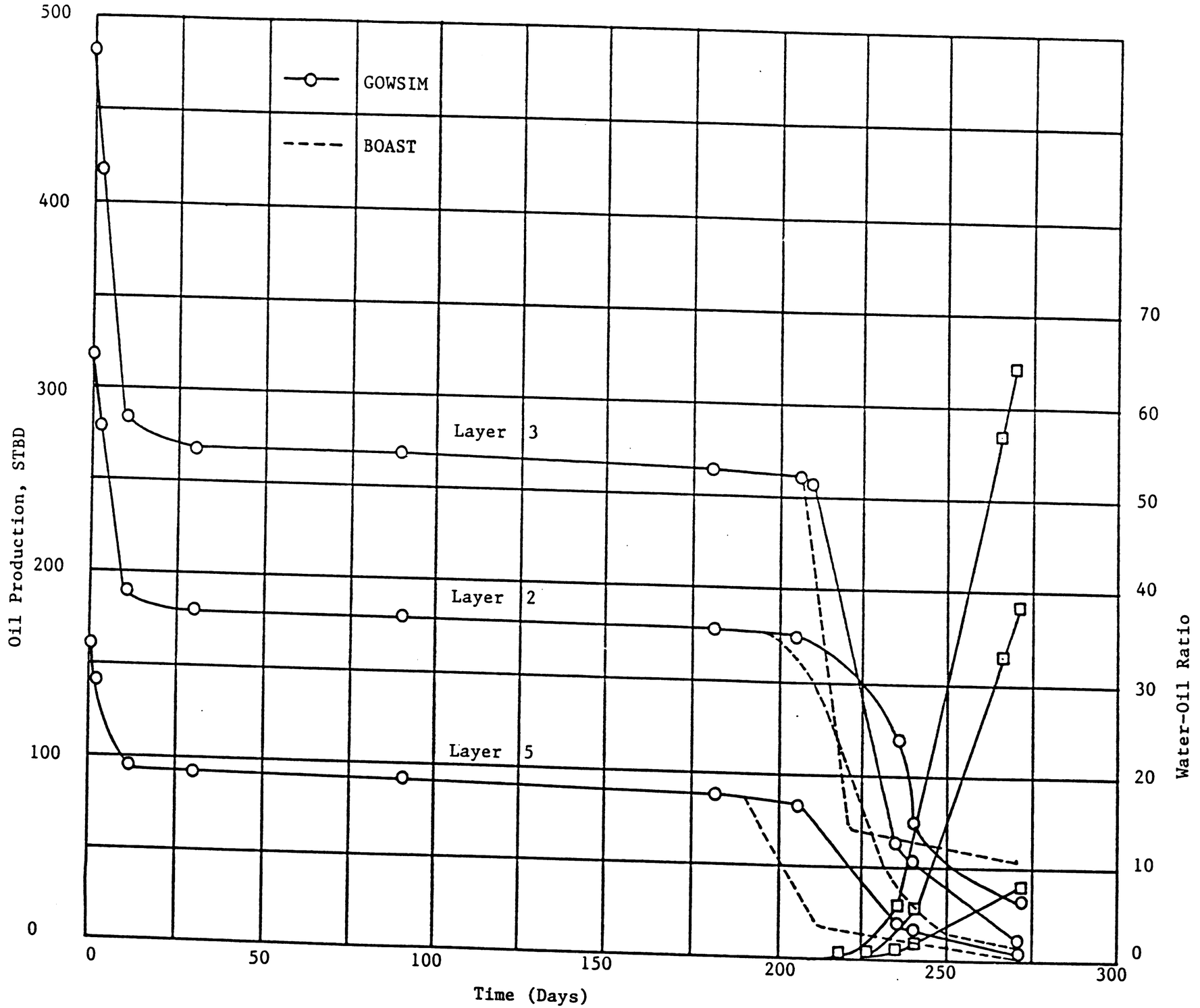

GOWSIM
BOAST
Layer 3
Layer 2
Layer 5
Oil Production, STBD
Water-Oil Ratio
Time (Days)
500
400
300
200
100
0
70
60
50
40
30
20
10
0
0
50
100
150
200
250
300

FIGURE (6.6)

Problem 4: Two-Dimensional, x-z Cross-Section Model Showing Production from Oil-Patch and Gas-Patch Reservoirs with Fluid Contact Levels at Different Depths.

In this problem, an oil well and a gas well are producing under a rate constraint of 100 STBD and 10 MSCFD respectively. Oil production is from the well (I = 3, J = 1) completed in the uppermost layer while gas production from the well located at (I = 9, J = 1, K = 1). Equilibrium pressure initializations at constant saturations initialization are made by reading the pressures and saturations block-by-block basis. The oil-water contact and gas-water contact are found at the depths of 8405 ft and 8445 ft respectively. The complete nature of the problem is illustrated in Fig. (6.7). Input data for this problem is presented in Table 6.6. The calculations are plotted in Fig. (6.8) for average reservoir pressure versus time. An approximately constant pressure decline in the reservoir is maintained throughout the time. Care must be taken to use small time steps in this problem.

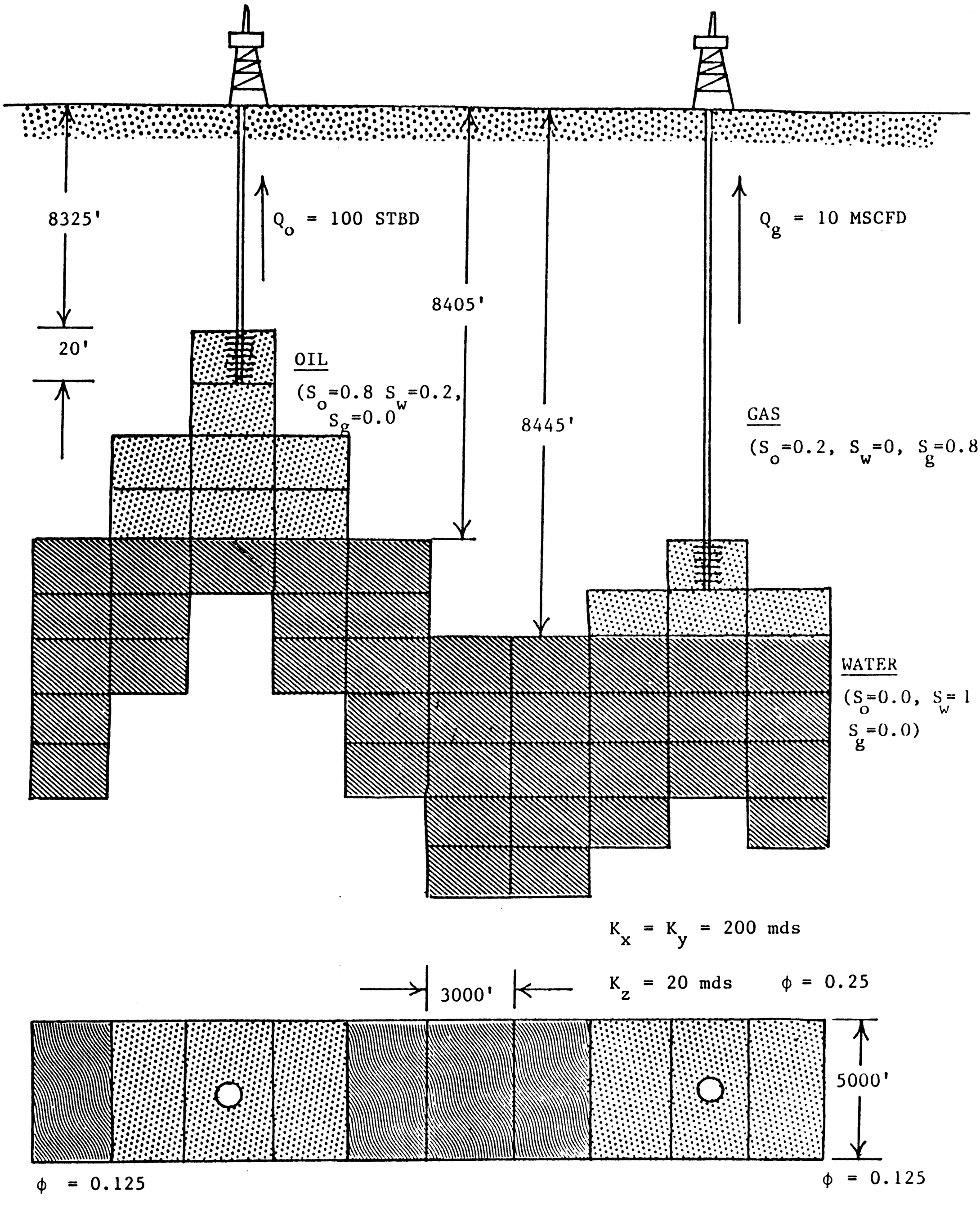

8325'
20'
Q_o = 100 STBD
8405'
8445'
Q_g = 10 MSCFD
OIL
(S_o=0.8 S_w=0.2, S_g=0.0
GAS
(S_o=0.2, S_w=0, S_g=0.8)
WATER
(S_o=0.0, S_w=1 S_g=0.0)
K_x = K_y = 200 mds
K_z = 20 mds
ϕ = 0.25
3000'
5000'
ϕ = 0.125
ϕ = 0.125

FIGURE (6.7)

TABLE 6.6 INPUT DATA FOR PROBLEM 4

```
2D, CROSS-SECTIONAL 1 OIL WELL PLUS I GAS WELL (RATE CONSTR)
10,1,5
GRID BLOCK LENGTHS
-1,-1,-1
3000.
5000.
20.
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
CAPROCK BASE DEPTHS
1
8405.,8365.,8325.,8365.,8405.,8445.,8445.,8425.,8405.,8425.
POROSITY AND PERMEABILITY
-1,-1,-1,-1
.25
200.
200.
20.
POROSITY AND PERMEABILITY MODIFICATION CARDS
10,0,0,0,1
1,1,1,.125
1,1,2,.125
1,1,3,.125
1,1,4,.125
1,1,5,.125
10,1,1,.125
10,1,2,.125
10,1,3,.125
10,1,4,.125
10,1,5,.125
TRANSMISSIBILITY MODIFICATIONS
0,0,0,0
  SAT     KRO      KRW       KRG         PCOW       PCGO
-.1,0.0,0.0,0.0,0.0,0.0
.1,0.0,0.0,0.0,0.0,0.0
.2,.00147,0.0,.075,0.0,0.0
.3,.00228,.0122,.19,0.0,0.0
.4,.037,.0244,.41,0.0,0.0
.5,.0571,.0336,.72,0.0,0.0
.6,.134,.0672,.87,0.0,0.0
.7,.207,.1344,.94,0.0,0.0
.8,.604,.2688,.9667,0.0,0.0
.9,1.,.4704,.9933,0.0,0.0
1.1,1.,.5,1.,0.0,0.0
   PBO     VSLOPE      BSLOPE        RSLOPE        PMAX
4014.7,0.0,-.000001,0.0,9014.7,0
    P        MUO         BO          RSO
14.7,2.,1.5,1.
4014.7,2.,1.5,1.
9014.7,2.,1.5,1.
    P        MUW         BW          RSW
14.7,1.,1.,0.0
4014.7,1.,1.,0.0
9014.7,1.,1.,0.0
    P        MUG         BG          CR
14.7,.008,.9358,.000003
264.7,.0096,.067902,.000003
514.7,.0112,.035228,.000003
1014.7,.014,.017951,.000003
2014.7,.0189,.009063,.000003
2514.7,.0208,.007266,.000003
3014.7,.0228,.006064,.000003
4014.7,.0268,.004554,.000003
5014.7,.0309,.003644,.000003
9014.7,.047,.002167,.000003
    RHOSCO      RHOSCW      RHOSCG
46.244,62.238,.0647
EQUILIBRIUM PRESSURE INITIALIZATION/CONSTANT SATURATIONS
1,1
4819.,4809.,4800.,4809.,4819.,4837.,4837.,4831.,4829.,4831
4828.,4813.,4804.,4813.,4828.,4845.,4845.,4837.,4831.,4837.
4837.,4819.,4809.,4819.,4837.,4854.,4854.,4845.,4837.,4845.
4845.,4828.,4813.,4828.,4845.,4863.,4863.,4854.,4845.,4854.
4854.,4837.,4819.,4837.,4854.,4871.,4871.,4863.,4854.,4863.
0.0,.8,.8,.8,0.0,0.0,0.0,.2,.2,.2
0.0,.8,.8,.8,0.0,0.0,0.0,0.0,.2,0.0
0.0,0.0,.8,0.0,0.0,0.0,0.0,0.0,0.0,0.0
0.0,0.0,.8,0.0,0.0,0.0,0.0,0.0,0.0,0.0
0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
1.,.2,.2,.2,1.,1.,1.,0.0,0.0,0.0
1.,.2,.2,.2,1.,1.,1.,1.,0.0,1.
1.,1.,.2,1.,1.,1.,1.,1.,1.,1.
1.,1.,.2,1.,1.,1.,1.,1.,1.,1.
1.,1.,1.,1.,1.,1.,1.,1.,1.,1.
KSN1    KSM1     KCO1      KTR     KCOFF
0,0,0,0,0
```

TABLE 6.6 Continued

```
NMAX  FACT1  FACT2   TMAX   WORMAX   GORMAX    PAMIN    PAMAX
500,1.2,.5,120.,20.,500000.,14.7,10000.
KSOL   MITER   OMEGA   TOL   TOL1     DSMAX     DPMAX
2,100,1.7,.1,0.0,.05,100.
RECURRENT DATA
1,1,1,1,1,1,1,1,1
1.,1.,1.,0,0,0
RATES
2
'PROD'3,1,1,1,1,100.,0.0,0.0,0.0
.25,0.0,0.0
'PRO2',9,1,1,1,3,0.0,0.0,10.0,0.0
.25,0.0,0.0
0,5,0,0,0,0,0,0,0
1.,1.,1.,0,0,0
0,1,1,1,1,1,1,0,0
1.,.1,30.,0,0,0
0,30,0,0,0,0,0,0,0
1.,.1,30.,0,0,0
0,1,1,1,1,1,1,0,0
1.,.1,30.,0,0,0
0,15,0,0,0,0,0,0,0
1.,.1,30.,0,0,0
0,1,1,1,1,1,1,0,0
1.,.1,30.,0,0,0
0,15,0,0,0,0,0,0,0
1.,.1,30.,0,0,0
0,1,1,1,1,1,1,0,0
1.,.1,30.,0,0,0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
```

TABLE 6.6 Continued

```
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,5,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,10,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,1,1,1,1,1,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,1,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
0,15,0,0,0,0,0,0,0
.25,1.,1.,0.0,0.0,0.0
```

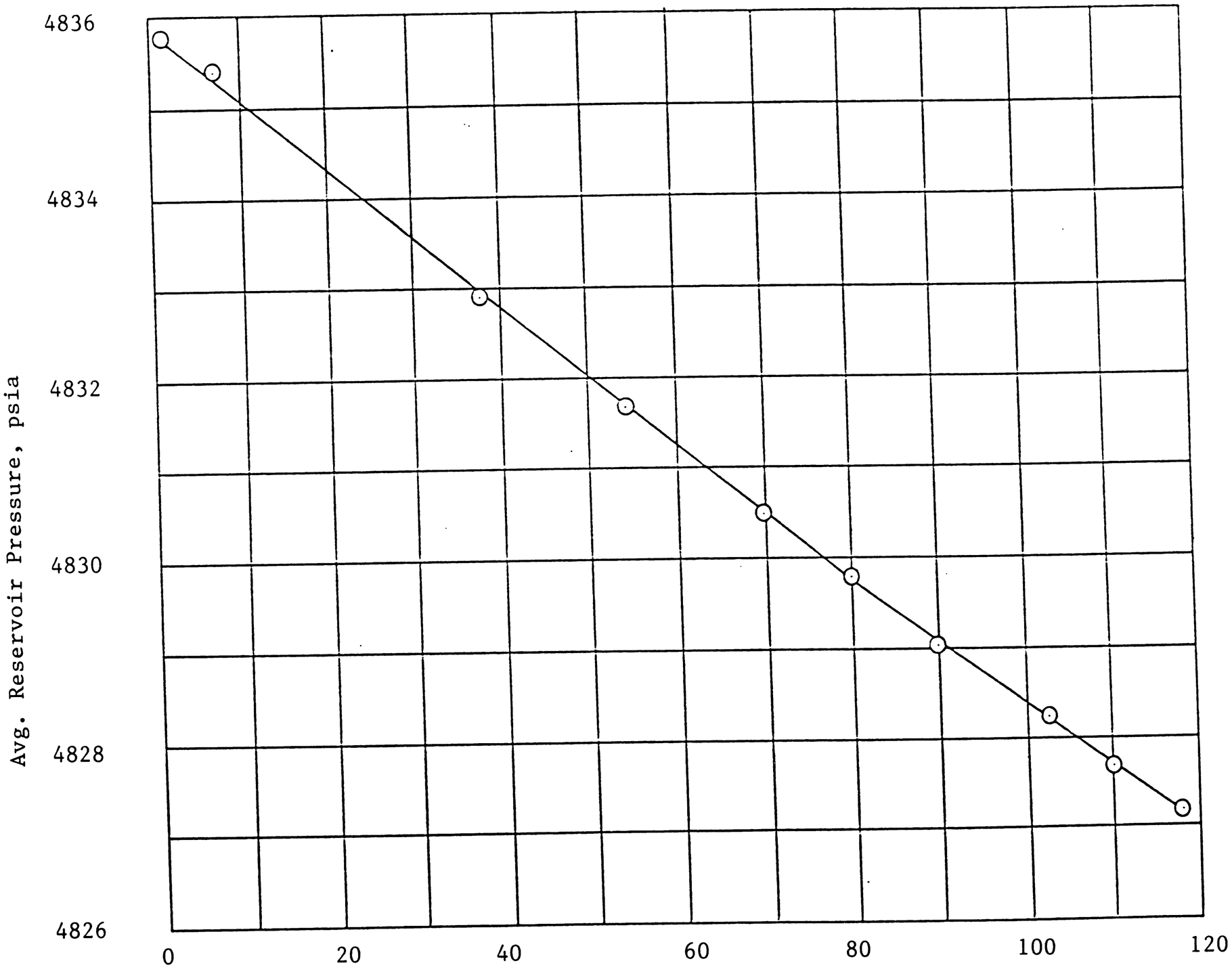

4836
4834
4832
4830
4828
4826
Avg. Reservoir Pressure, psia
0
20
40
60
80
100
120
Time, Days

FIGURE (6.8)

Problem 5: Linear, Three-Dimensional, Two-Phae Model Showing a Single Gas Well Producing from Gas-Water Reservoir

A linear, three-dimensional model grid (II = 5, JJ = 5, KK = 5) contains a single gas well at the centre of top layer producing under a rate constraint of 25,000 MSCFD. The well production is from layer 1 only. The depth to top of horizontal layer 1 is 8325 ft. The gas-water contact is located at the bottom of layer (2).The complete picture of the problem is illustrated in Fig. (6.9). The performance of a gas-water reservoir with the pertinent data given in Table 6.7 is studied using GWSIM simulator. The reservoir's pressure and temperature are 4800 psia and 180°F respectively. The pressure at the gas-water contact is found to be 4806.6 psia. The input data for this problem is shown in Table 6.8.

The average reservoir pressures after different elapsed times of production are calculated. Then the cummulative gas production verus P/Z are plotted in Fig. (6.10). The straight line nature of the graph clearly states that two-phase gas-water simulator can be used as a single phase reservoir simulator provided that the GWC should be far below the production location.

FIGURE (6.9) Gas-Water Reservoir

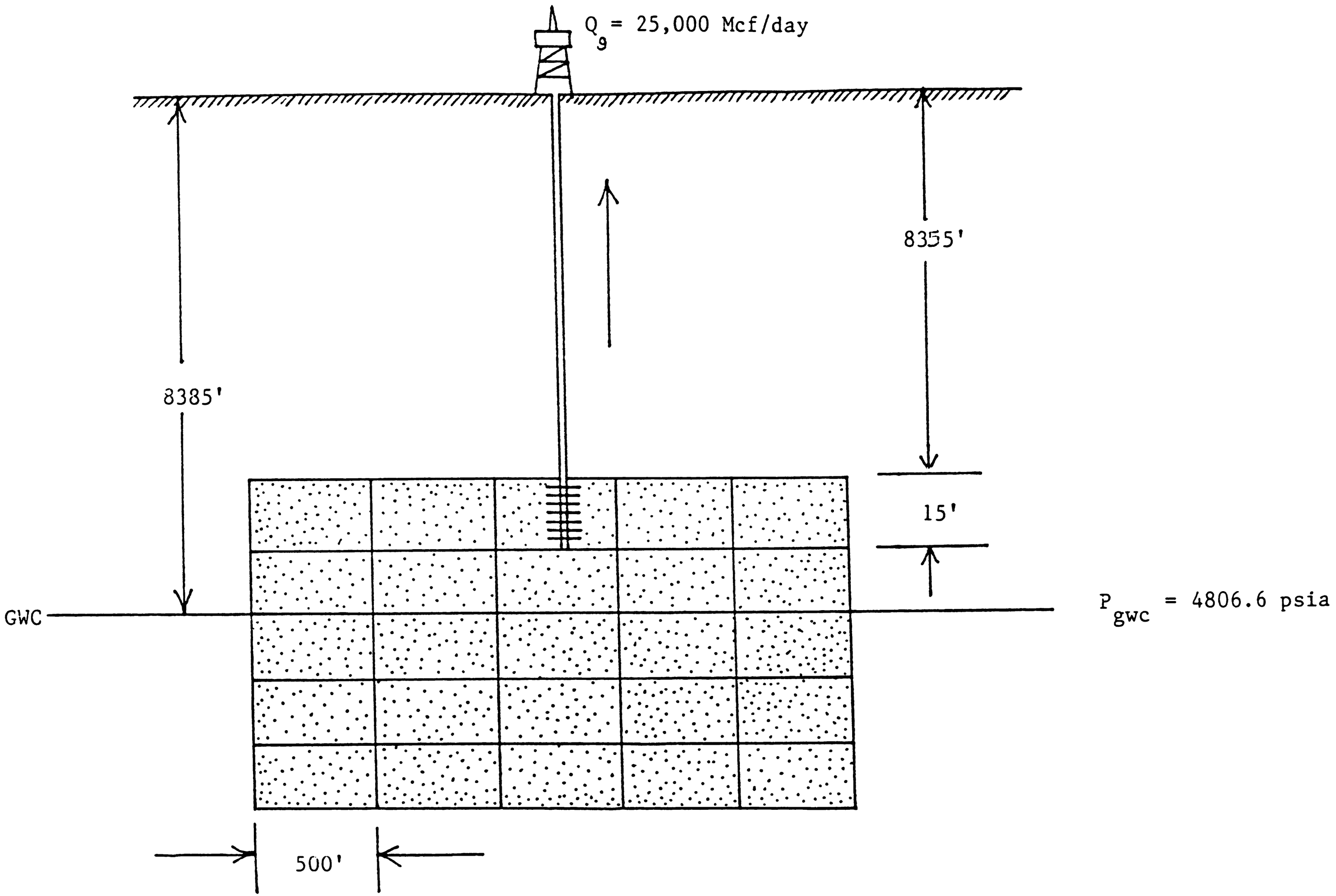


TABLE 6.7

GAS-WATER RESERVOIR SIMULATION

Data: $\bar{P}_{res}$ = 4800 psia

$\bar{T}_{res}$ = 180°F = 180 + 460 = 640°R

	M	y	P_c	T_c	M	P_c	T_c
CH_4	16	0.9	668	343	14.4	601.2	308.7
C_2H_6	30	0.07	708	550	2.1	49.56	38.5
C_3H_8	44	0.03	616	660	1.32	18.48	19.8
					17.82	669.24	367.0

$$\text{Gas Gravity} = \frac{17.82}{29} = 0.614$$

TABLE 6.7 Continued

At Constant Average Reservoir Temperature of 180°F

P	Z_{res}	$B_g = 18.112\ Z/P^*$
264.7	0.975	0.6671×10^{-1}
514.7	0.96	0.3378×10^{-1}
1014.7	0.935	0.1669×10^{-1}
2014.7	0.89	0.8001×10^{-2}
2514.7	0.885	0.6374×10^{-2}
3014.7	0.89	0.5347×10^{-2}
4014.7	0.94	0.4241×10^{-2}
5014.7	1.0	0.3612×10^{-2}
9014.7	1.32	0.2652×10^{-2}

After a Simulation Time of 1 Year:

P	Q_g	Z	P/Z
4603.3	0.675×10^6	0.98	4697
4362.7	1.425×10^6	0.96	4544
4102.8	2.175×10^6	0.94	4365
3894.8	2.925×10^6	0.93	4188
3717.1	3.675×10^6	0.925	4018
3523.3	4.425×10^6	0.915	3849
3309.9	5.175×10^6	0.905	4018
3075.1	5.925×10^6	0.894	3441
2898.1	6.675×10^6	0.889	3261
2739.1	7.425×10^6	0.887	3087
2559.7	8.175×10^6	0.885	2891
2409.2	8.925×10^6	0.887	2716

*$B_g = 0.0283\ TZ/P = 0.0283 \times 640\ Z/P = 18.112\ Z/P$

TABLE 6.8 INPUT DATA FOR PROBLEM 5

THREE DIMENSIONAL, GAS-WATER RESERVOIR SIMULATOR

```
5,5,5
GRID BLOCK LENGTHS
-1,-1,-1
1500.0
1500.0
15.0
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
CAPROCK BASE DEPTHS
-1
8355.
POROSITY AND PERMEABILITY
-1,-1,-1,-1
0.25
200.0
200.0
20.0
POROSITY AND PERMEABILITY MODIFICATION CARDS
0,0,0,0,0
TRANSMISSIBILITY MODIFICATINS
0,0,0,0
      SAT           KRW          KRO          PCOW
-0.1,0.0,0.0,0.99
0.1,0.0,0.0,.99
0.2,0.0,0.075,.99
0.3,.0122,.19,.85
0.4,.0244,.41,.65
0.5,.0366,.72,.45
0.6,.0672,.87,.395
0.7,.1344,.94,.32
0.8,.2688,.9667,.25
0.9,.4704,.9935,0.0
1.1,0.5,1.0,0.0
   PMAX
9014.7,0
       P             VISW         BW             RSW
14.7,1.0,1.0,0.0
4014.7,1.0,1.0,0.0
9014.7,1.0,1.0,0.0
       P             VISO         BO                CR
14.7,0.008,0.9358,0.000003
264.7,0.0096,0.06671,0.000003
514.7,0.0112,0.03378,0.000003
1014.7,0.014,0.01667,0.000003
2014.7,0.0189,0.008001,0.000003
2514.7,0.0208,0.006374,0.000003
3014.7,0.0228,0.005347,0.000003
4014.7,0.0268,0.004241,0.000003
5014.7,0.0309,0.003612,0.000003
9014.7,0.047,0.002652,0.000003
  RHOSCW       RHOSCO
62.238,0.0647
EQUILIBRIUM PRESSURE INITIALIZATION/CONST SAT
0,0
4806.6,8385.0
0.20
```

TABLE 6.8 Continued

```
KSN1    KSM1      KCO1      KTR       KCOFF
0,0,,0,0,0
NMAX   FACT1    FACT2     TMAX      WGRMAX     PAMIN     PAMAX
500,1.2,0.5,3650.0,0.000001,14.7,100000.0
KSOL   NITER   OMEGA    TOL     TOL1     DSMAX     DPMAX
2,100,1.7,0.1,0.0,0.05,150.0
RECURRENT DATA
1,1,1,1,1,1,1
3.,3.,3.
RATES
1
'PROD',3,3,1,1,3,0.0,25000.0,0.0
.25,23.955,0.0
0,7,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
```

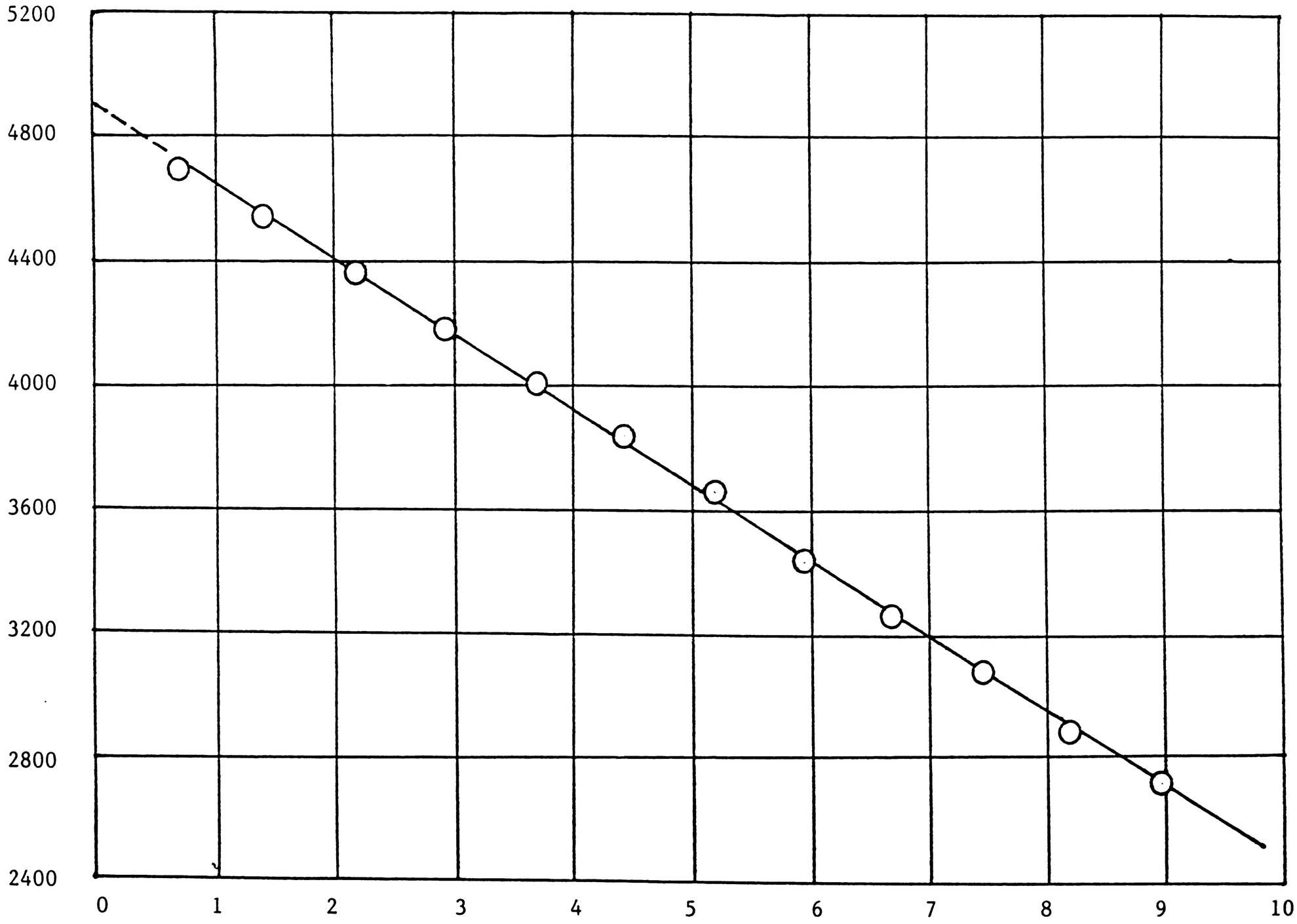

5200
4800
4400
4000
3600
3200
2800
2400
P/Z
0
1
2
3
4
5
6
7
8
9
10
Qg, Cummulative Gas Production, MSCF x 10^6

FIGURE (6.10)

Problem 6: Two-Dimensional, Cross-Section Model Grid of Gas-Water Reservoir With Two Different Fluid Contact Levels

In this problem, cross section model grid (II = 10, JJ = 1, KK = 3) contains 2 gas wells at the location (3, 1, 1) and (9, 1, 1). The grid dimensions are $\Delta x = 3000'$, $\Delta y = 3000'$, $\Delta z = 50'$. The well located at (3, 1, 1) is under a rate constraint of 10 MSCFD while the other well is under a PI and FBHP control at a pressure of 4000 psia. The well block pressures and saturations are read in block-by-block basis to have two different fluid contact levels in the model grid. The complete picture of the problem is illustrated in Figure (6.11). The input data file for this problem is shown in Table 6.9.

The average reservoir pressures are reported in table and plotted against elapsed time of simulation in Figure (6.12). The method of solution is BAND.

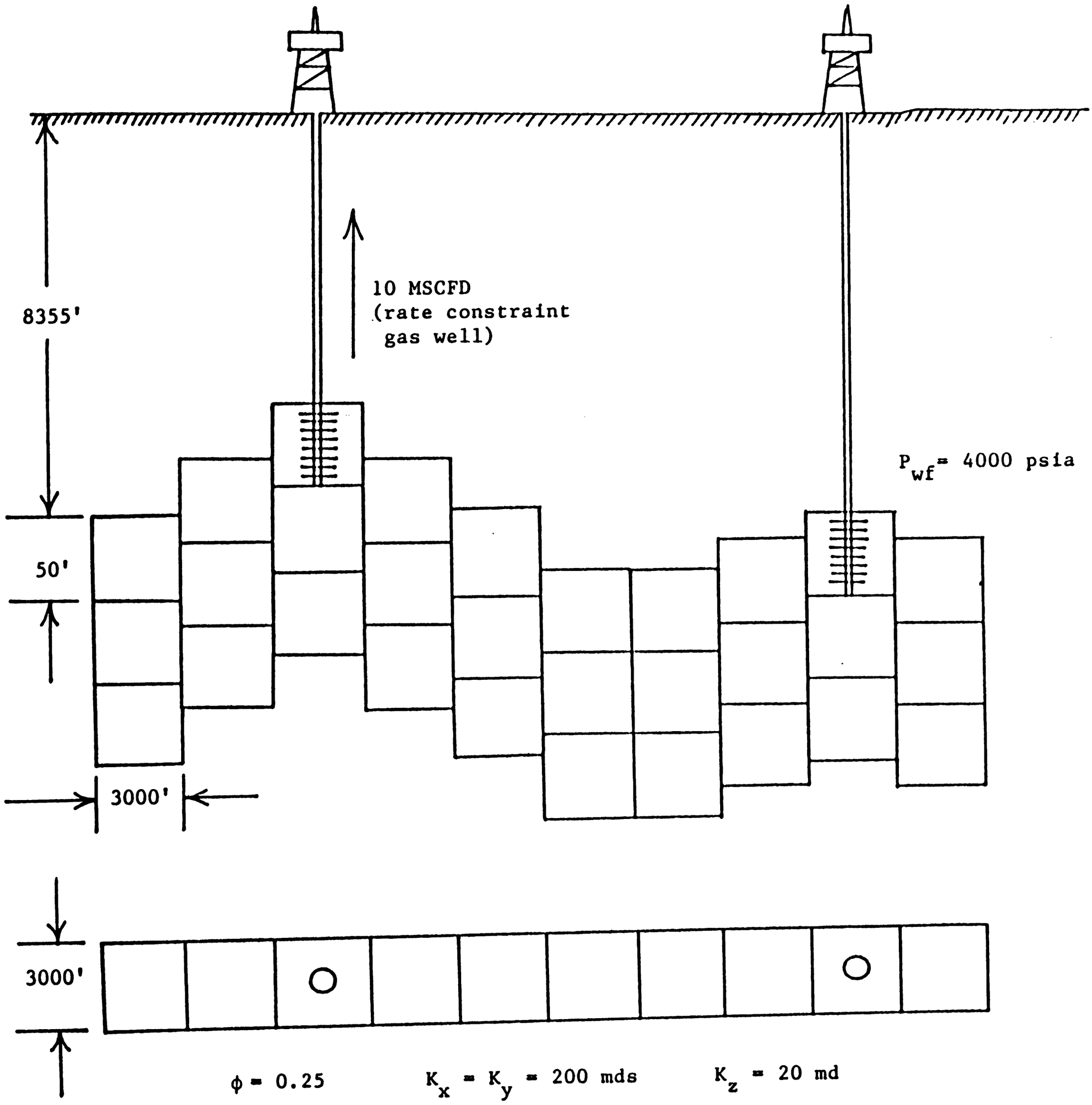

8355'
10 MSCFD
(rate constraint
gas well)
P_{wf} = 4000 psia
50'
3000'
3000'
ϕ = 0.25
$K_x = K_y$ = 200 mds
K_z = 20 md

FIGURE (6.11)

TABLE 6.9 INPUT DATA FOR PROBLEM 6

```
                  THREE DIMENSIONAL, GAS-WATER RESERVOIR SIMULATOR
                  ------------------------------------------------
10,1,3
GRID BLOCK LENGTHS
-1,-1,-1
3000.0
3000.0
50.0
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
CAPROCK BASE DEPTHS
1
8355.,8325.,8295.,8325.,8355.,8385.,8385.,8370.,8355.,8370.
POROSITY AND PERMEABILITY
-1,-1,-1,-1
0.25
200.0
200.0
20.0
POROSITY AND PERMEABILITY MODIFICATION CARDS
6,0,0,0,0
1,1,1,.1
1,1,2,.1
1,1,3,.1
10,1,1,.1
10,1,2,.1
10,1,3,.1
TRANSMISSIBILITY MODIFICATINS
0,0,0,0
    SAT         KRW        KRG         PCGW
-0.1,0.0,0.0,0.99
0.1,0.0,0.0,.99
0.2,0.0,0.075,.99
0.3,.0122,.19,.85
0.4,.0244,.41,.65
0.5,.0366,.72,.45
0.6,.0672,.87,.395
0.7,.1344,.94,.32
0.8,.2688,.9667,.25
0.9,.4704,.9935,0.0
1.1,0.5,1.0,0.0
  PMAX
9014.7,0
    P           VISW       BW          RSW
14.7,1.0,1.0,0.0
4014.7,1.0,1.0,0.0
9014.7,1.0,1.0,0.0
     P          VISG       BG               CR
14.7,0.008,0.9350,0.000003
264.7,0.0096,0.06671,0.000003
514.7,0.0112,0.03378,0.000003
1014.7,0.014,0.01669,0.000003
2014.7,0.0189,0.008001,0.000003
2514.7,0.0208,0.006374,0.000003
3014.7,0.0228,0.005347,0.000003
4014.7,0.0260,0.004241,0.000003
5014.7,0.0309,0.003612,0.000003
9014.7,0.047,0.002652,0.000003
  RHOSCW     RHOSCG
62.238,0.0647
EQUILIBRIUM PRESSURE INITIALIZATION/CONST SAT
1,1
4818.,4812.,4809.,4812.,4818.,4831.,4831.,4827.,4825.,4827.
4824.,4814.,4810.,4814.,4824.,4837.,4837.,4831.,4827.,4831.
4831.,4818.,4812.,4818.,4831.,4844.,4844.,4837.,4831.,4837.
1.,.2,.2,.2,1.,1.,1.,.2,.2,.2
1.,.2,.2,.2,1.,1.,1.,1.,.2,1.
1.,1.,.2,1.,1.,1.,1.,1.,1.,1.
KSN1    KSM1     KC01     KTR      KCOFF
0,0,,0,0,0
NMAX  FACT1    FACT2     TMAX     WGRMAX    PAMIN    PAMAX
500,1.2,0.5,3650.0,0.000001,14.7,100000.0
KSOL  MITER  OMEGA    TOL     TOL1    DSMAX    DPMAX
1,100,1.7,0.1,0.0,0.05,150.0
RECURRENT DATA
1,1,1,1,1,1,1
3.,3.,3.
RATES
2
'PRO1',3,1,1,1,3,0.0,10.0,0.0
.25,23.955,0.0
'PRO2',9,1,1,1,-3,0.0,0.0,0.0
.25,23.955,4000.
0,7,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0,0
3.,3.,3.
0,1,1,1,1,1,1
3.,3.,3.
```

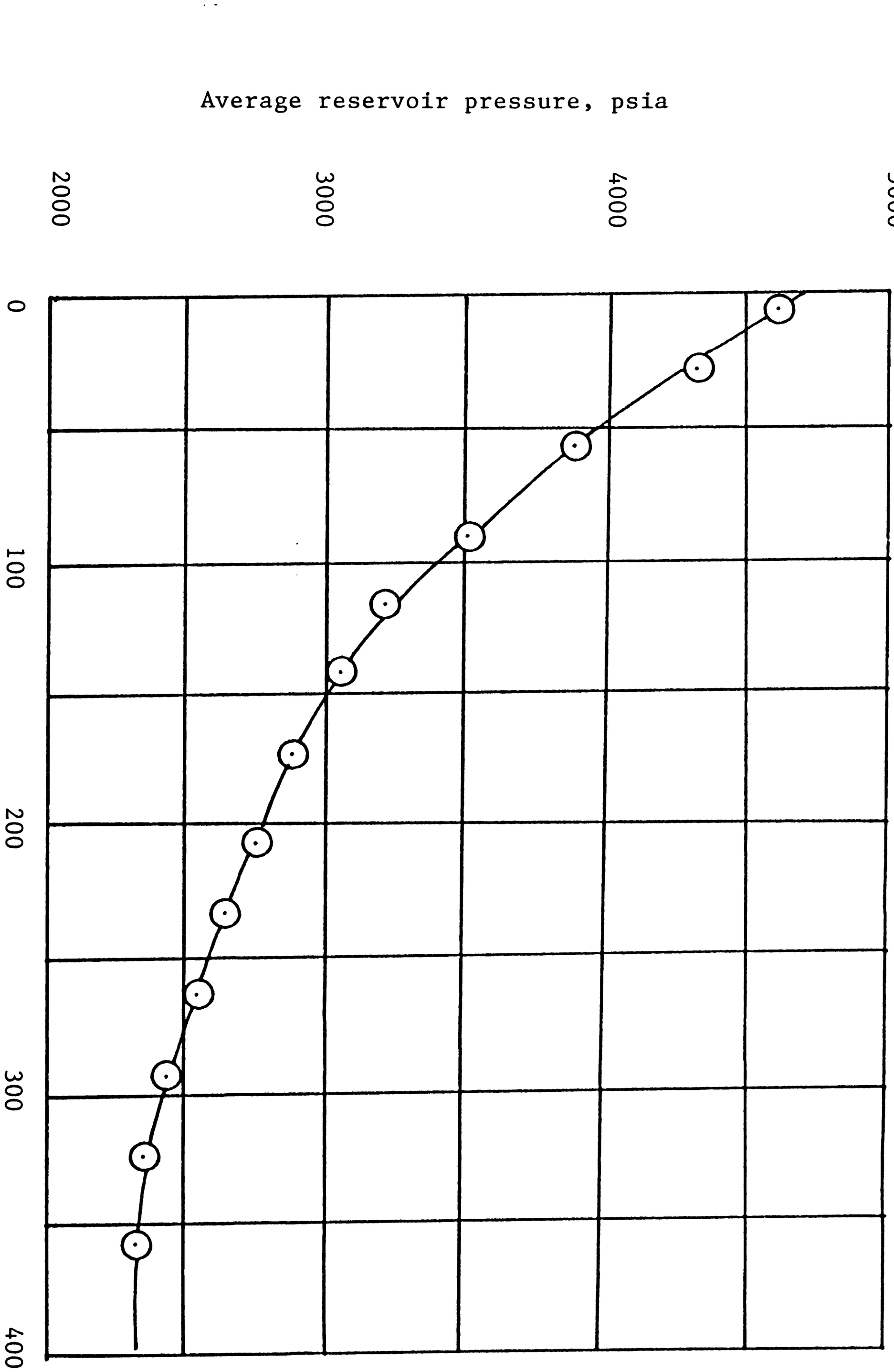

Average reservoir pressure, psia
2000
3000
4000
5000
0
100
200
300
400
Time, days

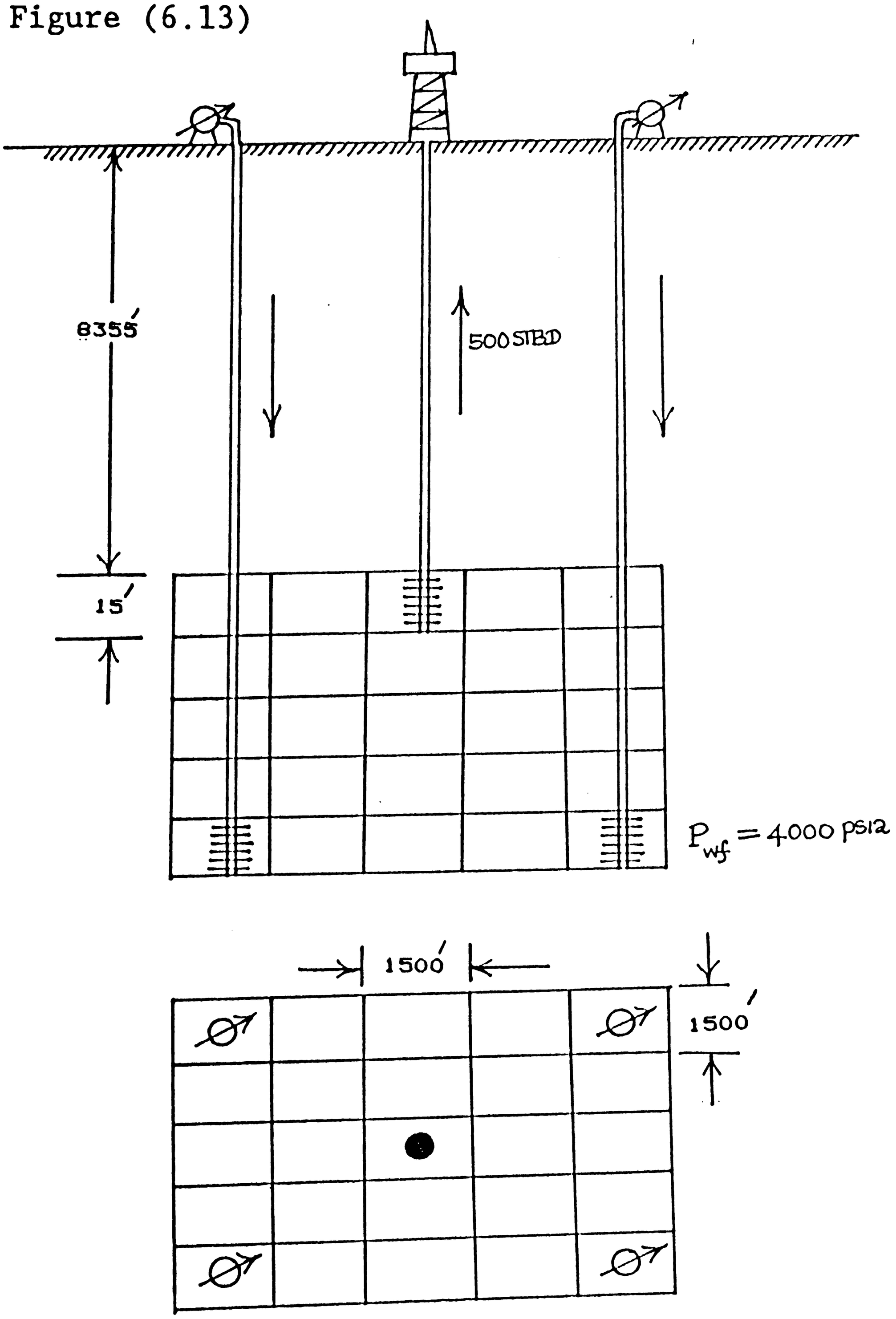
FIGURE (6.12)

Problem 7: Three-Dimensional Water Reservoir Model

In this problem, three-dimensional model grid (II = 5, JJ = 5, KK = 5) contains 5 water wells of which one well is producing water under a rate constraint of 500 STBD while the rest are injection wells under the PI and FBHP control at a pressure of 4000 psia. The grid dimensions and other pertinent data are shown in Figure (6.13). The input data for this problem is shown in Table (6.10).

Solution method for this problem is BAND. The average reservoir pressure distribution through simulation time is shown in Table (6.11). It is plotted in Figure (6.14).

Figure (6.13)



producing water

injection wells

TABLE 6.10 INPUT DATA FOR PROBLEM 7

```
                THREE DIMENSIONAL, WATER RESERVOIR SIMULATOR
                --------------------------------------------
5,5,5
GRID BLOCK LENGTHS
-1,-1,-1
1500.0
1500.0
15.0
GRID BLOCK LENGTH MODIFICATIONS
0,0,0,0
CAPROCK BASE DEPTHS
0
8355.0
POROSITY AND PERMEABILITY
-1,-1,-1,-1
0.25
200.0
200.0
20.0
POROSITY AND PERMEABILITY MODIFICATION CARDS
0,0,0,0,0
TRANSMISSIBILITY MODIFICATINS
0,0,0,0
  PMAX
9014.7
     P         VISW      BW        CRT
14.7,1.0,1.0,.000003
4014.7,1.0,1.0,.000003
9014.7,1.0,1.0,.000003
  RHOSCW
62.238
EQUILIBRIUM PRESSURE INITIALIZATION/
0,0
4806.6,8355.0
0.2
KSN1   KSM1    KCO1    KTR     KCOFF
0,0,0,0,0
NMAX  FACT1   FACT2    TMAX        PAMIN    PAMAX
500,1.2,0.5,365.0,14.7,100000.0
KSOL  MITER  OMEGA   TOL   TOL1         DSMAX      DPMAX
1,100,1.7,0.1,0.0,0.05,150.0
RECURRENT DATA
1,1,1,1,1,1
3.,3.,3.
RATES
5
'PROD',3,3,1,1,2,500.0
.25,23.955,0.0
'INJ',1,1,5,1,-2,0.0
.25,23.955,4000.0
'INJ',5,1,5,1,-2,0.0
.25,23.955,4000.0
'INJ',1,5,5,1,-2,0.0
.25,23.955,4000.0
'INJ',5,5,5,1,-2,0.0
.25,23.955,4000.0
0,7,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
0,9,0,0,0,0
3.,3.,3.
0,1,1,1,1,1
3.,3.,3.
```

TABLE 6.11

AVERAGE RESERVOIR PRESSURE VS ELAPSED TIME OF SIMULATION

Time, days	Avg. Res. Pressure, Psia
3	4819.75
27	4793.68
57	4759.93
87	4730.36
117	4703.64
147	4677.21
177	4646.92
207	4613.90
237	4581.68
267	4556.00
297	4529.70
327	4501.13
357	4468.57

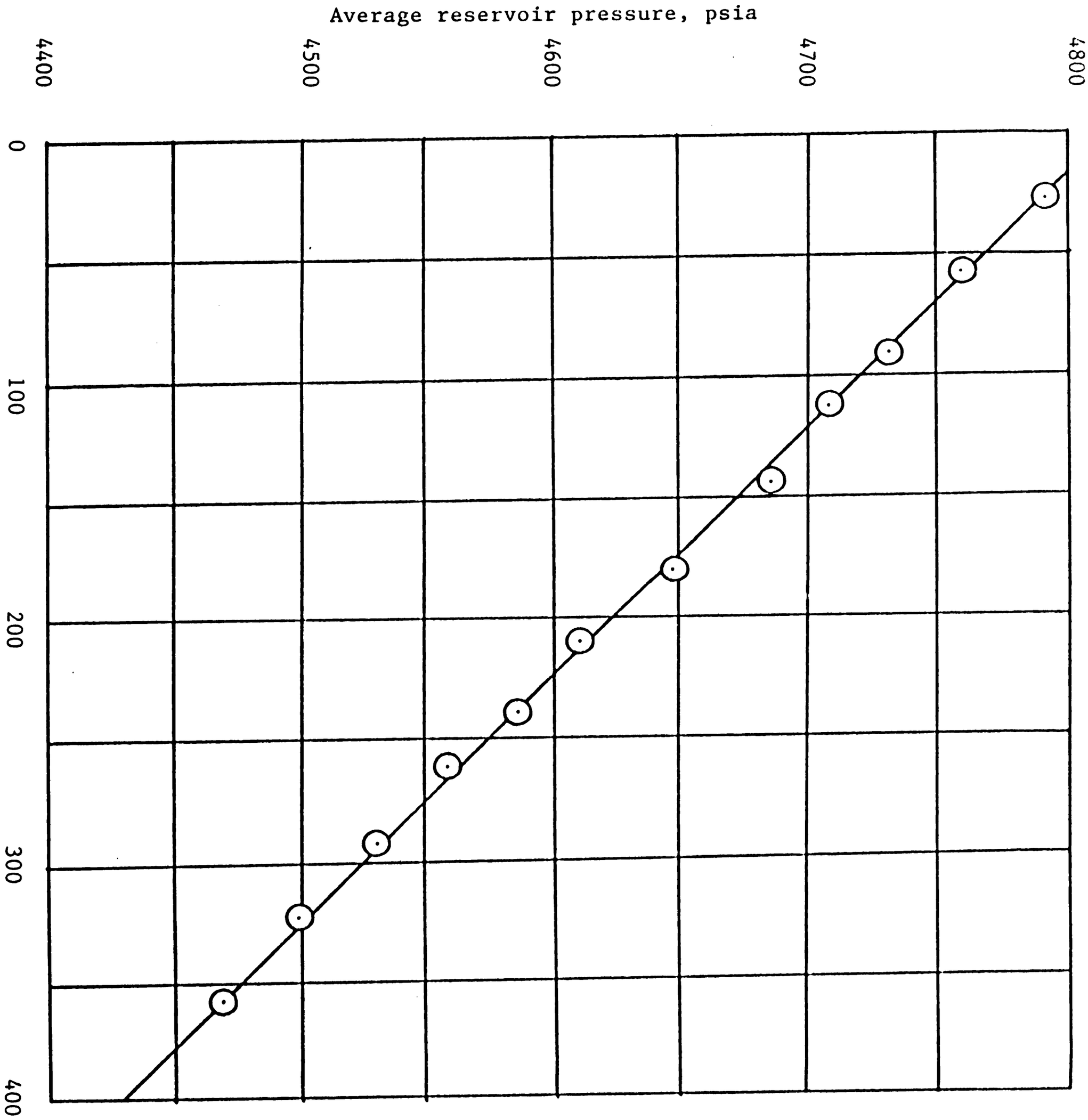

Average reservoir pressure, psia
4400
4500
4600
4700
4800
0
100
200
300
400
Time, days

FIGURE (6.14)

CONCLUSIONS

Based on the results of simulation of different kinds of models using GOWSIM, GWSIM and WSIM simulators we make the following conclusions:

1. GOWSIM is designed and tested primarily for three-dimensional, three-phase black oil reservoir. GWSIM is better to use for gas reservoir with active water drive. WSIM is used primarily for aquifer model.

2. All of the simulators can be redimensioned painlessly by changing the subsequent parameters in the common block file with no need to compile the source program again.

3. Input data without format can be read eliminating problem to use formatted data file.

4. Simulators contain the direct (BAND) and line successive overrelaxation with additive corrections (CLSOR) methods. We suggest to use BAND algorithm for the model with small number of grid blocks. It runs faster than CLSOR. However, CLSOR is preferred to be used for anisotropic reservoir model with large number of grid blocks.

5. As GOWSIM, GWSIM are primarily a pure IMPES simulator very small time-steps less than 1 day are required in some portions of the run to maintain a stable solution.

6. The GOWSIM, GWSIM programs are easy to use and understand, but still too large to run on a micro-computer system of which we are aware.

Appendices

A1: Computer Program GWSIM

A2: Computer Program GOWSIM

```
C
C
      CHARACTER IHEDIN
      CHARACTER*5 WELLID
      REAL KROT,KRWT,KRGT,MUWT,MUOT,MUGT,KX,KY,KZ,
     $  MUO,MUW,MUG,KRO,KRG,KRW,MBEWI,MBEGI,MCFGI,
     $  MBEO,MBEW,MBEG,MCFG,MBEOI,MIN,MCFG1,MCFOT
      INTEGER NX,NY,NZ,NXP,NYP,NZP,NTE,NU,NPMAX,NROW
C THIS VERSION ALLOWS ALL RE-DIMENSIONING TO BE DONE
C SIMPLY BY CHANGING THE SUBSEQUENT PARAMETERS AS DESIRED.
      PARAMETER (NX=10,NY=10,NZ=10,NXP=11,NYP=11,NZP=11,NTE=25,
     $NW=20,NPMAX=10,NROW=1000)
      COMMON /COEFF/AW(NX,NY,NZ),AE(NX,NY,NZ),AN(NX,NY,NZ),
     $AS(NX,NY,NZ),AT(NX,NY,NZ),AB(NX,NY,NZ),E(NX,NY,NZ),B(NX,NY,NZ)
      COMMON /SAT/SWN(NX,NY,NZ),SGN(NX,NY,NZ),SON(NX,NY,NZ),
     $SW1(NX,NY,NZ),SG1(NX,NY,NZ),SO1(NX,NY,NZ),A1(NX,NY,NZ),
     $A2(NX,NY,NZ),A3(NX,NY,NZ),SUM(NX,NY,NZ),
     $GAM(NX,NY,NZ),QS(NX,NY,NZ)
      COMMON /PARM/KX(NX,NY,NZ),KY(NX,NY,NZ),KZ(NX,NY,NZ),
     $TX(NXP,NY,NZ),TY(NX,NYP,NZ),TZ(NX,NY,NZP)
      COMMON /FACT/WW(NXP,NY,NZ),WE(NXP,NY,NZ),WS(NX,NYP,NZ),
     $WN(NX,NYP,NZ),WT(NX,NY,NZP),WB(NX,NY,NZP),OW(NXP,NY,NZ),
     $OE(NXP,NY,NZ),ON(NX,NYP,NZ),OS(NX,NYP,NZ),OT(NX,NY,NZP),
     $OB(NX,NY,NZP)
      COMMON /RPRT/PC(NX,NY,NZ),PN(NX,NY,NZ),SU(NX,NY,NZ),
     $SG(NX,NY,NZ),SO(NX,NY,NZ),PCGOT(NTE),PCOUT(NTE),PWT(NTE),
     $MUWT(NTE),BWT(NTE),BWPT(NTE),RSWT(NTE),RSWPT(NTE),PGT(NTE),
     $MUGT(NTE),BOT(NTE),BOPT(NTE),POT(NTE),MUOT(NTE),BOT(NTE),
     $BOPT(NTE),RSOT(NTE),RSOPT(NTE),CRT(NTE),SAT(NTE),KRWT(NTE),
     $KRGT(NTE),KROT(NTE),PCGWT(NTE)
      COMMON /RATES/PID(NW,NZ),PWF(NW,NZ),PWFC(NW,NZ),KIP(NW),
     $GMW(NW,NZ),GMG(NW,NZ),GMO(NW,NZ),WELLID(NW),LAYER(NW),
     $QVW(NW),QVG(NW),QVO(NW),QVT(NW),CUMW(NW,NZ),CUMG(NW,NZ),
     $CUMO(NW,NZ)
      COMMON /SOLN/GWWT(NX,NY,NZ),GGUT(NX,NY,NZ),GOUT(NX,NY,NZ),
     $QOWG(NX,NY,NZ),QO(NX,NY,NZ),QW(NX,NY,NZ),QG(NX,NY,NZ)
      COMMON /POSD/DX(NX,NY,NZ),DY(NX,NY,NZ),DZ(NX,NY,NZ),
     $IQN1(NW),IQN2(NW),IQN3(NW),IHEDIN(80),EL(NX,NY,NZ)
      COMMON /PHASE/PBOT(NX,NY,NZ),BU(NX,NY,NZ),BG(NX,NY,NZ),
     $BO(NX,NY,NZ),VP(NX,NY,NZ),CT(NX,NY,NZ),PBO,VSLOPE,
     $BSLOPE,RSLOPE,PMAXT,IREPRS,RHOSCO,RHOSCW,RHOSCG,MSAT,MPOT,
     $MPWT,MPGT,IOCODE
      COMMON /BAND1/BMAT(NROW,NROW),QVEC(NROW),PVEC(NROW),
     $GVEC(NROW)

      END
```

Program GWSIM

```
C
CC    ****************************************************************
CC    *                                                              *
CC    *                          GWSIM:                              *
CC    *        3D  GAS - WATER  RESERVOIR  SIMULATOR                 *
CC    *        U.N.S.W.  -  VERSION  2.0  (25-10-88)                 *
CC    *                                                              *
CC    *                                                              *
CC    *    GWSIM  IS A THREE-DIMENSIONAL  GAS - WATER  RESERVOIR     *
CC    *    SIMULATOR  USING A FINITE-DIFFERENCE, IMPLICIT PRESS-     *
CC    *    URE - EXPLICIT SATURATION  (IMPES) METHOD.  IT CONTA-     *
CC    *    INS  DIRECT (BAND)  AND  ITERATIVE  LINE  SUCCESSIVE     *
CC    *    OVERRELAXATION  WITH  ADDITIVE  CORRECTIONS  (CLSOR)     *
CC    *    SOLUTION METHODS  FOR SOLVING  A SYSTEM OF  ALGEBRAIC    *
CC    *    EQUATIONS.   THE WELL MODEL IN  GWSIM ALLOWS  SPECIF-    *
CC    *    ICATION  OF  RATE OR  PRESSURE  CONSTRAINTS  ON  WELL    *
CC    *    PERFORMANCE, AND  THE USER  IS  FREE TO ADD OR RECOM-    *
CC    *    PLETE  WELLS DURING  SIMULATION.  ALL  INPUT DATA        *
CC    *    VALUES ARE  READ UNDER  FREE FORMAT.  REDIMENSIONING     *
CC          CAN  BE DONE  EASILY BY  CHANGING  SUBSEQUENT  PARA-     *
CC    *    METERS  IN  THE COMMON BLOCK FILE  WITH  NO  NEED  TO    *
CC    *    RECOMPILE  THE  MAIN  PROGRAM.                           *
C     *                                                              *
CC    *                                                              *
CC    ****************************************************************
CC
CC
CC    ================================================================
C
CC        PROGRAM GWSIM
C
CC    ================================================================
CC
CC        NOMENCLATURE:
CC
CC            AB(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW6-RSW))*AWB+
CC                          BG(I,J,K)*AGB
CC            AE(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW2-RSW))*AWE+
CC                          BG(I,J,K)*AGE
CC            AN(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW4-RSW))*AWN+
CC                          BG(I,J,K)*AGN
CC            AS(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW3-RSW))*AWS+
CC                          BG(I,J,K)*AGS
CC            AW(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW1-RSW))*AWW+
CC                          BG(I,J,K)*AGW
CC            AT(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW5-RSW))*AWT+
CC                          BG(I,J,K)*AGT
CC            A1(I,J,K)  :  KX(I,J,K)*DY(I,J,K)*DZ(I,J,K)
CC            A2(I,J,K)  :  KY(I,J,K)*DX(I,J,K)*DZ(I,J,K)
C             A3(I,J,K)  :  KZ(I,J,K)*DX(I,J,K)*DY(I,J,K)
CC            AWW        :  TX(I,J,K)*MW1
CC            AGW        :  TX(I,J,K)*MG1
CC            AWE        :  TX(I+1,J,K)*MW2
CC            AGE        :  TX(I+1,J,K)*MG2
CC            AWS        :  TY(I,J,K)*MW3
CC            AGS        :  TY(I,J,K)*MG3
CC            AWN        :  TY(I,J+1,K)*MW4
CC            AGN        :  TY(I,J+1,K)*MG4
CC            AWT           TZ(I,J,K)*MW5
CC            AGT        :  TZ(I,J,K)*MG5
CC            AWB        :  TZ(I,J,K+1)*MW6
CC            AGB        :  TZ(I,J,K+1)*MG6
CC            B(I,J,K)   :  QOWG(I,J,K)-VP(I,J,K)*P(I,J,K)*CT(I,J,K)
CC                          /DELT
CC            BGT        :  VARIABLE GAS FORMATION VOLUME FACTOR AT
CC                          PRESSURE (PGT) IN PVT DATA TABLE
CC            BWT        :  VARIABLE WATER FORMATION VOLUME FACTOR AT
CC                          PRESSURE (PWT) IN PVT DATA TABLE
CC            BGPT       :  SLOPE dBG/dP
CC            BWPT       :  SLOPE dBW/dP
CC            CGI        :  CUMMULATIVE GAS INJECTION (MSCF)
CC            CGP        :  CUMMULATIVE GAS PRODUCTION (MSCF)
CC            CRT        :  PRESSURE DEPENDENT ROCK COMPRESSIBILITY,
CC                          /PSIA
CC            CT         :  TOTAL COMPRESSIBILITY
CC            CWP        :  CUMMULATIVE WATER PRODUCTION, STB
C             CWI        :  CUMMULATIVE WATER INJECTION, STB
C             DELTO      :  TIME STEP, DAYS
CC            DPMAX      :  MAXIMUM PRESSURE CHANGE PERMITTED OVER A
CC                          TIME STEP, PSIA
CC            DSMAX      :  MAXIMUM SATURATION CHANGE PERMITTED OVER A
CC                          TIME STEP, FRACTION
CC            DX(I,J,K)  :  VARIABLE X-DIRECTION GRID DIMENSION, FT.
CC            DY(I,J,K)  :  VARIABLE Y-DIRECTION GRID DIMENSION, FT.
CC            DZ(I,J,K)  :  VARIABLE Z-DIRECTION GRID DIMENSION, FT.
CC            E(I,J,K)   :  -SUM(I,J,K)-GAM(I,J,K)
CC            EL(I,J,K)  :  VARIABLE NODE MIDPOINT ELEVATIONS, FT.
CC            ETI        :  ELAPSED TIME, DAYS
CC            FACT1      :  FACTOR FOR INCREASING TIME STEP SIZE UNDER
CC                          AUTOMATIC TIME STEP CONTROL. SET FACT1=1
CC                          FOR FIXED TIME STEP SIZE.  A COMMON VALUE
CC                          FOR FACT1 IS 1.25
```

```
C
C       FACT2       : FACTOR FOR DECREASING TIME STEP SIZE UNDER
C                     AUTOMATIC TIME STEP CONTROL.  SET FACT2=1
C                     FOR FIXED TIME STEP SIZE.  A COMMON VALUE
C                     FOR FACT2 IS 0.5
C       GAM(I,J,K): VP(I,J,K)*CT(I,J,K)/DELT
C       GGWT        : AGW*GGW1+AGE*GGW2+AGS*GGW3+AGN*GGW4+AGT*
C                     GGW5+AGB*GGW6+RSW1A*AW1+RSW2A*AW2+RSW3A*
C                     AW3+RSW4A*AW4+RSW5A*AW5+RSW6A*AW6
C       GMG         : KRG/MUG
C       GMW         : KRW/MUW
C       GWC         : GAS - WATER CONTACT, FT.
C       GWWT        : AW1+AW2+AW3+AW4+AW5+AW5
C       ICHANG      : NUMBER OF TIME STEPS FOR WHICH THE OUTPUT
C                     CONTROL AND TIME STEP CONTROL INFORMATION
C                     WILL APPLY
C       II          : NUMBER OF GRID BLOCKS IN THE X-DIRECTION
C       IOCODE      : OUTPUT DATA FILE CODE
C       IPMAP       : OUTPUT CODE TO CONTROL PRINTING OF THE MAP
C                     OF GRID BLOCK PRESSURES
C       IQN1        : X-COORDINATE OF GRID BLOCK CONTAINING THE
C                     WELL UNDER CONSIDERATION
C       IQN2        : Y-COORDINATE OF GRID BLOCK CONTAINING THE
C                     WELL UNDER CONSIDERATION
C       IQN3        : LAYER NUMBER OF THE UPPERMOST COMPLETION
C                     LAYER FOR THE WELL UNDER CONSIDERATION
C       ISGMAP      : OUTPUT CODE TO CONTROL PRINTING OF THE
C                     GRID BLOCK GAS SATURATIONS
C       ISUMRY      : OUTPUT CODE TO CONTROL PRINTING OF THE
C                     TIME STEP SUMMARY REPORT
C       ISWMAP      : OUTPUT CODE TO CONTROL PRINTING OF THE
C                     GRID BLOCK WATER SATURATIONS
C       IWLCNG      : CODE TO TELL PROGRAM WHETHER OR NOT THE WELL
C                     INFORMATION CARDS SHOULD BE READ THIS TIME
C                     STEP
C       IWLREP      : OUTPUT CODE TO CONTROL PRINTING OF THE WELL
C                     REPORT
C       KCO1        : COMPRESSIBILITY AND FORMATION VOLUME FACTOR
C                     DEBUG OUTPUT CONTROL
C       KCOFF       : DENSITY AND SATURATION DEBUG OUTPUT CONTROL
C       KIP         : CODE FOR SPECIFYING BOTH WELL TYPE AND
C                     WHETHER THE WELL'S PRODUCTION (INJECTION)
C                     PERFORMANCE IS DETERMINED BY SPECIFYING RATES
C                     OR BY SPECIFYING FLOWING BOTTOM-HOLE PRESSURE
C                     AND ALSO WHETHER AN EXPLICIT OR IMPLICIT
C                     PRESSURE CALCULATION IS TO BE MADE.
C       KK          : NUMBER OF GRID BLOCKS IN THE Z-DIRECTION
C       KPI         : PRESSURE INITIALIZATION CODE
C       KRG         : GAS PHASE RELATIVE PERMEABILITY, FRACTION
C       KRW         : WATER PHASE RELATIVE PERMEABILITY, FRACTION
C       KSM1        : SOLUTION MATRIX DEBUG OUTPUT CONTROL CODE
C       KSN1        : CLSOR PARAMETER DEBUG OUTPUT CONTROL CODE
C       KTR         : TRANSMISSIBILITY DEBUG OUTPUT CONTROL CODE
C       KX(I,J,K)   : VARIABLE VALUE OF X-DIRECTION PERMEABILITY
C       KY(I,J,K)   : VARIABLE VALUE OF Y-DIRECTION PERMEABILITY
C       KZ(I,J,K)   : VARIABLE VALUE OF Z-DIRECTION PERMEABILITY
C       LAYER       : TOTAL NUMBER OF CONSECUTIVE COMPLETION LAYERS
C                     STARTING WITH AND INCLUDING IQN3
C       MBEG        : GAS MATERIAL BALANCE, PER CENT
C       MBEW        : WATER MATERIAL BALANCE, PER CENT
C       MITER       : MAXIMUM NUMBER OF CLSOR ITERATIONS PER TIME
C                     STEP.  A TYPICAL VALUE FOR MITER IS 250
C       MPGT        : MAXIMUM NUMBER OF ENTRY VALUE FOR GAS PHASE
C                     PRESSURE IN PVT DATA TABLE, PSIA
C       MPWT        : MAXIMUM NUMBER OF ENTRY VALUE FOR WATER
C                     PHASE PRESSURE IN PVT DATA TABLE, PSIA
C       MUG         : GAS VISCOSITY, CP
C       MUW         : WATER VISCOSITY, CP
C       MW1         : 4.0*KRW1/((BW(I-1,J,K)+BW(I,J,K))*(MUW1+
C                     MUW))
C       MG1         : 4.0*KRG1/((BG(I-1,J,K)+BG(I,J,K))*(MUG1+
C                     MUG))
C       MW2         : 4.0*KRW2/((BW(I+1,J,K)+BW(I,J,K))*(MUW2+
C                     MUW))
C       MG2         : 4.0*KRG2/((BG(I+1,J,K)+BG(I,J,K))*(MUG2+
C                     MUG))
C       MW3         : 4.0*KRW3/((BW(I,J-1,K)+BW(I,J,K))*(MUW3+
C                     MUW))
C       MG3         : 4.0*KRG3/((BG(I,J-1,K)+BG(I,J,K))*(MUG3+
C                     MUG))
C       MW4         : 4.0*KRW4/((BW(I,J+1,K)+BW(I,J,K))*(MUW4+
C                     MUW))
C       MG4         : 4.0*KRG4/((BG(I,J+1,K)+BG(I,J,K))*(MUG4+
C                     MUG))
C       MW5         : 4.0*KRW5/((BW(I,J,K-1)+BW(I,J,K))*(MUW5+
C                     MUW))
C       MG5         : 4.0*KRG5/((BG(I,J,K-1)+BG(I,J,K))*(MUG5+
C                     MUG))
C       MW6         : 4.0*KRW6/((BW(I,J,K+1)+BW(I,J,K))*(MUW6+
C                     MUW))
C       MG6         : 4.0*KRG6/((BG(I,J,K+1)+BG(I,J,K))*(MUG6+
C                     MUG))
C       NN          : MAXIMUM NUMBER OF TIME STEPS ALLOWED BEFORE
C                     RUN IS TERMINATED
C       NTE         : MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN
C                     ALL PVT TABLES
C       NVQN        : NUMBER OF WELLS FOR WHICH WELL INFORMATION
C                     IS TO BE READ
```

```
C
C          NW          : MAXIMUM NUMBER OF WELLS ALLOWED TO BE READ
C          NX          : MAXIMUM NUMBER OF X-DIRECTION BLOCKS ALLOWED
C                        TO BE READ
C          NY          : MAXIMUM NUMBER OF Y-DIRECTION BLOCKS ALLOWED
C                        TO BE READ
C          NZ          : MAXIMUM NUMBER OF Z-DIRECTION BLOCKS ALLOWED
C                        TO BE READ
C          ODGIP       : ORIGINAL DISSOLVED GAS IN PLACE
C          OFGIP       : ORIGINAL FREE GAS IN PLACE
C          OMEGA       : INITIAL CLSOR ACCLERATION PARAMETER.  INITIAL
C                        VALUE FOR OMEGA SHOULD BE IN THE RANGE 1 AND
C                        2.  A TYPICAL INITIAL VALUE FOR OMEGA WOULD
C                        BE 1.7.  THE PROGRAM WILL OPTIMIZE AS THE
C                        SOLUTION PROCEEDS.
C          OWIP        : ORIGINAL WATER IN PLACE
C          PAMIN       : LIMITING MINIMUM FIELD AVERAGE PRESSURE, PSIA
C          PAMAX       : LIMITING MAXIMUM FIELD AVERAGE PRESSURE,
C                        PSIA
C          PAVG        : CURRENT AVERAGE RESERVOIR PRESSURE, PSIA
C          PAVGO       : PREVIOUS AVERAGE RESERVOIR PRESSURE, PSIA
C          PCGWT       : GAS-WATER CAPILLARY PRESSURE DATA
C          PGT         : GAS PHASE PRESSURE, PSIA
C                        THE LAST ENTRY OF PGT IN THE TABLE MUST
C                        BE PMAXT
C          PGWC        : PRESSURE AT GAS-WATER CONTACT, PSIA
C          PN          : PRESSURE AT TIME LEVEL N
C          PWF         : FLOWING BOTTOM-HOLE PRESSURE, PSIA
C          PWT         : WATER PHASE PRESSURE, PSIA
C                        THE LAST ENTRY OF PWT IN THE TABLE MUST
C                        BE PMAXT
C          QVG         : GAS RATE, MCF/D
C          QOWG        : (BW(I,J,K)-BG(I,J,K)*RSW)*(-GWWT(I,J,K)+
C                        QW(I,J,K))+BG(I,J,K)*(-GGWT(I,J,K)+
C                        QG(I,J,K))
C          QVW         : WATER RATE, STB/D
C          QVT         : TOTAL FLUID RATE, STB/D
C          RESVOL      : RESERVOIR VOLUME
C          RSW         : GAS IN WATER SOLUTION RATIO, SCF/STB
C          RSW1        : GAS IN WATER SOLUTION RATIO AT PRESSURE
C                        P(I-1,J,K)
C          RSW1A       : 0.5*(RSW1+RSW)
C          SAT         : VALUE OF PHASE SATURATION.  THE FIRST ENTRY
C                        OF SAT IN THE TABLE MUST BE -0.1 AND THE
C                        LAST ENTRY BE 1.1.   READ EACH SATURATION
C                        AS A FRACTION
C          SKIN        : LAYER SKIN FACTOR, DIMENSIONLESS
C          SG          : GAS SATURATION
C          SUM         : AW(I,J,K)+AE(I,J,K)+AS(I,J,K)+AN(I,J,K)+
C                        AT(I,J,K)+AB(I,J,K)
C          SW          : WATER SATURATION
C          SWC         : CONNATE WATER SATURATION
C          TMAX        : MAXIMUM REAL TIME TO BE SIMULATED DURING
C                        THE RUN, DAYS
C          TODGIP      : TOTAL SOLUTION GAS IN PLACE, BILLION SCF
C          TOFGIP      : TOTAL FREE GAS IN PLACE, BILLION SCF
C          TOL         : MAXIMUM ACCEPTABLE PRESSURE CHANGE FOR
C                        CONVERGENCE OF CLSOR ITERATIONS, PSI.
C                        A TYPICAL VALUE FOR TOL WOULD BE 0.1
C          TOL1        : PARAMETER FOR DETERMINING WHEN TO CHANGE
C                        OMEGA.  A TYPICAL VALUE FOR TOL1 WOULD BE
C                        0.001.  IF TOL1 =0, THE INITIAL VALUE
C                        ENTERED AS OMEGA ABOVE WILL BE USED FOR
C                        THE ENTIRE RUN
C          TOWIP       : TOTAL WATER IN PLACE, MILLION STB
C          TX(I,J,K)   : TRANSMISSIBILTY  REFERRING TO FLOW ACROSS
C                        THE BOUNDARY BETWEEN BLOCKS (I-1) AND (I)
C                        AND IS CALCULATED BY
C                        0.012656*A1(I-1,J,K)*A1(I,J,K)/(DX(I-1,J,K)
C                        *A1(I,J,K)+DX(I,J,K)*A1(I-1,J,K))
C          TY(I,J,K)   : TRANSMISSIBILITY REFERRING TO FLOW ACROSS
C                        THE BOUNDARY BETWEEN BLOCKS (J-1) AND (J)
C          TZ(I,J,K)   : TRANSMISSIBILITY REFERRING TO FLOW ACROSS
C                        THE BOUNDARY BETWEEN BLOCKS (K-1) AND (K)
C          VPP         : RESERVOIR PORE VOLUME
C          WB(I,J,K)   : AWB
C          WE(I,J,K)   : AWE
C          WN(I,J,K)   : AWN
C          WRAD        : WELLBORE RADIUS, FT
C          WS(I,J,K)   : AWS
C          WT(I,J,K)   : AWT
C          WW(I,J,K)   : AWW
C
C-----------------------------------------------------------------------
C
      INCLUDE 'COMGWSIM.FOR'
      DIMENSION  OWIP(NZ),ODGIP(NZ),OFGIP(NZ)

      DATA  CWP,CGP,CWI,CGI/0.0,0.0,0.0,0.0/
      DATA  ETI,FT,FTMAX,KTR/0.0,0.0,0.0,0/
      DATA  SCFW,SCFG,SCFG1,KRG/0.0,0.0,0.0,0.0/
      DATA  MCFG1,MCFGT,PAVG,PAVGO/0.0,0.0,0.0,0.0/
      DATA  MCFG,MCFGI,MUW,MUG,KRW/0.0,0.0,0.0,0.0,0.0/
      DATA  MBEWI,MBEGI,MBEW,MBEG/0.0,0.0,0.0,0.0/
```

```
      DO  I=1,NW
          IQN1(I)   = 0
          IQN2(I)   = 0
          IQN3(I)   = 0
          KIP(I)    = 0
          LAYER(I)  = 0
          QVG(I)    = 0.0
          QVW(I)    = 0.0
          QVT(I)    = 0.0

          DO  J=1,NZ
              CUMG(I,J)  = 0.0
              CUMW(I,J)  = 0.0
              GMG(I,J)   = 0.0
              GMW(I,J)   = 0.0
              PWF(I,J)   = 0.0
              PWFC(I,J)  = 0.0
          END DO
      END DO

      DO  I=1,NTE
          BGT(I)    = 0.0
          BGPT(I)   = 0.0
          BWT(I)    = 0.0
          BWPT(I)   = 0.0
          CRT(I)    = 0.0
          KRGT(I)   = 0.0
          KRWT(I)   = 0.0
          MUGT(I)   = 0.0
          MUWT(I)   = 0.0
          PCGWT(I)  = 0.0
          PGT(I)    = 0.0
          PWT(I)    = 0.0
          RSWT(I)   = 0.0
          RSWPT(I)  = 0.0
          SAT(I)    = 0.0
      END DO

      DO  700  I=1,IM
      DO  700  J=1,JM
      DO  700  K=1,KM
          AE(I,J,K)     = 0.0
          AW(I,J,K)     = 0.0
          AN(I,J,K)     = 0.0
          AS(I,J,K)     = 0.0
          AT(I,J,K)     = 0.0
          AB(I,J,K)     = 0.0
          A1(I,J,K)     = 0.0
          A2(I,J,K)     = 0.0
          A3(I,J,K)     = 0.0
          B(I,J,K)      = 0.0
          BG(I,J,K)     = 0.0
          BW(I,J,K)     = 0.0
          CT(I,J,K)     = 0.0
          DX(I,J,K)     = 0.0
          DY(I,J,K)     = 0.0
          DZ(I,J,K)     = 0.0
          E(I,J,K)      = 0.0
          EL(I,J,K)     = 0.0
          GAM(I,J,K)    = 0.0
          GGWT(I,J,K)   = 0.0
          GWWT(I,J,K)   = 0.0
          KX(I,J,K)     = 0.0
          KY(I,J,K)     = 0.0
          KZ(I,J,K)     = 0.0
          P(I,J,K)      = 0.0
          PN(I,J,K)     = 0.0
          QG(I,J,K)     = 0.0
          QOWG(I,J,K)   = 0.0
          QS(I,J,K)     = 0.0
          QW(I,J,K)     = 0.0
          SG(I,J,K)     = 0.0
          SG1(I,J,K)    = 0.0
          SGN(I,J,K)    = 0.0
          SW(I,J,K)     = 0.0
          SW1(I,J,K)    = 0.0
          SWN(I,J,K)    = 0.0
          SUM(I,J,K)    = 0.0
          TX(I,J,K)     = 0.0
          TY(I,J,K)     = 0.0
          TZ(I,J,K)     = 0.0
          VP(I,J,K)     = 0.0
          WE(I,J,K)     = 0.0
          WW(I,J,K)     = 0.0
          WN(I,J,K)     = 0.0
          WS(I,J,K)     = 0.0
          WT(I,J,K)     = 0.0
          WB(I,J,K)     = 0.0
```

```
            IF (I.EQ.NX) THEN
               TX(I+1,J,K) = 0.0
               WE(I+1,J,K) = 0.0
               WW(I+1,J,K) = 0.0
            END IF

            IF (J.EQ.NY) THEN
               TY(I,J+1,K) = 0.0
               WN(I,J+1,K) = 0.0
               WS(I,J+1,K) = 0.0
            END IF

            IF (K.EQ.NZ) THEN
               TZ(I,J,K+1) = 0.0
               WT(I,J,K+1) = 0.0
               WB(I,J,K+1) = 0.0
            END IF
  700   CONTINUE
C -----------------------------------------------------------------
C          OPEN INPUT AND OUTPUT FILES.
C -----------------------------------------------------------------
      OPEN(20,FILE='GWSIM.DAT',ACCESS='SEQUENTIAL',STATUS='OLD')
      WRITE(*,31)
      READ(*,32) IOCODE

      OPEN(IOCODE,FILE='GWSIM.OUT',ACCESS='SEQUENTIAL',
     $     STATUS='NEW')
C -----------------------------------------------------------------
C          OUTPUT HEADER.
C -----------------------------------------------------------------
      WRITE(IOCODE,36)
C -----------------------------------------------------------------
C          ESTABLISH RESERVOIR BLOCK DIMENSIONS AND GEOMETRY.
C -----------------------------------------------------------------
      CALL  GRID1 (II,JJ,KK)
C -----------------------------------------------------------------
C          ESTABLISH POROSITY AND PERMEABILITY DISTRIBUTION.
C -----------------------------------------------------------------
      CALL  PARM1 (II,JJ,KK)
C -----------------------------------------------------------------
C          CALCULATE INTERBLOCK TRANSMISSIBILITIES.
C -----------------------------------------------------------------
      CALL  TRAN1 (II,JJ,KK,KTR)
C -----------------------------------------------------------------
C          ESTABLISH EMPIRICAL DATA TABLES.
C -----------------------------------------------------------------
      CALL  TABLE
C -----------------------------------------------------------------
C          ESTABLISH INITIAL CONDITIONS.
C -----------------------------------------------------------------
      CALL  UINIT1 (KPI,II,JJ,KK,CUMPW,MBEW,CUMPG,MBEG,
     $              SWC,GWC,PGWC,CUMIW,CUMIG)
C -----------------------------------------------------------------
C          SOLUTION METHOD, DEBUG PRINT, AND TIME STEP CONTROL.
C -----------------------------------------------------------------
      CALL  CODES (IOCODE,KSM1,KSN1,KCO1,NN,FACT1,FACT2,
     $             TMAX,KSOL,MITER,OMEGA,TOL,TOL1,KSN,
     $             KSM,KCO,KTR,KCOFF,DSMAX,DPMAX,WGRMAX,
     $             PAMIN,PAMAX)
C -----------------------------------------------------------------

      D5615 = 1.0/5.615
      D288  = 1.0/288.0
      D144  = 1.0/144.0

      READ(20,69) (IHEDIN(L), L=1,80)
```

```
      NMAX  = NN+1
      NITER = 0

      DO N=1,NMAX
         NLOOP = N
         IF (FT.GE.FTMAX) THEN
            READ(20,*,END=1001) IWLCNG,ICHANG,IWLREP,ISUMRY,
     $                          IPMAP,ISWMAP,ISGMAP
            READ(20,*) DAY,DTMIN,DTMAX
            IF (IWLCNG.NE.0) THEN
               CALL  NODES (NVQN,WRAD,SKIN)
            END IF

            DELT  = DAY
            FTMAX = ETI+ICHANG*DELT
         END IF

         IF (N.NE.1) THEN
            IF (DSMC.LT.DSMAX .AND. DPMC.LT.DPMAX .AND.
     $          NITER.LT.MITER)     DELT = DELT*FACT1
            IF (DSMC.GT.DSMAX .OR. DPMC.GT.DPMAX .OR.
     $          NITER.GE.MITER)     DELT = DELT*FACT2
            IF (DELT.LT.DTMIN)      DELT = DTMIN
            IF (DELT.GT.DTMAX)      DELT = DTMAX
            IF (ETI+DELT.GT.FTMAX) DELT = FTMAX-ETI
         ELSE
            DELTO = DELT
         END IF

         FT = ETI+DELT
         IF (ETI+DELT*0.5.GE.TMAX) GO TO 1001
         ITFLAG = 0

060      DIV1 = 1.0/DELT
         IF (N.GT.1 .OR. ITFLAG.GT.0) GO TO 105
            RESVOL = 0.0
            SCFG   = 0.0
            SCFG1  = 0.0
            DO K=1,KK
               OWIP(K)  = 0.0
               ODGIP(K) = 0.0
               OFGIP(K) = 0.0
               DO 100 J=1,JJ
               DO 100 I=1,II
                  PP = P(I,J,K)
                  VP(I,J,K) = VP(I,J,K)*DX(I,J,K)*DY(I,J,K)*
     $                        DZ(I,J,K)
                  RESVOL    = RESVOL+VP(I,J,K)
                  CALL  INTERP (NTE,PWT,RSWT,MPWT,PP,RSW)
                  CALL  INTERP (NTE,PWT,BWT,MPWT,PP,BW(I,J,K))
                  CALL  INTERP (NTE,PGT,BGT,MPGT,PP,BG(I,J,K))
                  SCFW  = SCFW+VP(I,J,K)*SW(I,J,K)/BW(I,J,K)
                  SCFG  = SCFG+VP(I,J,K)*SG(I,J,K)/BG(I,J,K)
                  SCFG1 = SCFG1+VP(I,J,K)*RSW*SW(I,J,K)/BW(I,J,K)
                  CALL  NTERP1 (NTE,PWT,BWPT,MPWT,PP,BWDER)
                  CALL  NTERP1 (NTE,PWT,RSWPT,MPWT,PP,RSWDER)
                  CALL  NTERP1 (NTE,PGT,BGPT,MPGT,PP,BGDER)
                  CALL  INTERP (NTE,PGT,CRT,MPGT,PP,CR)

                  CW = -(BWDER-BG(I,J,K)*RSWDER)/BW(I,J,K)
                  CG = -BGDER/BG(I,J,K)
                  CX = CT(I,J,K)
                  CT(I,J,K) = CR+CW*SW(I,J,K)+CG*SG(I,J,K)
                  IF (CT(I,J,K).LE.0.0) THEN
                     CT(I,J,K) = CX
                     WRITE(*,85) I,J,K
                  END IF

                  OWIP(K)  = OWIP(K)+D5615*0.000001*SWN(I,J,K)*
     $                       VP(I,J,K)/BW(I,J,K)
                  ODGIP(K) = ODGIP(K)+0.001*0.000001*RSW*
     $                       SWN(I,J,K)*VP(I,J,K)/BW(I,J,K)
                  OFGIP(K) = OFGIP(K)+0.001*0.000001*SGN(I,J,K)*
     $                       VP(I,J,K)/BG(I,J,K)

                  IF (KCO1.NE.0) THEN
                     WRITE(IOCODE,87) I,J,K,VP(I,J,K),CT(I,J,K),
     $                                BW(I,J,K),SW(I,J,K),
     $                                BG(I,J,K),SG(I,J,K)
                  END IF
100               CONTINUE
                  WRITE(IOCODE,110) K,OWIP(K),ODGIP(K),OFGIP(K)
               END DO


               TOWIP  = 0.0
               TODGIP = 0.0
               TOFGIP = 0.0

               DO K=1,KK
                  TOWIP  = TOWIP+OWIP(K)
                  TODGIP = TODGIP+ODGIP(K)
                  TOFGIP = TOFGIP+OFGIP(K)
               END DO
               WRITE(IOCODE,115) TOWIP,TODGIP,TOFGIP

               STBW  = SCFW*D5615
               MCFG  = SCFG*0.001
               MCFG1 = SCFG1*0.001
               STBWI = STBW
               MCFGI = MCFG+MCFG1
               IF (MCFGI.LE.1.E-7 .AND. MCFGT.LE.1.E-7)
     $                MBEG = 0.0
105      CONTINUE
```

```
      IF (N.EQ.1 .AND. ITFLAG.LE.0)
     $   CALL  PRTPS (NLOOP,KPI,II,JJ,KK,PAVGO,PAVG,CWP,
     $                CWI,CGP,CGI,MBEW,MBEG,DELTO,WPR,
     $                GPR,WIR,GIR,ETI,CWGR,WGR,IPMAP,
     $                ISWMAP,ISGMAP)
      IF (N.EQ.NMAX) GO TO 1001
C ---------------------------------------------------------------------
C       ESTABLISH RATES AND CALCULATE BHFP.
C ---------------------------------------------------------------------
      IF (NVQN.NE.0) THEN
         CALL  QRATE (NVQN)
         WRITE(*,125)
      END IF
C ---------------------------------------------------------------------
C       CALCULATE SEVEN DIAGONAL MATRIX FOR PRESSURE SOLUTION.
C ---------------------------------------------------------------------
      CALL  SOLMAT (II,JJ,KK,DIV1,D288,D144,KSM,
     $              KSM1,NLOOP,NN,KCOFF)
      WRITE(*,130)
C ---------------------------------------------------------------------
C       MODIFY MATRIX ELEMENTS FOR WELLS UNDER
C       IMLICIT CONTROL.
C ---------------------------------------------------------------------
      IF (NVQN.NE.0) THEN
         CALL  PRATEI (NVQN)
         WRITE(*,140)
      END IF
C ---------------------------------------------------------------------
C       CALCULATE NEW PRESSURE DISTRIBUTION.
C ---------------------------------------------------------------------
      WRITE(*,675) FT
      WRITE(*,676) DELT
      IF (KSOL.EQ.1) THEN
         CALL  BANDIN (II,JJ,KK)
      ELSE IF (KSOL.EQ.2) THEN
         CALL  CLSOR (II,JJ,KK,OMEGA,TOL,TOL1,MITER,
     $                DELT,DELTO,KSN,NLOOP,NITER)
      ELSE
         CONTINUE
      END IF
C ---------------------------------------------------------------------
C       CALCULATE IMPLICIT RATES.
C ---------------------------------------------------------------------
      IF (NVQN.NE.0) THEN
         CALL  PRATEO (NVQN)
      END IF
C ---------------------------------------------------------------------
C       CALCULATE NEW FLUID SATURATIONS.
C ---------------------------------------------------------------------
      SCFW   = 0.0
      SCFG   = 0.0
      SCFG1  = 0.0
      RESVOL = 0.0

      DO 400 K=1,KK
      DO 400 J=1,JJ
      DO 400 I=1,II
         PPN =  PN(I,J,K)
         PP  =  P(I,J,K)
         CALL   INTERP (NTE,PWT,RSWT,MPWT,PP,RSW)
         CALL   INTERP (NTE,PGT,CRT,MPGT,PPN,CR)
         CALL   INTERP (NTE,PWT,BWT,MPWT,PP,BBW)
         CALL   INTERP (NTE,PGT,BGT,MPGT,PP,BBG)
```

```
                VPP      = VP(I,J,K)*(1.0+CR*(P(I,J,K)-PPN))
                RESVOL   = RESVOL+VPP
                DP1      = 0.0
                DP2      = 0.0
                DP3      = 0.0
                DP4      = 0.0
                DP5      = 0.0
                DP6      = 0.0
                IF ((I-1).GT.0)   DP1 = P(I-1,J,K)-PP
                IF ((I+1).LE.II)  DP2 = P(I+1,J,K)-PP
                IF ((J-1).GT.0)   DP3 = P(I,J-1,K)-PP
                IF ((J+1).LE.JJ)  DP4 = P(I,J+1,K)-PP
                IF ((K-1).GT.0)   DP5 = P(I,J,K-1)-PP
                IF ((K+1).LE.KK)  DP6 = P(I,J,K+1)-PP

                DAWDP       = WW(I,J,K)*DP1+WE(I,J,K)*DP2+WS(I,J,K)*
     $                        DP3+WN(I,J,K)*DP4+WT(I,J,K)*DP5+
     $                        WB(I,J,K)*DP6
                SW(I,J,K)   = ((DAWDP+GWWT(I,J,K)-QW(I,J,K))*DELT+
     $                        VP(I,J,K)*SWN(I,J,K)/BW(I,J,K))*BBW/VPP
                SG(I,J,K)   = 1.0-SW(I,J,K)

                IF (SG(I,J,K).LE.0.0)   SG(I,J,K)=0.0
                IF (KCOFF.NE.0) THEN
                   RHW1   = VPP*SW(I,J,K)/BBW
                   RHW2   = VP(I,J,K)*SWN(I,J,K)/BW(I,J,K)
                   DIFFW  = RHW1-RHW2
                   RHG1   = VPP*SG(I,J,K)/BBG
                   RHG2   = VP(I,J,K)*SGN(I,J,K)/BG(I,J,K)
                   DIFFG  = RHG1-RHG2

                   WRITE(IOCODE,733)
                   WRITE(IOCODE,87)  I,J,K,P(I,J,K),SW(I,J,K),
     $                                SWN(I,J,K),SG(I,J,K),SGN(I,J,K),VPP
                   WRITE(IOCODE,87)  I,J,K,GWWT(I,J,K),QW(I,J,K)
                   WRITE(IOCODE,87)  I,J,K,DAWDP,DELT
                   WRITE(IOCODE,87)  I,J,K,RHW1,RHW2,DIFFW
                   WRITE(IOCODE,87)  I,J,K,RHG1,RHG2,DIFFG,GGWT(I,J,K),
     $                                QG(I,J,K)
                END IF

                VP(I,J,K)  = VPP
                BW(I,J,K)  = BBW
                BG(I,J,K)  = BBG
                   SCFW    = SCFW+VP(I,J,K)*SW(I,J,K)/BW(I,J,K)
                   SCFG    = SCFG+VP(I,J,K)*SG(I,J,K)/BG(I,J,K)
                  SCFG1    = SCFG1+VP(I,J,K)*RSW*SW(I,J,K)/BW(I,J,K)

                   CALL   NTERP1 (NTE,PWT,BWPT,MPWT,PP,BWDER)
                   CALL   NTERP1 (NTE,PWT,RSWPT,MPWT,PP,RSWDER)
                   CALL   NTERP1 (NTE,PGT,BGPT,MPGT,PP,BGDER)
                   CALL   INTERP (NTE,PGT,CRT,MPGT,PP,CR)

                   CW = -(BWDER-BG(I,J,K)*RSWDER)/BW(I,J,K)
                   CG = -BGDER/BG(I,J,K)
                   CX = CT(I,J,K)
                   CT(I,J,K) = CR+CW*SW(I,J,K)+CG*SG(I,J,K)

                   IF (CT(I,J,K).LE.0.0) THEN
                      CT(I,J,K) = CX
                      WRITE(*,743)  I,J,K
                   END IF

                   IF (N.EQ.KCO)   WRITE(IOCODE,87)  I,J,K,CR,CW,RSW,CG
 400            CONTINUE
C ---------------------------------------------------------------------

C               AUTO TIME STEP CONTROL.  CALCULATE PRESSURE AT
C               MAXIMUM SATURATION.

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

                PPM = 0.0
                SWM = 0.0
                SGM = 0.0

                DO 240 K=1,KK
                DO 240 J=1,JJ
                DO 240 I=1,II
                   DPW = P(I,J,K)-PN(I,J,K)
                   DSW = SW(I,J,K)-SWN(I,J,K)
                   DSG = SG(I,J,K)-SGN(I,J,K)
                   IF (ABS(DPW).GT.ABS(PPM)) THEN
                      PPM = DPW
                      IP  = I
                      JP  = J
                      KP  = K
                   END IF
                   IF (ABS(DSW).GT.ABS(SWM)) THEN
                      SWM = DSW
                      ISW = I
                      JSW = J
                      KSW = K
                   END IF
```

```
            IF (ABS(DSG).GT.ABS(SGM)) THEN
                SGM = DSG
                ISG = I
                JSG = J
                KSG = K
            END IF
  240   CONTINUE

        DPMC = ABS(PPM)
        DSMC = ABS(SWM)
        IF (DSMC.LT.ABS(SGM)) THEN
            DSMC = ABS(SGM)
            IS   = ISG
            JS   = JSG
            KS   = KSG
        ELSE
            IS   = ISW
            JS   = JSW
            KS   = KSW
        END IF
C -------------------------------------------------------------------
C       REPEAT TIME STEP.
C -------------------------------------------------------------------
        IF (DSMC.LT.DSMAX .AND. DPMC.LT.DPMAX .AND.
     $      NITER.LT.MITER)      GO TO 402
        IF (DELT.LE.DTMIN .OR. FACT2.GE.1.0)    GO TO 402

        ITFLAG = ITFLAG+1
        DELT   = DELT*FACT2
        IF (DELT.LT.DTMIN)      DELT = DTMIN
        FT = ETI+DELT
        IF (FT.GT.FTMAX)        DELT = FTMAX-ETI
C -------------------------------------------------------------------
C       RESET VARIABLES.
C -------------------------------------------------------------------
        DO 250 I=1,II
        DO 250 J=1,JJ
        DO 250 K=1,KK
            P(I,J,K)  = PN(I,J,K)
            SW(I,J,K) = SWN(I,J,K)
            SG(I,J,K) = SGN(I,J,K)
  250   CONTINUE

        IF (NITER.LE.MITER) THEN
            IF (DSMC.GT.DSMAX) THEN
                WRITE(IOCODE,753) DSMC,IS,JS,KS
            END IF
            IF (DPMC.GT.DPMAX) THEN
                WRITE(IOCODE,763) DPMC,IP,JP,KP
            END IF
        END IF
        GO TO 1060
  402   CONTINUE

        DO 410 I=1,II
        DO 410 J=1,JJ
        DO 410 K=1,KK
            IF (P(I,J,K).LE.PN(I,J,K)) THEN
                IP = I+1
                IM = I-1
                JP = J+1
                JM = J-1
                KP = K+1
                KM = K-1
                IF (IP.LE.II) THEN
                    IF (SGN(IP,J,K).GT.0.0001) GO TO 410
                END IF
                IF (IM.GE.1) THEN
                    IF (SGN(IM,J,K).GT.0.0001) GO TO 410
                END IF
                IF (JP.LE.JJ) THEN
                    IF (SGN(I,JP,K).GT.0.0001) GO TO 410
                END IF
                IF (JM.GE.1) THEN
                    IF (SGN(I,JM,K).GT.0.0001) GO TO 410
                END IF
                IF (KP.LE.KK) THEN
                    IF (SGN(I,J,KP).GT.0.0001) GO TO 410
                END IF
                IF (KM.GE.1) THEN
                    IF (SGN(I,J,KM).GT.0.0001) GO TO 410
                END IF
                SG(I,J,K) = 0.0
            END IF
  410   CONTINUE
```

```
            IF (IREPRS.NE.1) THEN
               DO 50 I=1,II
               DO 50 J=1,JJ
               DO 50 K=1,KK
                  IF (SG(I,J,K).GT.0.0001) THEN
                     PP = P(I,J,K)
                     CALL  INTERP (NTE,PGT,BGT,MPGT,PP,BBG)
                  END IF
 50            CONTINUE
            END IF
C ---------------------------------------------------------------------
C           UPDATE OLD FLUID VOLUMES FOR MATERIAL BALANCE.
C ---------------------------------------------------------------------
            STBWI = STBW
            MCFGI = MCFGT
C ---------------------------------------------------------------------
C           UDATE NEW FLUID VOLUMES.
C ---------------------------------------------------------------------
            STBW  = SCFW*D5615
            MCFG  = SCFG*0.001
            MCFG1 = SCFG1*0.001
            MCFGT = MCFG+MCFG1
C ---------------------------------------------------------------------
C           DEBUG PRINT OF PRESENT AND FUTURE P,SW, AND
C           SG VALUES.
C ---------------------------------------------------------------------
            IF (KCOFF.NE.0) THEN
               DO 290 K=1,KK
               DO 290 J=1,JJ
               DO 290 I=1,II
                  WRITE(IOCODE,87) I,J,K,PN(I,J,K),SWN(I,J,K),
     $                             SGN(I,J,K)
                  WRITE(IOCODE,87) I,J,K,P(I,J,K),SW(I,J,K),SG(I,J,K)
 290           CONTINUE
            END IF
C ---------------------------------------------------------------------
C           WELL REPORT.
C ---------------------------------------------------------------------
            IJ  = 0
            TGR = 0.0
            TWR = 0.0
            TGC = 0.0
            TWC = 0.0

            DO J=1,NVQN
               WGR = 0.0
               IQ1 = IQN1(J)
               IQ2 = IQN2(J)
               IQ3 = IQN3(J)
               IJ  = IJ+1
               LAY = IQ3+LAYER(J)-1

               DO K=IQ3,LAY
                  QWW = QW(IQ1,IQ2,K)*D5615
                  QGG = QG(IQ1,IQ2,K)*0.001
                  CUMW(J,K) = CUMW(J,K)+QWW*DELT*0.001
                  CUMG(J,K) = CUMG(J,K)+QGG*DELT*0.001
                  IF (IWLREP.NE.0) THEN
                     IF (IJ.EQ.1 .AND. K.EQ.IQ3) THEN
                        WRITE(IOCODE,773) FT
                        WRITE(IOCODE,783)
                     END IF
                     IF (QGG.NE.0.0) THEN
                        WGR = QWW/(QGG*1000.0)
                     END IF

                     WRITE(IOCODE,793) WELLID(J),IQN1(J),IQN2(J),
     $                                 K,PWFC(J,K),PWF(J,K),PID(J,K),
     $                                 QGG,QWW,WGR,CUMG(J,K),
     $                                 CUMW(J,K)
                     TGR = TGR+QGG
                     TWR = TWR+QWW
                     TGC = TGC+CUMG(J,K)
                     TWC = TWC+CUMW(J,K)
                  END IF
               END DO
            END DO

            IF (IWLREP.NE.0) THEN
               WRITE(IOCODE,803) TGR,TWR,TGC,TWC
            END IF
```

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

C          CALCULATE MATERIAL BALANCE ERRORS AND AVERAGE
C          RESERVOIR PRESSURE.

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

           DELTO = DELT
           ETI   = ETI+DELT
           CALL  MATBAL (II,JJ,KK,STBW,STBWI,MCFG,MCFGI,MBEW,
     $                   MBEG,DELTO,RESVOL,WP,GP,WI,GI,PAVGO,
     $                   PAVG,NLOOP,WPR,GPR,WIR,GIR,D5615,CWP,
     $                   CGP,CWI,CGI,MCFG1,MCFGT,CWGR,WGR)

           IF (WGR.GT.WGRMAX) GO TO 1002
           IF (PAVG.LT.PAMIN) GO TO 1004
           IF (PAVG.GT.PAMAX) GO TO 1005

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

C          SUMMARY REPORT.

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

           IF (ISUMRY.NE.0) THEN
              NLP = N+1
              CALL  PRTPS (NLP,KPI,II,JJ,KK,PAVGO,PAVG,CWP,
     $                     CWI,CGP,CGI,MBEW,MBEG,DELTO,WPR,
     $                     GPR,WIR,GIR,ETI,CWGR,WGR,IPMAP,
     $                     ISWMAP,ISGMAP)
           END IF

           IF (N.EQ.KCO .OR. KCO1.EQ.0) GO TO 500
              DO 300 K=1,KK
              DO 300 J=1,JJ
              DO 300 I=1,II
                 WRITE(IOCODE,87) I,J,K,VP(I,J,K),CT(I,J,K),
     $                            BW(I,J,K),SW(I,J,K),BG(I,J,K),
     $                            SG(I,J,K)
 300          CONTINUE
 500       CONTINUE

           IF (N.EQ.KSN) KSN=KSN+KSN1
           IF (N.EQ.KSM) KSM=KSM+KSM1
           IF (N.EQ.KCO) KCO=KCO+KCO1

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

C          UPDATE ARRAYS.

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

           DO 1150 K=1,KK
           DO 1150 J=1,JJ
           DO 1150 I=1,II
              QW(I,J,K)  = 0.0
              QG(I,J,K)  = 0.0
              PN(I,J,K)  = P(I,J,K)
              SWN(I,J,K) = SW(I,J,K)
              SGN(I,J,K) = SG(I,J,K)
1150       CONTINUE
        END DO

1002  WRITE(IOCODE,2003)
        GO TO 1001
1004  WRITE(IOCODE,2004)
        GO TO 1001
1005  WRITE(IOCODE,2005)
1001  CONTINUE


31     FORMAT(/1X,'ENTER IOCODE ..........')
32     FORMAT(I4)
36     FORMAT(//3X,69('*')/3X,'*',67X,'*'/3X,'*',67X,'*'/3X,
     $         '*',17X,'                GWSIM: ',31X,'*'/3X,'*',
     $         15X,'3D GAS-WATER RESERVOIR SIMULATOR',20X,
     $         '*'/3X,'*',16X,'U.N.S.W.-VERSION 2.0 (25-10-88)',
     $         20X,'*'/3X,'*',67X,'*'/3X,'*',67X,'*'/3X,69('*')///)
69     FORMAT(80A1)
85     FORMAT(' ',' RESET TO PREVIOUS TOTAL COMP. AT POSN.',3I5)
87     FORMAT(1X,3I3,8E15.6)
110    FORMAT(3(/),10X,'LAYER',I3,' INITIAL FLUID VOLUMES'/10X,
     $         29('-')//13X,'WATER IN PLACE',T40,':',1X,F10.4,1X,
     $         'MILLION STB.'/13X,'SOLUTION GAS IN PLACE',T40,':',
     $         1X,F10.4,1X,'BILLION SCF.'/13X,'FREE GAS IN PLACE',
     $         T40,':',1X,F10.4,1X,'BILLION SCF.')
115    FORMAT(5(/),10X,'TOTAL FLUID VOLUMES IN RESERVOIR'/10X,
     $         32('-')//13X,'WATER IN PLACE',T40,':',F10.4,1X,
     $         'MILLION STB.'/13X,'SOLUTION GAS IN PLACE',T40,':',
     $         F10.4,1X,'BILLION SCF.'/13X,'FREE GAS IN PLACE',
     $         T40,':',F10.4,1X,'BILLION SCF.'//)
125    FORMAT(' ',' QRATE DONE !')
130    FORMAT(' ',' SOLMAT DONE !')
140    FORMAT(' ',' PRATEI DONE !')
675    FORMAT(' ',' ELAPSED TIME = ',F6.2,' DAYS FROM',
     $         ' BEGINNING OF SIMULATION')
```

```
676   FORMAT(' ','DELT = ',F8.3)
733   FORMAT(//)
743   FORMAT(' ','RESET TOTAL COM. TO PREVIOUS VALUE AT',
     $         ' POSN.',3I5)
753   FORMAT(' ','TIME STEP RETREAT-CHANGE IN SATURATION(',
     $        F6.4,') AT',3I5,' EXCEEDS USER DEFINED MAXIMUM')
763   FORMAT(' ','TIME STEP RETREAT-CHANGE IN PRESSURE(',
     $        F6.4,') AT',3I5,' EXCEEDS USER DEFINED MAXIMUM')
773   FORMAT(5(/),10X,' WELL REPORT FOR ALL ACTIVE WELLS ',
     $        11X,32('-')//20X,'ELAPSED TIME =',F8.3,
     $        ' DAYS FROM BEGINNING OF SIMULATION'///)
783   FORMAT(3(/),48X,8('-'),' RATE ',8('-'),5X,'--- CUMMULATIVE',
     $        '---'/5X,'WELL   LOCATION',4X,'CALC     SEC',
     $        '     SPEC',4X,'GAS      WATER       WGR',5X,
     $        '   GAS      WATER'/4X,' ID',5X,' I   J   K',
     $        '     BHFP     BHFP      PI',5X,'MCF/D       ',
     $        'STB/D',4X,'STB/SCF',2X,'MMSCF     MSTB'///)
793   FORMAT(5X,A5,1X,3I3,2F8.0,F8.3,2F9.0,F8.0,4X,2F8.0)
803   FORMAT(6X,102('_')/5X,'TOTALS',33X,2F9.0,14X,2F8.0//)
2003  FORMAT(/5X,'MAXIMUM WGR HAS BEEN EXCEEDED --- ',
     $        ' SIMULATION IS BEING TERMINATED'//)
2004  FORMAT(/5X,'MINIMUM AVERAGE RESERVOIR PRESSURE WAS NOT',
     $        ' ACHIEVED --- SIMULATION IS BEING TERMINATED'//)
2005  FORMAT(/5X,'MAXIMUM AVERAGE RESERVOIR PRESSURE HAS BEEN',
     $        ' EXCEEDED --- SIMULATION IS BEING TERMINATED'//)
      CLOSE(20)
      CLOSE(IOCODE)
      STOP
      END


C
C ======================================================================

      SUBROUTINE  BANDIN (II,JJ,KK)

C ======================================================================

C     THIS SUBROUTINE FIRST SETS UP FOR NORMAL ORDERING OF THE
C     MATRIX AND THEN SOLVES THE ALGEBRAIC EQUATIONS USING
C     DIRECT GAUSSIAN (BAND) ALGORITHM.

C     NOMENCLATURE:

C        AW(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW1-RSW))*AWW+
C                      BG(I,J,K)*AGW
C        AE(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW2-RSW))*AWE+
C                      BG(I,J,K)*AGE
C        AS(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW3-RSW))*AWS+
C                      BG(I,J,K)*AGS
C        AN(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW4-RSW))*AWN+
C                      BG(I,J,K)*AGN
C        AT(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW5-RSW))*AWT+
C                      BG(I,J,K)*AGT
C        AB(I,J,K)  :  (BW(I,J,K)+0.5*BG(I,J,K)*(RSW6-RSW))*AWB+
C                      BG(I,J,K)*AGB
C        BMAT       :  MATRIX AFTER NORMAL ORDERING OF THE COEFFI-
C                      CIENT MATRIX OF PRESSURE EQUATION
C        IC         :  COUNTER
C        ICM        :  IC-1
C        ICMM       :  IC-KK
C        ICMMM      :  IC-JJ*KK
C        ICP        :  IC+1
C        ICPP       :  IC+KK
C        ICPPP      :  IC+JJ*KK
C        II         :  NUMBER OF THE X-DIRECTION GRID BLOCKS
C        JJ         :  NUMBER OF THE Y-DIRECTION GRID BLOCKS
C        KK         :  NUMBER OF THE Z-DIRECTION GRID BLOCKS
C        NBAND      :  2*JJ*KK+1
C        NMAXI      :  II*JJ*KK
C        NROW       :  NX*NY*NZ
C        P(I,J,K)   :  PRESSURE ARRAY RETURNED TO MAIN PROGRAM
C        PVEC       :  PRESSURE ARRAY CALCULATED BY BAND ALGORITHM
C        QVEC       :  B(I,J,K)
C
C ----------------------------------------------------------------------

      INCLUDE 'COMGWSIM.FOR'

      NMAXI = II*JJ*KK

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

C         INITIALSIZE FACTOR MATRICES

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

      DO I=1,NMAXI
         DO J=1,NMAXI
            BMAT(I,J) = 0.0
         END DO
      END DO
```

```
      IC = 0
      DO 110 I=1,II
      DO 110 J=1,JJ
      DO 110 K=1,KK
         IC     = IC+1
         ICM    = IC-1
         ICMM   = IC-KK
         ICMMM  = IC-JJ*KK
         ICP    = IC+1
         ICPP   = IC+KK
         ICPPP  = IC+JJ*KK

         IF (ICMMM.LT.1) GO TO 115
         BMAT(IC,ICMMM) = AW(I,J,K)
115      IF (ICMM.LT.1 .OR. JJ.EQ.1) GO TO 120
         BMAT(IC,ICMM) = AS(I,J,K)
120      IF (ICM.LT.1 .OR. KK.EQ.1) GO TO 130
         BMAT(IC,ICM) = AT(I,J,K)
130      BMAT(IC,IC) = E(I,J,K)

         IF (ICP.GT.NMAXI .OR. KK.EQ.1) GO TO 140
         BMAT(IC,ICP) = AB(I,J,K)
140      IF (ICPP.GT.NMAXI .OR. JJ.EQ.1) GO TO 145
         BMAT(IC,ICPP) = AN(I,J,K)
145      IF (ICPPP.GT.NMAXI) GO TO 150
         BMAT(IC,ICPPP) = AE(I,J,K)
150      QVEC(IC) = B(I,J,K)
110   CONTINUE

      NBAND = 2*JJ*KK+1

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

      CALL   BAND (NROW,NMAXI,NBAND,BMAT,QVEC,PVEC,GVEC)

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

      IC = 0
      DO 200 I=1,II
      DO 200 J=1,JJ
      DO 200 K=1,KK
         IC      = IC+1
         P(I,J,K) = PVEC(IC)
200   CONTINUE

      RETURN
      END

C
C ======================================================================
      SUBROUTINE  BAND (NROW,NMAXI,NBAND,BMAT,QVEC,PVEC,GVEC)
C ======================================================================
      DIMENSION   BMAT(NROW,NMAXI),QVEC(NMAXI),PVEC(NMAXI),
     $            GVEC(NROW)
C -----------------------------------------------------------------------
C         BEGIN BAND ALGORITHM
C -----------------------------------------------------------------------
      NW1 = (NBAND-1)/2

      DO 1110 I=1,NMAXI
         L1 = I-NW1
         IF (L1.LT.1) L1=1

         DO 1120 J=L1,I
            IF (J.EQ.1) GO TO 1120
            JM  = J-1
            ADD = 0.0

            DO 1130 K=L1,JM
               IF (L1.GT.JM) GO TO 1125
               ADD = ADD+BMAT(I,K)*BMAT(K,J)
1130        CONTINUE

1125        BMAT(I,J) = BMAT(I,J)-ADD
1120     CONTINUE

         IF (I.EQ.NMAXI) GO TO 1110
         IP = I+1
         IM = I-1
         L2 = I+NW1

         IF (L2.GT.NMAXI) L2 = NMAXI

         DO 1140 J=IP,L2
            ADD = 0.0
            IF (I.EQ.1) GO TO 1140
            L3 = J-NW1
            IF (L3.LT.1) L3 = 1
```

```
                DO 1150 K=L3,IM
                   IF (L3.GT.IM) GO TO 1140
                   ADD = ADD+BMAT(I,K)*BMAT(K,J)
 1150           CONTINUE

 1140           BMAT(I,J) = (BMAT(I,J)-ADD)/BMAT(I,I)
 1110   CONTINUE
C ---------------------------------------------------------------------

C          FORWARD SOLUTION

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

       GVEC(1) = QVEC(1)/BMAT(1,1)

       DO I=2,NMAXI
          L1 = I-NW1
          IF (L1.LT.1) L1 = 1
          IM  = I-1
          ADD = 0.0

          DO K=L1,IM
             ADD = ADD+BMAT(I,K)*GVEC(K)
          END DO

          GVEC(I) = (QVEC(I)-ADD)/BMAT(I,I)
       END DO

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

C          BACKWARD SOLUTION

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

       PVEC(NMAXI) = GVEC(NMAXI)

       DO I=2,NMAXI
          INVI  = NMAXI+1-I
          INVIP = INVI+1
          L2    = INVI+NW1
          IF (L2.GT.NMAXI) L2 = NMAXI
          ADD = 0.0

          DO K=INVIP,L2
             ADD = ADD+BMAT(INVI,K)*PVEC(K)
          END DO
          PVEC(INVI) = GVEC(INVI)-ADD
       END DO

       RETURN
       END

C
C ======================================================================

       SUBROUTINE  CLSOR (II,JJ,KK,OMEGA,TOL,TOL1,MITER,
      $                   DELT,DELTO,KSN,N,NITER)

C ======================================================================

C      THE SUBROUTINE ITERATES THE SPARSE MATRIX, SIMULTANEOUS
C      EQUATIONS BY LINE SUCCESSIVE OVER RELAXATION METHOD
C      (CLSOR) TO NEW PRESSURE DISTRIBUTION UNTIL THE SPECIFIED
C      CONVERGENCE CRITERIAS ARE SATISFIED.

C      NOMENCLATURE:

C          AZL       : THE LOWER DIAGONAL MATRIX OF THE COEFFICIENT
C                      VECTOR OF THE PRESSURE EQUATION
C          BZL       : THE MAIN DIAGONAL MATRIX OF THE COEFFICIENT
C                      VECTOR OF THE PRESSURE EQUATION
C          CZL       : THE UPPER DIAGONAL MATRIX OF THE COEFFICIENT
C                      VECTOR OF THE PRESSURE EQUATION
C          DELTO     : TIME STEP, DAYS
C          DZL       : THE COLUMN VECTOR OF THE PRESSURE EQUATION
C          II        : NUMBER OF THE X-DIRECTION GRID BLOCKS
C          JJ        : NUMBER OF YHE Y-DIRECTION GRID BLOCKS
C          KK        : NUMBER OF THE Z-DIRECTION GRID BLOCKS
C          KSN       : CLSOR PARAMETER DEBUG OUTPUT CONTROL
C          MITER     : MAXIMUM NUMBER OF CLSOR ITERATIONS PER TIME
C                      STEP.  A TYPICAL VALUE FOR MITER IS 250
C          NITER     : ITERATION COUNTER
C          OMEGA     : INITIAL CLSOR ACCELERATION PARAMETER.  INITIAL
C                      VALUE FOR OMEGA SHOULD BE IN THE RANGE 1 AND
C                      2.   A TYPICAL INITIAL VALUE FOR OMEGA WOULD
C                      1.7.  THE PROGRAM WILL OPTIMIZE AS THE
C                      SOLUTION PROCEEDS.
```

```
C          TOL       : MAXIMUM ACCEPTABLE PRESSURE CHANGE FOR
CC                     CONVERGENCE OF CLSOR ITERATIONS, PSI.
CCC                    A TYPICAL VALUE FOR TOL WOULD BE 0.1.
CCCC       TOL1      : PARAMETER FOR DETERMINING WHEN TO CHANGE
CCCCC                  OMEGA.  A TYPICAL VALUE FOR TOL1 WOULD BE
CCCCCC                 0.001.  IF TOL1=0, THE INITIAL VALUE ENTERED
CCCCCCC                AS OMEGA ABOVE WILL BE USED FOR THE ENTIRE RUN.
CCCCCC     UZL       : SOLUTION VECTOR OF THE PRESSURE EQUATION
C ------------------------------------------------------------------

      INCLUDE 'COMGWSIM.FOR'

      DIMENSION  UM(NPMAX),AZL(NPMAX),BZL(NPMAX),CZL(NPMAX),
     $           DZL(NPMAX),UZL(NPMAX),BETA(NPMAX),
     $           GAMMA(NPMAX),W(NPMAX)


      DIV   = DELT/DELTO
      NITER = 0
      DMAX  = 1.0
      RHO1  = 0.0
      THETA = 0.0

  110 TW      = 1.0 - OMEGA
      DMAXO   = DMAX
      THETAO = THETA
      IF (NITER .LT. MITER) THEN
         NITER = NITER + 1
         DMAX  = 0.0
         IF (NITER.EQ.1 .OR. NITER MOD 10.EQ.0) THEN
            IF (II .NE. 1) THEN
               DO I=1,II
                  AZL(I) = 0.0
                  BZL(I) = 0.0
                  CZL(I) = 0.0
                  DZL(I) = 0.0
               END DO

               DO 3030 I=1,II
               DO 3030 K=1,KK
               DO 3030 J=1,JJ
                  IM = I-1
                  IP = I+1
                  IF (I .EQ. 1) IM=1
                  IF (I .EQ. II) IP=II

                  JM = J-1
                  JP = J+1
                  IF (J .EQ. 1) JM=1
                  IF (J .EQ. JJ) JP=JJ

                  KM = K-1
                  KP = K+1
                  IF (K .EQ. 1) KM=1
                  IF (K .EQ. KK) KP=KK

                  AZL(I) = AZL(I)+AW(I,J,K)
                  BZL(I) = BZL(I)+E(I,J,K)+AT(I,J,K)+AB(I,J,K)+
     $                     AS(I,J,K)+AN(I,J,K)
                  CZL(I) = CZL(I)+AE(I,J,K)
                  DZL(I) = DZL(I)-(AT(I,J,K)*P(I,J,KM)+
     $                     AS(I,J,K)*P(I,JM,K)+AW(I,J,K)*
     $                     P(IM,J,K)+AB(I,J,K)*P(I,J,KP)+
     $                     AN(I,J,K)*P(I,JP,K)+AE(I,J,K)*
     $                     P(IP,J,K)+E(I,J,K)*P(I,J,K)-B(I,J,K))
 3030          CONTINUE
C ------------------------------------------------------------------

               CALL LTRI(II,AZL,BZL,CZL,DZL,UZL,BETA,GAMMA,
     $                   W,NPMAX)
C ------------------------------------------------------------------

               DO 3040 I=1,II
               DO 3040 K=1,KK
               DO 3040 J=1,JJ
                  P(I,J,K) = P(I,J,K)+UZL(I)
 3040          CONTINUE
            END IF

            IF (JJ .NE. 1) THEN
               DO J=1,JJ
                  AZL(J) = 0.0
                  BZL(J) = 0.0
                  CZL(J) = 0.0
                  DZL(J) = 0.0
               END DO
```

```
      DO 3060 J=1,JJ
      DO 3060 K=1,KK
      DO 3060 I=1,II
         IM = I-1
         IP = I+1
         IF (I .EQ. 1) IM=1
         IF (I .EQ. II) IP=II

         JM = J-1
         JP = J+1
         IF (J .EQ. 1) JM=1
         IF (J .EQ. JJ) JP=JJ

         KM = K-1
         KP = K+1
         IF (K .EQ. 1) KM=1
         IF (K .EQ. KK) KP=KK

         AZL(J) = AZL(J)+AS(I,J,K)
         BZL(J) = BZL(J)+E(I,J,K)+AT(I,J,K)+AB(I,J,K)+
     $            AW(I,J,K)+AE(I,J,K)
         CZL(J) = CZL(J)+AN(I,J,K)
         DZL(J) = DZL(J)-(AT(I,J,K)*P(I,J,KM)+AS(I,J,K)*
     $            P(I,JM,K)+AW(I,J,K)*P(IM,J,K)+
     $            AB(I,J,K)*P(I,J,KP)+AN(I,J,K)*
     $            P(I,JP,K)+AE(I,J,K)*P(IP,J,K)+
     $            E(I,J,K)*P(I,J,K)-B(I,J,K))
3060     CONTINUE
C -----------------------------------------------------------------------
         CALL LTRI(JJ,AZL,BZL,CZL,DZL,UZL,BETA,
     $             GAMMA,W,NPMAX)
C -----------------------------------------------------------------------
         DO 3070 J=1,JJ
         DO 3070 K=1,KK
         DO 3070 I=1,II
            P(I,J,K) = P(I,J,K)+UZL(J)
3070     CONTINUE
      END IF

      IF (KK .NE. 1) THEN
         DO K=1,KK
            AZL(K) = 0.0
            BZL(K) = 0.0
            CZL(K) = 0.0
            DZL(K) = 0.0
         END DO

      DO 3090 K=1,KK
      DO 3090 I=1,II
      DO 3090 J=1,JJ
         IM = I-1
         IP = I+1
         IF (I .EQ. 1) IM=1
         IF (I .EQ. II) IP=II

         JM = J-1
         JP = J+1
         IF (J .EQ. 1) JM=1
         IF (J .EQ. JJ) JP=JJ

         KM = K-1
         KP = K+1
         IF (K .EQ. 1) KM=1
         IF (K .EQ. KK) KP=KK

         AZL(K) = AZL(K)+AT(I,J,K)
         BZL(K) = BZL(K)+E(I,J,K)+AW(I,J,K)+AE(I,J,K)+
     $            AS(I,J,K)+AN(I,J,K)
         CZL(K) = CZL(K)+AB(I,J,K)
         DZL(K) = DZL(K)-(AT(I,J,K)*P(I,J,KM)+AS(I,J,K)*
     $            P(I,JM,K)+AW(I,J,K)*P(IM,J,K)+
     $            AB(I,J,K)*P(I,J,KP)+AN(I,J,K)*
     $            P(I,JP,K)+AE(I,J,K)*P(IP,J,K)+
     $            E(I,J,K)*P(I,J,K)-B(I,J,K))
3090     CONTINUE
C -----------------------------------------------------------------------
         CALL LTRI(KK,AZL,BZL,CZL,DZL,UZL,BETA,
     $             GAMMA,W,NPMAX)
C -----------------------------------------------------------------------
         DO 4010 K=1,KK
         DO 4010 J=1,JJ
         DO 4010 I=1,II
            P(I,J,K) = P(I,J,K)+UZL(K)
4010     CONTINUE
      END IF
   END IF
```

```
      DO 200 K=1,KK
      DO 200 J=1,JJ
         DO I=1,II
            UM(I) = P(I,J,K)
            AZL(I) = AW(I,J,K)
            BZL(I) = E(I,J,K)
            CZL(I) = AE(I,J,K)
            DZL(I) = B(I,J,K)
            IF (JJ .NE. 1) THEN
               JM = J-1
               JP = J+1
               IF (J .EQ. 1) JM=1
               IF (J .EQ. JJ) JP=JJ
               DZL(I) = DZL(I)-AS(I,J,K)*P(I,JM,K)-AN(I,J,K)*
     $                    P(I,JP,K)
            END IF

            IF (KK .NE. 1) THEN
               KM = K-1
               KP = K+1
               IF (K .EQ. 1) KM=1
               IF (K .EQ. KK) KP=KK
               DZL(I) = DZL(I)-AT(I,J,K)*P(I,J,KM)-AB(I,J,K)*
     $                    P(I,J,KP)
            END IF
         END DO
C -------------------------------------------------------------------

         CALL LTRI(II,AZL,BZL,CZL,DZL,UZL,BETA,
     $              GAMMA,W,NPMAX)
C -------------------------------------------------------------------

         WRITE(*,76) NITER
         DO I=1,II
            GSLSOR    = UZL(I)
            P(I,J,K) = TW*UM(I)+OMEGA*GSLSOR
            ARG       = P(I,J,K)-UM(I)
            DM        = ABS(ARG)
            IF (DM .GT. DMAX) DMAX=DM
         END DO
200   CONTINUE

      IF (TOL1 .NE. 0.0) THEN
         THETA = DMAX/DMAXO
         DELTA = THETA-THETAO
         ARG   = DELTA
         ARG   = ABS(ARG)
      ELSE
         GO TO 25
      END IF

      IF (ARG .LE. TOL1) THEN
         OM = OMEGA-1.0
      ELSE
         GO TO 25
      END IF

      IF (THETA.EQ.0.0 .OR. OMEGA.EQ.0.0) THEN
         RHO1 = 1.0
         GO TO 25
      END IF


      RHO1 = (THETA+OM)*(THETA+OM)/(THETA*OMEGA*OMEGA)

      IF (RHO1 .LT. 1.0) THEN
         ARG   = 1.0-RHO1
         OMEGA = 2.0/(1.0+SQRT(ARG))
      ELSE
         GO TO 25
      END IF

25    IF (DMAX .GT. TOL) GO TO 110
      WRITE(IOCODE,40) NITER,OMEGA,DMAX,THETA,RHO1
      ELSE
      WRITE(IOCODE,30) NITER,TOL,DMAX
      END IF


30    FORMAT(5(/),10X,'CONVERGENCE (LSOR) WAS NOT REACHED IN',
     $       I5,' ITERATIONS'//20X,'TOL = ',F10.7//20X,
     $       'DMAX = ',F15.7/)
40    FORMAT(5(/),10X,'CONVERGENCE (LSOR) HAS BEEN REACHED ',
     $       'AFTER',I3,' ITERATIONS'//15X,'OMEGA',T25,':',
     $       F6.3/15X,'DMAX',T25,':',F10.6/15X,'THETA',T25,
     $       ':',F10.6/15X,'RHO1',T25,':',F10.6/)
76    FORMAT('    ','LTRI  :',I5)
      RETURN
      END
```

```
C
C ==========================================================================

      SUBROUTINE  CODES (IOCODE,KSM1,KSN1,KCO1,NN,FACT1,FACT2,
     $                   TMAX,KSOL,MITER,OMEGA,TOL,TOL1,KSN,
     $                   KSM,KCO,KTR,KCOFF,DSMAX,DPMAX,WGRMAX,
     $                   PAMIN,PAMAX)

C ==========================================================================

C       THE SUBROUTINE CALLS FOR SEVERAL CODES THAT CONTROL DIA-
C       GNOSTIC OUTPUT FOR USE IN PROGRAM DEBUGGING.  THESE CODES
C       ARE NORMALLY ALWAYS SET EQUAL TO ZEROS.  IT ALSO CALLS FOR
C       RUN CONTROL PARAMETERS AND SOLUTION METHOD CONTROL PARA-
C       METERS.

C       NOMENCLATURE:

CC            DPMAX     :  MAXIMUM PRESSURE CHANGE PERMITTED OVER
CC                           A TIME STEP, PSI.
CC                           THE TIME STEP SIZE WILL BE REDUCED BY
CC                           FACT2 IF THE PRESSURE CHANGE IN ANY GRID
CC                           BLOCK EXCEEDS DPMAX DURING A TIME STEP.
CC            DSMAX     :  MAXIMUM SATURATION CHANGE PERMITTED OVER
CC                           A TIME STEP, FRACTION.
CC                           THE TIME STEP SIZE WILL BE REDUCED BY
CC                           FACT2 IF THE SATURATION CHANGE OF ANY
CC                           PHASE IN ANY GRID BLOCK EXCEEDS DSMAX
CC                           DURING A TIME STEP.
CC            FACT1     :  FACTOR FOR INCREASING TIME STEP SIZE UNDER
CC                           AUTOMATIC TIME STEP CONTROL.
CC                           SET FACT1 = 1.0 FOR FIXED TIME STEP SIZE.
CC                           A COMMON VALUE FOR FACT1 IS 1.25.
CC            FACT2     :  FACTOR FOR DECREASING TIME STEP SIZE UNDER
CC                           AUTOMATIC TIME STEP CONTROL.
CC                           SET FACT2 = 1.0 FOR FIXED TIME STEP SIZE.
CC                           A COMMON VALUE FOR FACT2 IS 0.5.
CC            IOCODE    :  CODE FOR INPUT FILE.
C             IHEDIN    :  STORAGE MEMORY FOR INPUT TITLE CARD.
C             KSM1      :  SOLUTION MATRIX DEBUG OUTPUT CONTROL.
C             KSN1      :  LSOR PARAMETER DEBUG OUTPUT CONTROL.
C             KCO1      :  COMPRESSIBILITY AND VOLUME FACTORS DEBUG
CC                           OUTPUT CONTROL.
CC            KSOL      :  SOLUTION METHOD CODE.
C                            KSOL = 1 MEANS 'DIRECT SOLUTION - BAND
CC                           ALGORITHM'.
CC                           KSOL = 2 MEANS 'LSOR METHOD'.
CC                           KSOL = 3 MEANS 'DIRECT SOLUTION - D4
CC                           ALGORITHM'.
C             KTR       :  TRANSMISSIBILITY DEBUG OUTPUT CONTROL.
C             KCOFF     :  DENSITY AND SATURATION DEBUG OUTPUT CONTROL.
C             MITER     :  MAXIMUM NUMBER OF LSOR ITERATIONS PER TIME
CC                           STEP.
CC                           A TYPICAL VALUE FOR MITER IS 250.
C             NN        :  MAXIMUM NUMBER OF TIME STEPS ALLOWED BEFORE
CC                           RUN IS TERMINATED.
C             OMEGA     :  INITIAL LSOR ACCELERATION PARAMETER,
CC                           INITIAL VALUE FOR OMEGA SHOULD BE IN THE
CC                           RANGE 1 AND 2.
CC                           A TYPICAL INITIAL VALUE FOR OMEGA WOULD BE
CC                           1.7.  THE PROGRAM WILL OPTIMIZE OMEGA AS THE
CC                           SOLUTION PROCEEDS.
C             PAMIN     :  LIMITING MINIMUM FIELD AVERAGE PRESSURE,
CC                           PSIA.
C             PAMAX     :  LIMITING MAXIMUM FIELD AVERAGE PRESSURE,
CC                           PSIA.
CC                           RUN WILL BE TERMINATED WHEN AVERAGE RESER-
CC                           VOIR PRESSURE EXCEEDS PAMAX.
C             TMAX      :  MAXIMUM REAL TIME TO BE SIMULATED DURING
CC                           THIS RUN, DAYS.
CC                           RUN WILL BE TERMINATED WHEN TIME EXCEEDS
C                            TMAX.
C             TOL       :  MAXIMUM ACCEPTABLE PRESSURE CHANGE FOR CON-
CC                           VERGENCE OF LSOR ITERATIONS, PSI.
CC                           A TYPICAL VALUE FOR TOL WOULD BE 0.1.
C             TOL1      :  PARAMETER FOR DETERMINING WHEN TO CHANGE
CC                           OMEGA.
CC                           A TYPICAL VALUE FOR TOL1 WOULD BE 0.001.
CC                           IF TOL1 IS SET TO ZERO, THE INITIAL VALUE
CC                           ENTERTED AS OMEGA ABOVE WILL BE USED FOR THE
CC                           ENTIRE RUN.
C             WGRMAX    :  LIMITING MAXIMUM FIELD GAS-WATER RATIO,
C                            SCF/STB.

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

      CHARACTER IHEDIN(80)

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

C         READ CODES FOR CONTROLLING DIAGNOSTIC OUTPUT.

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

```
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KSN1,KSM1,KCO1,KTR,KCOFF
      WRITE(*,5) IOCODE

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

C         READ RUN CONTROL PARAMETERS.

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

      WRITE(IOCODE,50)
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) NN,FACT1,FACT2,TMAX,WGRMAX,PAMIN,PAMAX
      WRITE(IOCODE,59) NN,FACT1,FACT2,TMAX,WGRMAX,PAMIN,PAMAX

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

C         READ SOLUTION METHOD CONTROL PARAMETERS.

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

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KSOL,MITER,OMEGA,TOL,TOL1,DSMAX,DPMAX
      WRITE(IOCODE,71)

      IF (KSOL .EQ. 1) THEN
         WRITE(IOCODE,73)
      ELSE IF (KSOL .EQ. 2) THEN
         WRITE(IOCODE,75) MITER,OMEGA,TOL,TOL1
      ELSE
         CONTINUE
      END IF

      WRITE(IOCODE,79) DSMAX,DPMAX
      KSN = KSN1
      KSM = KSM1
      KCO = KCO1


 5    FORMAT(///' ***** VALUE OF IOCODE ***** ',I5/)
 50   FORMAT(7(/),30X,'RUN CONTROL PARAMETERS'/30X,22('_')//)
 59   FORMAT(10X,'MAXIMUM NUMBER OF TIME STEPS',T50,'(NN)',
     $          T60,':',I8,
     $       /10X,'FACTOR FOR INCREASING TIME STEP',T50,'(FACT1)',
     $          T60,':',F11.2,
     $       /10X,'FACTOR FOR DECREASING TIME STEP',T50,'(FACT2)',
     $          T60,':',F11.2,
     $       /10X,'MAXIMUM SIMULATION TIME',T50,'(TMAX)',
     $          T60,':',F11.2,1X,'DAYS.',
     $       /10X,'MAXIMUM RESERVOIR WGR/TIME STEP',T50,'(WGRMAX)',
     $          T60,':',F11.2,1X,'STB/SCF.',
     $       /10X,'MINIMUM AVG. RESERVOIR PR/TIME STEP',T50,
     $          '(PAMIN)',T60,':',F11.2,1X,'PSIA.',
     $       /10X,'MAXIMUM AVG. RESERVOIR PR/TIME STEP',T50,
     $          '(PAMAX)',T60,':',F11.2,1X,'PSIA.'///)
 69   FORMAT(80A1)
 71   FORMAT(30X,'SOLUTION METHOD SPECIFICATIONS'/30X,30('_')//)
 73   FORMAT(20X,'SOLUTION METHOD IS BAND.')
 75   FORMAT(20X,'SOLUTION METHOD IS CLSOR.'//10X,
     $       'MAXIMUM NUMBER OF ITERATIONS',T50,'(MITER)',T60,
     $        ':',I5/10X,
     $       'INITIAL ACCELERATION PARAMETER',T50,'(OMEGA)',T60,
     $        ':',F10.4/10X,
     $       'MAXIMUM PRESSURE RESIDUAL',T50,'(TOL)',T60,':',
     $        F10.4,1X,'PSI.'/10X,
     $       'PARAMETER FOR CHANGING OMEGA',T50,'(TOL1)',T60,
     $        ':',F10.4///)
 79   FORMAT(20X,'AUTOMATIC TIME STEP CRITERIA.'//10X,
     $          'MAXIMUM ALLOWED SATURATION CHANGE',T50,
     $          '(DSMAX)',T60,':',F10.4/10X,
     $          'MAXIMUM ALLOWED PRESSURE CHANGE',T50,
     $          '(DPMAX)',T60,':',F10.4///)
      RETURN
      END

C
C ================================================================

      SUBROUTINE  GRID1 (II,JJ,KK)

C ================================================================

C     THE SUBROUTINE READS THE NUMBER OF GRID BLOCKS FOR THE SIM-
C     ULATOR IN THREE-DIMENSION GRID SYSTEM, AND ESTABLISHES GRID
C     DIMENSIONS USING THE CONTROL CODES.  THE PROGRAM COULD MOD-
C     IFY TO GRID DIMENSIONS IF NECESSARY.  ALSO CALCULATED ARE
C     NODE MIDPOINT ELEVATIONS OF ALL BLOCKS IN THE MODEL GRID.
```

```
C       NOMENCLATURE:
C
C           DXC           :   A SINGLE CONSTANT VALUE OF X-DIRECTION
CC                            GRID DIMENSION (FT) FOR KDX = -1.
CC                            THE X-DIRECTION GRID DIMENSIONS ARE THE
CC                            SAME FOR ALL BLOCKS IN THE MODEL GRID.
CC          DYC           :   A SINGLE CONSTANT VALUE OF Y-DIRECTION
CC                            GRID DIMENSION (FT) FOR KDY = -1.
CC                            THE Y-DIRECTION GRID DIMENSIONS ARE THE
CC                            SAME FOR ALL BLOCKS IN THE MODEL GRID.
CC          DZC           :   A SINGLE CONSTANT VALUE OF Z-DIRECTION
CC                            GRID DIMENSION (FT) FOR KDZ = -1.
CC                            THE Z-DIRECTION GRID DIMENSIONS ARE THE
CC                            SAME FOR ALL BLOCKS IN THE MODEL GRID.
CC          DX(I,J,K)     :   VARIABLE X-DIRECTION GRID DIMENSION IN FT.
CC                            THE X-DIRECTION GRID DIMENSIONS FOR EVERY
CC                            BLOCK IN LAYER 1 MUST BE READ, AND THEN
C                             ASSIGNED TO ALL OTHER LAYERS IN THE MODEL
CC                            GRID IF KDX = +1.
CC          DY(I,J,K)     :   VARIABLE Y-DIRECTION GRID DIMENSION IN FT.
CC                            THE Y-DIRECTION GRID DIMENSIONS FOR EVERY
CC                            BLOCK IN LAYER 1 MUST BE READ, AND THEN
CC                            ASSIGNED TO ALL OTHER LAYERS IN THE MODEL
CC                            GRID IF KDY = +1.
CC          DZ(I,J,K)     :   VARIABLE Z-DIRECTION GRID DIMENSION IN FT.
CC                            THE Z-DIRECTION DIMENSIONS FOR ALL BLOCKS
CC                            IN THE MODEL GRID MUST BE READ IF KDZ = +1.
CC          ELEV          :   A SINGLE CONSTANT DEPTH VALUE (FT) TO TOP
CC                            OF GRID BLOCKS IN LAYER 1.
CC                            ELEV MUST BE READ IF KEL = 0.
CC          EL(I,J,K)     :   VARIABLE NODE MIDPOINT ELEVATIONS (FT) OF
CC                            ALL BLOCKS IN THE MODEL GRID.
CC          IDCODE        :   PRINT CODE FOR MODIFICATION TO GRID
CC                            DIMENSIONS.
CC                            IDCODE = 0 MEANS 'DO NOT PRINT', AND
CC                            IDCODE = 1 MEANS 'PRINT'.
CC          I             :   X-COORDINATE OF BLOCK TO BE MODIFIED.
CC          II            :   NUMBER OF X-DIRECTION BLOCKS USED IN THE
CC                            MODEL GRID.
CC          IHEDIN(L)     :   STORAGE FOR THE TITLE CARD.
CC          J             :   Y-COORDINATE OF BLOCK TO BE MODIFIED.
CC          JJ            :   NUMBER OF Y-DIRECTION BLOCKS USED IN THE
CC                            MODEL GRID.
CC          K             :   Z-DIRECTION OF BLOCK TO BE MODIFIED.
CC          KK            :   NUMBER OF Z-DIRECTION BLOCKS USED IN THE
C                             MODEL GRID.
CC          KDX           :   CONTROL CODE FOR INPUT OF X-DIRECTION
CC                            GRID DIMENSIONS.
CC          KDY           :   CONTROL CODE FOR INPUT OF Y-DIRECTION
CC                            GRID DIMENSIONS.
CC          KDZ           :   CONTROL CODE FOR INPUT OF Z-DIRECTION
CC                            GRID DIMENSIONS.
CC          NUMDX         :   NUMBER OF GRID BLOCKS WHERE DX IS TO BE
CC                            CHANGED.
CC          NUMDY         :   NUMBER OF GRID BLOCKS WHERE DY IS TO BE
CC                            CHANGED.
CC          NUMDZ         :   NUMBER OF GRID BLOCKS WHERE DZ IS TO BE
CC                            CHANGED.
CC          NX            :   MAXIMUM NUMBER OF X-DIRECTION BLOCKS
CC                            ALLOWED TO USE IN THE MODEL GRID.
CC          NY            :   MAXIMUM NUMBER OF Y-DIRECTION BLOCKS
CC                            ALLOWED TO USE IN THE MODEL GRID.
CC          NZ            :   MAXIMUM NUMBER OF Z-DIRECTION BLOCKS
CC                            ALLOWED TO USE IN THE MODEL GRID.
CC          RDXL(I)       :   VARIABLE X-DIRECTION DIMENSIONS (FT) FOR
CC                            II BLOCKS IN THE FIRST ROW (J=1) OF
CC                            LAYER 1.
CC                            RDXL FOR II BLOCKS MUST BE READ, AND THEN
CC                            ASSIGNED TO ALL OTHER ROWS AND ALL OTHER
CC                            LAYERS IN THE MODEL GRID IF KDX = 0.
CC          RDYL(J)       :   VARIABLE Y-DIRECTION DIMENSIONS (FT) FOR
C                             JJ BLOCKS IN THE FIRST COLUMN (I=1) OF
CC                            LAYER 1.
CC                            RDYL FOR JJ BLOCKS MUST BE READ, AND THEN
CC                            ASSIGNED TO ALL OTHER COLUMNS AND ALL
CC                            OTHER LAERS IN THE MODEL GRID IF KDY = 0.
CC          RDZL(K)       :   VARIABLE Z-DIRECTION DIMENSIONS (FT) WITH
CC                            ONLY ONE VALUE FOR EACH LAYER IN THE
CC                            MODEL GRID.
CC                            RDZL FOR KK LAYERS MUST BE READ IF KDZ=0.
CC          VAREL(I,J)    :   A SEPARATE DEPTH VALUE (FT) TO TOP OF EACH
CC                            GRID BLOCK IN LAYER 1. VAREL FOR II*JJ
CC                            BLOCKS MUST BE READ IF KEL = 1.
CC          ZSUM(I,J)     :   TEMPORARY STORAGE FOR SUMMATION OF THICK-
C                             NESS OF BLOCKS IN THE MODEL GRID.

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

```
      INCLUDE 'COMGWSIM.FOR'
      DIMENSION     ZSUM(NX,NY), VAREL(NX,NY), RDXL(NX),
     $              RDYL(NY), RDZL(NZ)
C -----------------------------------------------------------------
C
C         READ NUMBER OF BLOCKS IN THE MODEL GRID IN THE
C         X,Y, AND Z-DIRECTIONS.
C
C -----------------------------------------------------------------
      READ(20,69) (IHEDIN(L), L=1,80)
      WRITE(IOCODE,70) (IHEDIN(L), L=1,80)
      READ(20,69) (IHEDIN(L), L=1,80)
      WRITE(IOCODE,70) (IHEDIN(L), L=1,80)
      READ(20,*) II,JJ,KK
      WRITE(IOCODE,25) II,JJ,KK
C -----------------------------------------------------------------
C
C         READ INPUT CODES FOR DX, DY, AND DZ.
C
C -----------------------------------------------------------------
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KDX,KDY,KDZ
C -----------------------------------------------------------------
C
C         ESTABLISH GRID BLOCK LENGTH (DX) DISTRIBUTION.
C
C -----------------------------------------------------------------
      WRITE(IOCODE,26)
      IF (KDX .LT. 0) THEN
         READ(20,*) DXC
         DO 175 K=1,KK
         DO 175 J=1,JJ
         DO 175 I=1,II
           DX(I,J,K) = DXC
 175     CONTINUE
         WRITE(IOCODE,29) DXC

      ELSE IF (KDX .EQ. 0) THEN
         READ(20,*) (RDXL(I), I=1,II)
         DO 187 K=1,KK
         DO 187 J=1,JJ
         DO 187 I=1,II
           DX(I,J,K) = RDXL(I)
 187     CONTINUE
         DO I=1,II
           WRITE(IOCODE,511) I,RDXL(I)
         END DO

      ELSE
         WRITE(IOCODE,43)
         K=1
         WRITE(IOCODE,38) K
         DO J=1,JJ
           READ(20,*) (DX(I,J,K), I=1,II)
           WRITE(IOCODE,72) (DX(I,J,K), I=1,II)
         END DO

         DO 194 K=2,KK
           WRITE(IOCODE,38) K
         DO 194 J=1,JJ
         DO 193 I=1,II
 193       DX(I,J,K) = DX(I,J,1)
           WRITE(IOCODE,72) (DX(I,J,K), I=1,II)
 194     CONTINUE
      END IF
C -----------------------------------------------------------------
C
C         ESTABLISH GRID BLOCK WIDTH (DY) DISTRIBUTION.
C
C -----------------------------------------------------------------
      IF (KDY .LT. 0) THEN
         READ(20,*) DYC
         DO 202 K=1,KK
         DO 202 J=1,JJ
         DO 202 I=1,II
           DY(I,J,K) = DYC
 202     CONTINUE
         WRITE(IOCODE,33) DYC

      ELSE IF (KDY .EQ. 0) THEN
         READ(20,*) (RDYL(J), J=1,JJ)
         DO 205 K=1,KK
         DO 205 J=1,JJ
         DO 205 I=1,II
           DY(I,J,K) = RDYL(J)
 205     CONTINUE
         DO J=1,JJ
           WRITE(IOCODE,512) J,RDYL(J)
         END DO
```

```
      ELSE
          WRITE(IOCODE,47)
          K=1
          WRITE(IOCODE,38) K
          DO 215 I=1,II
            READ(20,*) (DY(I,J,K), J=1,JJ)
            WRITE(IOCODE,72) (DY(I,J,K), J=1,JJ)
215       CONTINUE

          DO 214 K=2,KK
            WRITE(IOCODE,38) K
          DO 214 J=1,JJ
          DO 213 I=1,II
213         DY(I,J,K) = DY(I,J,1)
            WRITE(IOCODE,72) (DY(I,J,K), I=1,II)
214       CONTINUE
      END IF
C -----------------------------------------------------------------
C         ESTABLISH GRID BLOCK DEPTH (DZ) DISTRIBUTION.
C -----------------------------------------------------------------
      IF (KDZ .LT. 0) THEN
          READ(20,*) DZC
          DO 230 K=1,KK
          DO 230 J=1,JJ
          DO 230 I=1,II
            DZ(I,J,K) = DZC
230       CONTINUE
          WRITE(IOCODE,36) DZC

      ELSE IF (KDZ .EQ. 0) THEN
          READ(20,*) (RDZL(K), K=1,KK)
          DO 235 K=1,KK
          DO 235 J=1,JJ
          DO 235 I=1,II
            DZ(I,J,K) = RDZL(K)
235       CONTINUE

          DO K=1,KK
            WRITE(IOCODE,513) K,RDZL(K)
          END DO

      ELSE
          WRITE(IOCODE,48)
          DO 240 K=1,KK
            WRITE(IOCODE,38) K
          DO 240 J=1,JJ
            READ(20,*) (DZ(I,J,K), I=1,II)
            WRITE(IOCODE,72) (DZ(I,J,K), I=1,II)
240       CONTINUE
      END IF
C -----------------------------------------------------------------
C         MODIFICATION TO GRID DIMENSIONS.
C         READ NUMBER OF GRID BLOCKS TO BE MODIFIED AND
C         INPUT PRINT CODE.
C -----------------------------------------------------------------
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) NUMDX,NUMDY,NUMDZ,IDCODE

      IF (NUMDX .NE. 0) THEN
          WRITE(IOCODE,31)
          DO L=1,NUMDX
            READ(20,*) I,J,K,DX(I,J,K)
            WRITE(IOCODE,32) I,J,K,DX(I,J,K)
          END DO

          IF (IDCODE .EQ. 1) THEN
              WRITE(IOCODE,143)
              DO 853 K=1,KK
                WRITE(IOCODE,38) K
              DO 853 J=1,JJ
                WRITE(IOCODE,72) (DX(I,J,K), I=1,II)
853           CONTINUE
          END IF
      END IF

      IF (NUMDY .NE. 0) THEN
          WRITE(IOCODE,34)
          DO L=1,NUMDY
            READ(20,*) I,J,K,DY(I,J,K)
            WRITE(IOCODE,32) I,J,K,DY(I,J,K)
          END DO

          IF (IDCODE .EQ. 1) THEN
              WRITE(IOCODE,147)
              DO 855 K=1,KK
                WRITE(IOCODE,38) K
              DO 855 J=1,JJ
                WRITE(IOCODE,72) (DY(I,J,K), I=1,II)
855           CONTINUE
          END IF
      END IF
```

```
      IF (NUMDZ .NE. 0) THEN
         WRITE(IOCODE,37)
         DO L=1,NUMDZ
           READ(20,*) I,J,K,DZ(I,J,K)
           WRITE(IOCODE,32) I,J,K,DZ(I,J,K)
         END DO

         IF (IDCODE .EQ. 1) THEN
            WRITE(IOCODE,148)
            DO 857 K=1,KK
              WRITE(IOCODE,38) K
            DO 857 J=1,JJ
              WRITE(IOCODE,72) (DZ(I,J,K), I=1,II)
857         CONTINUE
         END IF
      END IF

C -----------------------------------------------------------------
C        ESTABLISH NODE MIDPOINT ELEVATION (EL).
C        READ INPUT CODE FOR NODE ELEVATION.
C -----------------------------------------------------------------

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KEL

      IF (KEL .NE. 1) THEN
         READ(20,*) ELEV
         DO 910 I=1,II
         DO 910 J=1,JJ
           VAREL(I,J) = ELEV
910      CONTINUE
      ELSE
         DO J=1,JJ
           READ(20,*) (VAREL(I,J), I=1,II)
         END DO
      END IF

      DO 923 I=1,II
      DO 923 J=1,JJ
        ZSUM(I,J) = 0.0
923   CONTINUE

      DO 926 K=1,KK
      DO 926 J=1,JJ
      DO 926 I=1,II
        DEL       = ZSUM(I,J) + 0.5*DZ(I,J,K)
        EL(I,J,K) = VAREL(I,J)+DEL
        ZSUM(I,J) = ZSUM(I,J)+DZ(I,J,K)
926   CONTINUE
      WRITE(IOCODE,390)

      DO 600 K=1,KK
        WRITE(IOCODE,38) K
      DO 600 J=1,JJ
        WRITE(IOCODE,72) (EL(I,J,K), I=1,II)
600   CONTINUE


25    FORMAT(///25X,'NUMBER OF GRID BLOCKS'/25X,21('_')//30X,
     $       'II',T35,':',I4/30X,'JJ'T35,':',I4/30X,'KK',
     $       T35,':',I4///)
26    FORMAT(/25X,'GRID BLOCK DIMENSIONS'/25X,21('_')/)
29    FORMAT(27X,'DXC',T32,':',1X,F8.2,' FT.')
31    FORMAT(/2X,'GRID BLOCK LENGTH (DX) NODE MODIFICATION'/2X,
     $       '          I    J    K          NEW DX VALUE')
32    FORMAT(2X,3I5,5X,F14.4)
33    FORMAT(27X,'DYC',T32,':',1X,F8.2,' FT.')
34    FORMAT(/2X,'GRID BLOCK WIDTH (DY) NODE MODIFICATION'/2X,
     $       '          I    J    K          NEW DY VALUE')
36    FORMAT(27X,'DZC',T32,':',1X,F8.2,' FT.')
37    FORMAT(/2X,'GRID BLOCK DEPTH (DZ) NODE MODIFICATION'/2X,
     $       '          I    J    K          NEW DZ VALUE')
38    FORMAT(/20X,'LAYER (',I2,')'/20X,10('-')/)
43    FORMAT(/5X,'GRID BLOCK LENGTH (DX) DISTRIBUTION'/)
47    FORMAT(/5X,'GRID BLOCK WIDTH (DY) DISTRIBUTION'/)
48    FORMAT(/5X,'GRRID BLOCK DEPTH (DZ) DISTRIBUTION'/)
69    FORMAT(80A1)
70    FORMAT(2X,80A1)
72    FORMAT(25X,20F6.0)
143   FORMAT(/5X,'MODIFIED (DX) DISTRIBUTION'/)
147   FORMAT(/5X,'MODIFIED (DY) DISTRIBUTION'/)
148   FORMAT(/5X,'MODIFIED (DZ) DISTRIBUTION'/)
390   FORMAT(///20X,'NODE MIDPOINT ELEVATION (EL) IN FT.'/20X,
     $       35('_')/)
511   FORMAT(5X,'DXC(',I2,',J,K)',T20,':',F8.2/)
512   FORMAT(5X,'DYC(I,',I2,',K)',T20,':',F8.2/)
513   FORMAT(5X,'DZC(I,J,',I2,')',T20,':',F8.2/)
      RETURN
      END
```

```
C
C ======================================================================

      SUBROUTINE INTERP (NTE,X,Y,N,XO,YO)

C ======================================================================

C     THE SUBROUTINE CALLS FOR LINEAR INTERPOLATION TO CALCULATE
C     YO WHEN XO VALUE IS KNOWN.  THE VALUES OF X,Y ARRAY FOR
C     N NUMBER OF DATA ENTRY IS GIVEN IN TABLE.

C     NOMENCLATURE:

C          N          :  NUMBER OF DATA ENTRY IN TABLE.
C          NTE        :  MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN
C                        ALL THE PDVT TABLES.
C          X          :  VALUES OF X ARRAY
C          XO         :  SPECIFIC VALUE OF X WHERE YO IS TO BE
C                        DETERMINED.
C          Y          :  VALUES OF Y ARRAY.
C          YO         :  CALCULATED VALUE OF Y AT KNOWN XO VALUE.

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

      DIMENSION X(NTE),Y(NTE)
      IF (XO .GE. X(N)) THEN
         YO = Y(N)
      ELSE
         I=2
 10      IF (XO .LT. X(I)) THEN
            YO = Y(I-1)+(XO-X(I-1))*(Y(I)-Y(I-1))/(X(I)-X(I-1))
         ELSE
            I=I+1
            GO TO 10
         END IF
      END IF
      RETURN
      END


C
C ======================================================================

      SUBROUTINE  LTRI(N,A,BI,C,D,X,BETA,GAMMA,W,NPMAX)

C ======================================================================

C     THE SUBROUTINE SOLVES THE TRIDIAGONAL MATRIX USING THOMAS
C     ALGORITHM.

C     NOMENCLATURE :

C          A      : THE LOWER DIAGONAL MATRIX
C          BI     : THE MAIN DIAGONAL MATRIX
C          C      : THE UPPER DIAGONAL MATRIX
C          D      : THE RIGHT HAND SIDE COLUMN VECTOR
C          N      : THE NUMBER OF EQUATIONS TO BE SOLVED
C          NPMAX  : THE MAXIMUM NUMBER OF GRID BLOCKS USED IN
C                   THE COMMON BLOCK FILE AND IS USUALLY TAKEN
C                   AS NX.
C          X      : SOLTION VECTOR
C
C ----------------------------------------------------------------------

      DIMENSION  A(NPMAX),BI(NPMAX),C(NPMAX),D(NPMAX),X(NPMAX),
     $           BETA(NPMAX),GAMMA(NPMAX),W(NPMAX)

      BETA(1)  = BI(1)
      GAMMA(1) = D(1)/BI(1)
      NM       = N-1

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

C        COMPUTE FORWARD SOLUTION.

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

      DO I=1,NM
         W(I)      = C(I)/BETA(I)
         IP        = I+1
         BETA(IP)  = BI(IP)-A(IP)*W(I)
      END DO

      DO I=2,N
         IM        = I-1
         GAMMA(I)  = (D(I)-A(I)*GAMMA(IM))/BETA(I)
      END DO

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

C        COMPUTE BACK SOLUTION.

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

      X(N) = GAMMA(N)
      DO J=1,NM
         I     = N-J
         IP    = I+1
         X(I)  = GAMMA(I)-W(I)*X(IP)
      END DO

      RETURN
      END
```

```
C
C ==========================================================================
      SUBROUTINE  MATBAL (II,JJ,KK,STBW,STBWI,MCFG,MCFGI,MBEW,
     $                    MBEG,DELTO,RESVOL,WP,GP,WI,GI,PAVGO,
     $                    PAVG,N,WPR,GPR,WIR,GIR,D5615,CWP,
     $                    CGP,CWI,CGI,MCFG1,MCFGT,CWGR,WGR)
C ==========================================================================
C
C       THE SUBROUTINE CALCULATES THE PRESENT AND PREVIOUS AVERAGE
C       RESERVOIR PRESSURES WITH TIME-STEP MATERIAL BALANCE ERROR
C       CHECK.
C
C       NOMENCLATURE:
C
C            CGI       : CUMMULATIVE GAS INJECTION, MSCF.
C            CGP       : CUMMULATIVE GAS PRODUCTION, MSCF
C            CWGR      : CUMMULATIVE WATER--GAS PRODUCTION RATIO
C            CWI       : CUMMULATIVE WATER INJECTION, STB
C            CWP       : CUMMULATIVE WATER PRODUCTION, STB
C            DELTO     :  TIME STEP, DAYS.
C            D5615     :  1/5.615
C            FACT      :  D5615*DELTO
C            II        :  NUMBER OF THE X-DIRECTION BLOCKS USED IN THE
C                         MODEL GRID.
C            JJ        :  NUMBER OF THE Y-DIRECTION BLOCKS USED IN THE
C                         MODEL GRID.
C            KK        :  NUMBER OF THE Z-DIRECTION BLOCKS USED IN THE
C                         MODEL GRID.
C            MBEG      :  GAS MATERIAL BALANCE ERROR, PER CENT.
C            MBEW      :  WATER MATERIAL BALANCE ERROR, PER CENT.
C            PAVG      :  CURRENT AVERAGE RESERVOIR PRESSURE, PSIA.
C            PAVGO     :  PREVIOUS AVERAGE RESERVOIR PRESSURE, PSIA.
C ---------------------------------------------------------------------------
      INCLUDE 'COMGWSIM.FOR'

      FACT = D5615*DELTO
      PAVGO = 0.0
      PAVG = 0.0
      WP = 0.0
      GP = 0.0
      WI = 0.0
      GI = 0.0

      DO 100 K=1,KK
      DO 100 J=1,JJ
      DO 100 I=1,II
         PAVGO = PAVGO+PN(I,J,K)*VP(I,J,K)
         PAVG  = PAVG+P(I,J,K)*VP(I,J,K)
         IF (QW(I,J,K) .GT. 0.0) THEN
            WP = WP+QW(I,J,K)*FACT
         ELSE IF (QW(I,J,K) .LT. 0.0) THEN
            WI = WI+QW(I,J,K)*FACT
         ELSE
            CONTINUE
         END IF

         IF (QG(I,J,K) .GT. 0.0) THEN
            GP = GP+QG(I,J,K)*DELTO
         ELSE IF (QG(I,J,K) .LT. 0.0) THEN
            GI = GI+QG(I,J,K)*DELTO
         ELSE
            CONTINUE
         END IF

  100 CONTINUE

      CWP = CWP+WP
      CGP = CGP+GP*0.001
      CWI = CWI+WI
      CGI = CGI+GI*0.001
C ---------------------------------------------------------------------------
C          CONVERT SCF TO MCF.
C ---------------------------------------------------------------------------
      GP = GP*0.001
      GI = GI*0.001
      DIV = 1.0/DELTO
      WPR = WP*DIV
      GPR = GP*DIV
      WIR = WI*DIV
      GIR = GI*DIV
      PAVG = PAVG/RESVOL
      PAVGO = PAVGO/RESVOL

      DENOM2 = STBWI-WP-WI
      IF (DENOM2 .GE. 1.E-7) THEN
         MBEW = ((STBW/(STBWI-WP-WI))-1.0)*100.0
      END IF
```

```
      DENOM3 = MCFGI-GP-GI
      IF (DENOM3 .GE. 1.E-7) THEN
         MBEG = (MCFGT/(MCFGI-GP-GI)-1.0)*100.0
      END IF

      IF (WP .NE. 0.0) THEN
         WGR = WP/(GP*1000.0)
      ELSE
         WGR = 0.0
      END IF

      IF (CWP .NE. 0.0) THEN
         CWGR = CWP/(CGP*1000.0)
      ELSE
         CWGR = 0.0
      END IF

      GP = GP*0.001
      GI = GI*0.001

      RETURN
      END

C
C =====================================================================

      SUBROUTINE  NODES (NVQN,WRAD,SKIN)

C =====================================================================

C     THE SUBROUTINE CALLS FOR TOTAL NUMBER OF WELLS FOR WHICH
C     WELL INFORMATION IS TO BE READ.  THE WELL INFORMATION
C     INCLUDES WELL NAME,(X,Y) COORDINATES OF GRID BLOCKS CONT-
C     AINING THE WELLLS, CODE FOR SPECIFYING WELL TYPE, ETC.

C     NOMENCLATURE:

C          IHEDIN    :  STORAGE MEMORY FOR INPUT TITLE CARD.
C          IQN1      :  X-COORDINATE OF GRID BLOCK CONTAINING THE
C                       WELL FOR WHICH WELL INFORMATION IS TO BE
C                       READ.
C          IQN2      :  Y-COORDINATE OF GRID BLOCK CONTAINING THE
C                       WELL FOR WHICH WELL INFORMATION IS TO BE
C                       READ.
C          IQN3      :  LAYER NUMBER OF THE UPPERMOST COMPLETION
C                       LAYER FOR THIS WELL.
C          KIP       :  CODE FOR SPECIFYING BOTH WELL TYPE AND
C                       WHETHER THE WELL'S PRODUCTION (INJECTION)
C                       PERFORMANCE IS DETERMINED BY SPECIFYING
C                       RATES OR BY SPECIFYING FLOWING BOTTOMHOLE
C                       PRESSURE AND ALSO WHETHER AN EXPLICIT OR
C                       IMPLICIT PRESSURE CALCULATION IS TO BE
C                       MADE.
C          LAYER     :  TOTAL NUMBER OF CONSECUTIVE COMPLETION
C                       LAYERS, STARTING WITH AND INCLUDING IQN3.
C          NVQN      :  TOTAL NUMBER OF WELLS FOR WHICH WELL
C                       INFORMATION IS TO BE READ.
C          PID       :  LAYER FLOW INDEX.
C          PWF       :  FLOWING BOTTOMHOLE PRESSURE, PSIA.
C          QVW       :  WATER RATE, STB/D.
C                       NON-ZERO ONLY IF KIP = 2.
C          QVG       :  GAS RATE, MCF/D.
C                       NON-ZERO ONLY IF KIP = 3.
C          QVT       :  TOTAL FLUID RATE, STB/D.
C                       NON-ZERO ONLY IF KIP = 1 .
C          SKIN      :  LAYER SKIN FACTOR.
C          WRAD      :  WELL RADIUS, FT.

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

      INCLUDE 'COMGWSIM.FOR'

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

C        ESTABLISH RATE-SPECIFIED AND PRESSURE-SPECIFIED WELLS.

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

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) NVQN

      IF (NVQN .NE. 0) THEN
         WRITE(IOCODE,67)
         DO J=1,NVQN
            READ(20,*) WELLID(J), IQN1(J), IQN2(J), IQN3(J),
     $                 LAYER(J), KIP(J), QVW(J), QVG(J), QVT(J)
            IQ1 = IQN1(J)
            IQ2 = IQN2(J)
            IQ3 = IQN3(J)
            LAY = IQ3+LAYER(J)-1
```

```
            DO K=IQ3,LAY
               READ(20,*) WRAD, SKIN, PWF(J,K)
               IF (SKIN .GE. 500.) THEN
                  PID(J,K) = 0.0
               ELSE
                  IF (DX(IQ1,IQ2,K) .EQ. DY(IQ1,IQ2,K)) THEN
                     DENOM = ALOG(+0.121*SQRT(DX(IQ1,IQ2,K)*
     $                        DY(IQ1,IQ2,K))/WRAD) + SKIN
                  ELSE
                     DENOM = ALOG(+0.0849*SQRT(DX(IQ1,IQ2,K)*
     $                        DX(IQ1,IQ2,K)+DY(IQ1,IQ2,K)*
     $                        DY(IQ1,IQ2,K))/WRAD) + SKIN
                  END IF

                  PID(J,K) = 0.00708*KX(IQ1,IQ2,K)*DZ(IQ1,IQ2,K)
     $                        / DENOM
               END IF

               WRITE(IOCODE,70) IQN1(J),IQN2(J),K,QVW(J),QVG(J),
     $                          QVT(J),PWF(J,K),PID(J,K),WRAD,SKIN
            END DO
         END DO

         WRITE(IOCODE,33)
         DO J=1,NVQN
            IQ3 = IQN3(J)
            LAY = IQ3+LAYER(J)-1

            DO K=IQ3,LAY
               IF (KIP(J) .EQ. 1) THEN
                  WRITE(IOCODE,95) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. 2) THEN
                  WRITE(IOCODE,96) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. 3) THEN
                  WRITE(IOCODE,97) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. -1) THEN
                  WRITE(IOCODE,105) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. -2) THEN
                  WRITE(IOCODE,106) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. -3) THEN
                  WRITE(IOCODE,107) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. -11) THEN
                  WRITE(IOCODE,115) IQN1(J),IQN2(J),K
               ELSE IF (KIP(J) .EQ. -12) THEN
                  WRITE(IOCODE,116) IQN1(J),IQN2(J),K
               ELSE
                  WRITE(IOCODE,117) IQN1(J),IQN2(J),K
               END IF
            END DO
         END DO
      END IF


33    FORMAT(/)
67    FORMAT(4(//),20X,'RESERVOIR CONTAINS THE FOLLOWING RATE',
     $       ' NODES'/20X,43('-')//16X,
     $       'WATER',7X,'GAS',8X,'TOTAL',4X,'BHFP',13X,
     $       'WRAD',4X,'SKIN'/8X,'NODE',3X,'(STBD)',6X,
     $       '(MCFD)',6X,'(RBD)',3X,'(PSIA)',4X,'PID',5X,
     $       '(FT)',3X,'FACTOR'/8X,'----',3X,'------',6X,
     $       '------',6X,'-----',3X,'------',4X,'---',5X,
     $       '----',3X,'------'/)
69    FORMAT(80A1)
70    FORMAT(4X,3I3,F8.1,F12.1,F11.1,F9.1,F8.2,F6.3,F9.3)
95    FORMAT(5X,'BLOCK',3I3,'  CONTAINS A RATE SPECIFIED ',
     $       'PRODUCING WELL.')
96    FORMAT(5X,'BLOCK',3I3,'  CONTAINS A RATE SPECIFIED ',
     $       'WATER INJECTION WELL.')
97    FORMAT(5X,'BLOCK',3I3,'  CONTAINS A RATE SPECIFIED ',
     $       'GAS INJECTION WELL.')
105   FORMAT(5X,'BLOCK',3I3,'  CONTAINS AN EXPLICIT PRESSURE',
     $       ' SPECIFIED PRODUCING WELL.')
106   FORMAT(5X,'BLOCK',3I3,'  CONTAINS AN EXPLICIT PRESSURE',
     $       ' SPECIFIED WATER INJECTION WELL.')
107   FORMAT(5X,'BLOCK',3I3,'  CONTAINS AN EXPLICIT PRESSURE',
     $       ' SPECIFIED GAS INJECTION WELL.')
115   FORMAT(5X,'BLOCK',3I3,'  CONTAINS AN IMPLICIT PRESSURE',
     $       ' SPECIFIED PRODUCING WELL.')
116   FORMAT(5X,'BLOCK',3I3,'  CONTAINS AN IMPLICIT PRESSURE',
     $       ' SPECIFIED WATER INJECTION WELL.')
117   FORMAT(5X,'BLOCK',3I3,'  CONTAINS AN IMLICIT PRESSURE',
     $       ' SPECIFIED GAS INJECTION WELL.')
      RETURN
      END
```

```
C
C
C ==========================================================================

      SUBROUTINE NTERP1 (NTE,X,Y,N,XO,YO)

C ==========================================================================

C     THE SUBROUTINE CALLS FOR LINEAR INTERPOLATION TO CALCULATE
C     YO WHEN XO VALUE IS KNOWN.  THE VALUES OF X,Y ARRAY FOR
C     N NUMBER OF DATA ENTRY IS GIVEN IN TABLE.

C     NOMENCLATURE:
C          N         :  NUMBER OF DATA ENTRY IN TABLE.
C          NTE       :  MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN
C                       ALL PVT TABLES.
C          X         :  VALUES OF X ARRAY.
C          XO        :  SPECIFIC VALUE OF X WHERE YO IS TO BE
C                       DETERMINED.
C          Y         :  VALUES OF Y ARRAY.
C          YO        :  CALCULATED VALUE OF Y AT KNOWN XO VALUE.

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

      DIMENSION X(NTE),Y(NTE)

      IF (XO .GE. X(N)) THEN
         YO = Y(N)
      ELSE
         I = 2
 10      IF (XO .LT. X(I)) THEN
            YO = Y(I)
         ELSE
            I=I+1
            GO TO 10
         END IF
      END IF

      RETURN
      END
```

```
C
C ======================================================================
      SUBROUTINE PARM1 (II,JJ,KK)
C ======================================================================
C     THE SUBROUTINE ESTABLISHES THE POROSITY AND PERMABILITY
C     DISTRIBUTIONS IN THE GRID MODEL USING THE CODES FOR COTRO-
C     LLING POROSITY AND PERMEABILITY DATA.  THE PROGRAM COULD
C     MODIFY TO POROSITY AND PERMEABILITY DISTRIBUTIONS IF NECE-
C     SSARY.
C     NOMENCLATURE:
C        IPCODE      :  CONTROL CODE FOR PRINTING MODIFIED POROS-
C                       ITY AND PERMEABILITY DISTRIBUTIONS.
C                       IPCODE = 0 MEANS 'DO NOT PRINT', AND
C                       IPCODE = 1 MEANS 'PRINT'.
C        IHEDIN(L)   :  STORAGE FOR THE TITLE CARD.
C        I           :  X-COORDINATE OF BLOCK TO BE MODIFIED.
C        II          :  NUMBER OF X-DIRECTION BLOCKS USED IN THE
C                       MODEL GRID.
C        J           :  Y-COORDINATE OF BLOCK TO BE MODIFIED.
C        JJ          :  NUMBER OF Y-DIRECTION BLOCKS USED IN THE
C                       MODEL GRID.
C        K           :  Z-COORDINATE OF BLOCK TO BE MODIFIED.
C        KK          :  NUMBER OF Z-DIRECTION BLOCKS USED IN THE
C                       MODEL GRID.
C        KPH         :  CONTROL CODE FOR POROSITY DATA INPUT.
C        KKX         :  CONTROL CODE FOR X-DIRECTION PERMEABILITY
C                       DATA INPUT.
C        KKY         :  CONTROL CODE FOR Y-DIRECTION PERMEABILITY
C                       DATA INPUT.
C        KKZ         :  CONTROL CODE FOR Z-DIRECTION PERMEABILITY
C                       DATA INPUT.
C        KXC         :  A SINGLE CONSTANT VALUE OF X-DIRECTION
C                       PERMEABILITY IN MILLIDARCIES (MD).
C                       KXC IS READ AND ASSIGNED TO ALL BLOCKS IN
C                       THE MODEL GRID IF KKX = -1.
C        KYC         :  A SINGLE CONSTANT VALUE OF Y-DIRECTION
C                       PERMEABILITY IN MILLIDARCIES (MD).
C                       KYC IS READ AND ASSIGNED TO ALL BLOCKS IN
C                       THE MODEL GRID IF KKY = -1.
C        KZC         :  A SINGLE CONSTANT VALUE OF Z-DIRECTION
C                       PERMEABILITY IN MILLIDARCIES (MD).
C                       KZC IS READ AND ASSIGNED TO ALL BLOCKS IN
C                       THE MODEL GRID IF KKZ = -1.
C        KX(I,J,K)   :  VARIABLE VALUE OF X-DIRECTION PERMEABIL-
C                       ITY IN MD.
C                       A SEPARATE VALUE OF X-DIRECTION PERMEABI-
C                       LITY FOR II*JJ*KK BLOCKS MUST BE READ IF
C                       KKX = +1.
C        KY(I,J,K)   :  VARIABLE VALUE OF Y-DIRECTION PERMEABIL-
C                       ITY IN MD.
C                       A SEPARATE VALUE OF Y-DIRECTION PERMEABI-
C                       LITY FOR II*JJ*KK BLOCKS MUST BE READ IF
C                       KKY = +1.
C        KZ(I,J,K)   :  VARIABLE VALUE OF Z-DIRECTION PERMEABIL-
C                       ITY IN MD.
C                       A SEPARATE VALUE OF Z-DIRECTION PERMEABI-
C                       LITY FO II*JJ*KK BLOCKS MUST BE READ IF
C                       KKZ = +1.
C        NUMP        :  NUMBER OF GRID BLOCKS WHERE POROSITY
C                       VALUE IS TO BE CHANGED.
C        NUMKX       :  NUMBER OF GRID BLOCKS WHERE KX(I,J,K)
C                       VALUE IS TO BE CHANGED.
C        NUMKY       :  NUMBER OF GRID BLOCKS WHERE KY(I,J,K)
C                       VALUE IS TO BE CHANGED.
C        NUMKZ       :  NUMBER OF GRID BLOCKS WHERE KZ(I,J,K)
C                       VLUE IS TO BE CHANGED.
C        NX          :  MAXIMUM NUMBER OF X-DIRECTION BLOCKS
C                       ALLOWED TO USE IN THE MODEL GRID.
C        NY          :  MAXIMUM NUMBER OF Y-DIRECTION BLOCKS
C                       ALLOWED TO USE IN THE MODEL GRID.
C        NZ          :  MAXIMUM NUMBER OF Z-DIRECTION BLOCKS
C                       ALLOWED TO USE IN THE MODEL GRID.
C        PHIC        :  A SINGLE CONSTANT VALUE OF POROSITY IN
C                       FRACTION.
C                       PHIC IS READ AND ASSIGNED TO ALL BLOCKS IN
C                       THE MODEL GRID IF KPH = -1.
C        RPHL(K)     :  DIFFERENT BUT CONSTANT POROSITY VALUES FOR
C                       INDIVIDUAL LAYERS.
C                       RPHL VALUES FOR K = 1 TO KK LAYERS ARE
C                       READ IF KPH = 0.
C        RKXL(K)     :  DIFFERENT BUT CONSTANT, X-DIRECTION PER-
C                       MEABILITY VALUES FOR INDIVIDUAL LAYERS.
C                       RKXL VALUES FOR K = 1 TO KK LAYERS ARE
C                       READ IF KKX = 0.
C        RKYL(K)     :  DIFFERENT BUT CONSTANT, Y-DIRECTION PER-
C                       MEABILITY VALUES FOR INDIVIDUAL LAYERS.
C                       RKYL VALUES FOR K = 1 TO KK LAYERS ARE
C                       READ IF KKY = 0.
C        RKZL(K)     :  DIFFERENT BUT CONSTANT, Z-DIRECTION PER-
C                       MEABILITY VALUES FOR INDIVIDUAL LAYERS.
C                       RKZL VALUES FOR K = 1 TO KK LAYERS ARE
C                       READ IF KKZ = 0.
C        VP(I,J,K)   :  VARIABLE POROSITY VALUES IN THE MODEL
C                       GRID.
C                       A SEPARATE VALUE OF VP IS READ FOR EACH
C                       BLOCK IN THE MODEL GRID IF KPH = +1.
```

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

      INCLUDE 'COMGWSIM.FOR'
      DIMENSION   RPHL(NZ), RKXL(NZ), RKYL(NZ), RKZL(NZ)
      REAL  KXC,KYC,KZC

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

C         READ INPUT CODES FOR POROSITY AND PERMEABILITY
C         DISTRIBUTIONS.

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

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KPH,KKX,KKY,KKZ

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

C         ESTABLISH POROSITY (VP) DISTRIBUTION.

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

      WRITE(IOCODE,25)
      IF (KPH .LT. 0) THEN
         READ(20,*) PHIC
         DO 140 K=1,KK
         DO 140 J=1,JJ
         DO 140 I=1,II
           VP(I,J,K) = PHIC
140      CONTINUE
         WRITE(IOCODE,26) PHIC

      ELSE IF (KPH .EQ. 0) THEN
         READ(20,*) (RPHL(K), K=1,KK)
         DO 550 K=1,KK
         DO 550 J=1,JJ
         DO 550 I=1,II
           VP(I,J,K) = RPHL(K)
550      CONTINUE
         DO K=1,KK
         WRITE(IOCODE,510) K,RPHL(K)
         END DO

      ELSE
         DO 160 K=1,KK
           WRITE(IOCODE,38) K
         DO 160 J=1,JJ
           READ(20,*) (VP(I,J,K), I=1,II)
           WRITE(IOCODE,73) (VP(I,J,K), I=1,II)
160      CONTINUE
      END IF

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

C         ESTABLISH X-DIRECTION PERMEABILITY (KX) DISTRIBUTION.

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

      WRITE(IOCODE,28)
      IF (KKX .LT. 0) THEN
         READ(20,*) KXC
         DO 175 K=1,KK
         DO 175 J=1,JJ
         DO 175 I=1,II
           KX(I,J,K) = KXC
175      CONTINUE
         WRITE(IOCODE,29) KXC

      ELSE IF (KKX .EQ. 0) THEN
         READ(20,*) (RKXL(K), K=1,KK)
         DO 187 K=1,KK
         DO 187 J=1,JJ
         DO 187 I=1,II
           KX(I,J,K) = RKXL(K)
187      CONTINUE
         DO K=1,KK
           WRITE(IOCODE,511) K,RKXL(K)
         END DO

      ELSE
         WRITE(IOCODE,43)
         DO 192 K=1,KK
           WRITE(IOCODE,38) K
         DO 192 J=1,JJ
           READ(20,*) (KX(I,J,K), I=1,II)
           WRITE(IOCODE,72) (KX(I,J,K), I=1,II)
192      CONTINUE
      END IF

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

C         ESTABLISH Y-DIRECTION PERMEABILITY (KY) DISTRIBUTION.

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

```
      IF (KKY .LT. 0) THEN
         READ(20,*) KYC
         DO 202 K=1,KK
         DO 202 J=1,JJ
         DO 202 I=1,II
           KY(I,J,K) = KYC
202      CONTINUE
         WRITE(IOCODE,33) KYC

      ELSE IF (KKY .EQ. 0) THEN
         READ(20,*) (RKYL(K), K=1,KK)
         DO 205 K=1,KK
         DO 205 J=1,JJ
         DO 205 I=1,II
           KY(I,J,K) = RKYL(K)
205      CONTINUE
         DO K=1,KK
           WRITE(IOCODE,512) K,RKYL(K)
         END DO

      ELSE
         WRITE(IOCODE,47)
         DO 212 K=1,KK
           WRITE(IOCODE,38) K
         DO 212 J=1,JJ
           READ(20,*) (KY(I,J,K), I=1,II)
           WRITE(IOCODE,72) (KY(I,J,K), I=1,II)
212      CONTINUE
      END IF

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

C        ESTABLISH Z-DIRECTION PERMEABILITY (KZ) DISTRIBUTION.

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

      IF (KKZ .LT. 0) THEN
         READ(20,*) KZC
         DO 230 K=1,KK
         DO 230 J=1,JJ
         DO 230 I=1,II
           KZ(I,J,K) = KZC
230      CONTINUE
         WRITE(IOCODE,36) KZC

      ELSE IF (KKZ .EQ. 0) THEN
         READ(20,*) (RKZL(K), K=1,KK)
         DO 235 K=1,KK
         DO 235 J=1,JJ
         DO 235 I=1,II
           KZ(I,J,K) = RKZL(K)
235      CONTINUE
         DO K=1,KK
           WRITE(IOCODE,513) K,RKZL(K)
         END DO

      ELSE
         WRITE(IOCODE,48)
         DO 240 K=1,KK
           WRITE(IOCODE,38) K
         DO 240 J=1,JJ
           READ(20,*) (KZ(I,J,K), I=1,II)
           WRITE(IOCODE,72) (KZ(I,J,K), I=1,II)
240      CONTINUE
      END IF

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

C        ESTABLISH POROSITY AND PERMEABILITY MODIFICATIONS.

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

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) NUMP, NUMKX, NUMKY, NUMKZ, IPCODE

      IF (NUMP .NE. 0) THEN
         WRITE(IOCODE,27)
         DO L=1,NUMP
           READ(20,*) I,J,K,VP(I,J,K)
           WRITE(IOCODE,32) I,J,K,VP(I,J,K)
         END DO
         IF (IPCODE .EQ. 1) THEN
            WRITE(IOCODE,125)
            DO 851 K=1,KK
              WRITE(IOCODE,38) K
            DO 851 J=1,JJ
              WRITE(IOCODE,73) (VP(I,J,K), I=1,II)
851         CONTINUE
         END IF
      END IF
```

```
      IF (NUMKX .NE. 0) THEN
         WRITE(IOCODE,31)
         DO L=1,NUMKX
           READ(20,*) I,J,K,KX(I,J,K)
           WRITE(IOCODE,32) I,J,K,KX(I,J,K)
         END DO
         IF (IPCODE .EQ. 1) THEN
            WRITE(IOCODE,143)
            DO 853 K=1,KK
              WRITE(IOCODE,38) K
            DO 853 J=1,JJ
              WRITE(IOCODE,72) (KX(I,J,K), I=1,II)
853         CONTINUE
         END IF
      END IF

      IF (NUMKY .NE. 0) THEN
         WRITE(IOCODE,34)
         DO L=1,NUMKY
           READ(20,*) I,J,K,KY(I,J,K)
           WRITE(IOCODE,32) I,J,K,KY(I,J,K)
         END DO
         IF (IPCODE .EQ. 1) THEN
            WRITE(IOCODE,147)
            DO 855 K=1,KK
              WRITE(IOCODE,38) K
            DO 855 J=1,JJ
              WRITE(IOCODE,72) (KY(I,J,K), I=1,II)
855         CONTINUE
         END IF
      END IF

      IF (NUMKZ .NE. 0) THEN
         WRITE(IOCODE,37)
         DO L=1,NUMKZ
           READ(20,*) I,J,K,KZ(I,J,K)
           WRITE(IOCODE,32) I,J,K,KZ(I,J,K)
         END DO
         IF (IPCODE .EQ. 1) THEN
            WRITE(IOCODE,148)
            DO 857 K=1,KK
              WRITE(IOCODE,38) K
            DO 857 J=1,JJ
              WRITE(IOCODE,72) (KZ(I,J,K), I=1,II)
857         CONTINUE
         END IF
      END IF


   25 FORMAT(////25X,'POROSITY VALUES'/25X,15('_')//)
   26 FORMAT(27X,'PHIC',T33,':',F8.4//)
   27 FORMAT(/2X,'POROSITY NODE MODIFICATION'/2X,
     $       '      I    J    K          NEW PHI VALUE')
   28 FORMAT(//25X,'PERMEABILITY VALUES'/25X,19('_')//)
   29 FORMAT(27X,'KXC',T32,':',F8.2,1X,'MDS.')
   31 FORMAT(/2X,'PERMEABILITY (KX) NODE MODIFICATION'/2X,
     $       '      I    J    K          NEW KX VALUE')
   32 FORMAT(2X,3I5,5X,F14.4)
   33 FORMAT(27X,'KYC',T32,':',F8.2,1X,'MDS.')
   34 FORMAT(/2X,'PERMEABILITY (KY) NODE MODIFICATION'/2X,
     $       '      I    J    K          NEW KY VALUE')
   36 FORMAT(27X,'KZC',T32,':',F8.2,1X,'MDS.'//)
   37 FORMAT(/2X,'PERMEABILITY (KZ) NODE MODIFICATION'/2X,
     $       '      I    J    K          NEW KZ VALUE')
   38 FORMAT(/2X,'LAYER(',I2,')'/)
   43 FORMAT(/5X,'X-DIRECTION PERMEABILITY (KX) DISTRIBUTION'/)
   47 FORMAT(/5X,'Y-DIRECTION PERMEABILITY (KY) DISTRIBUTION'/)
   48 FORMAT(/5X,'Z-DIRECTION PERMEABILITY (KZ) DISTRIBUTION'/)
   69 FORMAT(80A1)
   72 FORMAT(2X,20F6.2)
   73 FORMAT(2X,20F6.4)
  125 FORMAT(/5X,'MODIFIED POROSITY DISTRIBUTION'/)
  143 FORMAT(/5X,'MODIFIED PERMEABILITY (KX) DISTRIBUTION'/)
  147 FORMAT(/5X,'MODIFIED PERMEABILITY (KY) DISTRIBUTION'/)
  148 FORMAT(/5X,'MODIFIED PERMEABILITY (KZ) DISTRIBUTION'/)
  510 FORMAT(5X,'VP(I,J,',I2,')',T20,':',1X,F8.5/)
  511 FORMAT(5X,'KX(I,J,',I2,')',T20,':',1X,F8.2/)
  512 FORMAT(5X,'KY(I,J,',I2,')',T20,':',1X,F8.2/)
  513 FORMAT(5X,'KZ(I,J,',I2,')',T20,':',1X,F8.2/)

      RETURN
      END
```

```
 31   FORMAT(2X,'TRANSMISSIBILITY VALUES :'/2X,25('_')//5X,
     $       'TRANSMISSIBILITY (TX) NODE MODIFICATION'/5X,
     $       '          I     J     K            NEW TX VALUE')
 32   FORMAT(2X,3I5,5X,F14.4)
 34   FORMAT(5X,'TRANSMISSIBILITY (TY) NODE MODIFICATION'/5X,
     $       '          I     J     K            NEW TY VALUE')
 37   FORMAT(5X,'TRANSMISSIBILITY (TZ) NODE MODIFICATION'/5X,
     $       '          I     J     K            NEW TZ VALUE')
 38   FORMAT(2X,'LAYER(',I2,')'/)
 41   FORMAT(2X,3I5,3F15.3)
 43   FORMAT(2X,'     I     J     K        A1              A2',
     $       '              A3'/)
 44   FORMAT(5X,'MODIFIED TRANSMISSIBILITY (TX) DISTRIBUTION'/)
 47   FORMAT(5X,'MODIFIED TRANSMISSIBILITY (TY) DISTRIBUTION'/)
 48   FORMAT(5X,'MODIFIED TRANSMISSIBILITY (TZ) DISTRIBUTION'/)
 49   FORMAT(2X,'     I     J     K        TX              TY',
     $       '              TZ'/)
 69   FORMAT(80A1)
 72   FORMAT(2X,20F6.1)

      RETURN
      END

C
C ================================================================
      SUBROUTINE UINIT1 (KPI,II,JJ,KK,CUMPW,MBEW,CUMPG,MBEG,
     $                   SWC,GWC,PGWC,CUMIW,CUMIG)
C ================================================================
C
C     THE SUBROUTINE CALLS FOR TWO OPTIONS OF PRESSURE AND SAT-
C     URATION INITIALIZATION. THE INITIAL PRESSURE DISTRIBUTION
C     CAN EITHER BE CALCULATED BY THE PROGRAM FOR EQUILIBRIUM
C     CONDITIONS GIVEN THE LOCATION OF THE GAS/WATER CONTACT AND
C     THE PRESSURE AT THE CONTACT, OR THE INITIAL PRESSURE
C     DISTRIBUTION CAN BE READ ON A BLOCK-BY-BLOCK BASIS. THE
C     SATURATIONS (SW,SG) CAN EITHER BE READ AS CONSTANT VALUES
C     OVER THE ENTIRE GRID, OR THE ENTIRE SW AND SG DISTRIBUTIONS
C     ARE READ ON A BLOCK-BY-BLOCK BASIS.
C
C     NOMENCLATURE:
C
C        BBG ...       :  GAS FORMATION VOLUME FACTOR AT PRESSURE,
C                         PGWC IN RCF/SCF.
C        BBW           :  WATER FORMATION VOLUME FACTOR AT PRESSURE,
C                         PGWC IN RB/STB.
C        BGT           :  VARIABLE GAS FORMATION VOLUME FACTOR AT
C                         PRESSURE, PGT IN PVT DATA TABLE.
C        BWT           :  VARIABLE WATER FORMATION VOLUME FACTOR AT
C                         PRESSURE, PWT IN PVT DATA TABLE.
C        CUMIW         :  CUMMULATIVE WATER INJECTION, STB.
C        CUMIG         :  CUMMULATIVE GAS INJECTION, MSCF.
C        CUMPW         :  CUMMULATIVE WATER PRODUCTION, STB.
C        CUMPG         :  CUMMULATIVE GAS PRODUCTION, MSCF.
C        EL(I,J,K)     :  VARIABLE NODE MIDPOINT ELEVATION (FT) OF
C                         ALL BLOCKS IN THE MODEL GRID.
C        GWC           :  GAS/WATER CONTACT, FT.
C        IHEDIN(L)     :  STORAGE MEMORY FOR INPUT TITLE CARD.
C        II            :  NUMBER OF BLOCKS IN THE X-DIRECTION IN
C                         THE MODEL GRID.
C        JJ            :  NUMBER OF BLOCKS IN THE Y-DIRECTION IN
C                         THE MODEL GRID.
C        KK            :  NUMBER OF BLOCKS IN THE Z-DIRECTION IN
C                         THE MODEL GRID.
C        KPI           :  PRESSURE INITIALIZATION CODE.
C                         KPI = 0 MEANS 'USE EQUILIBRIUM PRESSURE
C                         INITIALIZATION'. INPUT REQUIRED WILL BE
C                         PGWC, PRESSURE AT GAS/WATER CONTACT.
C                         KPI = 1 MEANS 'USE NON-EQUILIBRIUM PRESS-
C                         URE INITIALIZATION'. PRESSURES PN(I,J,K)
C                         FOR EACH BLOCK MUST BE READ ON A BLOCK-
C                         BY-BLOCK BASIS. PN(I,J,K) FOR II*JJ*KK
C                         VALUES MUST BE READ.
C        KSI           :  SATURATION INITIALIZATION CODE.
C                         KSI = 0 MEANS 'INITIAL WATER AND GAS
C                         SATURATIONS ARE CONSTANT OVER THE ENTIRE
C                         MODEL GRID'. SWC MUST BE READ.
C                         KSI = 1 MEANS 'WATER SATURATION MUST BE
C                         READ FOR EACH GRID BLOCK ON A BLOCK-BY-
C                         BLOCK BASIS'.
C        MBEG          :  GAS MATERIAL BALANCE, PER CENT.
C        MBEW          :  WATER MATERIAL BALANCE, PER CENT.
C        MPGT          :  MAXIMUM NUMBER OF ENTRY VALUE FOR GAS
C                         PHASE PRESSURE, PSIA.
C        MPWT          :  MAXIMUM NUMBER OF ENTRY VALUE FOR WATER
C                         PHASE PRESSURE, PSIA.
C        NTE           :  MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN
C                         ALL PVT TABLES.
C        PGWC          :  PRESSURE AT GAS/WATER CONTACT, PSIA.
C        PGT           :  VARIABLE GAS PHASE PRESSURE IN PVT
C                         TABLE, PSIA.
C        PWT           :  VARIABLE WATER PHASE PRESSURE IN PVT
C                         TABLE, PSIA.
C        RHOG          :  GAS DENSITY AT PRESSURE P(I,J,K).
C        RHOSCG        :  GAS DENSITY AT STANDARD CONDITION.
C        RHOW          :  WATER DENSITY AT PRESSURE P(I,J,K).
C        RHOSCW        :  WATER DENSITY AT STANDARD CONDITION.
C        RSW           :  GAS IN WATER SOLUTION RATIO AT PRESSURE,
C                         PGWC.
C        RSWT          :  GAS IN WATER SOLUTION RATIO AT PGT IN ALL
C                         PVT TABLES.
C        SG(I,J,K)     :  GAS SATURATION ARRAY.
C        SGN(I,J,K)    :  INITIAL GAS PHASE SATURATION ARRAY.
C        SW(I,J,K)     :  WATER SATURATION ARRAY.
C        SWN(I,J,K)    :  INITIAL WATER PHASE SATURATION ARRAY.
C        SWC           :  CONNATE WATER SATURATION.
C
C ----------------------------------------------------------------
```

```
C
C ==========================================================================

      SUBROUTINE  PRATEI(NVQN)

C ==========================================================================

C     THE SUBROUTINE MODIFIES THE MATRIX ELEMENTS (B) AND (E)
C     FOR THE WELLS BEFORE IMPLICIT PRESSURE CALCULATION.

C     NOMENCLATURE:

C          B(I,J,K): QQWG(I,J,K)-VP(I,J,K)*P(I,J,K)*CT(I,J,K)
C                    /DELT
C          BBG     : GAS FORMATION VOLUME FACTOR AT PRESSURE PPN
C          BBW     : WATER FORMATION VOLUME FACTOR AT PRESSURE PPN
C          BGT     : VARIABLE GAS FORMATION VOLUME FACTOR AT
C                    PRESSURE (PGT) IN PVT DATA TABLE
C          BWT     : VARIABLE WATER FORMATION VOLUME FACTOR AT
C                    PRESSURE (PWT) IN PVT DATA TABLE
C          E(I,J,K): -SUM(I,J,K)-GAM(I,J,K)
C          GMG     : KRG/MUG
C          GMW     : KRW/MUW
C          IQN1    : X-COORDINATE OF GRID BLOCK CONTAINING THE WELL
C                    UNDER CONSIDERATION
C          IQN2    : Y-COORDINATE OF GRID BLOCK CONTAINING THE WELL
C                    UNDER CONSIDERATION
C          IQN3    : LAYER NUMBER OF THE UPPERMOST COMPLETION
C                    LAYER FOR THE WELL UNDER CONSIDERATION
C          KIP     : CODE FOR SPECIFYING BOTH WELL TYPE AND WHETHER
C                    THE WELL'S PRODUCTION (INJECTION) PERFORMANCE
C                    IS DETERMINED BY SPECIFYING RATES OR BY
C                    SPECIFYING FLOWING BOTTOM-HOLE PRESSURE AND
C                    ALSO WHETHER AN EXPLICIT OR IMPLICIT PRESSURE
C                    CALCULATION IS TO BE MADE.
C          LAYER   : TOTAL NUMBER OF CONSECUTIVE COMPLETION LAYERS
C                    STARTING WITH AND INCLUDING IQN3
C          MPGT    : MAXIMUM NUMBER OF ENTRY VALUE FOR GAS PHASE
C                    PRESSURE IN PVT DATA TABLE, PSIA
C          MPWT    : MAXIMUM NUMBER OF ENTRY VALUE FOR WATER PHASE
C                    PRESSURE IN PVT DATA TABLE, PSIA
C          NVQN    : NUMBER OF WELLS FOR WHIH WELL INFORMATION IS
C                    TO BE READ
C          NTE     : MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN ALL
C                    PVT DATA TABLES
C          PID     : LAYER FLOW INDEX
C          PPN     : P(IQ1,IQ2,K)
C          PWT     : WATER PHASE PRESSURE, PSIA
C          RSW     : GAS IN WATER SOLUTION RATIO, SCF/STB

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

      INCLUDE 'COMGWSIM.FOR'

      DO J = 1,NVQN
         IF (KIP(J) .LT. -10) THEN
            IQ1 = IQN1(J)
            IQ2 = IQN2(J)
            IQ3 = IQN3(J)
            LAY = IQ3+LAYER(J)-1

            DO K = IQ3,LAY
               PPN = PN(IQ1,IQ2,K)
               CALL INTERP(NTE,PWT,BWT,MPWT,PPN,BBW)
               CALL INTERP(NTE,PGT,BGT,MPGT,PPN,BBG)
               CALL INTERP(NTE,PWT,RSWT,MPWT,PPN,RSW)
               CPIW = GMW(J,K)*PID(J,K)*5.615*(BBW-BBG*RSW)/BBW
               CPIG = GMG(J,K)*PID(J,K)*5.615
               CPI  = CPIW+CPIG
               B(IQ1,IQ2,K) = B(IQ1,IQ2,K)-CPI*PWF(J,K)
               E(IQ1,IQ2,K) = E(IQ1,IQ2,K)-CPI
            END DO
         END IF
      END DO

      RETURN
      END
```

```
C
C ==========================================================================
      SUBROUTINE  PRATEO(NVQN)
C ==========================================================================
C     THIS SUBROUTINE CALCULATES THE WELL'S PRODUCTION (INJEC-
C     TION) RATE WHICH IS UNDER PID AND FBHP CONTROL.
C --------------------------------------------------------------------------
      INCLUDE 'COMGWSIM.FOR'

      DO J=1,NVQN
         IF (KIP(J) .LT. -10) THEN
            IQ1 = IQN1(J)
            IQ2 = IQN2(J)
            IQ3 = IQN3(J)
            LAY = IQ3+LAYER(J)-1

            DO K=IQ3,LAY
               PP = P(IQ1,IQ2,K)
               PPN = PN(IQ1,IQ2,K)
               CALL INTERP(NTE,PWT,RSWT,MPWT,PPN,RSWN)
               CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
               RSWAV = 0.5*(RSW+RSWN)
               FACTOR = PID(J,K)*5.615*(PP-PWF(J,K))
               IF (KIP(J) .EQ. -13) THEN
                  QG(IQ1,IQ2,K) = (GMW(J,K)+GMG(J,K))/
     $                            BG(IQ1,IQ2,K)*FACTOR
               ELSE IF (KIP(J) .EQ. -12) THEN
                  QW(IQ1,IQ2,K) = (GMW(J,K)+GMG(J,K))/
     $                            BW(IQ1,IQ2,K)*FACTOR
               ELSE IF (KIP(J) .EQ. -11) THEN
                  QW(IQ1,IQ2,K) = GMW(J,K)/BW(IQ1,IQ2,K)*FACTOR
                  QG(IQ1,IQ2,K) = GMG(J,K)/BG(IQ1,IQ2,K)*FACTOR
     $                            +RSWAV*QW(IQ1,IQ2,K)
               ELSE
                  CONTINUE
               END IF
            END DO
         END IF
      END DO
      RETURN
      END


C
C ==========================================================================
      SUBROUTINE PRTPS (NLOOP,KPI,II,JJ,KK,PAVGO,PAVG,CWP,CWI,
     $                  CGP,CGI,MBEW,MBEG,DELTO,WPR,GPR,WIR,
     $                  GIR,ETI,CWGR,WGR,IPMAP,ISWMAP,ISGMAP)
C ==========================================================================
C     THIS SUBROUTINE ESTABLISHES THE PRESSURE AND SATURATION
C     DISTRIBUTIONS WITHIN THE RESERVOIR, AND MAKES A SUMMARY
C     REPORT.

C     NOMENCLATURE :

C        CGI       : CUMMULATIVE GAS INJECTION, MSCF
C        CWGR      : CUMMULATIVE PRODUCING WGR
C        CWI       : CUMMULATIVE WATER INJECTION, STB
C        CWP       : CUMMULATIVE WATER PRODUCTION, STB
C        DELTO     : TIME STEP, DAYS
C        DPW       : P(I,J,K)-PN(I,J,K)
C        DSG       : SG(I,J,K)-SGN(I,J,K)
C        DSW       : SW(I,J,K)-SWN(I,J,K)
C        ETI       : ELAPSED TIME, DAYS
C        GIR       : GAS INJECTION RATE, MSCF/D
C        GPR       : GAS PRODUCTION RATE, MSCF/D
C        IPMAP     : OUTPUT CODE TO CONTROL PRINTING OF THE MAP
C                    OF GRID BLOCK PRESSURES
C        ISGMAP    : OUTPUT CODE TO CONTROL PRINTING OF THE
C                    GRID BLOCK GAS SATURATIONS
C        ISWMAP    : OUTPUT CODE TO CONTROL PRINTING OF THE
C                    GRID BLOCK WATER SATURATIONS
C        KPI       : PRESSURE INITIALIZATION CODE
C        MBEG      : GAS MATERIAL BALANCE, PER CENT
C        MBEW      : WATER MATERIAL BALANCE, PER CENT
C        PAVG      : CURRENT AVERAGE RESERVOIR PRESSURE, PSIA
C        PAVGO     : PREVIOUS AVERAGE RESERVOIR PRESSURE, PSIA
C        PPM       : PRESSURE DPMAX
C        SGM       : GAS DSMAX
C        SWM       : WATER DSMAX
C        WGRM      : PRODUCING WATER-GAS RATIO
C        WIR       : WATER INJECTION RATE
C        WPR       : WATER PRODUCTION RATE

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

```
      INCLUDE 'COMGWSIM.FOR'

      PPM = 0.0
      SWM = 0.0
      SGM = 0.0
      IF (NLOOP .NE. 1) THEN
         DO 240 K=1,KK
         DO 240 J=1,JJ
         DO 240 I=1,II
            DPW = P(I,J,K)-PN(I,J,K)
            DSW = SW(I,J,K)-SWN(I,J,K)
            DSG = SG(I,J,K)-SGN(I,J,K)
         IF (ABS(DPW) .GT. ABS(PPM)) THEN
            PPM = DPW
            IPM = I
            JPM = J
            KPM = K
         END IF

         IF (ABS(DSW) .GT. ABS(SWM)) THEN
            SWM = DSW
            IWM = I
            JWM = J
            KWM = K
         END IF

         IF (ABS(DSG) .GT. ABS(SGM)) THEN
            SGM = DSG
            IGM = I
            JGM = J
            KGM = K
         END IF
240      CONTINUE

         WRITE(IOCODE,5)
         NLM = NLOOP-1
         WGRM = WGR
         WRITE(IOCODE,110) ETI,NLM,DELTO,PAVG,PAVGO,IPM,JPM,KPM,
     $                     PPM,IGM,JGM,KGM,SGM,IWM,JWM,KWM,SWM

         WRITE(IOCODE,111) MBEG,MBEW,GPR,CGP,WPR,CWP,
     $                     GIR,CGI,WIR,CWI,WGRM,CWGR
       END IF

       IF (NLOOP.EQ.1 .AND. KPI.EQ.0) THEN
          WRITE(IOCODE,120)
       ELSE
          WRITE(IOCODE,61)
       END IF

       IF (IPMAP .EQ. 0 .AND. NLOOP .NE. 1) GO TO 315
          DO 310 K=1,KK
            WRITE(IOCODE,51) K
          DO 310 J=1,JJ
            WRITE(IOCODE,41) (P(I,J,K), I=1,II)
310       CONTINUE

       IF (NLOOP.EQ.1 .AND. KPI.EQ.0) THEN
          WRITE(IOCODE,121)
       ELSE
          WRITE(IOCODE,81)
       END IF

315    IF (ISWMAP.EQ.0 .AND. NLOOP.NE.1) GO TO 432
          DO 430 K=1,KK
            WRITE(IOCODE,51) K
          DO 430 J=1,JJ
            WRITE(IOCODE,101) (SW(I,J,K), I=1,II)
430       CONTINUE

       IF (NLOOP.EQ.1 .AND. KPI.EQ.0) THEN
          WRITE(IOCODE,122)
       ELSE
          WRITE(IOCODE,91)
       END IF

432    IF (ISGMAP.EQ.0 .AND. NLOOP.NE.1) GO TO 450
          DO 440 K=1,KK
            WRITE(IOCODE,51) K
          DO 440 J=1,JJ
            WRITE(IOCODE,101) (SG(I,J,K), I=1,II)
440       CONTINUE
450    CONTINUE

       IF (NLOOP .NE. 1) WRITE(IOCODE,7)
```

```
      INCLUDE 'COMGWSIM.FOR'
C -----------------------------------------------------------------------
      CUMPW  = 0.0
      CUMPG  = 0.0
      CUMIW  = 0.0
      CUMIG  = 0.0
      MBEW   = 0.0
      MBEG   = 0.0
C -----------------------------------------------------------------------
C         READ CODES FOR CONTROLLING PRESSURE AND SATURATION
C         INITIALIZATION.
C -----------------------------------------------------------------------
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KPI,KSI

      IF (KPI .EQ. 0) THEN
         READ(20,*) PGWC,GWC
         DO 200 K=1,KK
         DO 200 J=1,JJ
         DO 200 I=1,II
            IF (EL(I,J,K) .LT. GWC) THEN
               CALL INTERP (NTE,PGT,BGT,MPGT,PGWC,BBG)
               RHOG = RHOSCG/BBG
               PN(I,J,K) = PGWC+RHOG*(EL(I,J,K)-GWC)/144.
            ELSE
               CALL INTERP (NTE,PWT,BWT,MPWT,PGWC,BBW)
               CALL INTERP (NTE,PWT,RSWT,MPWT,PGWC,RSW)
               RHOW = (RHOSCW+RSW*RHOSCG)/BBW
               PN(I,J,K) = PGWC+RHOW*(EL(I,J,K)-GWC)/144.
            END IF
200      CONTINUE
      ELSE
         DO 3009 K=1,KK
         DO 3009 J=1,JJ
           READ(20,*) (PN(I,J,K), I=1,II)
3009     CONTINUE
      END IF

      DO 3012 I=1,II
      DO 3012 J=1,JJ
      DO 3012 K=1,KK
        P(I,J,K) = PN(I,J,K)
3012  CONTINUE
C -----------------------------------------------------------------------
C         INITIALIZE SATURATION ARRAY.
C -----------------------------------------------------------------------
      IF (KSI .EQ. 0) THEN
         READ(20,*) SWC
         DO 30 K=1,KK
         DO 30 J=1,JJ
         DO 30 I=1,II
            IF (EL(I,J,K) .GT. GWC) THEN
               SWN(I,J,K) = 1.0
               SGN(I,J,K) = 0.0
               SW(I,J,K) = SWN(I,J,K)
               SG(I,J,K) = SGN(I,J,K)
            ELSE
               SWN(I,J,K) = SWC
               SGN(I,J,K) = 1.0-SWN(I,J,K)
               SW(I,J,K) = SWN(I,J,K)
               SG(I,J,K) = SGN(I,J,K)
            END IF
            IF (SG(I,J,K) .LT. 0.0)  SG(I,J,K)=0.0
30       CONTINUE
      ELSE
         DO 3020 K=1,KK
         DO 3020 J=1,JJ
           READ(20,*) (SWN(I,J,K), I=1,II)
3020     CONTINUE

         DO 3030 K=1,KK
         DO 3030 J=1,JJ
         DO 3030 I=1,II
           SGN(I,J,K) = 1.0-SWN(I,J,K)
           IF (SGN(I,J,K) .LT. 0.0) SGN(I,J,K)=0.0
           SW(I,J,K) = SWN(I,J,K)
           SG(I,J,K) = SGN(I,J,K)
3030     CONTINUE
      END IF

69    FORMAT(80A1)

      RETURN
      END
C ----------------------------------------------------------- END.
```

```
      IF (KK .GT. 1) THEN
            DO 60 K=1,KK
            DO 60 J=1,JJ
            DO 60 I=1,II
            IF (KZ(I,J,K-1) .GT. 0.0001 .AND. KZ(I,J,K) .GT.
     $      0.0001) THEN
                  TZ(I,J,K) = 0.012656*A3(I,J,K-1)*A3(I,J,K)/
     $                        (DZ(I,J,K-1)*A3(I,J,K)+DZ(I,J,K)*
     $                         A3(I,J,K-1))
            END IF
60          CONTINUE
      END IF

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

C        ESTABLISH TRANSMISSIBILITY MODIFICATIONS.
C        READ NUMBER OF BLOCKS WHERE TRANSMISSIBILITY VALUES
C        ARE TO BE CHANGED, AND INPUT PRINT CODE.

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

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) NUMTX, NUMTY, NUMTZ, ITCODE

      IF (NUMTX .GT. 0) THEN
         WRITE(IOCODE,31)
         DO L=1,NUMTX
            READ(20,*) I,J,K,TX(I,J,K)
            WRITE(IOCODE,32) I,J,K,TX(I,J,K)
         END DO
         IF (ITCODE .EQ. 1) THEN
            WRITE(IOCODE,44)
            DO 853 K=1,KK
               WRITE(IOCODE,38) K
            DO 853 J=1,JJ
               WRITE(IOCODE,72) (TX(I,J,K), I=1,II)
853         CONTINUE
         END IF
      END IF

      IF (NUMTY .GT. 0) THEN
         WRITE(IOCODE,34)
         DO L=1,NUMTY
            READ(20,*) I,J,K,TY(I,J,K)
            WRITE(IOCODE,32) I,J,K,TY(I,J,K)
         END DO
         IF (ITCODE .EQ. 1) THEN
            WRITE(IOCODE,47)
            DO 855 K=1,KK
               WRITE(IOCODE,38) K
            DO 855 J=1,JJ
               WRITE(IOCODE,72) (TY(I,J,K), I=1,II)
855         CONTINUE
         END IF
      END IF

      IF (NUMTZ .GT. 0) THEN
         WRITE(IOCODE,37)
         DO L=1,NUMTZ
            READ(20,*) I,J,K,TZ(I,J,K)
            WRITE(IOCODE,32) I,J,K,TZ(I,J,K)
         END DO
         IF (ITCODE .EQ. 1) THEN
            WRITE(IOCODE,48)
            DO 857 K=1,KK
               WRITE(IOCODE,38) K
            DO 857 J=1,JJ
               WRITE(IOCODE,72) (TZ(I,J,K), I=1,II)
857         CONTINUE
         END IF
      END IF

      IF (KTR .EQ. 1) THEN
         WRITE(IOCODE,43)
         DO 42 K=1,KK
         DO 42 J=1,JJ
            WRITE(IOCODE,41) I,J,K,A1(I,J,K),A2(I,J,K),A3(I,J,K)
42       CONTINUE
         WRITE(IOCODE,49)
         DO 505 K=1,KK
         DO 505 J=1,JJ
         DO 505 I=1,II
            WRITE(IOCODE,41) I,J,K,TX(I,J,K),TY(I,J,K),TZ(I,J,K)
505      CONTINUE
      END IF
```

```
C      NOMEMCLATURE:

C          A1(I,J,K)   :  KX(I,J,K) * DY(I,J,K) * DZ(I,J,K)
C          A2(I,J,K)   :  KY(I,J,K) * DX(I,J,K) * DZ(I,J,K)
C          A3(I,J,K)   :  KZ(I,J,K) * DX(I,J,K) * DY(I,J,K)
C          DX(I,J,K)   :  VARIABLE X-DIRECTION DIMENSION OF BLOCK
C                         IN THE MODEL GRID, FT.
C          DY(I,J,K)   :  VARIABLE Y-DIRECTION DIMENSION OF BLOCK
C                         IN THE MODEL GRID, FT.
C          DZ(I,J,K)   :  VARIABLE Z-DIRECTION DIMENSION OF BLOCK
C                         IN THE MODEL GRID, FT.
C          IHEDIN(L)   :  STORAGE MEMORY FOR INPUT TITLE CARD.
C          ITCODE      :  PRINT CODE FOR MODIFICATION TO TRANSMI-
C                         SSIBILITY DISTRIBUTIONS.
C                         ITCODE = 0 MEANS 'DO NOT PRINT', AND
C                         ITCODE = 1 MEANS 'PRINT'.
C          I           :  X-COORDINATE OF BLOCK TO BE MODIFIED.
C          II          :  NUMBER OF BLOCKS IN THE X-DIRECTION IN
C                         THE MODEL GRID.
C          J           :  Y-COORDINATE OF BLOCK TO BE MODIFIED.
C          JJ          :  NUMBER OF BLOCKS IN THE Y-DIRECTION IN
C                         THE MODEL GRID.
C          K           :  Z-COORDINATE OF BLOCK TO BE MODIFIED.
C          KK          :  NUMBER OF BLOCKS IN THE Z-DIRECTION IN
C                         THE MODEL GRID.
C          KX(I,J,K)   :  VARIABLE X-DIRECTION PERMEABILITY OF
C                         BLOCK IN THE MODEL GRID IN MILLIDARCIES.
C          KY(I,J,K)   :  VARIABLE Y-DIRECTION PERMEABILITY OF
C                         BLOCK IN THE MODEL GRID IN MILLIDARCIES.
C          KZ(I,J,K)   :  VARIABLE Z-DIRECTION PERMEABILITY OF
C                         BLOCK IN THE MODEL GRID IN MILLIDARCIES.
C          KTR         :  TRANSMISSIBILITY DEBUG OUTPUT CONTROL.
C                         KTR = 0 MEANS 'DO NOT PRINT', AND
C                         KTR = 1 MEANS 'PRINT'.
C          NUMTX       :  NUMBER OF GRID BLOCKS WHERE TX IS TO BE
C                         CHANGED.
C          NUMTY       :  NUMBER OF GRID BLOCKS WHERE TY IS TO BE
C                         CHANGED.
C          NUMTZ       :  NUMBER OF GRID BLOCKS WHERE TZ IS TO BE
C                         CHANGED.
C          TX(I,J,K)   :  VARIABLE COEFFICIENT OF X-DIRECTION TRAN-
C                         SMISSIBILITY.
C                         TX IS CALCULATED AS 0.012656* A1(I-1,J,K)*
C                         A1(I,J,K)/(DX(I-1,J,K)* A1(I,J,K) +
C                         DX(I,J,K)* A1(I-1,J,K))
C          TY(I,J,K)   :  VARIABLE COEFFICIENT OF Y-DIRECTION TRAN-
C                         SMISSIBILITY.
C                         TY IS CALCULATED AS 0.012656* A2(I,J-1,K)*
C                         A2(I,J,K)/(DY(I,J-1,K)* A2(I,J,K) +
C                         DY(I,J,K)* A2(I,J-1,K)).
C          TZ(I,J,K)   :  VARIABLE COEFFICIENT OF Z-DIRECTION TRAN-
C                         SMISSIBILITY.
C                         TZ IS CALCULATED AS 0.012656* A3(I,J,K-1)*
C                         A3(I,J,K)/(DZ(I,J,K-1)* A3(I,J,K) +
C                         DZ(I,J,K)* A3(I,J,K-1)).

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

      INCLUDE 'COMGWSIM.FOR'

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

C         CALCULATE THE COEFFICIENTS OF TRANSMISSIBILITY FOR
C         BLOCKS IN THE MODEL GRID.

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

      DO 30 K=1,KK
      DO 30 J=1,JJ
      DO 30 I=1,II

         A1(I,J,K) = KX(I,J,K)*DY(I,J,K)*DZ(I,J,K)
         A2(I,J,K) = KY(I,J,K)*DX(I,J,K)*DZ(I,J,K)
         A3(I,J,K) = KZ(I,J,K)*DX(I,J,K)*DY(I,J,K)
 30   CONTINUE

      IF (II .GT. 1) THEN
           DO 50 K=1,KK
           DO 50 J=1,JJ
           DO 50 I=1,II
           IF (KX(I-1,J,K) .GT. 0.0001 .AND. KX(I,J,K) .GT.
     $     0.0001) THEN
                TX(I,J,K) = 0.012656*A1(I-1,J,K)*A1(I,J,K)/
     $                      (DX(I-1,J,K)*A1(I,J,K)+DX(I,J,K)*
     $                       A1(I-1,J,K))
           END IF
50         CONTINUE
      END IF

      IF (JJ .GT. 1) THEN
           DO 55 K=1,KK
           DO 55 J=1,JJ
           DO 55 I=1,II
           IF (KY(I,J-1,K) .GT. 0.0001 .AND. KY(I,J,K) .GT.
     $     0.0001) THEN
                TY(I,J,K) = 0.012656*A2(I,J-1,K)*A2(I,J,K)/
     $                      (DY(I,J-1,K)*A2(I,J,K)+DY(I,J,K)*
     $                       A2(I,J-1,K))
           END IF
55         CONTINUE
      END IF
```

```
5     FORMAT(6(/),12X,58('*')/12X,'*',56X,'*'/12X,'*',56X,'*',
     $       /12X,'*',7X,'SUMMARY REPORT : UNSW GWSIM  (VERSION',
     $       ' 2.0)',7X,'*'/12X,'*',56X,'*'/12X,'*',56X,'*',
     $       /12X,58('*')///)
7     FORMAT(///2X,30('*'),' END OF REPORT ',30('*'),6(/))
41    FORMAT(25X,20F6.0)
51    FORMAT(/20X,'LAYER (',I2,')'/20X,10('-')/)
61    FORMAT(4(/),20X,'PRESENT RESERVOIR PRESSURE DISTRIBUTION',
     $       /20X,39('-')/)
81    FORMAT(7(/),20X,'PRESENT WATER SATURATION'/20X,24('-')/)
91    FORMAT(7(/),20X,'PRESENT GAS SATURATION'/20X,22('-')/)
101   FORMAT(25X,20F6.3)
110   FORMAT(17X,'ELAPSED TIME',T49,':',F12.2,1X,'DAYS.',
     $       /17X,'TIME STEP NUMBER',T49,':',I9,
     $       /17X,'TIME STEP',T49,':',F12.2,1X,'DAYS.',
     $       /17X,'CURRENT AVG. RESERVOIR PR',T49,':',F12.2,1X,
     $       'PSIA.',
     $       /17X,'PREVIOUS AVG. RESERVOIR PR',T49,':',F12.2,1X,
     $       'PSIA.',
     $       /17X,'PRESSURE DPMAX',2X,'(',3I3,' )',T49,':',F15.5,
     $       /17X,'GAS DSMAX',7X,'(',3I3,' )',T49,':',F15.5,
     $       /17X,'WATER DSMAX',5X,'(',3I3,' )',T49,':',F15.5/)
111   FORMAT(17X,'GAS MATERIAL BALANCE',T49,':',E12.4,
     $       1X,'PER CENT.',
     $       /17X,'WATER MATERIAL BALANCE',T49,':',E12.4,1X,
     $       'PER CENT.',
     $       //17X,'GAS PRODUCTION RATE',T49,':',E12.4,1X,'MSCF/D.',
     $       /17X,'CUMMULATIVE GAS PRODUCTION',T49,':',E12.4,
     $       1X,'MSCF.',
     $       /17X,'WATER PRODUCTION RATE',T49,':',E12.4,1X,'STB/D.',
     $       /17X,'CUMMULATIVE WATER PRODUCTION',T49,':',E12.4,
     $       1X,'STB.',
     $       /17X,'GAS INJECTION RATE',T49,':',E12.4,1X,'MSCF/D.',
     $       /17X,'CUMMULATIVE GAS INJECTION',T49,':',E12.4,
     $       1X,'MSCF.',
     $       /17X,'WATER INJECTION RATE',T49,':',E12.4,1X,'STB/D.',
     $       /17X,'CUMMULATIVE WATER INJECTION',T49,':',E12.4,
     $       1X,'STB.',
     $       /17X,'PRODUCING WATER-GAS RATIO',T49,':',E12.4,1X,
     $       'STB/SCF.',
     $       /17X,'CUMMULATIVE PRODUCING WGR',T49,':',E12.4,1X,
     $       'STB/SCF'//)
120   FORMAT(4(/),20X,'INITIAL RESERVOIR PRESSURE DISTRIBUTION',
     $       /20X,39('-')/)
121   FORMAT(4(/),20X,'INITIAL WATER SATURATION DISTRIBUTION',
     $       /20X,36('-')/)
122   FORMAT(4(/),20X,'INITIAL GAS SATURATION DISTRIBUTION',
     $       /20X,34('-')/)

      RETURN
      END

C
C ======================================================================

      SUBROUTINE  QRATE (NVQN)

C ======================================================================

C     THIS SUBROUTINE CALCULATES THE LAYERED FLOW RATES OF THE
C     INDIVIDUAL PHASES FOR THE WELL UNDER RATE CONSTRAINT, AND
C     THEN DETERMINES THE FLOWING BOTTOM-HOLE PRESSURE OF THAT
C     WELL.     IF THE WELL IS UNDER PID AND FBHP CONTROL, IT
C     CALCULATES THE LAYERED FLOW RATES OF THE INDIVIDUAL PHASES
C     FOR EXPLICIT PRESSURE CALCULATION.

C     NOMENCLATURE :

C       ALPHAW      : GMW/GMT
C       ALPHAG      : GMG/GMT
C       GMG         : KRG/MUG
C       GMW         : KRW/MUW
C       GMT         : GMG+GMW
C       IQ1         : IQN1(I,J,K)
C       IQ2         : IQN2(I,J,K)
C       IQ3         : IQN3(I,J,K)
C       KRG         : GAS PHASE RELATIVE PERMEABILITY
C       KRW         : WATER PHASE RELATIVE PERMEABILITY
C       LAYER       : TOTAL NUMBER OF CONSECUTIVE COMPLETION LAYERS
C                     STARTING WITH AND INCLUDING IQN3
C       MUG         : GAS VISCOSITY
C       MUW         : WATER VISCOSITY
C       NTE         : MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN ALL
C                     PVT TABLES
C       NVQN        : NUMBER OF WELLS FOR WHICH WELL INFORMATION IS
C                     TO BE READ
C       PGT         : GAS PHASE PRESSURE ENTRY IN PVT TABLE
C       PP          : P(IQ1,IQ2,K)
C       PPN         : PN(IQ1,IQ2,K)
C       PWFC        : FLOWING BOTTOM-HOLE PRESSURE OF RATE CONSTRA-
C                     INTED WELL
C       PWT         : WATER PHASE PRESSURE ENTRY IN PVT TABLE
C       QG          : LAYERED GAS FLOW RATE
C       QW          : LAYERED WATER FLOW RATE
C       QVG         : WELL'S GAS PRODUCTION RATE
C       QVW         : WELL'S WATER PRODUCTION RATE (RATE CONSTRAINT)
C       SG          : GAS SATURATION
C       SW          : WATER SATURATION

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

```
      INCLUDE 'COMGWSIM.FOR'

      DO J = 1,NVQN
         IQ1 = IQN1(J)
         IQ2 = IQN2(J)
         IQ3 = IQN3(J)
         LAY = IQ3+LAYER(J)-1

         DO K = IQ3,LAY
            PWFC(J,K) = -1.0
            PP = P(IQ1,IQ2,K)
            CALL INTERP (NTE,PWT,MUWT,MPWT,PP,MUW)
            CALL INTERP (NTE,PGT,MUGT,MPGT,PP,MUG)
            SSW = SW(IQ1,IQ2,K)
            SSG = SG(IQ1,IQ2,K)
            CALL INTERP (NTE,SAT,KRWT,MSAT,SSW,KRW)
            CALL INTERP (NTE,SAT,KRGT,MSAT,SSG,KRG)
            GMW(J,K) = KRW/MUW
            GMG(J,K) = KRG/MUG
         END DO

         IF (KIP(J) .EQ. 1) THEN
            QDENOM = 0.0
            ALPHAW = 0.0
            ALPHAG = 0.0
            DO K = IQ3,LAY
               PP = P(IQ1,IQ2,K)
               CALL INTERP (NTE,PWT,BWT,MPWT,PP,BBW)
               CALL INTERP (NTE,PGT,BGT,MPGT,PP,BBG)
               CALL INTERP (NTE,PWT,RSWT,MPWT,PP,RSW)
               QDENOM = QDENOM+PID(J,K)*GMW(J,K)/BBW
               GMT = GMW(J,K)+GMG(J,K)
               ALPHAW = ALPHAW+GMW(J,K)/GMT
               ALPHAG = ALPHAG+GMG(J,K)/GMT
            END DO

            IF (QVT(J) .NE. 0.0) THEN
               TOTOR = QVT(J)*ALPHAW/(ALPHAW+ALPHAG)
            ELSE
               TOTOR = QVW(J)
            END IF

            DO K = IQ3,LAY
               QW(IQ1,IQ2,K) = TOTOR*5.615*PID(J,K)*GMW(J,K)
     $                         /(BBW*QDENOM)
               QG(IQ1,IQ2,K) = QW(IQ1,IQ2,K)*GMG(J,K)*BBW
     $                         /(BBG*GMW(J,K))+RSW*QW(IQ1,IQ2,K)
            END DO

         ELSE IF (KIP(J) .GT. 1) THEN
            QDENOM = 0.0
            DO K = IQ3,LAY
               QDENOM = QDENOM+PID(J,K)*(GMW(J,K)+GMG(J,K))
            END DO

            DO K = IQ3,LAY
               IF (KIP(J) .EQ. 2) THEN
                  QW(IQ1,IQ2,K) = QVW(J)*5.615*PID(J,K)*(GMW(J,K)
     $                            + GMG(J,K))/QDENOM
               ELSE
                  QG(IQ1,IQ2,K) = QVG(J)*1000.*PID(J,K)
     $                            * (GMW(J,K)+GMG(J,K))/QDENOM
               END IF
            END DO

         ELSE
            CONTINUE
         END IF
      END DO

      DO J =1,NVQN
         IQ1 = IQN1(J)
         IQ2 = IQN2(J)
         IQ3 = IQN3(J)
         LAY = IQ3+LAYER(J)-1

         IF (KIP(J) .GE. 0) THEN
            DO K = IQ3,LAY
               PWFC(J,K) = 0.0
               IF (PID(J,K) .GT. 0.0001) THEN
                  PP = P(IQ1,IQ2,K)
                  CALL INTERP (NTE,PWT,BWT,MPWT,PP,BBW)
                  CALL INTERP (NTE,PGT,BGT,MPGT,PP,BBG)
                  CALL INTERP (NTE,PWT,RSWT,MPWT,PP,RSW)
                  FAC = PID(J,K)*5.615

                  IF (KIP(J) .EQ. 2) THEN
                     GMTB = (GMW(J,K)+GMG(J,K))/BBW
                     PWFC(J,K) = PP-QW(IQ1,IQ2,K)/(FAC*GMTB)
                  ELSE IF (KIP(J) .EQ. 3) THEN
                     GMTB = (GMG(J,K)+GMW(J,K))/BBG
                     PWFC(J,K) = PP-QG(IQ1,IQ2,K)/(FAC*GMTB)
                  ELSE
                     GMTB = GMW(J,K)/BBW+GMG(J,K)/BBG
                     SOLN = RSW*QW(IQ1,IQ2,K)
                     QT   = QW(IQ1,IQ2,K)+QG(IQ1,IQ2,K)
                     PWFC(J,K) = PP-(QT-SOLN)/(FAC*GMTB)
                  END IF
```

```
            END IF
            END DO
        ELSE
            DO K = IQ3,LAY
                PPN = PN(IQ1,IQ2,K)
                CALL INTERP (NTE,PWT,BWT,MPWT,PPN,BBW)
                CALL INTERP (NTE,PGT,BGT,MPGT,PPN,BBG)
                CALL INTERP (NTE,PWT,RSWT,MPWT,PPN,RSW)

                IF (KIP(J) .EQ. -1) THEN
                    QW(IQ1,IQ2,K) = PID(J,K)*5.615*GMW(J,K)*
     $                              (PPN-PWF(J,K))/BBW
                    IF (PPN .LE. PWF(J,K)) QW(IQ1,IQ2,K)=0.0
                    QG(IQ1,IQ2,K) = QW(IQ1,IQ2,K)*GMG(J,K)*BBW
     $                              /(BBG*GMW(J,K))+RSW*QW(IQ1,IQ2,K)
                ELSE IF (KIP(J) .EQ. -2) THEN
                    QW(IQ1,IQ2,K) = PID(J,K)*5.615*(GMW(J,K)+
     $                              GMG(J,K))*(PPN-PWF(J,K))/BBW
                    IF (PPN .GE. PWF(J,K)) QW(IQ1,IQ2,K)=0.0
                ELSE IF (KIP(J) .EQ. -3) THEN
                    QG(IQ1,IQ2,K) = PID(J,K)*5.615*(GMW(J,K)+
     $                              GMG(J,K))*(PPN-PWF(J,K))/BBG
                    IF (PPN .GE. PWF(J,K)) QG(IQ1,IQ2,K)=0.0
                ELSE
                    CONTINUE
                END IF

            END DO
        END IF
    END DO
    RETURN
    END


C
C ==========================================================================

      SUBROUTINE  SOLMAT (II,JJ,KK,DIV1,D288,D144,KSM,
     $                    KSM1,N,NN,KCOFF)

C ==========================================================================

C     THIS SUBROUTINE ESTABLISHES  AW,AE,AS,AN,AT,AB,E, AND B.

      INCLUDE 'COMGWSIM.FOR'

      REAL  MUG1,MUG2,MUG3,MUG4,MUG5,MUG6,
     $      MUW1,MUW2,MUW3,MUW4,MUW5,MUW6,
     $      MG1,MG2,MG3,MG4,MG5,MG6,
     $      MW1,MW2,MW3,MW4,MW5,MW6,
     $      KRG1,KRG2,KRG3,KRG4,KRG5,KRG6,
     $      KRW1,KRW2,KRW3,KRW4,KRW5,KRW6

      DATA  MG1,MG2,MG3,MG4,MG5,MG6/6*0.0/
      DATA  MW1,MW2,MW3,MW4,MW5,MW6/6*0.0/
      DATA  RSW1,RSW2,RSW3,RSW4,RSW5,RSW6/6*0.0/
      DATA  GGW1,GGW2,GGW3,GGW4,GGW5,GGW6/6*0.0/
      DATA  GWW1,GWW2,GWW3,GWW4,GWW5,GWW6/6*0.0/


      DO 200 K=1,KK
      DO 200 J=1,JJ
      DO 200 I=1,II
          PP = P(I,J,K)
          CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
          CALL INTERP(NTE,PWT,MUWT,MPWT,PP,MUW)
          CALL INTERP(NTE,PGT,MUGT,MPGT,PP,MUG)
          SSW = SW(I,J,K)
          SSG = SG(I,J,K)
          RW = (RHOSCW+RSW*RHOSCG)/BW(I,J,K)
          RG = RHOSCG/BG(I,J,K)
          IF (I .NE. 1) THEN
              P1 = P(I-1,J,K)
              CALL INTERP(NTE,PWT,RSWT,MPWT,P1,RSW1)
              CALL INTERP(NTE,PWT,MUWT,MPWT,P1,MUW1)
              CALL INTERP(NTE,PGT,MUGT,MPGT,P1,MUG1)
              SW1S = SW(I-1,J,K)
              SG1S = SG(I-1,J,K)
              RW1 = (RHOSCW+RSW1*RHOSCG)/BW(I-1,J,K)
              RG1 = RHOSCG/BG(I-1,J,K)
              FACT = -D288*(EL(I-1,J,K)-EL(I,J,K))
              GWW1 = (RW1+RW)*FACT
              GGW1 = (RG1+RG)*FACT
              P11 = P1-PP
              HW1 = P11+GWW1
              HG1 = P11+GGW1
              IF (HW1 .GE. 0.0) THEN
                  CALL INTERP(NTE,SAT,KRWT,MSAT,SW1S,KRW1)
              ELSE
                  CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW1)
              END IF

              IF (HG1 .GE. 0.0) THEN
                  CALL INTERP(NTE,SAT,KRGT,MSAT,SG1S,KRG1)
              ELSE
                  CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG1)
              END IF

              MW1 = 4.0*KRW1/((BW(I-1,J,K)+BW(I,J,K))*(MUW1+MUW))
              MG1 = 4.0*KRG1/((BG(I-1,J,K)+BG(I,J,K))*(MUG1+MUG))
      END IF
```

```
AWW = TX(I,J,K)*MW1
AGW = TX(I,J,K)*MG1

IF (I .NE. II) THEN
   P2 = P(I+1,J,K)
   CALL INTERP(NTE,PWT,RSWT,MPWT,P2,RSW2)
   CALL INTERP(NTE,PWT,MUWT,MPWT,P2,MUW2)
   CALL INTERP(NTE,PGT,MUGT,MPGT,P2,MUG2)
   SW2 = SW(I+1,J,K)
   SG2 = SG(I+1,J,K)
   RW2 = (RHOSCW+RSW2*RHOSCG)/BW(I+1,J,K)
   RG2 = RHOSCG/BG(I+1,J,K)
   FACT = -D288*(EL(I+1,J,K)-EL(I,J,K))
   GWW2 = (RW2+RW)*FACT
   GGW2 = (RG2+RG)*FACT
   P22 = P2-PP
   HW2 = P22+GWW2
   HG2 = P22+GGW2
   IF (HW2 .GE. 0.0) THEN
      CALL INTERP(NTE,SAT,KRWT,MSAT,SW2,KRW2)
   ELSE
      CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW2)
   END IF
   IF (HG2. GE. 0.0) THEN
      CALL INTERP(NTE,SAT,KRGT,MSAT,SG2,KRG2)
   ELSE
      CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG2)
   END IF
   MW2 = 4.0*KRW2/((BW(I+1,J,K)+BW(I,J,K))*(MUW2+MUW))
   MG2 = 4.0*KRG2/((BG(I+1,J,K)+BG(I,J,K))*(MUG2+MUG))
END IF

AWE = TX(I+1,J,K)*MW2
AGE = TX(I+1,J,K)*MG2

IF (J .NE. 1) THEN
   P3 =P(I,J-1,K)
   CALL INTERP(NTE,PWT,RSWT,MPWT,P3,RSW3)
   CALL INTERP(NTE,PWT,MUWT,MPWT,P3,MUW3)
   CALL INTERP(NTE,PGT,MUGT,MPGT,P3,MUG3)
   SW3 = SW(I,J-1,K)
   SG3 = SG(I,J-1,K)
   RW3 = (RHOSCW+RSW3*RHOSCG)/BW(I,J-1,K)
   RG3 = RHOSCG/BG(I,J-1,K)
   FACT = -D288*(EL(I,J-1,K)-EL(I,J,K))
   GWW3 = (RW3+RW)*FACT
   GGW3 = (RG3+RG)*FACT
   P33 = P3-PP
   HW3 = P33+GWW3
   HG3 = P33+GGW3
   IF (HW3 .GE. 0.0) THEN
      CALL INTERP(NTE,SAT,KRWT,MSAT,SW3,KRW3)
   ELSE
      CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW3)
   END IF

   IF (HG3 .GE. 0.0) THEN
      CALL INTERP(NTE,SAT,KRGT,MSAT,SG3,KRG3)
   ELSE
      CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG3)
   END IF
   MW3 = 4.0*KRW3/((BW(I,J-1,K)+BW(I,J,K))*(MUW3+MUW))
   MG3 = 4.0*KRG3/((BG(I,J-1,K)+BG(I,J,K))*(MUG3+MUG))
END IF

AWS = TY(I,J,K)*MW3
AGS = TY(I,J,K)*MG3

IF (J .NE. JJ) THEN
   P4 = P(I,J+1,K)
   CALL INTERP(NTE,PWT,RSWT,MPWT,P4,RSW4)
   CALL INTERP(NTE,PWT,MUWT,MPWT,P4,MUW4)
   CALL INTERP(NTE,PGT,MUGT,MPGT,P4,MUG4)
   SW4 = SW(I,J+1,K)
   SG4 = SG(I,J+1,K)
   RW4 = (RHOSCW+RSW4*RHOSCG)/BW(I,J+1,K)
   RG4 = RHOSCG/BG(I,J+1,K)
   FACT = -D288*(EL(I,J+1,K)-EL(I,J,K))
   GWW4 = (RW4+RW)*FACT
   GGW4 = (RG4+RG)*FACT
   P44 = P4-PP
   HW4 = P44+GWW4
   HG4 = P44+GGW4
   IF (HW4 .GE. 0.0) THEN
      CALL INTERP(NTE,SAT,KRWT,MSAT,SW4,KRW4)
   ELSE
      CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW4)
   END IF

   IF (HG4 .GE. 0.0) THEN
      CALL INTERP(NTE,SAT,KRGT,MSAT,SG4,KRG4)
   ELSE
      CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG4)
   END IF
   MW4 = 4.0*KRW4/((BW(I,J+1,K)+BW(I,J,K))*(MUW4+MUW))
   MG4 = 4.0*KRG4/((BG(I,J+1,K)+BG(I,J,K))*(MUG4+MUG))
END IF

AWN = TY(I,J+1,K)*MW4
AGN = TY(I,J+1,K)*MG4
```

```
      IF (K .NE. 1) THEN
         P5 = P(I,J,K-1)
         CALL INTERP(NTE,PWT,RSWT,MPWT,P5,RSW5)
         CALL INTERP(NTE,PWT,MUWT,MPWT,P5,MUW5)
         CALL INTERP(NTE,PGT,MUGT,MPGT,P5,MUG5)
         SW5 = SW(I,J,K-1)
         SG5 = SG(I,J,K-1)
         RW5 = (RHOSCW+RSW5*RHOSCG)/BW(I,J,K-1)
         RG5 = RHOSCG/BG(I,J,K-1)
         FACT = -D288*(EL(I,J,K-1)-EL(I,J,K))
         GWW5 = (RW5+RW)*FACT
         GGW5 = (RG5+RG)*FACT
         P55 = P5-PP
         HW5 = P55+GWW5
         HG5 = P55+GGW5
         IF (HW5 .GE. 0.0) THEN
            CALL INTERP(NTE,SAT,KRWT,MSAT,SW5,KRW5)
         ELSE
            CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW5)
         END IF

         IF (HG5 .GE. 0.0) THEN
            CALL INTERP(NTE,SAT,KRGT,MSAT,SG5,KRG5)
         ELSE
            CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG5)
         END IF
         MW5 = 4.0*KRW5/((BW(I,J,K-1)+BW(I,J,K))*(MUW5+MUW))
         MG5 = 4.0*KRG5/((BG(I,J,K-1)+BG(I,J,K))*(MUG5+MUG))
      END IF

      AWT = TZ(I,J,K)*MW5
      AGT = TZ(I,J,K)*MG5

      IF (K .NE. KK) THEN
         P6 = P(I,J,K+1)
         CALL INTERP(NTE,PWT,RSWT,MPWT,P6,RSW6)
         CALL INTERP(NTE,PWT,MUWT,MPWT,P6,MUW6)
         CALL INTERP(NTE,PGT,MUGT,MPGT,P6,MUG6)
         SW6 = SW(I,J,K+1)
         SG6 = SG(I,J,K+1)
         RW6 = (RHOSCW+RSW6*RHOSCG)/BW(I,J,K+1)
         RG6 = RHOSCG/BG(I,J,K+1)
         FACT = -D288*(EL(I,J,K+1)-EL(I,J,K))
         GWW6 = (RW6+RW)*FACT
         GGW6 = (RG6+RG)*FACT
         P66 = P6-PP
         HW6 = P66+GWW6
         HG6 = P66+GGW6
         IF (HW6 .GE. 0.0) THEN
            CALL INTERP(NTE,SAT,KRWT,MSAT,SW6,KRW6)
         ELSE
            CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW6)
         END IF

         IF (HG6 .GE. 0.0) THEN
            CALL INTERP(NTE,SAT,KRGT,MSAT,SG6,KRG6)
         ELSE
            CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG6)
         END IF
         MW6 = 4.0*KRW6/((BW(I,J,K+1)+BW(I,J,K))*(MUW6+MUW))
         MG6 = 4.0*KRG6/((BG(I,J,K+1)+BG(I,J,K))*(MUG6+MUG))
      END IF

      AWB = TZ(I,J,K+1)*MW6
      AGB = TZ(I,J,K+1)*MG6

      RSW1A = 0.5*(RSW1+RSW)
      RSW2A = 0.5*(RSW2+RSW)
      RSW3A = 0.5*(RSW3+RSW)
      RSW4A = 0.5*(RSW4+RSW)
      RSW5A = 0.5*(RSW5+RSW)
      RSW6A = 0.5*(RSW6+RSW)

      AW1 = AWW*GWW1
      AW2 = AWE*GWW2
      AW3 = AWS*GWW3
      AW4 = AWN*GWW4
      AW5 = AWT*GWW5
      AW6 = AWB*GWW6

      GWWT(I,J,K) = AW1+AW2+AW3+AW4+AW5+AW6
      GGWT(I,J,K) = AGW*GGW1+AGE*GGW2+AGS*GGW3+AGN*GGW4+
     $              AGT*GGW5+AGB*GGW6+RSW1A*AW1+RSW2A*AW2+
     $              RSW3A*AW3+RSW4A*AW4+RSW5A*AW5+RSW6A*AW6
      QOWG(I,J,K) = (BW(I,J,K)-BG(I,J,K)*RSW)*(-GWWT(I,J,K)+
     $              QW(I,J,K))+BG(I,J,K)*(-GGWT(I,J,K)+QG(I,J,K))

      AW(I,J,K) = (BW(I,J,K)+0.5*BG(I,J,K)*(RSW1-RSW))*AWW+
     $            BG(I,J,K)*AGW
      AE(I,J,K) = (BW(I,J,K)+0.5*BG(I,J,K)*(RSW2-RSW))*AWE+
     $            BG(I,J,K)*AGE
      AS(I,J,K) = (BW(I,J,K)+0.5*BG(I,J,K)*(RSW3-RSW))*AWS+
     $            BG(I,J,K)*AGS
      AN(I,J,K) = (BW(I,J,K)+0.5*BG(I,J,K)*(RSW4-RSW))*AWN+
     $            BG(I,J,K)*AGN
      AT(I,J,K) = (BW(I,J,K)+0.5*BG(I,J,K)*(RSW5-RSW))*AWT+
     $            BG(I,J,K)*AGT
      AB(I,J,K) = (BW(I,J,K)+0.5*BG(I,J,K)*(RSW6-RSW))*AWB+
     $            BG(I,J,K)*AGB
```

```
      WW(I,J,K) = AWW
      WE(I,J,K) = AWE
      WS(I,J,K) = AWS
      WN(I,J,K) = AWN
      WT(I,J,K) = AWT
      WB(I,J,K) = AWB

      IF (KCOFF .NE. 0) THEN
         WRITE(IOCODE,33)
         WRITE(IOCODE,2)  I,J,K,MW1,MW2,MW3,MW4,MW5,MW6
         WRITE(IOCODE,2)  I,J,K,MG1,MG2,MG3,MG4,MG5,MG6
         WRITE(IOCODE,2)  I,J,K,AWW,AWE,AWS,AWN,AWT,AWB,
     $                    BW(I,J,K),RSW
         WRITE(IOCODE,2)  I,J,K,AGW,AGE,AGS,AGN,AGT,AGB,
     $                    BG(I,J,K)
         WRITE(IOCODE,2)  I,J,K,GWWT(I,J,K),QW(I,J,K),
     $                    GGWT(I,J,K),QG(I,J,K),QOWG(I,J,K)
      END IF
  200 CONTINUE
C -----------------------------------------------------------------
C       CALCULATE MAIN DIAGONAL AND RHS VECTOR.
C -----------------------------------------------------------------
      DO 300 K=1,KK
      DO 300 J=1,JJ
      DO 300 I=1,II
         SUM(I,J,K) = AW(I,J,K)+AE(I,J,K)+AS(I,J,K)+AN(I,J,K)+
     $                AT(I,J,K)+AB(I,J,K)
         GAM(I,J,K) = VP(I,J,K)*CT(I,J,K)*DIV1
         E(I,J,K)   = -SUM(I,J,K)-GAM(I,J,K)
         B(I,J,K)   = QOWG(I,J,K)-GAM(I,J,K)*P(I,J,K)
  300 CONTINUE

      IF (KSM1 .EQ. 0) RETURN
      IF (N.NE.1 .AND. N.NE.NN .AND. N.NE.KSM) RETURN

      WRITE(IOCODE,4)
      DO 500 K=1,KK
      DO 500 J=1,JJ
      DO 500 I=1,II
        WRITE(IOCODE,2) I,J,K,AT(I,J,K),AS(I,J,K),AW(I,J,K),
     $                  E(I,J,K),AE(I,J,K),AN(I,J,K),AB(I,J,K),
     $                  B(I,J,K)
  500  CONTINUE


    2   FORMAT(1X,3I3,8E15.6)
    4   FORMAT(//3X,'NODE',6X,'AT(I,J,K)',5X,'AS(I,J,K)',5X,
     $          'AW(I,J,K)',5X,'E(I,J,K)',5X,'AE(I,J,K)',5X,
     $          'AN(I,J,K)',5X,'AB(I,J,K)',5X,'B(I,J,K)'/)
   33   FORMAT(//)
        RETURN
        END
```

```
C
C ==========================================================================

      SUBROUTINE TABLE

C ==========================================================================

C     THE SUBROUTINE ESTABLISHES THE RELATIVE PERMEABILITY AND
C     CAPILLARY PRESSURE TABLES, AND FLUID PVT DATA TABLES.

C     NOMENCLATURE:

C          BWT(L)     :  VARIABLE WATER FORMATION VOLUME FACTOR
C                        VALUE, RB/STB.
C          BGT(L)     :  VARIABLE GAS FORMATION FACTOR VALUE,
C                        RCF/SCF.
C          BWPT(L)    :  SLOPE dBW/dP.
C                        BWPT IS CALCULATED AS (BWT(L)-BWT(L-1))/
C                        (PWT(L)-PWT(L-1)).
C          BGPT(L)    :  SLOPE dBG/dP.
C                        BGPT IS CALCULATED AS (BGT(L)-BGT(L-1))/
C                        (PGT(L)-PGT(L-1)).
C          CRT(L)     :  PRESSURE DEPENDENT ROCK COMPRESSIBILITY,
C                        /PSIA.
C          IHEDIN(L)  :  STORAGE MEMORY FOR INPUT TITLE CARD.
C          L          :  COUNT NUMBER.
C          KRGT(L)    :  GAS PHASE RELATIVE PERMEABILITY
C          KRWT(L)    :  WATER PHASE RELATIVE PERMEABILITY.
C          MSAT       :  MAXIMUM NUMBER OF ENTRY VALUE FOR PHASE
C                        SATURATION.
C          MUWT(L)    :  WATER VISCOSITY, CENTIPOISES.
C          MPWT       :  MAXIMUM NUMBER OF ENTRY VALUE FOR WATER
C                        PHASE PRESSURE, PSIA.
C          MUGT(L)    :  GAS VISCOSITY, CPS.
C          MPGT       :  MAXIMUM NUMBER OF ENTRY VALUE FOR GAS
C                        PHASE PRESSURE, PSIA.
C          NTE        :  MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN
C                        ALL PVT TABLES.
C          PCGWT(L)   :  GAS/WATER CAPILLARY PRESSURE, PSI.
C          PMAXT      :  MAXIMUM PRESSURE ENTRY IN ALL PVT TABLES,
C                        PSIA.
C          PWT(L)     :  WATER PHASE PRESSURE, PSIA.
C                        THE LAST ENTRY OF PWT IN THE TABLE MUST
C                        BE PMAXT.
C          PGT(L)     :  GAS PHASE PRESSURE, PSIA.
C                        THE LAST ENTRY OF PGT IN THE TABLE MUST
C                        BE PMAXT.
C          RSWT(L)    :  GAS IN WATER SOLUTION RATIO, SCF/STB.
C          RHOSCW     :  STOCK TANK WATER DENSITY, LB/CU FT.
C          RHOSCG     :  GAS DENSITY AT STANDARD CONDITIONS,
C                        LB/CU FT.
C          RSWPT(L)   :  SLOPE dRSW/dP.
C                        RSWPT IS CALCULATED AS (RSWT(L)-RSWT(L-1))
C                        /(PWT(L)-PWT(L-1)).
C          SAT(L)     :  VALUE OF PHASE SATURATION IN FRACTION.
C                        FIRST ENTRY OF SAT SHOULD BE -0.10, AND
C                        THE LAST ENTRY MUST BE 1.10.
C                        SAT REFERS TO THE SATURATION OF EACH PAR-
C                        TICULAR PHASE. FOR EXAMPLE, SAT=0.2 MEANS
C                        KRWT WOULD REFER TO WATER RELATIVE PERM-
C                        EABILITY IN THE PRESENCE OF 20 PERCENT
C                        WATER SATURATION, KRGT(20% GAS SATURATION)
C                        AND PCGWT(20% GAS SATURATION).

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

      INCLUDE 'COMGWSIM.FOR'

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

C         ESTABLISH RELATIVE PERMEABILITY AND CAPILLARY
C         PRESSURE TABLES.

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

      WRITE(IOCODE,111)
      READ(20,69) (IHEDIN(L), L=1,80)
      DO L=1,NTE
         READ(20,*) SAT(L),KRWT(L),KRGT(L),PCGWT(L)
         WRITE(IOCODE,21) SAT(L),KRWT(L),KRGT(L),PCGWT(L)
         IF (SAT(L).GE.1.1) GO TO 10
      END DO
 10   MSAT = L

      WRITE(IOCODE,112)
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) PMAXT
      WRITE(IOCODE,31) PMAXT

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

C         ESTABLISH WATER DATA TABLE.

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

      WRITE(IOCODE,113)
      READ(20,69) (IHEDIN(L), L=1,80)
      DO L=1,NTE
         READ(20,*) PWT(L),MUWT(L),BWT(L),RSWT(L)
         WRITE(IOCODE,21) PWT(L),MUWT(L),BWT(L),RSWT(L)
         IF (PWT(L).GE.PMAXT) GO TO 30
      END DO
 30   MPWT = L
```

```
C ------------------------------------------------------------------
C          ESATBLISH GAS AND ROCK DATA TABLE.
C ------------------------------------------------------------------
      WRITE(IOCODE,114)
      READ(20,69) (IHEDIN(L), L=1,80)
      DO L=1,NTE
         READ(20,*) PGT(L),MUGT(L),BGT(L),CRT(L)
         WRITE(IOCODE,24) PGT(L),MUGT(L),BGT(L),CRT(L)
         IF (PGT(L).GE.PMAXT) GO TO 40
      END DO
 40   MPGT = L
C ------------------------------------------------------------------
C          WATER AND GAS DENSITIES AT STOCK TANK CONDITIONS.
C ------------------------------------------------------------------
      WRITE(IOCODE,115)
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) RHOSCW,RHOSCG
      WRITE(IOCODE,401) RHOSCW,RHOSCG
C ------------------------------------------------------------------
C          CALCULATE SLOPES dBW/dP, dRSW/dP, AND dBG/dP.
C ------------------------------------------------------------------
      WRITE(IOCODE,222)
      DO L=2,MPWT
         BWPT(L)=(BWT(L)-BWT(L-1))/(PWT(L)-PWT(L-1))
         RSWPT(L)=(RSWT(L)-RSWT(L-1))/(PWT(L)-PWT(L-1))
         WRITE(IOCODE,224) PWT(L),BWT(L),BWPT(L),RSWT(L),RSWPT(L)
      END DO

      WRITE(IOCODE,444)
      DO L=2,MPGT
         BGPT(L)=(BGT(L)-BGT(L-1))/(PGT(L)-PGT(L-1))
         WRITE(IOCODE,445) PGT(L),BGT(L),BGPT(L)
      END DO


 21   FORMAT(20X,4F10.4)
 24   FORMAT(18X,F10.1,1X,F8.4,2X,E10.4,2X,E10.3)
 31   FORMAT(27X,'PMAXT'T37,'=',1X,F10.2,1X,'PSIA.'////)
 69   FORMAT(80A1)
 111  FORMAT(5(/),15X,'RELATIVE PERMEABILITY AND CAPILLARY',
     $       ' PRESSURE TABLE'/15X,50('_')///20X,'      SAT',
     $       '       KRW        KRG        PCGW'/20X,'      ---',
     $       '       ---        ----'/)
 112  FORMAT(////20X,'MAXIMUM PRESSURE ENTRY IN ALL PVT TABLES',
     $       /20X,40('_')//)
 113  FORMAT(30X,'WATER PVT TABLE'/30X,15('_')///25X,
     $       'P           VISW       BW         RSW'/25X,
     $       '-           ----       --         ---'/)
 114  FORMAT(///20X,'GAS PVT TABLE AND ROCK COMPRESSBILITY DATA',
     $      /20X,43('-')//26X,'P      VISG          BG           CR',
     $       /26X,'-      ----       --           --'/)
 115  FORMAT(6(/),20X,'WATER AND GAS DENSITIES AT STOCK',
     $       'TANK CONDITIONS'/20X,48('-')//)
 222  FORMAT(//25X,'SLOPE FOR COMPRESSIBILITY CALCULATION'/25X,
     $       36('_')///20X,'FOR WATER:'//23X,'P',7X,'BW',7X,
     $       'DBW/DP',8X,'RSW',
     $       4X,'DRSW/DP'/23X,'-',7X,'--',7X,'------',8X,'---',
     $       4X,'-------'/)
 224  FORMAT(20X,F7.1,F8.4,3X,E11.4,F8.1,2X,E11.4)
 401  FORMAT(30X,'RHOSCW',T40,':',1X,F10.4,1X,'LB/CU FT'/30X,
     $       'RHOSCG',T40,':',1X,F10.4,1X,'LB/CU FT'//)
 444  FORMAT(//20X,'FOR GAS:'//25X,'P',13X,'BG',12X,'DBG/DP'/25X,
     $       '-',13X,'--',12X,'------'/)
 445  FORMAT(20X,F9.1,1X,2E15.4)

      RETURN
      END

C
C ==================================================================
      SUBROUTINE TRAN1 (II,JJ,KK,KTR)
C ==================================================================

C     THE SUBROUTINE CALCULATES THE COEFFICIENTS OF TRANSMISSI-
C     BILITY IN THE X,Y, AND Z DIRECTION BLOCKS IN THE MODEL
C     GRID.  THE PROGRAM COULD MODIFY TO GRID TRANSMISSIBILITY
C     IF NECEESSARY.  THE CONVENTION USED IN SPECIFYING TRANSMI-
C     SSIBILITY MODIFICATION IS IN SUCH A WAY, FOR EXAMPLE,
C     IN GRID BLOCK(I,J,K) TX(I,J,K) REFERS TO FLOW ACROSS THE
C     BOUNDARY BETWEEN BLOCKS (I-1) AND (I).
```

```
C       NOMEMCLATURE:
C
C           A1(I,J,K)   :  KX(I,J,K) * DY(I,J,K) * DZ(I,J,K)
C           A2(I,J,K)   :  KY(I,J,K) * DX(I,J,K) * DZ(I,J,K)
C           A3(I,J,K)   :  KZ(I,J,K) * DX(I,J,K) * DY(I,J,K)
C           DX(I,J,K)   :  VARIABLE X-DIRECTION DIMENSION OF BLOCK
C                          IN THE MODEL GRID, FT.
C           DY(I,J,K)   :  VARIABLE Y-DIRECTION DIMENSION OF BLOCK
C                          IN THE MODEL GRID, FT.
C           DZ(I,J,K)   :  VARIABLE Z-DIRECTION DIMENSION OF BLOCK
C                          IN THE MODEL GRID, FT.
C           IHEDIN(L)   :  STORAGE MEMORY FOR INPUT TITLE CARD.
C           ITCODE      :  PRINT CODE FOR MODIFICATION TO TRANSMI-
C                          SSIBILITY DISTRIBUTIONS.
C                          ITCODE = 0 MEANS 'DO NOT PRINT', AND
C                          ITCODE = 1 MEANS 'PRINT'.
C           I           :  X-COORDINATE OF BLOCK TO BE MODIFIED.
C           II          :  NUMBER OF BLOCKS IN THE X-DIRECTION IN
C                          THE MODEL GRID.
C           J           :  Y-COORDINATE OF BLOCK TO BE MODIFIED.
C           JJ          :  NUMBER OF BLOCKS IN THE Y-DIRECTION IN
C                          THE MODEL GRID.
C           K           :  Z-COORDINATE OF BLOCK TO BE MODIFIED.
C           KK          :  NUMBER OF BLOCKS IN THE Z-DIRECTION IN
C                          THE MODEL GRID.
C           KX(I,J,K)   :  VARIABLE X-DIRECTION PERMEABILITY OF
C                          BLOCK IN THE MODEL GRID IN MILLIDARCIES.
C           KY(I,J,K)   :  VARIABLE Y-DIRECTION PERMEABILITY OF
C                          BLOCK IN THE MODEL GRID IN MILLIDARCIES.
C           KZ(I,J,K)   :  VARIABLE Z-DIRECTION PERMEABILITY OF
C                          BLOCK IN THE MODEL GRID IN MILLIDARCIES.
C           KTR         :  TRANSMISSIBILITY DEBUG OUTPUT CONTROL.
C                          KTR = 0 MEANS 'DO NOT PRINT', AND
C                          KTR = 1 MEANS 'PRINT'.
C           NUMTX       :  NUMBER OF GRID BLOCKS WHERE TX IS TO BE
C                          CHANGED.
C           NUMTY       :  NUMBER OF GRID BLOCKS WHERE TY IS TO BE
C                          CHANGED.
C           NUMTZ       :  NUMBER OF GRID BLOCKS WHERE TZ IS TO BE
C                          CHANGED.
C           TX(I,J,K)   :  VARIABLE COEFFICIENT OF X-DIRECTION TRAN-
C                          SMISSIBILITY.
C                          TX IS CALCULATED AS 0.012656* A1(I-1,J,K)*
C                          A1(I,J,K)/(DX(I-1,J,K)* A1(I,J,K) +
C                          DX(I,J,K)* A1(I-1,J,K))
C           TY(I,J,K)   :  VARIABLE COEFFICIENT OF Y-DIRECTION TRAN-
C                          SMISSIBILITY.
C                          TY IS CALCULATED AS 0.012656* A2(I,J-1,K)*
C                          A2(I,J,K)/(DY(I,J-1,K)* A2(I,J,K) +
C                          DY(I,J,K)* A2(I,J-1,K)).
C           TZ(I,J,K)   :  VARIABLE COEFFICIENT OF Z-DIRECTION TRAN-
C                          SMISSIBILITY.
C                          TZ IS CALCULATED AS 0.012656* A3(I,J,K-1)*
C                          A3(I,J,K)/(DZ(I,J,K-1)* A3(I,J,K) +
C                          DZ(I,J,K)* A3(I,J,K-1)).
C ---------------------------------------------------------------------

      INCLUDE 'COMGWSIM.FOR'

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

C         CALCULATE THE COEFFICIENTS OF TRANSMISSIBILITY FOR
C         BLOCKS IN THE MODEL GRID.

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

      DO 30 K=1,KK
      DO 30 J=1,JJ
      DO 30 I=1,II

         A1(I,J,K) = KX(I,J,K)*DY(I,J,K)*DZ(I,J,K)
         A2(I,J,K) = KY(I,J,K)*DX(I,J,K)*DZ(I,J,K)
         A3(I,J,K) = KZ(I,J,K)*DX(I,J,K)*DY(I,J,K)
30    CONTINUE

      IF (II .GT. 1) THEN
         DO 50 K=1,KK
         DO 50 J=1,JJ
         DO 50 I=1,II
         IF (KX(I-1,J,K) .GT. 0.0001 .AND. KX(I,J,K) .GT.
     $   0.0001) THEN
             TX(I,J,K) = 0.012656*A1(I-1,J,K)*A1(I,J,K)/
     $                   (DX(I-1,J,K)*A1(I,J,K)+DX(I,J,K)*
     $                    A1(I-1,J,K))
         END IF
50       CONTINUE
      END IF
```

```
      IF (JJ .GT. 1) THEN
            DO 55 K=1,KK
            DO 55 J=1,JJ
            DO 55 I=1,II
            IF (KY(I,J-1,K) .GT. 0.0001 .AND. KY(I,J,K) .GT.
     $      0.0001) THEN
                  TY(I,J,K) = 0.012656*A2(I,J-1,K)*A2(I,J,K)/
     $                        (DY(I,J-1,K)*A2(I,J,K)+DY(I,J,K)*
     $                         A2(I,J-1,K))
            END IF
55          CONTINUE
      END IF

      IF (KK .GT. 1) THEN
            DO 60 K=1,KK
            DO 60 J=1,JJ
            DO 60 I=1,II
            IF (KZ(I,J,K-1) .GT. 0.0001 .AND. KZ(I,J,K) .GT.
     $      0.0001) THEN
                  TZ(I,J,K) = 0.012656*A3(I,J,K-1)*A3(I,J,K)/
     $                        (DZ(I,J,K-1)*A3(I,J,K)+DZ(I,J,K)*
     $                         A3(I,J,K-1))
            END IF
60          CONTINUE
      END IF

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

C        ESTABLISH TRANSMISSIBILITY MODIFICATIONS.
C        READ NUMBER OF BLOCKS WHERE TRANSMISSIBILITY VALUES
C        ARE TO BE CHANGED, AND INPUT PRINT CODE.

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

      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) NUMTX, NUMTY, NUMTZ, ITCODE

      IF (NUMTX .GT. 0) THEN
         WRITE(IOCODE,31)
         DO L=1,NUMTX
           READ(20,*) I,J,K,TX(I,J,K)
           WRITE(IOCODE,32) I,J,K,TX(I,J,K)
         END DO
         IF (ITCODE .EQ. 1) THEN
            WRITE(IOCODE,44)
            DO 853 K=1,KK
              WRITE(IOCODE,38) K
            DO 853 J=1,JJ
              WRITE(IOCODE,72) (TX(I,J,K), I=1,II)
853         CONTINUE
         END IF

      IF (NUMTY .GT. 0) THEN
         WRITE(IOCODE,34)
         DO L=1,NUMTY
           READ(20,*) I,J,K,TY(I,J,K)
           WRITE(IOCODE,32) I,J,K,TY(I,J,K)
         END DO
         IF (ITCODE .EQ. 1) THEN
            WRITE(IOCODE,47)
            DO 855 K=1,KK
              WRITE(IOCODE,38) K
            DO 855 J=1,JJ
              WRITE(IOCODE,72) (TY(I,J,K), I=1,II)
855         CONTINUE
         END IF
      END IF

      IF (NUMTZ .GT. 0) THEN
         WRITE(IOCODE,37)
         DO L=1,NUMTZ
           READ(20,*) I,J,K,TZ(I,J,K)
           WRITE(IOCODE,32) I,J,K,TZ(I,J,K)
         END DO
         IF (ITCODE .EQ. 1) THEN
            WRITE(IOCODE,48)
            DO 857 K=1,KK
              WRITE(IOCODE,38) K
            DO 857 J=1,JJ
              WRITE(IOCODE,72) (TZ(I,J,K), I=1,II)
857         CONTINUE
         END IF
      END IF
```

```
      IF (KTR .EQ. 1) THEN
         WRITE(IOCODE,43)
         DO 42 K=1,KK
         DO 42 J=1,JJ
         DO 42 I=1,II
           WRITE(IOCODE,41) I,J,K,A1(I,J,K),A2(I,J,K),A3(I,J,K)
42       CONTINUE
         WRITE(IOCODE,49)
         DO 505 K=1,KK
         DO 505 J=1,JJ
         DO 505 I=1,II
           WRITE(IOCODE,41) I,J,K,TX(I,J,K),TY(I,J,K),TZ(I,J,K)
505      CONTINUE
      END IF

31    FORMAT(2X,'TRANSMISSIBILITY VALUES =',/2X,25('-')//5X,
     $       'TRANSMISSIBILITY (TX) NODE MODIFICATION',/5X,
     $       '     I     J     K              NEW TX VALUE')
32    FORMAT(2X,3I5,5X,F14.4)
34    FORMAT(5X,'TRANSMISSIBILITY (TY) NODE MODIFICATION',/5X,
     $       '     I     J     K              NEW TY VALUE')
37    FORMAT(5X,'TRANSMISSIBILITY (TZ) NODE MODIFICATION',/5X,
     $       '     I     J     K              NEW TZ VALUE')
38    FORMAT(2X,'LAYER(',I2,')',/)
41    FORMAT(2X,3I5,3F15.3)
43    FORMAT(2X,'     I     J     K          A1              A2',
     $       '              A3',/)
44    FORMAT(5X,'MODIFIED TRANSMISSIBILITY (TX) DISTRIBUTION',/)
47    FORMAT(5X,'MODIFIED TRANSMISSIBILITY (TY) DISTRIBUTION',/)
48    FORMAT(5X,'MODIFIED TRANSMISSIBILITY (TZ) DISTRIBUTION',/)
49    FORMAT(2X,'     I     J     K          TX              TY',
     $       '              TZ',/)
69    FORMAT(80A1)
72    FORMAT(2X,20F6.1)

      RETURN
      END


C
C ================================================================

      SUBROUTINE UINIT1 (KPI,II,JJ,KK,CUMPU,MBEW,CUMPO,MBEO,
     $                   SWC,SWC,PSWC,CUMIW,CUMIO)

C ================================================================

C     THE SUBROUTINE CALLS FOR TWO OPTIONS OF PRESSURE AND SAT-
C     URATION INITIALIZATION. THE INITIAL PRESSURE DISTRIBUTION
C     CAN EITHER BE CALCULATED BY THE PROGRAM FOR EQUILIBRIUM
C     CONDITIONS GIVEN THE LOCATION OF THE GAS/WATER CONTACT AND
C     THE PRESSURE AT THE CONTACT, OR THE INITIAL PRESSURE
C     DISTRIBUTION CAN BE READ ON A BLOCK-BY-BLOCK BASIS. THE
C     SATURATIONS (SW,SG) CAN EITHER BE READ AS CONSTANT VALUES
C     OVER THE ENTIRE GRID, OR THE ENTIRE SW AND SG DISTRIBUTIONS
C     ARE READ ON A BLOCK-BY-BLOCK BASIS.
```

```
C       NOMENCLATURE:

C           BBG             :   GAS FORMATION VOLUME FACTOR AT PRESSURE,
C                               PGUC IN RCF/SCF.
C           BBW             :   WATER FORMATION VOLUME FACTOR AT PRESSURE,
C                               PGUC IN RB/STB.
C           BGT             :   VARIABLE GAS FORMATION VOLUME FACTOR AT
C                               PRESSURE, PGT IN PVT DATA TABLE.
C           BWT             :   VARIABLE WATER FORMATION VOLUME FACTOR AT
C                               PRESSURE, PWT IN PVT DATA TABLE.
C           CUMIW           :   CUMMULATIVE WATER INJECTION, STB.
C           CUMIG           :   CUMMULATIVE GAS INJECTION, MSCF.
C           CUMPW           :   CUMMULATIVE WATER PRODUCTION, STB.
C           CUMPG           :   CUMMULATIVE GAS PRODUCTION, MSCF.
C           EL(I,J,K)       :   VARIABLE NODE MIDPOINT ELEVATION (FT) OF
C                               ALL BLOCKS IN THE MODEL GRID.
C           GUC             :   GAS/WATER CONTACT, FT.
C           IHEDINCL        :   STORAGE MEMORY FOR INPUT TITLE CARD.
C           II              :   NUMBER OF BLOCKS IN THE X-DIRECTION IN
C                               THE MODEL GRID.
C           JJ              :   NUMBER OF BLOCKS IN THE Y-DIRECTION IN
C                               THE MODEL GRID.
C           KK              :   NUMBER OF BLOCKS IN THE Z-DIRECTION IN
C                               THE MODEL GRID.
C           KPI             :   PRESSURE INITIALIZATION CODE.
C                               KPI = 0 MEANS 'USE EQUILIBRIUM PRESSURE
C                               INITIALIZATION'. INPUT REQUIRED WILL BE
C                               PGUC, PRESSURE AT GAS/WATER CONTACT
C                               KPI = 1 MEANS 'USE NON-EQUILIBRIUM PRESS-
C                               URE INITIALIZATION'. PRESSURES, PN(I,J,K)
C                               FOR EACH BLOCK MUST BE READ ON A BLOCK-
C                               BY-BLOCK BASIS. PN(I,J,K) FOR ITH,JTH,KTH
C                               VALUES MUST BE READ.
C           KSI             :   SATURATION INITIALIZATION CODE.
C                               KSI = 0 MEANS 'INITIAL WATER AND GAS
C                               SATURATIONS ARE CONSTANT OVER THE ENTIRE
C                               MODEL GRID'. SWC MUST BE READ.
C                               KSI = 1 MEANS 'WATER SATURATION MUST BE
C                               READ FOR EACH GRID BLOCK ON A BLOCK-BY-
C                               BLOCK BASIS'.
C           MBEG            :   GAS MATERIAL BALANCE, PER CENT.
C           MBEW            :   WATER MATERIAL BALANCE, PER CENT.
C           MPGT            :   MAXIMUM NUMBER OF ENTRY VALUE FOR GAS
C                               PHASE PRESSURE, PSIA.
C           MPWT            :   MAXIMUM NUMBER OF ENTRY VALUE FOR WATER
C                               PHASE PRESSURE, PSIA.
C           NTE             :   MAXIMUM NUMBER OF DATA ENTRY ALLOWED IN
C                               ALL PVT TABLES.
C           PGUC            :   PRESSURE AT GAS/WATER CONTACT, PSIA.
C           PGT             :   VARIABLE GAS PHASE PRESSURE IN PVT
C                               TABLE, PSIA.
C           PWT             :   VARIABLE WATER PHASE PRESSURE IN PVT
C                               TABLE, PSIA.
C           RHOG            :   GAS DENSITY AT PRESSURE P(I,J,K).
C           RHOSCG          :   GAS DENSITY AT STANDARD CONDITION.
C           RHOW            :   WATER DENSITY AT PRESSURE P(I,J,K).
C           RHOSCW          :   WATER DENSITY AT STANDARD CONDITION.
C           RSW             :   GAS IN WATER SOLUTION RATIO AT PRESSURE,
C                               PGWC.
C           RSWT            :   GAS IN WATER SOLUTION RATIO AT PGT IN ALL
C                               PVT TABLES.
C           SG(I,J,K)       :   GAS SATURATION ARRAY.
C           SGN(I,J,K)      :   INITIAL GAS PHASE SATURATION ARRAY.
C           SW(I,J,K)       :   WATER SATURATION ARRAY.
C           SWN(I,J,K)      :   INITIAL WATER PHASE SATURATION ARRAY.
C           SWC             :   CONNATE WATER SATURATION.

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

      INCLUDE 'COMGWSIM.FOR'

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

```
      CUMPW  = 0.0
      CUMPG  = 0.0
      CUMIW  = 0.0
      CUMIG  = 0.0
      MBEW   = 0.0
      MBEG   = 0.0
C -----------------------------------------------------------------------
C
C         READ CODES FOR CONTROLLING PRESSURE AND SATURATION
C         INITIALIZATION.
C
C -----------------------------------------------------------------------
      READ(20,69) (IHEDIN(L), L=1,80)
      READ(20,*) KPI,KSI

      IF (KPI .EQ. 0) THEN
         READ(20,*) PGWC,GUC
         DO 200 K=1,KK
         DO 200 J=1,JJ
         DO 200 I=1,II
            IF (CEL(I,J,K) .LT. GWC) THEN
               CALL INTERP (NTE,PGT,BGT,MPGT,PGUC,BBG)
               RHOG = RHOSCG/BBG
               PN(I,J,K) = PGWC+RHOG*(CEL(I,J,K)-GUC)/144.
            ELSE
               CALL INTERP (NTE,PUT,BUT,MPUT,PGUC,BUO)
               CALL INTERP (NTE,PUT,RSUT,MPUT,PGUC,RSO)
               RHOW = (RHOSCW+RSW*RHOSCG)/BBW
               PN(I,J,K) = PGUC+RHOW*(CEL(I,J,K)-GUC)/144.
            END IF
  200    CONTINUE
      ELSE
         DO 3009 K=1,KK
         DO 3009 J=1,JJ
            READ(20,*) (PN(I,J,K), I=1,II)
 3009    CONTINUE
      END IF

      DO 3012 I=1,II
      DO 3012 J=1,JJ
      DO 3012 K=1,KK
         P(I,J,K) = PN(I,J,K)
 3012 CONTINUE
C -----------------------------------------------------------------------
C
C         INITIALIZE SATURATION ARRAY.
C -----------------------------------------------------------------------
      IF (KSI .EQ. 0) THEN
         READ(20,*) SWC
         DO 30 K=1,KK
         DO 30 J=1,JJ
         DO 30 I=1,II
            IF (CEL(I,J,K) .GT. GUC) THEN
               SUN(I,J,K) = 1.0
               SGN(I,J,K) = 0.0
               SW(I,J,K) = SUN(I,J,K)
               SG(I,J,K) = SGN(I,J,K)
            ELSE
               SUN(I,J,K) = SWC
               SGN(I,J,K) = 1.0-SUN(I,J,K)
               SW(I,J,K) = SUN(I,J,K)
               SG(I,J,K) = SGN(I,J,K)
            END IF
            IF (SG(I,J,K) .LT. 0.0)  SG(I,J,K)=0.0
   30    CONTINUE
      ELSE
         DO 3020 K=1,KK
         DO 3020 J=1,JJ
            READ(20,*) (SUN(I,J,K), I=1,II)
 3020    CONTINUE

         DO 3030 K=1,KK
         DO 3030 J=1,JJ
         DO 3030 I=1,II
            SGN(I,J,K) = 1.0-SUN(I,J,K)
            IF (SGN(I,J,K) .LT. 0.0) SGN(I,J,K)=0.0
            SW(I,J,K) = SUN(I,J,K)
            SG(I,J,K) = SGN(I,J,K)
 3030    CONTINUE
      END IF

   69 FORMAT(80A1)

      RETURN
      END
C ------------------------------------------------------------------- END.
```

Program GOWSIM

```
      INTEGER NX,NY,NZ,NXP,NYP,NZP,NTE,NW,NPMAX
C THIS VERSION ALLOWS ALL RE-DIMENSIONING TO BE DONE
C SIMPLY BY CHANGING THE SUBSEQUENT PARAMETERS AS DESIRED.
      PARAMETER (NX=40,NY=5,NZ=5,NXP=41,NYP=6,NZP=6,NTE=25,
     $NW=40,NPMAX=40)
      COMMON /COEFF/AW(NX,NY,NZ),AE(NX,NY,NZ),AN(NX,NY,NZ),
     $AS(NX,NY,NZ),AT(NX,NY,NZ),AB(NX,NY,NZ),E(NX,NY,NZ),B(NX,NY,NZ)
      COMMON /SAT/SWN(NX,NY,NZ),SGN(NX,NY,NZ),SON(NX,NY,NZ),
     $SW1(NX,NY,NZ),SG1(NX,NY,NZ),SO1(NX,NY,NZ),A1(NX,NY,NZ),
     $A2(NX,NY,NZ),A3(NX,NY,NZ),SUM(NX,NY,NZ),
     $GAM(NX,NY,NZ),QS(NX,NY,NZ)
      COMMON /PARM/KX(NX,NY,NZ),KY(NX,NY,NZ),KZ(NX,NY,NZ),
     $TX(NXP,NY,NZ),TY(NX,NYP,NZ),TZ(NX,NY,NZP)
      COMMON /FACT/WW(NXP,NY,NZ),WE(NXP,NY,NZ),WS(NX,NYP,NZ),
     $WN(NX,NYP,NZ),WT(NX,NY,NZP),WB(NX,NY,NZP),OW(NXP,NY,NZ),
     $OE(NXP,NY,NZ),ON(NX,NYP,NZ),OS(NX,NYP,NZ),OT(NX,NY,NZP),
     $OB(NX,NY,NZP)
      COMMON /RPRT/P(NX,NY,NZ),PN(NX,NY,NZ),SW(NX,NY,NZ),
     $SG(NX,NY,NZ),SO(NX,NY,NZ),PCGOT(NTE),PCOWT(NTE),PWT(NTE),
     $MUWT(NTE),BWT(NTE),BWPT(NTE),RSWT(NTE),RSWPT(NTE),PGT(NTE),
     $MUGT(NTE),BGT(NTE),BGPT(NTE),POT(NTE),MUOT(NTE),BOT(NTE),
     $BOPT(NTE),RSOT(NTE),RSOPT(NTE),CRT(NTE),SAT(NTE),KRWT(NTE),
     $KRGT(NTE),KROT(NTE)
      COMMON /RATES/PID(NW,NZ),PWF(NW,NZ),PWFC(NW,NZ),KIP(NW),
     $GMW(NW,NZ),GMG(NW,NZ),GMO(NW,NZ),WELLID(NW),LAYER(NW),
     $QVW(NW),QVG(NW),QVO(NW),QVT(NW),CUMW(NW,NZ),CUMG(NW,NZ),
     $CUMO(NW,NZ)
      COMMON /SOLN/GWWT(NX,NY,NZ),GGWT(NX,NY,NZ),GOWT(NX,NY,NZ),
     $QOWG(NX,NY,NZ),QO(NX,NY,NZ),QW(NX,NY,NZ),QG(NX,NY,NZ)
      COMMON /POSD/DX(NX,NY,NZ),DY(NX,NY,NZ),DZ(NX,NY,NZ),
     $IQN1(NW),IQN2(NW),IQN3(NW),IHEDIN(80),EL(NX,NY,NZ)
      COMMON /PHASE/PBOT(NX,NY,NZ),BW(NX,NY,NZ),BG(NX,NY,NZ),
     $BO(NX,NY,NZ),VP(NX,NY,NZ),CT(NX,NY,NZ),PBO,VSLOPE,
     $BSLOPE,RSLOPE,PMAXT,IREPRS,RHOSCO,RHOSCW,RHOSCG,MSAT,MPOT,
     $MPWT,MPGT,IOCODE
C
C
      CHARACTER IHEDIN
      CHARACTER*5 WELLID
      REAL KROT,KRWT,KRGT,MUWT,MUOT,MUGT,KX,KY,KZ,
     $   MUO,MUW,MUG,KRO,KRG,KRW,MBEWI,MBEGI,MCFGI,
     $   MBEO,MBEW,MBEG,MCFG,MBEOI,MIN,MCFG1,MCFGT
```

```
C
CC
CC ==========================================================
C
      PROGRAM GOWSIM
C
CC ==========================================================
C
      INCLUDE 'COMMB3.FOR'
C
      DIMENSION  OOIP(NZ),OWIP(NZ),ODGIP(NZ),OFGIP(NZ)
C
CC    .... INITIALIZE SOME USEFUL ARRAYS
C
      DATA COP,CWP,CGP,CWI,CGI/0.0,0.0,0.0,0.0,0.0/
      DATA ETI,FT,FTMAX,PBO,KTR/0.0,0.0,0.0,0.0,0.0/
      DATA SCFO,SCFW,SCFG,SCFG1,KRG/0.0,0.0,0.0,0.0,0.0/
          DATA MCFG1,MCFGT,PAVG,PAVGO,MCFG / 0.0,0.0,0.0,0.0,0.0/
          DATA MCFGI,MUO,MUW,MUG,KRO,KRW /0.0,0.0,0.0,0.0,0.0,0.0/
          DATA MBEWI,MBEGI,MBEO,MBEW,MBEG /0.0,0.0,0.0,0.0,0.0/
          DATA MBEOI,MIN /0.0,0.0/
C
      DO 1016 I = 1,NW
          DO 1017 J = 1,NZ
              CUMO(I,J)=0.0
              CUMW(I,J)=0.0
              CUMG(I,J)=0.0
              PWF(I,J)=0.0
              PWFC(I,J)=0.0
              GMO(I,J)=0.0
              GMW(I,J)=0.0
              GMG(I,J)=0.0
 1017     CONTINUE
 1016 CONTINUE
C
      DO 1014 I = 1,NW
              LAYER(I)=0
              KIP(I)=0
              QVO(I)=0.0
              QVW(I)=0.0
              QVG(I)=0.0
              QVT(I)=0.0
              IQN1(I)=0
              IQN2(I)=0
              IQN3(I)=0
 1014 CONTINUE
C
      DO 1021 I = 1,NTE
              SAT(I)=0.0
              KROT(I)=0.0
              KRWT(I)=0.0
              KRGT(I)=0.0
              PCOWT(I)=0.0
              PCGOT(I)=0.0
              POT(I)=0.0
              MUOT(I)=0.0
              BOT(I)=0.0
              BOPT(I)=0.0
              RSOT(I)=0.0
              RSOPT(I)=0.0
              PWT(I)=0.0
              MUWT(I)=0.0
              BWT(I)=0.0
              BWPT(I)=0.0
              RSWT(I)=0.0
              RSWPT(I)=0.0
              PGT(I)=0.0
              MUGT(I)=0.0
              BGT(I)=0.0
              BGPT(I)=0.0
              CRT(I)=0.0
 1021 CONTINUE
C
      DO 700 I = 1,IM
      DO 700 J = 1,JM
      DO 700 K = 1,KM
          TX(I,J,K)=0.0
          TY(I,J,K)=0.0
          TZ(I,J,K)=0.0
          AW(I,J,K)=0.0
          AS(I,J,K)=0.0
          AT(I,J,K)=0.0
          AE(I,J,K)=0.0
          AN(I,J,K)=0.0
          AB(I,J,K)=0.0
          OW(I,J,K)=0.0
          OE(I,J,K)=0.0
          WW(I,J,K)=0.0
          WE(I,J,K)=0.0
          OS(I,J,K)=0.0
          ON(I,J,K)=0.0
          WS(I,J,K)=0.0
          WN(I,J,K)=0.0
          OT(I,J,K)=0.
          OB(I,J,K)=0.
          WT(I,J,K)=0.
          WB(I,J,K)=0.
```

```
                QO(I,J,K)=0.0
                QW(I,J,K)=0.0
                QG(I,J,K)=0.0
                VP(I,J,K)=0.0
                CT(I,J,K)=0.0
                KX(I,J,K)=0.0
                KY(I,J,K)=0.0
                KZ(I,J,K)=0.0
                DX(I,J,K)=0.0
                DY(I,J,K)=0.0
                DZ(I,J,K)=0.0
                A1(I,J,K)=0.0
                A2(I,J,K)=0.0
                A3(I,J,K)=0.0
                P(I,J,K)=0.0
                SO(I,J,K)=0.0
                SW(I,J,K)=0.0
                SG(I,J,K)=0.0
                BO(I,J,K)=0.0
                BW(I,J,K)=0.0
                BG(I,J,K)=0.0
                E(I,J,K)=0.0
                B(I,J,K)=0.0
                PBOT(I,J,K)=0.0
                PN(I,J,K)=0.0
                SON(I,J,K)=0.0
                SWN(I,J,K)=0.0
                SGN(I,J,K)=0.0
                SO1(I,J,K)=0.0
                SW1(I,J,K)=0.0
                SG1(I,J,K)=0.0
                SUM(I,J,K)=0.0
                GAM(I,J,K)=0.0
                QS(I,J,K)=0.0
                GOWT(I,J,K)=0.0
                GWWT(I,J,K)=0.0
                GGWT(I,J,K)=0.0
                QOWG(I,J,K)=0.0
                EL(I,J,K)=0.0
C
        IF (I.EQ.NX) THEN
                TX(I+1,J,K)=0.0
            OW(I+1,J,K)=0.
            OE(I+1,J,K)=0.
            WW(I+1,J,K)=0.
            WE(I+1,J,K)=0.
        END IF
C
        IF (J.EQ.NY) THEN
                TY(I,J+1,K)=0.0
            OS(I,J+1,K)=0.
            ON(I,J+1,K)=0.
            WS(I,J+1,K)=0.
            WN(I,J+1,K)=0.
        END IF
C
        IF (K.EQ.NZ) THEN
                TZ(I,J,K+1)=0.0
            OT(I,J,K+1)=0.
            OB(I,J,K+1)=0.
            WT(I,J,K+1)=0.
            WB(I,J,K+1)=0.
        END IF
  700   CONTINUE
C
CC   ------------------------------------------------------------------
CC
CC          OPEN INPUT AND OUTPUT FILES
CC
CC   ------------------------------------------------------------------
C
            OPEN(20,FILE='BOAST.DAT',ACCESS='SEQUENTIAL',STATUS='OLD')
        WRITE(*,3001)
 3001   FORMAT(/,1X,'ENTER IOCODE . . . ')
        READ(*,3002) IOCODE
 3002   FORMAT(I4)
            OPEN(IOCODE,FILE='BOAST.RES',ACCESS='SEQUENTIAL',STATUS='NEW')
C
CC   ------------------------------------------------------------------
CC
CC          OUTPUT HEADER
CC
CC   ------------------------------------------------------------------
C
            WRITE(IOCODE,3005)
            WRITE(IOCODE,3007)
            WRITE(IOCODE,3006)
 3007   FORMAT(30X,'*',17X,'                    BOAST:  ',30X,'*',/30X,'*',
     $  15X,'BLACK OIL APPLIED SIMULATION TOOL',19X,'*',
     $  /30X,'*',16X,'U.N.S.W.-VERSION 2.0 (02-07-85)',20X,'*')
 3005   FORMAT(//30X,69('*'),/30X,'*',67X,'*',/30X,'*',67X,'*')
 3006   FORMAT(30X,'*',67X,'*'/30X,'*',67X,'*'/30X,69('*'),///)
C
```

```
C     ------------------------------------------------------------------
C
C           ESTABLISH RESERVOIR AND BLOCK DIMENSIONS
C
C     ------------------------------------------------------------------
C
      CALL GRID1(II,JJ,KK)
C     ------------------------------------------------------------------
C
C           ESTABLISH POROSITY AND PERMEABILITY AT EACH ZONE
C
C     ------------------------------------------------------------------
C
      CALL PARM1(II,JJ,KK)
C     ------------------------------------------------------------------
C
C           CALCULATE INTERBLOCK TRANSMISSIBILITIES
C
C     ------------------------------------------------------------------
C
      CALL TRAN1(II,JJ,KK,KTR)
C     ------------------------------------------------------------------
C
C           EMPIRICAL DATA
C
C     ------------------------------------------------------------------
C
      CALL TABLE
C     ------------------------------------------------------------------
C
C     .... INITIALIZE BUBBLE POINT PRESSURE ARRAY
C
      DO 8 I = 1,II
      DO 8 J = 1,JJ
      DO 8 K = 1,KK
         PBOT(I,J,K) = PBO
    8 CONTINUE
C     ------------------------------------------------------------------
C
C           ESTABLISH INITIAL CONDITIONS
C
C     ------------------------------------------------------------------
C
      CALL UINIT1(KPI,II,JJ,KK,PWOC,CUMPO,MBEO,
     $            CUMPW,MBEW,CUMPG,MBEG,SOI,SWI,SGI,
     $            WOC,GOC,PGOC,CUMIW,CUMIG)
C     ------------------------------------------------------------------
C
C           SOLUTION METHOD, DEBUG PRINT, AND TIME STEP CONTROL
C
C     ------------------------------------------------------------------
C
      CALL  CODES(IOCODE,KSM1,KSN1,KCO1,NN,
     $            FACT1,FACT2,TMAX,KSOL,MITER,OMEGA,TOL,TOL1,
     $            KSN,KSM,KCO,KTR,KCOFF,DSMAX,DPMAX,
     $            WORMAX,GORMAX,PAMIN,PAMAX)
C     ------------------------------------------------------------------
C
      D5615=1.0/5.615
      D288=1.0/288.
      D144=1.0/144.
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      NMAX=NN+1
      NITER=0
C
      DO 1000 N = 1,NMAX
C     ------------------------------------------------------------------
C
C           NLOOP DEFINED TO AVOID INDEX MODIFICATION WARNINGS IN
C           CALLS TO MATBAL, SOLMAT, CLSOR.
C
C     ------------------------------------------------------------------
C
      NLOOP=N
C     ------------------------------------------------------------------
C
C           RECURRENT DATA
C
C     ------------------------------------------------------------------
C
      IF (FT.LT.FTMAX) GO TO 46
      READ(20,*,END=1001) IWLCNG,ICHANG,IWLREP,ISUMRY,
     $                    IPMAP,ISOMAP,ISWMAP,ISGMAP,IPBMAP
C
      READ(20,*) DAY,DTMIN,DTMAX,HOUR,MIN,SEC
      IF (IWLCNG.EQ.0) GO TO 45
      CALL  NODES(NVQN,WRAD,SKIN)
   45 CONTINUE
```

```
C
      DELT=DAY+HOUR/24.+MIN/1440.+SEC/86400.
      FTMAX=ETI+ICHANG*DELT
C
 46   IF (N.EQ.1) DELTO=DELT
      IF (N.EQ.1) GO TO 1050
 1049 IF (DSMC.LT.DSMAX.AND.DPMC.LT.DPMAX.AND.NITER.LT.MITER) DELT=
     $   DELT*FACT1
      IF (DSMC.GT.DSMAX.OR.DPMC.GT.DPMAX.OR.NITER.GE.MITER) DELT=
     $   DELT*FACT2
      IF (DELT.LT.DTMIN) DELT=DTMIN
      IF (DELT.GT.DTMAX) DELT=DTMAX
      IF (ETI+DELT .GT. FTMAX) DELT=FTMAX-ETI
 1050 FT=ETI+DELT
      IF (ETI+DELT*0.5 .GE. TMAX) GO TO 1001
C
CC ---------------------------------------------------------------
CC
CC         ITFLAG COUNTS THE NUMBER OF TIME STEP REPETITIONS
CC
CC ---------------------------------------------------------------
C
      ITFLAG=0
C
CC ---------------------------------------------------------------
CC
CC         RE-ENTRY POINT FOR REPEATED TIME STEP
CC
CC ---------------------------------------------------------------
C
 1060 CONTINUE
C
      DIV1=1.0/DELT
      IF (N.GT.1 .OR. ITFLAG.GT.0) GO TO 105
      RESVOL=0.0
      SCFO=0.0
      SCFG=0.0
      SCFG1=0.0
C
      DO 102 K = 1,KK
         OOIP(K)=0.0
         OWIP(K)=0.0
         ODGIP(K)=0.
         OFGIP(K)=0.
C
      DO 100 J = 1,JJ
      DO 100 I = 1,II
         PP=P(I,J,K)
         BPT=PBOT(I,J,K)
         VP(I,J,K)=VP(I,J,K)*DX(I,J,K)*DY(I,J,K)*DZ(I,J,K)
         RESVOL=RESVOL+VP(I,J,K)
C
CC ---------------------------------------------------------------
CC
CC         NOTE WE ARE ASSUMING INITIAL PHI IS AT INITIAL RESERVOIR PR
CC
CC ---------------------------------------------------------------
C
         CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PP,RSO)
         CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
         CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PP,BO(I,J,K))
         CALL INTERP(NTE,PWT,BWT,MPWT,PP,BW(I,J,K))
         CALL INTERP(NTE,PGT,BGT,MPGT,PP,BG(I,J,K))
C
         FF1=SO(I,J,K)/BO(I,J,K)
         FF2=SW(I,J,K)/BW(I,J,K)
         SCFO=SCFO+VP(I,J,K)*FF1
         SCFW=SCFW+VP(I,J,K)*FF2
         SCFG=SCFG+VP(I,J,K)*SG(I,J,K)/BG(I,J,K)
         SCFG1=SCFG1+VP(I,J,K)*(RSO*FF1+RSW*FF2)
C
         CALL NTERP1(NTE,POT,BOPT,MPOT,PP,BODER)
         CALL NTERP1(NTE,POT,RSOPT,MPOT,PP,RSODER)
         CALL NTERP1(NTE,PWT,BWPT,MPWT,PP,BWDER)
         CALL NTERP1(NTE,PWT,RSWPT,MPWT,PP,RSWDER)
         CALL NTERP1(NTE,PGT,BGPT,MPGT,PP,BGDER)
C
         IF (PP.GT.PBOT(I,J,K)) BODER=BSLOPE
         IF (PP.GT.PBOT(I,J,K)) RSODER=RSLOPE
         CO=-(BODER-BG(I,J,K)*RSODER)/BO(I,J,K)
         CW=-(BWDER-BG(I,J,K)*RSWDER)/BW(I,J,K)
         CG=-BGDER/BG(I,J,K)
C
         CALL INTERP(NTE,PGT,CRT,MPGT,PP,CR)
C
CC ---------------------------------------------------------------
CC
CC         MODIFICATION OF TOTAL BLOCK COMPRESSIBILITY TO THE
CC         PREVIOUS VALUE IN TIME IF THE CALCULATED VALUE IS
CC
CC ---------------------------------------------------------------
C
         CX = CT(I,J,K)
         CT(I,J,K)=CR + CO*SO(I,J,K) +CW*SW(I,J,K) + CG*SG(I,J,K)
C
            IF (CT(I,J,K).LE.0.0) THEN
               CT(I,J,K)=CX
               WRITE(*,847) I,J,K
 847  FORMAT(' ','RESET TO PREVIOUS TOTAL COMP. AT POSN.',3I5)
         END IF
```

```
C
      OOIP(K)=OOIP(K)+D5615*.000001*SON(I,J,K)*VP(I,J,K)/BO(I,J,K)
      OWIP(K)=OWIP(K)+D5615*.000001*SWN(I,J,K)*VP(I,J,K)/BW(I,J,K)
      ODGIP(K)=ODGIP(K)+.001*.000001*(RSO*SON(I,J,K)*VP(I,J,K)/
     $  BO(I,J,K) + RSW*SWN(I,J,K)*VP(I,J,K)/BW(I,J,K))
      OFGIP(K)=OFGIP(K)+.001*.000001*SGN(I,J,K)*VP(I,J,K)/BG(I,J,K)
C
      IF (KCO1.EQ.0) GO TO 100
      WRITE(IOCODE,21) I,J,K,VP(I,J,K),CTC(I,J,K),BO(I,J,K),SO(I,J,K),
     $  BW(I,J,K),SW(I,J,K),BG(I,J,K),SG(I,J,K)
 100  CONTINUE
C
      WRITE(IOCODE,110) K,OOIP(K),OWIP(K),ODGIP(K),OFGIP(K)
 110  FORMAT(/,1X,'LAYER',I3,' INITIAL FLUID VOLUMES: ',
     $ /,10X,'OIL IN PLACE (MILLION STB)',20X,F10.4,
     $ /,10X,'WATER IN PLACE (MILLION STB)',19X,F10.4,
     $ /,10X,'SOLUTION GAS IN PLACE (BILLION SCF)',14X,F10.4,
     $ /,10X,'FREE GAS IN PLACE (BILLION SCF)',17X,F10.4/)
 102  CONTINUE
C
      TOOIP=0.
      TOWIP=0.0
      TODGIP=0.
      TOFGIP=0.
C
      DO 103 K = 1,KK
         TOOIP=TOOIP+OOIP(K)
         TOWIP=TOWIP+OWIP(K)
         TODGIP=TODGIP+ODGIP(K)
         TOFGIP=TOFGIP+OFGIP(K)
 103  CONTINUE
C
      WRITE(IOCODE,115) TOOIP,TOWIP,TODGIP,TOFGIP
 115  FORMAT(/,1X,'TOTAL INITIAL FLUID VOLUMES IN RESERVOIR:',
     $ /,10X,'OIL IN PLACE  (MILLION STB)',20X,F10.4,
     $ /,10X,'WATER IN PLACE (MILLION STB)',19X,F10.4,
     $ /,10X,'SOLUTION GAS IN PLACE (BILLION SCF)',14X,F10.4,
     $ /,10X,'FREE GAS IN PLACE (BILLION SCF)',17X,F10.4,/)
C
      STBO=SCFO*D5615
      STBW=SCFW*D5615
      MCFG=SCFG*0.001
        MCFG1=SCFG1*0.001
      STBOI=STBO
      STBWI=STBW
      MCFGI=MCFG+MCFG1
C
      IF (MCFGI.LE.1.E-7 .AND. MCFGT.LE.1.E-7) MBEG=0.0
 105  IF (N.EQ.1 .AND. ITFLAG.LE.0)
     $    CALL PRTPS(NLOOP,KPI,II,JJ,KK,PAVGO,PAVG,
     $    COP,CWP,CWI,CGP,CGI,MBEO,MBEW,MBEG,DELTO,
     $    OPR,WPR,GPR,WIR,GIR,ETI,CWOR,CGOR,WOR,GOR,
     $    IPMAP,ISOMAP,ISWMAP,ISGMAP,IPBMAP)
 501  CONTINUE
C
      IF (N.EQ.NMAX) GO TO 1001
C
CCC   ------------------------------------------------------------------
CCC
CCC         ESTABLISH RATES AND CALCULATE BHFP ( IF PID IS NONZERO)
CCC
CCC   ------------------------------------------------------------------
C
      IF (NVQN.EQ.0) GO TO 1160
      CALL QRATE(NVQN)
        WRITE(*,671)
 671  FORMAT(' ',' QRATE DONE')
 1160 CONTINUE
C
CCC   ------------------------------------------------------------------
CCC
CCC         CALCULATE SEVEN DIAGONAL MATRIX FOR PRESSURE SOLUTION
CCC
CCC   ------------------------------------------------------------------
C
      CALL SOLMAT(II,JJ,KK,DIV1,D288,D144,
     $            KSM,KSM1,NLOOP,NN,KCOFF)
        WRITE(*,672)
 672  FORMAT(' ',' SOLMAT DONE')
C
```

```
C  ------------------------------------------------------------------
CCC
CC          MODIFY MATRIX ELEMENTS FOR WELLS UNDER IMPLICIT CONTROL
CCC
CC  ------------------------------------------------------------------
C
      IF (NVQN.EQ.0) GO TO 1170
      CALL PRATEI(NVQN)
           WRITE(*,673)
 673  FORMAT(' ','PRATEI DONE')
 1170 CONTINUE
C
CC  ------------------------------------------------------------------
CCC
CC          CALCULATE NEW PRESSURE DISTRIBUTION
CCC
C   ------------------------------------------------------------------
      WRITE(*,3336) FT
 3336 FORMAT(' ','ELAPSED TIME = ',F6.2,
     $       'DAYS FROM BEGINNING OF SIMULATION')
           WRITE(*,442) DELT
 442  FORMAT(' ','DELT=',F8.3)
C
      IF (KSOL.EQ.2) CALL CLSOR(II,JJ,KK,OMEGA,TOL,TOL1,MITER,
     $    DELT,DELTO,KSN,NLOOP,NITER)
C
CC  ------------------------------------------------------------------
CCC
CC          CALCULATE IMPLICIT RATES
CCC
CC  ------------------------------------------------------------------
C
      IF (NVQN.EQ.0) GO TO 2051
      CALL PRATEO(NVQN)
 2051 CONTINUE
C
CC  ------------------------------------------------------------------
CCC
CC          CALCULATE NEW FLUID SATURATIONS
CCC
CC  ------------------------------------------------------------------
CCC
C
      SCFO=0.0
      SCFW=0.0
      SCFG=0.0
      SCFG1=0.0
      RESVOL=0.0
C
      DO 400 K = 1,KK
      DO 400 J = 1,JJ
      DO 400 I = 1,II
      PPN=PN(I,J,K)
      PP=P(I,J,K)
      BPT=PBOT(I,J,K)
C
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PP,RSO)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
      CALL INTERP(NTE,PGT,CRT,MPGT,PPN,CR)
      CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PP,BBO)
      CALL INTERP(NTE,PWT,BWT,MPWT,PP,BBW)
      CALL INTERP(NTE,PGT,BGT,MPGT,PP,BBG)
C
      VPP=VP(I,J,K) * (1.0+CR*(P(I,J,K)-PPN))
      RESVOL=RESVOL+VPP
C
      DP1=0.0
      DP2=0.0
      DP3=0.0
      DP4=0.0
      DP5=0.0
      DP6=0.0
      IF ((I-1).GT.0)  DP1=P(I-1,J,K)-PP
      IF ((I+1).LE.II) DP2=P(I+1,J,K)-PP
      IF ((J-1).GT.0)  DP3=P(I,J-1,K)-PP
      IF ((J+1).LE.JJ) DP4=P(I,J+1,K)-PP
      IF ((K-1).GT.0)  DP5=P(I,J,K-1)-PP
      IF ((K+1).LE.KK) DP6=P(I,J,K+1)-PP
C
      DAODP=OW(I,J,K)*DP1+OE(I,J,K)*DP2+OS(I,J,K)*DP3
     $      +ON(I,J,K)*DP4+OT(I,J,K)*DP5+OB(I,J,K)*DP6
      DAWDP=WW(I,J,K)*DP1+WE(I,J,K)*DP2+WS(I,J,K)*DP3
     $      +WN(I,J,K)*DP4+WT(I,J,K)*DP5+WB(I,J,K)*DP6
      SW(I,J,K)=((DAWDP+GWWT(I,J,K)-QW(I,J,K))*DELT +VP(I,J,K)*
     $          SWN(I,J,K)/BW(I,J,K)) * BBW/VPP
      SO(I,J,K)=((DAODP+GOWT(I,J,K)-QO(I,J,K))*DELT +VP(I,J,K)*
     $          SON(I,J,K)/BO(I,J,K)) * BBO/VPP
      SG(I,J,K)=1.0-SO(I,J,K)-SW(I,J,K)
      IF (SG(I,J,K).GT.0.0) GO TO 405
 401  SG(I,J,K)=0.0
      SO(I,J,K)=1.0-SW(I,J,K)
 405  CONTINUE
C
      IF (KCOFF.EQ.0) GO TO 397
      RHO1=VPP*SO(I,J,K)/BBO
      RHO2=VP(I,J,K)*SON(I,J,K)/BO(I,J,K)
      DIFFO=RHO1-RHO2
C
      RHW1=VPP*SW(I,J,K)/BBW
      RHW2=VP(I,J,K)*SWN(I,J,K)/BW(I,J,K)
      DIFFW=RHW1-RHW2
```

```
C
      RHG1=VPP*SG(I,J,K)/BBG
      RHG2=VP(I,J,K)*SGN(I,J,K)/BG(I,J,K)
      DIFFG=RHG1-RHG2
C
      WRITE(IOCODE,33)
      WRITE(IOCODE,21)I,J,K,P(I,J,K),SO(I,J,K),SON(I,J,K),SW(I,J,K),
     $                  SWN(I,J,K),SG(I,J,K),SGN(I,J,K),VPP
      WRITE(IOCODE,21)I,J,K,GOWT(I,J,K),QO(I,J,K),GWWT(I,J,K),
     $                  QW(I,J,K)
      WRITE(IOCODE,21)I,J,K,DAODP,DAWDP,DELT
      WRITE(IOCODE,21)I,J,K,RHO1,RHO2,DIFFO
      WRITE(IOCODE,21)I,J,K,RHW1,RHW2,DIFFW
      WRITE(IOCODE,21)I,J,K,RHG1,RHG2,DIFFG,GGWT(I,J,K),QG(I,J,K)
C
  397 VP(I,J,K)=VPP
      BO(I,J,K)=BBO
      BW(I,J,K)=BBW
      BG(I,J,K)=BBG
      FF1=SO(I,J,K)/BO(I,J,K)
      FF2=SW(I,J,K)/BW(I,J,K)
      SCFO=SCFO+VP(I,J,K)*FF1
      SCFW=SCFW+VP(I,J,K)*FF2
      SCFG=SCFG+VP(I,J,K)*SG(I,J,K)/BG(I,J,K)
      SCFG1=SCFG1+VP(I,J,K)*(RSO*FF1+RSW*FF2)
C
      CALL NTERP1(NTE,POT,BOPT,MPOT,PP,BODER)
      CALL NTERP1(NTE,POT,RSOPT,MPOT,PP,RSODER)
      CALL NTERP1(NTE,PWT,BWPT,MPWT,PP,BWDER)
      CALL NTERP1(NTE,PWT,RSWPT,MPWT,PP,RSWDER)
      CALL NTERP1(NTE,PGT,BGPT,MPGT,PP,BGDER)
C
      IF (PP.GT.PBOT(I,J,K))BODER=BSLOPE
      IF (PP.GT.PBOT(I,J,K))RSODER=RSLOPE
      CO=-(BODER-BG(I,J,K)*RSODER)/BO(I,J,K)
      CW=-(BWDER-BG(I,J,K)*RSWDER)/BW(I,J,K)
      CG=-BGDER/BG(I,J,K)
      CALL INTERP(NTE,PGT,CRT,MPGT,PP,CR)
C
CCC ------------------------------------------------------------------
CCC
CCC      MODIFICATION OF BLOCK TOTAL COMPRESSIBILITY IF CALCULATED
CCC      VALUE IS NEGATIVE-RESET TO PREVIOUS TIME STEP VALUE.
CCC
CCC ------------------------------------------------------------------
C
      CX = CT(I,J,K)
      CT(I,J,K)=CR + CO*SO(I,J,K) +CW*SW(I,J,K) + CG*SG(I,J,K)
        IF (CT(I,J,K).LE.0.0) THEN
        CT(I,J,K)=CX
        WRITE(*,657) I,J,K
  657 FORMAT(' ','RESET TOTAL COMP. TO PREVIOUS VALUE AT POSN.',3I5)
      END IF
C
      IF (N.EQ.KCO)WRITE(IOCODE,21)I,J,K,CR,CO,RSO,CW,RSW,CG
  400 CONTINUE
C
CC ------------------------------------------------------------------
CC
CC      AUTO TIME STEP CONTROL CALC. OF PRESSURE AT SAT. MAXIMA.
CC
CC ------------------------------------------------------------------
C
      PPM=0.
      SOM=0.
      SWM=0.
      SGM=0.
C
      DO 240 K = 1,KK
      DO 240 J = 1,JJ
      DO 240 I = 1,II
         DPO=P(I,J,K)-PN(I,J,K)
         DSO=SO(I,J,K)-SON(I,J,K)
         DSW=SW(I,J,K)-SWN(I,J,K)
         DSG=SG(I,J,K)-SGN(I,J,K)
         IF (ABS(DPO).LE.ABS(PPM)) GO TO 210
         PPM=DPO
              IP=I
              JP=J
              KP=K
  210    IF (ABS(DSO).LE.ABS(SOM)) GO TO 220
         SOM=DSO
              ISO=I
              JSO=J
              KSO=K
  220    IF (ABS(DSW).LE.ABS(SWM)) GO TO 230
         SWM=DSW
              ISW=I
              JSW=J
              KSW=K
  230    IF (ABS(DSG).LE.ABS(SGM)) GO TO 240
         SGM=DSG
              ISG=I
              JSG=J
              KSG=K
  240 CONTINUE
```

```
C
      DPMC=ABS(PPM)
      DSMC=ABS(SOM)
           IF (DSMC.LT.ABS(SGM)) THEN
           DSMC=ABS(SGM)
           IS=ISG
           JS=JSG
           KS=KSG
           IF (DSMC.LT.ABS(SWM)) THEN
           DSMC=ABS(SWM)
           IS=ISW
           JS=JSW
           KS=KSW
      END IF
      ELSE
           IS=ISO
           JS=JSO
           KS=KSO
      END IF
C
CC ---------------------------------------------------------------------
CC
CC        REPEAT TIME STEP
CC
CC ---------------------------------------------------------------------
C
      IF (DSMC.LT.DSMAX.AND.DPMC.LT.DPMAX.AND.NITER.LT.MITER)
     $    GOTO 402
      IF (DELT.LE.DTMIN .OR.  FACT2.GE.1.0) GO TO 402
      ITFLAG=ITFLAG+1
      DELT=DELT*FACT2
C
      IF (DELT.LT.DTMIN) DELT=DTMIN
      FT=ETI+DELT
      IF (FT.GT.FTMAX) DELT=FTMAX-ETI
C
CC ---------------------------------------------------------------------
CC
CC          RESET VARIABLES
CC
CC ---------------------------------------------------------------------
C
      DO 250 I = 1,II
      DO 250 J = 1,JJ
      DO 250 K = 1,KK
          P(I,J,K)=PN(I,J,K)
          SO(I,J,K)=SON(I,J,K)
          SW(I,J,K)=SWN(I,J,K)
          SG(I,J,K)=SGN(I,J,K)
 250  CONTINUE
C
          IF (NITER.GT.MITER) GOTO 3176
          IF (DSMC.GT.DSMAX) THEN
          WRITE(IOCODE,3173) DSMC,IS,JS,KS
 3173 FORMAT(' ','TIME STEP RETREAT-CHANGE IN SATURATION(',
     $         F6.4,') AT',3I5,' EXCEEDS USER DEFINED MAXIMUM')
      END IF
C
          IF (DPMC.GT.DPMAX) THEN
          WRITE(IOCODE,3174) DPMC,IP,JP,KP
 3174 FORMAT(' ','TIME STEP RETREAT-CHANGE IN PRESSURE(',
     $         F6.1,') AT',3I5,' EXCEEDS USER DEFINED MAXIMUM')
      END IF
 3176 CONTINUE
C
      GO TO 1060
 402  CONTINUE
C
CC ---------------------------------------------------------------------
CC
CC        UNDERSATURATED GRID BLOCK SATURATION CALCULATION
CC
CC ---------------------------------------------------------------------
C
      DO 410 I = 1,II
      DO 410 J = 1,JJ
      DO 410 K = 1,KK
          IF (P(I,J,K).GT.PN(I,J,K)) GO TO 410
          IF (P(I,J,K).LT.PBOT(I,J,K)) GO TO 410
          IP=I+1
          IM=I-1
          JP=J+1
          JM=J-1
          KP=K+1
          KM=K-1
          IF (IP.GT.II) GO TO 412
          IF (SGN(IP,J,K).GT.0.0001) GO TO 410
412       IF (IM.LT.1) GO TO 414
          IF (SGN(IM,J,K).GT.0.0001) GO TO 410
414       IF (JP.GT.JJ) GO TO 416
          IF (SGN(I,JP,K).GT.0.0001) GO TO 410
416       IF (JM.LT.1) GO TO 418
          IF (SGN(I,JM,K).GT.0.0001) GO TO 410
418       IF (KP.GT.KK) GO TO 420
          IF (SGN(I,J,KP).GT.0.0001) GO TO 410
420       IF (KM.LT.1) GO TO 422
          IF (SGN(I,J,KM).GT.0.0001) GO TO 410
422       SG(I,J,K)=0.0
          SO(I,J,K)=1.0-SW(I,J,K)
410   CONTINUE
```

```
C
CCC ------------------------------------------------------------------
CCC
CCC         REPRESSURIZATION ALGORITHM
CCC
CCC ------------------------------------------------------------------
C
      IF (IREPRS.EQ.1) GO TO 51
      DO 50 I = 1,II
      DO 50 J = 1,JJ
      DO 50 K = 1,KK
         IF (SG(I,J,K).LE.0.0001) GO TO 50
         PP=P(I,J,K)
         IF (P(I,J,K).GT.PBOT(I,J,K)) PP=PBOT(I,J,K)
C
         CALL INTERP(NTE,POT,BOT,MPOT,PP,BBO)
         CALL INTERP(NTE,POT,RSOT,MPOT,PP,RSO)
         CALL INTERP(NTE,PGT,BGT,MPGT,PP,BBG)
         IF (SO(I,J,K).EQ.0.0) GO TO 50
         RSONEW=RSO + SG(I,J,K)*BBO/(SO(I,J,K)*BBG)
C
         CALL INTERP(NTE,RSOT,POT,MPOT,RSONEW,PBONEW)
         PBOT(I,J,K)=PBONEW
 50   CONTINUE
 51   CONTINUE
C
CCC ------------------------------------------------------------------
CCC
CCC         UPDATE OLD FLUID VOLUMES FOR MATERIAL BALANCE
CCC
CCC ------------------------------------------------------------------
C
      STBOI=STBO
      STBWI=STBW
      MCFGI=MCFGT
C
CCC ------------------------------------------------------------------
CCC
CCC         UPDATE NEW FLUID VOLUMES
CCC
CCC ------------------------------------------------------------------
C
      STBO=SCFO*D5615
      STBW=SCFW*D5615
      MCFG=SCFG*0.001
      MCFG1=SCFG1*0.001
      MCFGT=MCFG+MCFG1
C
CCC ------------------------------------------------------------------
CCC
CCC         DEBUG PRINT OF PRESENT AND FUTURE P, SO, SW, SG VALUES
CCC
CCC ------------------------------------------------------------------
C
      IF (KCOFF.EQ.0) GO TO 990
      DO 290 K = 1,KK
      DO 290 J = 1,JJ
      DO 290 I = 1,II
         WRITE(IOCODE,21) I,J,K,PN(I,J,K),SON(I,J,K),SWN(I,J,K),
     $                    SGN(I,J,K)
         WRITE(IOCODE,21) I,J,K,P(I,J,K),SO(I,J,K),SW(I,J,K),
     $                    SG(I,J,K)
 290  CONTINUE
 990  CONTINUE
C
CC ------------------------------------------------------------------
CC
CC         WELL REPORT ( ALL RATE & PRESSURE DATA APPLICABLE THIS STEP)
CC
CC ------------------------------------------------------------------
C
      IJ=0
      TOR=0.
      TGR=0.
      TWR=0.
      TOC=0.
      TGC=0.
      TWC=0.
C
      DO 2050 J = 1,NVQN
         GOR=0.
         WOR=0.
         IQ1=IQN1(J)
         IQ2=IQN2(J)
         IQ3=IQN3(J)
         IJ=IJ+1
         LAY=IQ3+(LAYER(J)-1)
C
      DO 2050 K = IQ3,LAY
         QOO=QO(IQ1,IQ2,K)*D5615
         QWW=QW(IQ1,IQ2,K)*D5615
         QGG=QG(IQ1,IQ2,K)*0.001
         CUMO(J,K)=CUMO(J,K)+QOO*DELT*0.001
         CUMW(J,K)=CUMW(J,K)+QWW*DELT*0.001
         CUMG(J,K)=CUMG(J,K)+QGG*DELT*0.001
         IF (IWLREP.EQ.0) GO TO 2050
         IF (IJ.EQ.1.AND.K.EQ.IQ3) WRITE(IOCODE,5911) FT
         IF (IJ.EQ.1.AND.K.EQ.IQ3) WRITE(IOCODE,5911)
5911 FORMAT(/,50X,'------- RATE -------',22X,'--- CUMULATIVE ---',
     $ /,13X,'WELL LOCATION',4X,'CALC     SPEC     SPEC',4X,
     $ 'OIL      GAS     WATER      GOR      WOR',5X,
     $ 14X,'ID',3X,'I   J   K     BHFP     BHFP      PI',
     $ 3X,'STB/D     MCF/D    STB/D',' SCF/STB    STB/STB  ',
     $ 'MSTB      MMCF      MSTB',/)
         IF (QOO.EQ.0.) GO TO 998
         GOR=QGG*1000./QOO
         WOR=QWW/QOO
998      WRITE(IOCODE,592) WELLID(J),IQN1(J),IQN2(J),K,PWF(J,K),
     $ PWF(J,K),PID(J,K),QOO,QGG,QWW,GOR,WOR,CUMO(J,K),CUMG(J,K),
     $ CUMW(J,K)
592  FORMAT(11X,A5,1X,3I3,2F8.0,F7.3,3F9.0,F7.0,F7.3,3F8.0)
```

```
C
            TOR=TOR+QOO
            TGR=TGR+QGG
            TWR=TWR+QWW
            TOC=TOC+CUMO(J,K)
            TGC=TGC+CUMG(J,K)
            TWC=TWC+CUMW(J,K)
 2050 CONTINUE
C
       IF (IWLREP.EQ.0) GO TO 2052
       WRITE(IOCODE,5912) TOR,TGR,TWR,TOC,TGC,TWC
 5912 FORMAT(12X,102('-'),/,
     $ 12X,'TOTALS',31X,3F9.0,14X,3F8.0,/)
 2052 CONTINUE
C
C  ------------------------------------------------------------
C
C          CALCULATE MATERIAL BALANCE ERRORS AND AVERAGE RESERVOIR PRESSURE
C
C  ------------------------------------------------------------
C
       DELTO=DELT
       ETI=ETI+DELT
       CALL MATBAL(II,JJ,KK,STBO,STBOI,STBW,STBWI,
      $            MCFG,MCFGI,MBEO,MBEW,MBEG,DELTO,RESVOL,OP,WP,GP,WI,
      $            GI,PAVGO,PAVG,NLOOP,OPR,WPR,GPR,WIR,GIR,D5615,
      $            COP,CWP,CGP,CWI,CGI,MCFG1,MCFGT,CWOR,WOR,CGOR,GOR)
C
       IF (WOR.GT.WORMAX) GO TO 1002
       IF (GOR.GT.GORMAX) GO TO 1003
       IF (PAVG.LT.PAMIN) GO TO 1004
       IF (PAVG.GT.PAMAX) GO TO 1005
C
C  ------------------------------------------------------------
C
C          SUMMARY REPORT
C
C  ------------------------------------------------------------
C
       IF (ISUMRY.EQ.0) GO TO 2057
       NLP=N+1
       CALL PRTPS(NLP,KPI,II,JJ,KK,PAVGO,PAVG,
      $           COP,CWP,CWI,CGP,CGI,MBEO,MBEW,MBEG,DELTO,
      $           OPR,WPR,GPR,WIR,GIR,ETI,CWOR,CGOR,WOR,GOR,
      $          IPMAP,ISOMAP,ISWMAP,IGGMAP,IPBMAP)
 2057 IF(N.EQ.KCO .OR. KCO1.EQ.0) GO TO 500
C
       DO 300 K = 1,KK
       DO 300 J = 1,JJ
       DO 300 I = 1,II
          WRITE(IOCODE,21)I,J,K,VP(I,J,K),CT(I,J,K),BO(I,J,K),SO(I,J,K),
      $                   BW(I,J,K),SW(I,J,K),BG(I,J,K),SG(I,J,K)
 300   CONTINUE
C
 500   CONTINUE
C
       IF (N.EQ.KSN) KSN=KSN+KSN1
       IF (N.EQ.KSM) KSM=KSM+KSM1
       IF (N.EQ.KCO) KCO=KCO+KCO1
C
C  ------------------------------------------------------------
C
C          UPDATE    ARRAYS
C
C  ------------------------------------------------------------
C
C
       DO 1150 K = 1,KK
       DO 1150 J = 1,JJ
       DO 1150 I = 1,II
          QO(I,J,K)=0.0
          QW(I,J,K)=0.0
          QG(I,J,K)=0.0
           PN(I,J,K)=P(I,J,K)
           SON(I,J,K)=SO(I,J,K)
           SWN(I,J,K)=SW(I,J,K)
           SGN(I,J,K)=SG(I,J,K)
 1150 CONTINUE
 1000 CONTINUE
C
 1002 WRITE(IOCODE,2002)
       GO TO 1001
 1003 WRITE(IOCODE,2003)
       GO TO 1001
 1004 WRITE(IOCODE,2004)
       GO TO 1001
 1005 WRITE(IOCODE,2005)
 1001 CONTINUE
C
 2002 FORMAT(/15X,'MAXIMUM WOR HAS BEEN EXCEEDED --- SIMULATION',
     $' IS BEING TERMINATED',//)
 2003 FORMAT(/15X,'MAXIMUM GOR HAS BEEN EXCEEDED --- SIMULATION',
     $' IS BEING TERMINATED',//)
 2004 FORMAT(/15X,'MINIMUM AVERAGE RESERVOIR PRESSURE WAS NOT',
     $' ACHIEVED --- SIMULATION IS BEING TERMINATED'//)
 2005 FORMAT(/15X,'MAXIMUM AVERAGE RESERVOIR PRESSURE HAS BEEN',
     $' EXCEEDED --- SIMULATION IS BEING TERMINATED'//)
 1234 FORMAT(1X,10E13.6)
 33    FORMAT(//)
 591   FORMAT(///5X,10('*'),' WELL REPORT FOR ALL ACTIVE WELLS ',4X,
     $ 'ELAPSED TIME =',F11.6,' DAYS FROM BEGINNING OF SIMULATION',
     $10('*'),//)
 21    FORMAT(1X,3I3,8E15.6)
 69    FORMAT(80A1)
C
C  ....CLOSE I/O FILES
C
          CLOSE(20)
          CLOSE(IOCODE)
       STOP
       END
```

```
C
CC ================================================================
C
      SUBROUTINE   CODES(IOCODE,KSM1,KSN1,KCO1,NN,
     $             FACT1,FACT2,TMAX,KSOL,MITER,OMEGA,TOL,TOL1,
     $             KSN,KSM,KCO,KTR,KCOFF,DSMAX,DPMAX,
     $             WORMAX,GORMAX,PAMIN,PAMAX)
C
CC ================================================================
C
      CHARACTER IHEDIN(80)
      READ(20,69) (IHEDIN(IH),IH=1,80)
      READ(20,*) KSN1,KSM1,KCO1,KTR,KCOFF
      WRITE(*,5) IOCODE
    5 FORMAT(///,'***** VALUE OF IOCODE ***** ',I5/)
C
CC ----------------------------------------------------------------
CC
CC       EVERY KSM1'TH STEP SOLUTION MATRIX WILL BE WRITTEN
CC       EVERY KSN1'TH STEP CLSOR DATA WILL BE WRITTEN
CC       IF KTR IS NONZERO, TRANSMISSIBILITIES WILL BE WRITTEN
CC
CC ----------------------------------------------------------------
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      READ(20,*)NN,FACT1,FACT2,TMAX,WORMAX,GORMAX,PAMIN,PAMAX
      WRITE(IOCODE,59)NN,FACT1,FACT2,TMAX,WORMAX,GORMAX,PAMIN,PAMAX
C
CC ----------------------------------------------------------------
CC
CC       NN      :  MAXIMUM NUMBER OF TIME STEPS
CC       FACT1   :  FACTOR FOR INCREASING TIME STEP
CC       FACT2   :  FACTOR FOR DECREASING TIME STEP
CC       TMAX    :  MAXIMUM SIMULATION TIME
CC
CC ----------------------------------------------------------------
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      READ(20,*)KSOL,MITER,OMEGA,TOL,TOL1,DSMAX,DPMAX
      IF (KSOL.EQ.2) WRITE(IOCODE,75) MITER,OMEGA,TOL,TOL1
      WRITE(IOCODE,79) DSMAX,DPMAX
   75 FORMAT(//15X,'SOLUTION METHOD IS CLSOR:',
     $/20X,'MAXIMUM NUMBER OF ITERATIONS          (MITR) = ',5X,I5,
     $/20X,'INITIAL ACCELERATION PARAMETER       (OMEGA) = ',F10.4,
     $/20X,'MAXIMUM PRESSURE RESIDUAL              (TOL) = ',F10.4,
     $/20X,'PARAMETER FOR CHANGING OMEGA          (TOL1) = ',F10.4)
   79 FORMAT(/15X,'AUTOMATIC TIME STEP CRITERIA:',
     $/20X,'MAXIMUM ALLOWED SATURATION CHANGE   (DSMAX) = ',F10.4,
     $/20X,'MAXIMUM ALLOWED PRESSURE  CHANGE    (DPMAX) = ',F10.4/)
   59 FORMAT(///15X,'MAXIMUM NUMBER OF TIME STEPS =',I5/
     $15X,'FACTOR FOR INCREASING DELT -',F10.4,3X,
     $'WHEN DSMAX AND DPMAX NOT EXCEEDED.',/,
     $15X,'FACTOR FOR DECREASING DELT -',F10.4,3X,
     $'WHEN DSMAX AND DPMAX IS EXCEEDED.',/,
     $15X,'MAXIMUM SIMULATION TIME =',F11.2/,
     $15X,'MAXIMUM RESERVOIR WOR/TIME-STEP =',F8.1,' STB/STB'/
     $15X,'MAXIMUM RESERVOIR GOR/TIME-STEP =',F8.1,' SCF/STB'/
     $15X,'MINIMUM AVERAGE RESERVOIR PRESSURE/TIME-STEP =',F8.1/
     $15X,'MAXIMUM AVERAGE RESERVOIR PRESSURE/TIME-STEP =',F8.1//)
   69 FORMAT(80A1)
      KSN=KSN1
      KSM=KSM1
      KCO=KCO1
C
      RETURN
      END
C
```

```
C  ==========================================================================
C
      SUBROUTINE  GRID1(II,JJ,KK)
C
C  ==========================================================================
C
      INCLUDE 'COMMB3.FOR'
C
      DIMENSION ZSUM(NX,NY),VAREL(NX,NY),RDXL(NX),RDYL(NY),RDZL(NZ)
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      WRITE(IOCODE,70) (IHEDIN(IH),IH=1,80)
      READ(20,*)II,JJ,KK
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
C
C  --------------------------------------------------------------------------
C
C         READ INPUT CODES FOR DX,DY,DZ
C
C  --------------------------------------------------------------------------
C
      READ(20,*) KDX,KDY,KDZ
C
C  --------------------------------------------------------------------------
C
C         ESTABLISH GRID BLOCK LENGTH (DX) DISTRIBUTION
C
C  --------------------------------------------------------------------------
C
      IF (KDX.GE.0) GO TO 180
      READ(20,*) DXC
      DO 175 K=1,KK
      DO 175 J=1,JJ
      DO 175 I=1,II
  175 DX(I,J,K)=DXC
      WRITE(IOCODE,56)
      WRITE(IOCODE,29) DXC
      GO TO 195
C
  180 IF (KDX.GT.0) GO TO 185
      READ(20,*)(RDXL(I),I=1,II)
      DO 187 K=1,KK
      DO 187 J=1,JJ
      DO 187 I=1,II
  187 DX(I,J,K)=RDXL(I)
      DO 182 I=1,II
      WRITE(IOCODE,511) I,RDXL(I)
  182 CONTINUE
      GO TO 195
C
  185 WRITE(IOCODE,43)
      K=1
      WRITE(IOCODE,38) K
      DO 190 J=1,JJ
      READ(20,*)(DX(I,J,K),I=1,II)
  190 WRITE(IOCODE,72)(DX(I,J,K),I=1,II)
  192 CONTINUE
      DO 194 K=2,KK
      WRITE(IOCODE,38) K
      DO 194 J=1,JJ
      DO 193 I=1,II
  193 DX(I,J,K)=DX(I,J,1)
  194 WRITE(IOCODE,72) (DX(I,J,K),I=1,II)
  195 CONTINUE
      WRITE(IOCODE,56)
C
C  --------------------------------------------------------------------------
C
C         ESTABLISH GRID BLOCK LENGTH (DY) DISTRIBUTION
C
C  --------------------------------------------------------------------------
C
      IF (KDY.GE.0) GO TO 200
      READ(20,*) DYC
      DO 202 K=1,KK
      DO 202 J=1,JJ
      DO 202 I=1,II
  202 DY(I,J,K)=DYC
      WRITE(IOCODE,56)
      WRITE(IOCODE,33) DYC
      GO TO 220
C
```

```
 200   IF (KDY.GT.0) GO TO 207
       READ(20,*) (RDYL(J),J=1,JJ)
       DO 205 K=1,KK
       DO 205 J=1,JJ
       DO 205 I=1,II
 205   DY(I,J,K)=RDYL(J)
       DO 210 J=1,JJ
       WRITE(IOCODE,512) J,RDYL(J)
 210   CONTINUE
       GO TO 220
C
 207   WRITE(IOCODE,47)
       K=1
       WRITE(IOCODE,38)K
       DO 215 I=1,II
       READ(20,*)(DY(I,J,K),J=1,JJ)
 215   WRITE(IOCODE,72)(DY(I,J,K),J=1,JJ)
       DO 214 K=2,KK
       WRITE(IOCODE,38) K
       DO 214 J=1,JJ
       DO 213 I=1,II
 213   DY(I,J,K)=DY(I,J,1)
       WRITE(IOCODE,72) (DY(I,J,K),I=1,II)
 214   CONTINUE
 212   CONTINUE
 220   CONTINUE
       WRITE(IOCODE,56)
C
CC ---------------------------------------------------------------------
CC
CC          ESTABLISH GRID BLOCK LENGTH (DZ) DISTRIBUTION
CC
CC ---------------------------------------------------------------------
C
       IF (KDZ.GE.0) GO TO 225
       READ(20,*) DZC
       DO 230 K=1,KK
       DO 230 J=1,JJ
       DO 230 I=1,II
 230   DZ(I,J,K)=DZC
       WRITE(IOCODE,56)
       WRITE(IOCODE,36)DZC
       GO TO 245
C
 225   IF (KDZ.GT.0) GO TO 232
       READ(20,*)(RDZL(K),K=1,KK)
       DO 235 K=1,KK
       DO 235 J=1,JJ
       DO 235 I=1,II
       DZ(I,J,K)=RDZL(K)
 235   CONTINUE
       DO 237 K=1,KK
       WRITE(IOCODE,513) K,RDZL(K)
 237   CONTINUE
       GO TO 245
C
 232   WRITE(IOCODE,48)
       DO 240 K=1,KK
       WRITE(IOCODE,38)K
       DO 242 J=1,JJ
       READ(20,*)(DZ(I,J,K),I=1,II)
 242   WRITE(IOCODE,72)(DZ(I,J,K),I=1,II)
 240   CONTINUE
 245   CONTINUE
       WRITE(IOCODE,56)
C
CC ---------------------------------------------------------------------
CC
CC          GRID BLOCK LENGTH MODIFICATIONS
CC
CC ---------------------------------------------------------------------
C
       READ(20,69) (IHEDIN(IH),IH=1,80)
       READ(20,*) NUMDX,NUMDY,NUMDZ,IDCODE
C
       IF (NUMDX.EQ.0) GO TO 711
       WRITE(IOCODE,31)
       DO 275 L=1,NUMDX
       READ(20,*)I,J,K,DX(I,J,K)
 275   WRITE(IOCODE,32)I,J,K,DX(I,J,K)
C
       IF (IDCODE.NE.1)GO TO 711
       WRITE(IOCODE,43)
       DO 853 K=1,KK
       WRITE(IOCODE,38)K
       DO 854 J=1,JJ
       WRITE(IOCODE,72) (DX(I,J,K),I=1,II)
 854   CONTINUE
 853   CONTINUE
 711   CONTINUE
C
```

```
      IF (NUMDY.EQ.0) GO TO 712
      WRITE(IOCODE,34)
      DO 276 L=1,NUMDY
      READ(20,*)I,J,DY(I,J,K)
  276 WRITE(IOCODE,32)I,J,K,DY(I,J,K)
      IF (IOCODE.NE.1) GO TO 712
      WRITE(IOCODE,47)
      DO 855 K=1,KK
      WRITE(IOCODE,38)K
      DO 856 J=1,JJ
      WRITE(IOCODE,72)(DY(I,J,K),I=1,II)
  856 CONTINUE
  855 CONTINUE
  712 CONTINUE
C
      IF (NUMDZ.EQ.0) GO TO 713
      WRITE(IOCODE,37)
      DO 277 L=1,NUMDZ
      READ(20,*)I,J,K,DZ(I,J,K)
  277 WRITE(IOCODE,32)I,J,K,DZ(I,J,K)
      IF (IOCODE.NE.1) GO TO 713
      WRITE(IOCODE,48)
      DO 857 K=1,KK
      WRITE(IOCODE,38)K
      DO 858 J=1,JJ
      WRITE(IOCODE,72)(DZ(I,J,K),I=1,II)
  858 CONTINUE
  857 CONTINUE
  713 CONTINUE
C
CC ---------------------------------------------------------------------
C
C          ESTABLISH NODE MID-POINT ELEVATION
C
CC ---------------------------------------------------------------------
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      READ(20,*) KEL
      IF (KEL.EQ.1) GO TO 920
      READ(20,*) ELEV
      DO 910 J=1,JJ
      DO 910 I=1,II
      VAREL(I,J)=ELEV
  910 CONTINUE
  920 IF (KEL.NE.1) GO TO 930
      DO 922 J=1,JJ
      READ(20,*)(VAREL(I,J),I=1,II)
  922 CONTINUE
  930 CONTINUE
C
      DO 923 I=1,II
      DO 923 J=1,JJ
      ZSUM(I,J)=0.0
  923 CONTINUE
C
      DO 926 K=1,KK
      DO 926 J=1,JJ
      DO 926 I=1,II
      DEL=ZSUM(I,J)+DZ(I,J,K)*0.5
      EL(I,J,K)=VAREL(I,J)+DEL
      ZSUM(I,J)=DZ(I,J,K)+ZSUM(I,J)
  926 CONTINUE
      WRITE(IOCODE,390)
      DO 600 K=1,KK
      WRITE(IOCODE,38)K
      DO 600 J=1,JJ
      WRITE(IOCODE,72)(EL(I,J,K),I=1,II)
  600 CONTINUE
  601 CONTINUE
C
```

```
 69    FORMAT(80A1)
 56    FORMAT(//)
 70    FORMAT(23X,83('*'),/,24X,('*'),81X,('*'),/,
     $ 24X,('*'),80A1,1X,('*'),/,24X,('*'),81X,
     $ ('*'),/,24X,83('*'),//)
 72    FORMAT(1X,20F6.0)
 29    FORMAT(15X,'GRID BLOCK LENGTH (DX) IS INITIALLY',
     $' SET AT',F10.4,' FOR ALL NODES'//)
 31    FORMAT(//15X,'********GRID BLOCK LENGTH (DX) NODE MODIFICATIONS',
     $'*********',//15X,'          I    J    K              NEW DX VALUE')
 32    FORMAT(15X,3I5,5X,F14.4)
 33    FORMAT(15X,'GRID BLOCK WIDTH (DY) IS INITIALLY',
     $' SET AT',F10.4,' FOR ALL NODES'//)
 34    FORMAT(//15X,'********GRID  BLOCK WIDTH (DY) NODE MODIFICATIONS',
     $'*********',//15X,'          I    J    K              NEW DY VALUE')
 36    FORMAT(15X,'GRID BLOCK DEPTH (DZ) IS INITIALLY ',
     $' SET AT',F10.4,' FOR ALL NODES'///)
 37    FORMAT(//15X,'*******GRID BLOCK DEPTH (DZ) NODE MODIFICATIONS',
     $'***********',//15X,'     I    J    K              NEW DZ VALUE')
 38    FORMAT(/1X,'K =',I2/)
390    FORMAT(//15X,'********** NODE MIDPOINT ELEVATIONS **********'/)
 43    FORMAT(//15X,'********GRID BLOCK LENGTH (DX) DISTRIBUTION******',
     $ /)
 47    FORMAT(//15X',********GRID BLOCK  WIDTH (DY) DISTRIBUTION******',
     $ /)
 48    FORMAT(//15X,'********GRID BLOCK  DEPTH (DZ) DISTRIBUTION******',
     $ /)
511    FORMAT(15X,'GRID SIZE (DX) IN COLUMN',I5,' IS INITIALLY SET AT'
     $,F8.2,' FOR ALL NODES',/)
512    FORMAT(15X,'GRID SIZE (DY) IN ROW   ',I5,' IS INITIALLY SET AT'
     $,F8.2,' FOR ALL NODES',/)
513    FORMAT(15X,'GRID SIZE (DZ) IN LAYER ',I5,' IS INITIALLY SET AT'
     $,F8.2,' FOR ALL NODES',/)
C
      RETURN
      END
C
```

```
C  ================================================================
C
      SUBROUTINE   TABLE
C
C  ================================================================
C
      INCLUDE 'COMMB3.FOR'
      WRITE(IOCODE,111)
C  ----------------------------------------------------------------
C
C         RELATIVE PERMEABILITY & CAPILLIARY PRESSURE TABLE
C
C  ----------------------------------------------------------------
C
      READ(20,1) (IHEDIN(IH),IH=1,80)
      WRITE(IOCODE,11) (IHEDIN(IH),IH=1,80)
      DO 5 I=1,NTE
      READ(20,*) SAT(I),KROT(I),KRWT(I),KRGT(I),PCOWT(I),PCGOT(I)
      WRITE(IOCODE,21) SAT(I),KROT(I),KRWT(I),KRGT(I),PCOWT(I),
     $                 PCGOT(I)
      IF (SAT(I).GE.1.10) GO TO 10
    5 CONTINUE
C
C  ----------------------------------------------------------------
C
C         BUBBLE POINT AND MAXIMUM PRESSURES
C
C  ----------------------------------------------------------------
C
   10 MSAT=I
      READ(20,1) (IHEDIN(IH),IH=1,80)
      WRITE(IOCODE,11) (IHEDIN(IH),IH=1,80)
      READ(20,*) PBO,VSLOPE,BSLOPE,RSLOPE,PMAXT,IREPRS
      WRITE(IOCODE,31)PBO,VSLOPE,BSLOPE,RSLOPE,PMAXT,IREPRS
C
C  ----------------------------------------------------------------
C
C         OIL DATA TABLE
C
C  ----------------------------------------------------------------
C
      READ(20,1) (IHEDIN(IH),IH=1,80)
      WRITE(IOCODE,11) (IHEDIN(IH),IH=1,80)
      DO 15 I=1,NTE
      READ(20,*) POT(I),MUOT(I),BOT(I),RSOT(I)
      WRITE(IOCODE,23)POT(I),MUOT(I),BOT(I),RSOT(I)
      RSOT(I)=0.17809*RSOT(I)
      IF (POT(I).GE.PMAXT) GO TO 20
   15 CONTINUE
C
C  ----------------------------------------------------------------
C
C         WATER DATA TABLE
C
C  ----------------------------------------------------------------
C
   20 MPOT=I
      READ(20,1) (IHEDIN(IH),IH=1,80)
      WRITE(IOCODE,11)(IHEDIN(IH),IH=1,80)
         DO 25 I=1,NTE
      READ(20,*) PWT(I),MUWT(I),BWT(I),RSWT(I)
      WRITE(IOCODE,23)PWT(I),MUWT(I),BWT(I),RSWT(I)
      RSWT(I)=0.17809*RSWT(I)
      IF (PWT(I).GE.PMAXT) GO TO 30
   25 CONTINUE
C
C  ----------------------------------------------------------------
C
C         GAS AND ROCK DATA TABLE
C
C  ----------------------------------------------------------------
C
```

```
 30     MPWT=I
        READ(20,1) (IHEDIN(IH),IH=1,80)
        WRITE(IOCODE,11) (IHEDIN(IH),IH=1,80)
        DO 35 I=1,NTE
        READ(20,*) PGT(I),MUGT(I),BGT(I),CRT(I)
        WRITE(IOCODE,24) PGT(I),MUGT(I),BGT(I),CRT(I)
        IF (PGT(I).GE.PMAXT) GO TO 40
 35     CONTINUE
C
 40     MPGT=I
        CONTINUE
C
CCC  ------------------------------------------------------------
CCC
CCC          OIL, WATER AND GAS DENSITIES AT STOCK TANK CONDITIONS
CCC
CCC  ------------------------------------------------------------
C
        READ(20,1) (IHEDIN(IH),IH=1,80)
        WRITE(IOCODE,11) (IHEDIN(IH),IH=1,80)
        READ(20,*) RHOSCO,RHOSCW,RHOSCG
        WRITE(IOCODE,4) RHOSCO,RHOSCW,RHOSCG
C
CCC  ------------------------------------------------------------
CCC
CCC          CALCULATE SLOPES dBO/dP, dRSO/dP, dBW/dP, dRSW/dP,
CCC          dBG/dP
CCC
CCC  ------------------------------------------------------------
C
        WRITE(IOCODE,222)
        WRITE(IOCODE,223)
        DO 100 I=2,MPOT
        DIV=1.0/(POT(I)-POT(I-1))
        BOPT(I)=(BOT(I)-BOT(I-1))*DIV
        RSOPT(I)=(RSOT(I)-RSOT(I-1))*DIV
 100    WRITE(IOCODE,224)POT(I),BOT(I),BOPT(I),RSOT(I),RSOPT(I)
        WRITE(IOCODE,333)
C
        DO 200 I=2,MPWT
        DIV=1.0/(PWT(I)-PWT(I-1))
        BWPT(I)=(BWT(I)-BWT(I-1))*DIV
        RSWPT(I)=(RSWT(I)-RSWT(I-1))*DIV
 200    WRITE(IOCODE,224)PWT(I),BWT(I),BWPT(I),RSWT(I),RSWPT(I)
C
        WRITE(IOCODE,444)
        DO 300 I=2,MPGT
        BGPT(I)=(BGT(I)-BGT(I-1))/(PGT(I)-PGT(I-1))
 300    WRITE(IOCODE,445)PGT(I),BGT(I),BGPT(I)
C
CCC  ------------------------------------------------------------
CCC
CCC          SLOPE AT POINT(I) IS BASED ON POINTS (I) AND (I-1)
CCC
CCC  ------------------------------------------------------------
C
  1     FORMAT(80A1)
  4     FORMAT(2X,8F10.4)
 21     FORMAT(1X,4F10.4,2F10.2)
 23     FORMAT(3X,F10.1,1X,F8.4,2X,F8.4,2X,F8.2)
 24     FORMAT(3X,F10.1,1X,F8.4,E10.4,E10.3)
 11     FORMAT(//,1X,80A1/)
 111    FORMAT(//15X,'***** EMPIRICAL DATA TABLE *****'//)
 31     FORMAT(F10.2,E10.3,1X,E10.3,2F10.2,I5)
 222    FORMAT(15X,'***** SLOPES FOR COMPRESSIBILITY CALCULATIONS',
     $  '*****')
 223    FORMAT(//16X,'P',7X,'BO',7X,'DBO/DP',7X,'RSO',
     $  7X,'DRSO/DP'/)
 445    FORMAT(10X,F9.1,1X,2E15.4)
 333    FORMAT(//16X,'P',7X,'BW',7X,'DBW/DP',7X,'RSW',
     $  7X,'DRSW/DP'/)
 224    FORMAT(13X,F7.1,F8.4,3X,E11.4,F8.1,2X,E11.4)
 444    FORMAT(//15X,'P',12X,'BG',12X,'DBG/DP')
C
        RETURN
        END
C
```

```
C  ==========================================================================
C
      SUBROUTINE  SOLMAT(II,JJ,KK,DIV1,D288,D144,
     $                        KSM,KSM1,N,NN,KCOFF)
C
CC ==========================================================================
C
      INCLUDE  'COMMB3.FOR'
C
      REAL  MUO4,MUW4,MUG4,MUO5,MUW5,MUG5,MUO6,MUW6,MUG6
     $     ,KRO1,KRW1,KRG1,KRO2,KRW2,KRG2,KRO3,KRW3,KRG3
     $     ,MUO1,MUW1,MUG1,MUO2,MUW2,MUG2,MUO3,MUW3,MUG3
     $     ,KRO4,KRW4,KRG4,KRO5,KRW5,KRG5,KRO6,KRW6,KRG6
     $     ,MO1,MW1,MG1,MO2,MW2,MG2,MO3,MW3,MG3
     $     ,MO4,MW4,MG4,MO5,MW5,MG5,MO6,MW6,MG6
C
          DATA  MO1,MO2,MO3,MO4,MO5,MO6  /6*0.0/
          DATA  MW1,MW2,MW3,MW4,MW5,MW6  /6*0.0/
          DATA  MG1,MG2,MG3,MG4,MG5,MG6  /6*0.0/
      DATA  RSO1,RSO2,RSO3,RSO4,RSO5,RSO6/6*0.0/
      DATA  RSW1,RSW2,RSW3,RSW4,RSW5,RSW6/6*0.0/
          DATA  GWW1,GWW2,GWW3,GWW4,GWW5,GWW6  /6*0.0/
          DATA  GOW1,GOW2,GOW3,GOW4,GOW5,GOW6  /6*0.0/
          DATA  GGW1,GGW2,GGW3,GGW4,GGW5,GGW6  /6*0.0/
          DATA  PCGO1,PCGO2,PCGO3,PCGO4,PCGO5,PCGO6  /6*0.0/
          DATA  PCOW1,PCOW2,PCOW3,PCOW4,PCOW5,PCOW6  /6*0.0/
C
      DO 200 K=1,KK
      DO 200 J=1,JJ
      DO 200 I=1,II
      PP=P(I,J,K)
           BPT=PBOT(I,J,K)
      CALL  INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PP,RSO)
      CALL  INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,PP,MUO)
      CALL  INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
      CALL  INTERP(NTE,PWT,MUWT,MPWT,PP,MUW)
      CALL  INTERP(NTE,PGT,MUGT,MPGT,PP,MUG)
C
      SSO=SO(I,J,K)
      SSW=SW(I,J,K)
      SSG=SG(I,J,K)
C
      CALL  INTERP(NTE,SAT,PCOWT,MSAT,SSW,PCOW)
      CALL  INTERP(NTE,SAT,PCGOT,MSAT,SSG,PCGO)
C
      RO=(RHOSCO + RSO*RHOSCG)/BO(I,J,K)
      RW=(RHOSCW + RSW*RHOSCG)/BW(I,J,K)
      RG=RHOSCG/BG(I,J,K)
C
      IF (I.EQ.1) GO TO 115
      P1=P(I-1,J,K)
      BPT=PBOT(I-1,J,K)
      CALL  INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,P1,RSO1)
      CALL  INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,P1,MUO1)
      CALL  INTERP(NTE,PWT,RSWT,MPWT,P1,RSW1)
      CALL  INTERP(NTE,PWT,MUWT,MPWT,P1,MUW1)
      CALL  INTERP(NTE,PGT,MUGT,MPGT,P1,MUG1)
C
      SO1S=SO(I-1,J,K)
      SW1S=SW(I-1,J,K)
      SG1S=SG(I-1,J,K)
      CALL  INTERP(NTE,SAT,PCOWT,MSAT,SW1S,PCOW1)
      CALL  INTERP(NTE,SAT,PCGOT,MSAT,SG1S,PCGO1)
C
      RO1=(RHOSCO + RSO1*RHOSCG)/BO(I-1,J,K)
      RW1=(RHOSCW + RSW1*RHOSCG)/BW(I-1,J,K)
      RG1=RHOSCG/BG(I-1,J,K)
C
      FACT=-D288*(EL(I-1,J,K)-EL(I,J,K))
      GOW1=(RO1+RO)*FACT
      GWW1=(RW1+RW)*FACT + PCOW-PCOW1
      GGW1=(RG1+RG)*FACT + PCGO1-PCGO
C
      P11=P1-PP
      HO1=P11+GOW1
      HW1=P11+GWW1
      HG1=P11+GGW1
```

```
C
      IF (HO1.GE.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SO1S,KRO1)
      IF (HO1.LT.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO1)
      IF (HW1.GE.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SW1S,KRW1)
      IF (HW1.LT.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW1)
      IF (HG1.GE.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SG1S,KRG1)
      IF (HG1.LT.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG1)
      MO1=4.0*KRO1/((BO(I-1,J,K)+BO(I,J,K)) * (MUO1+MUO))
      MW1=4.0*KRW1/((BW(I-1,J,K)+BW(I,J,K)) * (MUW1+MUW))
      MG1=4.0*KRG1/((BG(I-1,J,K)+BG(I,J,K)) * (MUG1+MUG))
C
  115 AOW=TX(I,J,K)*MO1
      AWW=TX(I,J,K)*MW1
      AGW=TX(I,J,K)*MG1
C
      IF (I.EQ.II) GO TO 125
      P2=P(I+1,J,K)
      BPT=PBOT(I+1,J,K)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,P2,RSO2)
      CALL INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,P2,MUO2)
      CALL INTERP(NTE,PWT,RSWT,MPWT,P2,RSW2)
      CALL INTERP(NTE,PWT,MUWT,MPWT,P2,MUW2)
      CALL INTERP(NTE,PGT,MUGT,MPGT,P2,MUG2)
      SO2=SO(I+1,J,K)
      SW2=SW(I+1,J,K)
      SG2=SG(I+1,J,K)
      CALL INTERP(NTE,SAT,PCOWT,MSAT,SW2,PCOW2)
      CALL INTERP(NTE,SAT,PCGOT,MSAT,SG2,PCGO2)
C
      RO2=(RHOSCO + RSO2*RHOSCG)/BO(I+1,J,K)
      RW2=(RHOSCW + RSW2*RHOSCG)/BW(I+1,J,K)
      RG2=RHOSCG/BG(I+1,J,K)
C
      FACT = -D288*(EL(I+1,J,K)-EL(I,J,K))
           GOW2=(RO2+RO)*FACT
           GWW2=(RW2+RW)*FACT+PCOW-PCOW2
           GGW2=(RG2+RG)*FACT+PCGO2-PCGO
C
           P22=P2-PP
           HO2=P22+GOW2
           HW2=P22+GWW2
           HG2=P22+GGW2
      IF (HO2.GE.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SO2,KRO2)
      IF (HO2.LT.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO2)
      IF (HW2.GE.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SW2,KRW2)
      IF (HW2.LT.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW2)
      IF (HG2.GE.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SG2,KRG2)
      IF (HG2.LT.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG2)
      MO2=4.0*KRO2/((BO(I+1,J,K)+BO(I,J,K)) * (MUO2+MUO))
      MW2=4.0*KRW2/((BW(I+1,J,K)+BW(I,J,K)) * (MUW2+MUW))
      MG2=4.0*KRG2/((BG(I+1,J,K)+BG(I,J,K)) * (MUG2+MUG))
C
  125 AOE=TX(I+1,J,K)*MO2
      AWE=TX(I+1,J,K)*MW2
      AGE=TX(I+1,J,K)*MG2
C
C
      IF (J.EQ.1) GO TO 135
      P3=P(I,J-1,K)
      BPT=PBOT(I,J-1,K)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,P3,RSO3)
      CALL INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,P3,MUO3)
      CALL INTERP(NTE,PWT,RSWT,MPWT,P3,RSW3)
      CALL INTERP(NTE,PWT,MUWT,MPWT,P3,MUW3)
      CALL INTERP(NTE,PGT,MUGT,MPGT,P3,MUG3)
      SO3=SO(I,J-1,K)
      SW3=SW(I,J-1,K)
      SG3=SG(I,J-1,K)
      CALL INTERP(NTE,SAT,PCOWT,MSAT,SW3,PCOW3)
      CALL INTERP(NTE,SAT,PCGOT,MSAT,SG3,PCGO3)
C
      RO3=(RHOSCO + RSO3*RHOSCG)/BO(I,J-1,K)
      RW3=(RHOSCW + RSW3*RHOSCG)/BW(I,J-1,K)
      RG3=RHOSCG/BG(I,J-1,K)
C
      FACT=-D288*(EL(I,J-1,K)-EL(I,J,K))
      GOW3=(RO3+RO)*FACT
      GWW3=(RW3+RW)*FACT  +  PCOW-PCOW3
      GGW3=(RG3+RG)*FACT  +  PCGO3-PCGO
C
```

```
      P33=P3-PP
      HO3=P33+GOW3
      HW3=P33+GWW3
      HG3=P33+GGW3
      IF (HO3.GE.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SO3,KRO3)
      IF (HO3.LT.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO3)
      IF (HW3.GE.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SW3,KRW3)
      IF (HW3.LT.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW3)
      IF (HG3.GE.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SG3,KRG3)
      IF (HG3.LT.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG3)
      MO3=4.0*KRO3/((BO(I,J-1,K)+BO(I,J,K)) * (MUO3+MUO))
      MW3=4.0*KRW3/((BW(I,J-1,K)+BW(I,J,K)) * (MUW3+MUW))
      MG3=4.0*KRG3/((BG(I,J-1,K)+BG(I,J,K)) * (MUG3+MUG))
C
  135 AOS=TY(I,J,K)*MO3
      AWS=TY(I,J,K)*MW3
      AGS=TY(I,J,K)*MG3
C
      IF (J.EQ.JJ) GO TO 140
      P4=P(I,J+1,K)
      BPT=PBOT(I,J+1,K)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,P4,RSO4)
      CALL INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,P4,MUO4)
      CALL INTERP(NTE,PWT,RSWT,MPWT,P4,RSW4)
      CALL INTERP(NTE,PWT,MUWT,MPWT,P4,MUW4)
      CALL INTERP(NTE,PGT,MUGT,MPGT,P4,MUG4)
      SO4=SO(I,J+1,K)
      SW4=SW(I,J+1,K)
      SG4=SG(I,J+1,K)
      CALL INTERP(NTE,SAT,PCOWT,MSAT,SW4,PCOW4)
      CALL INTERP(NTE,SAT,PCGOT,MSAT,SG4,PCGO4)
C
      RO4=(RHOSCO + RSO4*RHOSCG)/BO(I,J+1,K)
      RW4=(RHOSCW + RSW4*RHOSCG)/BW(I,J+1,K)
      RG4=RHOSCG/BG(I,J+1,K)
C
      FACT=-D288*(EL(I,J+1,K)-EL(I,J,K))
      GOW4=(RO4+RO)*FACT
      GWW4=(RW4+RW)*FACT + PCOW-PCOW4
      GGW4=(RG4+RG)*FACT + PCGO4-PCGO
C
      P44=P4-PP
      HO4=P44+GOW4
      HW4=P44+GWW4
      HG4=P44+GGW4
      IF (HO4.GE.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SO4,KRO4)
      IF (HO4.LT.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO4)
      IF (HW4.GE.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SW4,KRW4)
      IF (HW4.LT.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW4)
      IF (HG4.GE.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SG4,KRG4)
      IF (HG4.LT.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG4)
      MO4=4.0*KRO4/((BO(I,J+1,K)+BO(I,J,K)) * (MUO4+MUO))
      MW4=4.0*KRW4/((BW(I,J+1,K)+BW(I,J,K)) * (MUW4+MUW))
      MG4=4.0*KRG4/((BG(I,J+1,K)+BG(I,J,K)) * (MUG4+MUG))
C
  140 AON=TY(I,J+1,K)*MO4
      AWN=TY(I,J+1,K)*MW4
      AGN=TY(I,J+1,K)*MG4
C
C
      IF (K.EQ.1) GO TO 145
      P5=P(I,J,K-1)
      BPT=PBOT(I,J,K-1)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,P5,RSO5)
      CALL INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,P5,MUO5)
      CALL INTERP(NTE,PWT,RSWT,MPWT,P5,RSW5)
      CALL INTERP(NTE,PWT,MUWT,MPWT,P5,MUW5)
      CALL INTERP(NTE,PGT,MUGT,MPGT,P5,MUG5)
      SO5=SO(I,J,K-1)
      SW5=SW(I,J,K-1)
      SG5=SG(I,J,K-1)
      CALL INTERP(NTE,SAT,PCOWT,MSAT,SW5,PCOW5)
      CALL INTERP(NTE,SAT,PCGOT,MSAT,SG5,PCGO5)
      RO5=(RHOSCO + RSO5*RHOSCG)/BO(I,J,K-1)
      RW5=(RHOSCW + RSW5*RHOSCG)/BW(I,J,K-1)
      RG5=RHOSCG/BG(I,J,K-1)
C
```

```
      FACT=-D288*(EL(I,J,K-1)-EL(I,J,K))
      GOW5=(RO5+RO)*FACT
      GWW5=(RW5+RW)*FACT + PCOW-PCOW5
      GGW5=(RG5+RG)*FACT + PCG05-PCGO
      P55=P5-PP
      HO5=P55+GOW5
      HW5=P55+GWW5
      HG5=P55+GGW5
      IF (HO5.GE.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SO5,KRO5)
      IF (HO5.LT.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO5)
      IF (HW5.GE.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SW5,KRW5)
      IF (HW5.LT.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW5)
      IF (HG5.GE.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SG5,KRG5)
      IF (HG5.LT.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG5)
      MO5=4.0*KRO5/((BO(I,J,K-1)+BO(I,J,K)) * (MUO5+MUO))
      MW5=4.0*KRW5/((BW(I,J,K-1)+BW(I,J,K)) * (MUW5+MUW))
      MG5=4.0*KRG5/((BG(I,J,K-1)+BG(I,J,K)) * (MUG5+MUG))
C
C
  145 AOT=TZ(I,J,K)*MO5
      AWT=TZ(I,J,K)*MW5
      AGT=TZ(I,J,K)*MG5
C
      IF (K.EQ.KK) GO TO 150
      P6=P(I,J,K+1)
      BPT=PBOT(I,J,K+1)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,P6,RSO6)
      CALL INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,P6,MUO6)
      CALL INTERP(NTE,PWT,RSWT,MPWT,P6,RSW6)
      CALL INTERP(NTE,PWT,MUWT,MPWT,P6,MUW6)
      CALL INTERP(NTE,PGT,MUGT,MPGT,P6,MUG6)
      SO6=SO(I,J,K+1)
      SW6=SW(I,J,K+1)
      SG6=SG(I,J,K+1)
C
      CALL INTERP(NTE,SAT,PCOWT,MSAT,SW6,PCOW6)
      CALL INTERP(NTE,SAT,PCGOT,MSAT,SG6,PCGO6)
      RO6=(RHOSCO + RSO6*RHOSCG)/BO(I,J,K+1)
      RW6=(RHOSCW + RSW6*RHOSCG)/BW(I,J,K+1)
      RG6=RHOSCG/BG(I,J,K+1)
C
      FACT=-D288*(EL(I,J,K+1)-EL(I,J,K))
      GOW6=(RO6+RO)*FACT
      GWW6=(RW6+RW)*FACT + PCOW-PCOW6
      GGW6=(RG6+RG)*FACT + PCG06-PCGO
C
      P66=P6-PP
      HO6=P66+GOW6
      HW6=P66+GWW6
      HG6=P66+GGW6
      IF (HO6.GE.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SO6,KRO6)
      IF (HO6.LT.0.) CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO6)
      IF (HW6.GE.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SW6,KRW6)
      IF (HW6.LT.0.) CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW6)
      IF (HG6.GE.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SG6,KRG6)
      IF (HG6.LT.0.) CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG6)
      MO6=4.0*KRO6/((BO(I,J,K+1)+BO(I,J,K)) * (MUO6+MUO))
      MW6=4.0*KRW6/((BW(I,J,K+1)+BW(I,J,K)) * (MUW6+MUW))
      MG6=4.0*KRG6/((BG(I,J,K+1)+BG(I,J,K)) * (MUG6+MUG))
C
  150 AOB=TZ(I,J,K+1)*MO6
      AWB=TZ(I,J,K+1)*MW6
      AGB=TZ(I,J,K+1)*MG6
C
C
      RSO1A=0.5*(RSO1+RSO)
      RSO2A=0.5*(RSO2+RSO)
      RSO3A=0.5*(RSO3+RSO)
      RSO4A=0.5*(RSO4+RSO)
      RSO5A=0.5*(RSO5+RSO)
      RSO6A=0.5*(RSO6+RSO)
      RSW1A=0.5*(RSW1+RSW)
      RSW2A=0.5*(RSW2+RSW)
      RSW3A=0.5*(RSW3+RSW)
      RSW4A=0.5*(RSW4+RSW)
      RSW5A=0.5*(RSW5+RSW)
      RSW6A=0.5*(RSW6+RSW)
      AO1=AOW*GOW1
      AO2=AOE*GOW2
      AO3=AOS*GOW3
      AO4=AON*GOW4
      AO5=AOT*GOW5
      AO6=AOB*GOW6
      AW1=AWW*GWW1
      AW2=AWE*GWW2
      AW3=AWS*GWW3
      AW4=AWN*GWW4
      AW5=AWT*GWW5
      AW6=AWB*GWW6
```

```
C
      GOWT(I,J,K)= AO1 + AO2 + AO3 + AO4 + AO5 + AO6
      GWWT(I,J,K)= AW1 + AW2 + AW3 + AW4 + AW5 + AW6
      GGWT(I,J,K)=AGW*GGW1+AOE*GGW2+AGS*GGW3+AGN*GGW4+AGT*GGW5+
     $AGB*GGW6+RSO1A*AO1+RSO2A*AO2+RSO3A*AO3+RSO4A*AO4+RSO5A*AO5+
     $RSO6A*AO6+RSW1A*AW1+RSW2A*AW2+RSW3A*AW3+RSW4A*AW4+
     $RSW5A*AW5+RSW6A*AW6
C
          QOWG(I,J,K)=(BO(I,J,K)-BG(I,J,K)*RSO)*(-GOWT(I,J,K)+QO(I,J,K))
     $       +(BW(I,J,K)-BG(I,J,K)*RSW)*(-GWWT(I,J,K)+QW(I,J,K))+
     $             BG(I,J,K)*(-GGWT(I,J,K)+QG(I,J,K))
C
      AW(I,J,K)=(BO(I,J,K) + 0.5*BG(I,J,K)*(RSO1-RSO)) * AOW +
     $          (BW(I,J,K) + 0.5*BG(I,J,K)*(RSW1-RSW)) * AWW +
     $           BG(I,J,K)*AGW
      AE(I,J,K)=(BO(I,J,K) + 0.5*BG(I,J,K)*(RSO2-RSO)) * AOE +
     $          (BW(I,J,K) + 0.5*BG(I,J,K)*(RSW2-RSW)) * AWE +
     $           BG(I,J,K)*AGE
      AS(I,J,K)=(BO(I,J,K) + 0.5*BG(I,J,K)*(RSO3-RSO)) * AOS +
     $          (BW(I,J,K) + 0.5*BG(I,J,K)*(RSW3-RSW)) * AWS +
     $           BG(I,J,K)*AGS
      AN(I,J,K)=(BO(I,J,K) + 0.5*BG(I,J,K)*(RSO4-RSO)) * AON +
     $          (BW(I,J,K) + 0.5*BG(I,J,K)*(RSW4-RSW)) * AWN +
     $           BG(I,J,K)*AGN
      AT(I,J,K)=(BO(I,J,K) + 0.5*BG(I,J,K)*(RSO5-RSO)) * AOT +
     $          (BW(I,J,K) + 0.5*BG(I,J,K)*(RSW5-RSW)) * AWT +
     $           BG(I,J,K)*AGT
      AB(I,J,K)=(BO(I,J,K) + 0.5*BG(I,J,K)*(RSO6-RSO)) * AOB +
     $          (BW(I,J,K) + 0.5*BG(I,J,K)*(RSW6-RSW)) * AWB +
     $           BG(I,J,K)*AGB
C
      OW(I,J,K)=AOW
      OE(I,J,K)=AOE
      OS(I,J,K)=AOS
      ON(I,J,K)=AON
      OT(I,J,K)=AOT
      OB(I,J,K)=AOB
      WW(I,J,K)=AWW
      WE(I,J,K)=AWE
      WS(I,J,K)=AWS
      WN(I,J,K)=AWN
      WT(I,J,K)=AWT
      WB(I,J,K)=AWB
C
      IF (KCOFF.EQ.0) GO TO 200
      WRITE(IOCODE,33)
      WRITE(IOCODE,2)I,J,K,MO1,MO2,MO3,MO4,MO5,MO6
      WRITE(IOCODE,2)I,J,K,MW1,MW2,MW3,MW4,MW5,MW6
      WRITE(IOCODE,2)I,J,K,MG1,MG2,MG3,MG4,MG5,MG6
      WRITE(IOCODE,2)I,J,K,AOW,AOE,AOS,AON,AOT,AOB,BO(I,J,K),RSO
      WRITE(IOCODE,2)I,J,K,AWW,AWE,AWS,AWN,AWT,AWB,BW(I,J,K),RSW
      WRITE(IOCODE,2)I,J,K,AGW,AGE,AGS,AGN,AGT,AGB,BG(I,J,K)
      WRITE(IOCODE,2)I,J,K,GOWT(I,J,K),QO(I,J,K),GWWT(I,J,K),
     $QW(I,J,K),GGWT(I,J,K),QG(I,J,K),QOWG(I,J,K)
  200 CONTINUE
C
CCC   ------------------------------------------------------------------
CCC
CCC             CALCULATE  MAIN DIAGONAL  AND RHS  VECTOR
CCC
CCC   ------------------------------------------------------------------
C
      DO 300 K=1,KK
      DO 300 J=1,JJ
      DO 300 I=1,II
      SUM(I,J,K)=AW(I,J,K)+AE(I,J,K)+AS(I,J,K)+AN(I,J,K)+
     $           AT(I,J,K)+AB(I,J,K)
      GAM(I,J,K)=VP(I,J,K)*CT(I,J,K)*DIV1
      E(I,J,K)=-SUM(I,J,K) - GAM(I,J,K)
      B(I,J,K)=QOWG(I,J,K) - GAM(I,J,K)*P(I,J,K)
  300 CONTINUE
C
      IF (KSM1.EQ.0) RETURN
      IF (N.NE.1 .AND. N.NE.NN .AND. N.NE.KSM) RETURN
      WRITE(IOCODE,4)
C
      DO 404 K=1,KK
      DO 404 J=1,JJ
      DO 404 I=1,II
      WRITE(IOCODE,2)I,J,K,AT(I,J,K),AS(I,J,K),AW(I,J,K),E(I,J,K),
     $           AE(I,J,K),AN(I,J,K),AB(I,J,K),B(I,J,K)
  404 CONTINUE
C
```

```
 4     FORMAT(//3X,'NODE        AT(I,J,K)       AS(I,J,K)       AW(I,J,K)'
     $,'       E(I,J,K)     AE(I,J,K)       AN(I,J,K)       AB(I,J,K)',
     $'       B(I,J,K)'/)
 2     FORMAT(1X,3I3,8E15.6)
 33    FORMAT(//)
C
      RETURN
      END
C
CC ================================================================================
C
      SUBROUTINE   CLSOR (II,JJ,KK,OMEGA,TOL,TOL1,MITER,DELT,DELTO,
     $             KSN,N,NITER)
C
CC ================================================================================
C
      INCLUDE   'COMMB3.FOR'
C
CC --------------------------------------------------------------------------------
CC
C         THIS PROGRAM SOLVES A LINEAR SYSTEM OF THREE DIMENSIONAL
C         FINITE-DIFFERENCE EQUATIONS
C
C --------------------------------------------------------------------------------
C
        DIMENSION  UM(NPMAX),AZL(NPMAX),BZL(NPMAX),CZL(NPMAX),
     $    DZL(NPMAX),UZL(NPMAX),BETA(NPMAX),GAMMA(NPMAX),W(NPMAX)
C
      DIV=DELT/DELTO
      NITER=0
      DMAX=1.0
      RHO1=0.0
      THETA=0.0
C
 11   CONTINUE
      TW=1.0 - OMEGA
      DMAX0=DMAX
      THETA0=THETA
      IF (NITER.GE.MITER) WRITE(IOCODE,30) NITER,TOL,DMAX
      IF (NITER.GE.MITER) RETURN
      NITER=NITER+1
      DMAX=0.0
 2    CONTINUE
C
CC --------------------------------------------------------------------------------
CC
C         WATT'S 3-D METH. OF ADDITIVE CORRECTIONS APPLIED EVERY
C         TENTH ITERATION.
CC
C --------------------------------------------------------------------------------
C
          IF (NITER.EQ.1.OR.NITER MOD 10.EQ.0) THEN
          IF (II.EQ.1) GOTO 3045
          DO 3022 I=1,II
          AZL(I)=0.0
          BZL(I)=0.0
          CZL(I)=0.0
          DZL(I)=0.0
3022 CONTINUE
C
          DO 3030 I=1,II
          DO 3030 K=1,KK
          DO 3030 J=1,JJ
          IM=I-1
          IP=I+1
          IF (I.EQ.1) IM=1
          IF (I.EQ.II) IP=II
          JM=J-1
          JP=J+1
          IF (J.EQ.1) JM=1
          IF (J.EQ.JJ) JP=JJ
          KM=K-1
          KP=K+1
          IF (K.EQ.1) KM=1
          IF (K.EQ.KK) KP=KK
          AZL(I)=AZL(I)+AW(I,J,K)
          CZL(I)=CZL(I)+AE(I,J,K)
          BZL(I)=BZL(I)+E(I,J,K)+AT(I,J,K)+AB(I,J,K)+AS(I,J,K)+
     $       AN(I,J,K)
          DZL(I)=DZL(I)-(AT(I,J,K)*P(I,J,KM)+AS(I,J,K)*P(I,JM,K)+
     $       AW(I,J,K)*P(IM,J,K)+AB(I,J,K)*P(I,J,KP)+AN(I,J,K)*
     $P(I,JP,K)+AE(I,J,K)*P(IP,J,K)+E(I,J,K)*P(I,J,K)-B(I,J,K))
 3030 CONTINUE
 3031 CONTINUE
C
```

```
            CALL LTRI(II,AZL,BZL,CZL,DZL,UZL,BETA,GAMMA,U,NPMAX)
            DO 3040 I=1,II
            DO 3040 K=1,KK
            DO 3040 J=1,JJ
            P(I,J,K)=P(I,J,K)+UZL(I)
 3040 CONTINUE
C
 3045 IF (JJ.EQ.1) GOTO 3075
            DO 3050 J=1,JJ
            AZL(J)=0.0
            BZL(J)=0.0
            CZL(J)=0.0
            DZL(J)=0.0
 3050 CONTINUE
C
            DO 3060 J=1,JJ
            DO 3060 K=1,KK
            DO 3060 I=1,II
            IM=I-1
            IP=I+1
            IF (I.EQ.1) IM=1
            IF (I.EQ.II) IP=II
            JM=J-1
            JP=J+1
            IF (J.EQ.1) JM=1
            IF (J.EQ.JJ) JP=JJ
            KM=K-1
            KP=K+1
            IF (K.EQ.1) KM=1
            IF (K.EQ.KK) KP=KK
            AZL(J)=AZL(J)+AS(I,J,K)
            CZL(J)=CZL(J)+AN(I,J,K)
            BZL(J)=BZL(J)+E(I,J,K)+AT(I,J,K)+AB(I,J,K)+AW(I,J,K)+
     $         AE(I,J,K)
            DZL(J)=DZL(J)-(AT(I,J,K)*P(I,J,KM)+AS(I,J,K)*P(I,JM,K)+
     $    AW(I,J,K)*P(IM,J,K)+AB(I,J,K)*P(I,J,KP)+AN(I,J,K)*
     $P(I,JP,K)+AE(I,J,K)*P(IP,J,K)+E(I,J,K)*P(I,J,K)-B(I,J,K))
 3060 CONTINUE
 3061 CONTINUE
C
            CALL LTRI(JJ,AZL,BZL,CZL,DZL,UZL,BETA,GAMMA,W,NPMAX)
            DO 3070 J=1,JJ
            DO 3070 K=1,KK
            DO 3070 I=1,II
            P(I,J,K)=P(I,J,K)+UZL(J)
 3070 CONTINUE
 3075 IF (KK.EQ.1) GOTO 4020
C
            DO 3080 K=1,KK
            AZL(K)=0.0
            BZL(K)=0.0
            CZL(K)=0.0
            DZL(K)=0.0
 3080 CONTINUE
C
            DO 3090 K=1,KK
            DO 3090 I=1,II
            DO 3090 J=1,JJ
            IM=I-1
            IP=I+1
            IF (I.EQ.1) IM=1
            IF (I.EQ.II) IP=II
            JM=J-1
            JP=J+1
            IF (J.EQ.1) JM=1
            IF (J.EQ.JJ) JP=JJ
            KM=K-1
            KP=K+1
            IF (K.EQ.1) KM=1
            IF (K.EQ.KK) KP=KK
            AZL(K)=AZL(K)+AT(I,J,K)
            CZL(K)=CZL(K)+AB(I,J,K)
            BZL(K)=BZL(K)+E(I,J,K)+AW(I,J,K)+AE(I,J,K)+AS(I,J,K)+
     $         AN(I,J,K)
            DZL(K)=DZL(K)-(AT(I,J,K)*P(I,J,KM)+AS(I,J,K)*P(I,JM,K)+
     $      AW(I,J,K)*P(IM,J,K)+AB(I,J,K)*P(I,J,KP)+AN(I,J,K)*
     $P(I,JP,K)+AE(I,J,K)*P(IP,J,K)+E(I,J,K)*P(I,J,K)-B(I,J,K))
 3090 CONTINUE
 3091 CONTINUE
C
            CALL LTRI(KK,AZL,BZL,CZL,DZL,UZL,BETA,GAMMA,W,NPMAX)
            DO 4010 K=1,KK
            DO 4010 I=1,II
            DO 4010 J=1,JJ
            P(I,J,K)=P(I,J,K)+UZL(K)
 4010 CONTINUE
      END IF
 4020 CONTINUE
C
```

```
      DO 20 K=1,KK
      DO 20 J=1,JJ
      DO 15 I=1,II
      UM(I)=P(I,J,K)
C
 27   AZL(I)=AW(I,J,K)
      BZL(I)=E(I,J,K)
      CZL(I)=AE(I,J,K)
      DZL(I)=B(I,J,K)
      IF (JJ.EQ.1) GO TO 14
      JM=J-1
      JP=J+1
      IF (J.EQ.1) JM=1
      IF (J.EQ.JJ) JP=JJ
      DZL(I)=DZL(I) - AS(I,J,K)*P(I,JM,K) - AN(I,J,K)*P(I,JP,K)
 14   IF (KK.EQ.1) GO TO 15
      KM=K-1
      KP=K+1
      IF (K.EQ.1) KM=1
      IF (K.EQ.KK) KP=KK
      DZL(I)=DZL(I) - AT(I,J,K)*P(I,J,KM) - AB(I,J,K)*P(I,J,KP)
 15   CONTINUE
 17   CONTINUE
C
          CALL LTRI(II,AZL,BZL,CZL,DZL,UZL,BETA,GAMMA,W,NPMAX)
       WRITE(*,3176) NITER
3176  FORMAT(' ','LTRI',I5)
C
      DO 16 I=1,II
      GSLSOR=UZL(I)
      P(I,J,K) = TW*UM(I) + OMEGA*GSLSOR
      ARG=P(I,J,K) - UM(I)
      DM=ABS(ARG)
      IF (DM.GT.DMAX) DMAX=DM
 16   CONTINUE
 20   CONTINUE
 3    IF (TOL1.EQ.0.0) GO TO 25
      THETA=DMAX/DMAX0
      DELTA=THETA-THETA0
      ARG=DELTA
      ARG=ABS(ARG)
      IF (ARG.GT.TOL1) GO TO 25
      OM=OMEGA-1.0
C
C ------------------------------------------------------------------
C
C        ADDITIVE CORRECTIONS FOR 1D PROBLEMS GIVES AN EXACT
C        SOLUTION BUT ROUND-OFF ERROR USUALLY PRODUCES A DIFFER-
C        ENCE IN PRESSURE SOLUTION AFTER AN ITERATION OF LSOR
C        THE FOLLOWING CONDITION COVERS ZERO ACCELERATION
C        PARAMETER AND EXACT REPLICATION OF SOLUTION
C
C ------------------------------------------------------------------
C
          IF (THETA.EQ.0.0 .OR. OMEGA.EQ.0.0) THEN
          RHO1=1.0
      GO TO 25
      END IF
C
      RHO1=(THETA+OM)*(THETA+OM)/(THETA*OMEGA*OMEGA)
      IF (RHO1.GE.1.0) GO TO 25
      ARG=1.0-RHO1
      OMEGA=2.0/(1.0+SQRT(ARG))
C
 25   IF (DMAX.GT.TOL) GO TO 11
      WRITE(IOCODE,40)NITER,OMEGA,DMAX,THETA,RHO1
 300  CONTINUE
 40   FORMAT(5X,'CONVERGENCE (LSOR) HAS BEEN REACHED AFTER ',I3,
     $' ITERATIONS',5X,'OMEGA = ',F6.3/
     $5X,'DMAX = ',F10.6,5X,'THETA = ',F10.6,5X,'RHO1 = ',F10.6,/)
 30   FORMAT(15X,'CONVERGENCE (LSOR) WAS NOT REACHED IN ',I5,
     $' ITERATIONS'/15X,'TOL = ',F10.7,10X,'DMAX = ',F15.7)
C
      RETURN
      END
C
C ==================================================================
C
C ------------------------------------------------------------------
C
C     THIS PROGRAM SOLVES THE TRIDIAGONAL SYSTEM
C     GENERATED BY THE SYSTEM OF EQUATIONS :
C
C     A(I)*X(I-1) + BI(I)*X(I) + C(I)*X(I+1) = D(I)
C
      DIMENSION  A(NPMAX),BI(NPMAX),C(NPMAX),D(NPMAX),
     $           X(NPMAX),BETA(NPMAX),GAMMA(NPMAX),W(NPMAX)
C
      BETA(1)=BI(1)
      GAMMA(1)=D(1)/BI(1)
      NM=N-1
C
```

```
C  ------------------------------------------------------------------
CC
CCC          COMPUTE FORWARD SOLUTION
CCC
CC  ------------------------------------------------------------------
C
      DO 10 I=1,NM
      W(I)=C(I)/BETA(I)
      IP=I+1
 10   BETA(IP)=BI(IP)-A(IP)*W(I)
      DO 20 I=2,N
      IM=I-1
 20   GAMMA(I)=(D(I)-A(I)*GAMMA(IM))/BETA(I)
C  ------------------------------------------------------------------
CC
CCC          COMPUTE BACK SOLUTION
CCC
CC  ------------------------------------------------------------------
C
      X(N)=GAMMA(N)
      DO 30 J=1,NM
      I=N-J
      IP=I+1
        X(I)=GAMMA(I)-W(I)*X(IP)
 30   CONTINUE
C
      RETURN
      END
C
CC  ==================================================================
C
      SUBROUTINE  TRAN1(II,JJ,KK,KTR)
C
CC  ==================================================================
C
      INCLUDE  'COMMB3.FOR'
C
CC  ------------------------------------------------------------------
CC
C          THE FOLLOWING TRANSMISSIBILITIES ARE DIFFERENTIALLY EXACT
C          FOR UNIFORM GRID BLOCKS IN EACH DIRECTION, BUT GEOMETRIC
C          FACTORS FOR SECOND DIFFERENTIALS ARE AVERAGED FOR NON-
C          UNIFORM CASES -  A NECESSARY APPROXIMATION
CC
CC  ------------------------------------------------------------------
C
      DO 30 K=1,KK
      DO 30 J=1,JJ
      DO 30 I=1,II
      FACX=1.0
      FACY=1.0
      FACZ=1.0
      IF (I.GT.1 .AND. I.LT.II)
     $    FACX=4.*DX(I,J,K)/(2.*DX(I,J,K)+DX(I+1,J,K)+DX(I-1,J,K))
      IF (J.GT.1 .AND. J.LT.JJ)
     $    FACY=4.*DY(I,J,K)/(2.*DY(I,J,K)+DY(I,J+1,K)+DY(I,J-1,K))
      IF (K.GT.1 .AND. K.LT.KK)
     $    FACZ=4.*DZ(I,J,K)/(2.*DZ(I,J,K)+DZ(I,J,K+1)+DZ(I,J,K-1))
C
      A1(I,J,K)=FACX*KX(I,J,K)*DY(I,J,K)*DZ(I,J,K)
      A2(I,J,K)=FACY*KY(I,J,K)*DX(I,J,K)*DZ(I,J,K)
      A3(I,J,K)=FACZ*KZ(I,J,K)*DX(I,J,K)*DY(I,J,K)
 30   CONTINUE
C
      IF (II.EQ.1) GO TO 501
      DO 50 K=1,KK
      DO 50 J=1,JJ
      DO 50 I=2,II
      IF (KX(I-1,J,K).LE.0.0001 .AND. KX(I,J,K).LE.0.0001) GO TO 50
      TX(I,J,K)=.012656*A1(I-1,J,K)*A1(I,J,K)/
     $          (DX(I-1,J,K)*A1(I,J,K)+DX(I,J,K)*A1(I-1,J,K))
 50   CONTINUE
 501  CONTINUE
C
      IF (JJ.EQ.1) GO TO 551
      DO 55 K=1,KK
      DO 55 J=2,JJ
      DO 55 I=1,II
      IF (KY(I,J-1,K).LE.0.0001 .AND. KY(I,J,K).LE.0.0001) GO TO 55
        TY(I,J,K)=.012656*A2(I,J-1,K)*A2(I,J,K)/
     $            (DY(I,J-1,K)*A2(I,J,K)+DY(I,J,K)*A2(I,J-1,K))
 55   CONTINUE
 551  CONTINUE
```

```
C
      IF (KK.EQ.1) GO TO 61
      DO 60 K=2,KK
      DO 60 J=1,JJ
      DO 60 I=1,II
      IF (KZ(I,J,K-1).LE.0.0001 .AND. KZ(I,J,K).LE.0.0001) GO TO 60
        TZ(I,J,K)=.012656*A3(I,J,K-1)*A3(I,J,K)/
     $          (DZ(I,J,K-1)*A3(I,J,K)+DZ(I,J,K)*A3(I,J,K-1))
 60   CONTINUE
 61   CONTINUE
C
CC ---------------------------------------------------------------
CC
CC         TRANSMISSIBILITY MODIFICATIONS
CC
CC ---------------------------------------------------------------
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
 69   FORMAT(80A1)
      READ(20,*) NUMTX,NUMTY,NUMTZ,ITCODE
      IF (NUMTX.EQ.0) GO TO 100
      WRITE(IOCODE,31)
C
 31   FORMAT(//15X,'**********TRANSMISSIBILITY (TX) NODE MODIFICATION',
     $ '**********',//15X,'          I     J     K        NEW TX VALUE')
C
      DO 275 L=1,NUMTX
      READ(20,*)I,J,K,TX(I,J,K)
 275  WRITE(IOCODE,32)I,J,K,TX(I,J,K)
 32   FORMAT(15X,3I5,5X,F14.4)
 100  IF(ITCODE.NE.1) GO TO 711
      WRITE(IOCODE,44)
 44   FORMAT(//15X,'*********TRANSMISSIBILITY (TX) DISTRIBUTION ',
     $ '*********',/)
C
      DO 853 K=1,KK
      WRITE(IOCODE,38) K
 38   FORMAT(/1X,'K =',I2/)
      DO 854 J=1,JJ
 854  WRITE(IOCODE,72)(TX(I,J,K),I=1,II)
 72   FORMAT(1X,20F6.1)
 853  CONTINUE
 711  CONTINUE
C
      IF (NUMTY.EQ.0) GO TO 110
      WRITE(IOCODE,34)
 34   FORMAT(//15X,'*********TRANSMISSIBLITY (TY) NODE MODIFICATION',
     $ '*********',//15X,'          I     J     K        NEW TY VALUE')
C
      DO 276 L=1,NUMTY
      READ(20,*)I,J,K,TY(I,J,K)
 276  WRITE(IOCODE,32)I,J,K,TY(I,J,K)
 110  IF (ITCODE.NE.1) GO TO 713
      WRITE(IOCODE,47)
 47   FORMAT(//15X,'*********TRANSMISSIBILITY (TY) DISTRIBUTION',
     $ '*********')
C
      DO 855 K=1,KK
      WRITE(IOCODE,38) K
      DO 856 J=1,JJ
 856  WRITE(IOCODE,72)(TY(I,J,K),I=1,II)
 855  CONTINUE
 713  CONTINUE
C
      IF (NUMTZ.EQ.0) GO TO 120
      WRITE(IOCODE,37)
 37   FORMAT(//15X,'*********TRANSMISSIBILITY (TZ) NODE MODIFICATION',
     $ '*********',//15X,'          I     J     K            NEW TZ VALUE')
C
      DO 277 L=1,NUMTZ
      READ(20,*)I,J,K,TZ(I,J,K)
 277  WRITE(IOCODE,32)I,J,K,TZ(I,J,K)
 120  IF (ITCODE.NE.1) GO TO 717
      WRITE(IOCODE,48)
 48   FORMAT(//15X,'*********TRANSMISSIBILITY (TZ) DISTRIBUTION',
     $ '*********',/)
C
      DO 857 K=1,KK
      WRITE(IOCODE,38) K
      DO 858 J=1,JJ
 858  WRITE(IOCODE,72)(TZ(I,J,K),I=1,II)
 857  CONTINUE
 717  CONTINUE
C
 65   IF (KTR.EQ.0) RETURN
C
      WRITE(IOCODE,43)
      DO 42 K=1,KK
      DO 42 J=1,JJ
      DO 42 I=1,II
      WRITE(IOCODE,41)I,J,K,A1(I,J,K),A2(I,J,K),A3(I,J,K),
     $ TX(I,J,K),TY(I,J,K),TZ(I,J,K)
 41   FORMAT(1X,3I5,6F15.3)
 43   FORMAT(//1X,'     I     J     K          A1                A2          ',
     $ '          A3                TX                TY                TZ'/)
 42   CONTINUE
C
      RETURN
      END
C
```

```
C  ==========================================================================
C
      SUBROUTINE  QRATE(NVQN)
C
C  ==========================================================================
C
      INCLUDE  'COMMB3.FOR'
C
      DO 105 J=1,NVQN
      IQ1=IQN1(J)
      IQ2=IQN2(J)
      IQ3=IQN3(J)
      LAY=IQ3+(LAYER(J)-1)
C
      DO 1170 K=IQ3,LAY
      PWFC(J,K)=-1.0
      PP=P(IQ1,IQ2,K)
      BPT=PBOT(IQ1,IQ2,K)
      CALL INTPVT(NTE,BPT,VSLOPE,POT,MUOT,MPOT,PP,MUO)
      CALL INTERP(NTE,PWT,MUWT,MPWT,PP,MUW)
      CALL INTERP(NTE,PGT,MUGT,MPGT,PP,MUG)
      SSO=SO(IQ1,IQ2,K)
      SSW=SW(IQ1,IQ2,K)
      SSG=SG(IQ1,IQ2,K)
      CALL INTERP(NTE,SAT,KRWT,MSAT,SSW,KRW)
      CALL INTERP(NTE,SAT,KROT,MSAT,SSO,KRO)
      CALL INTERP(NTE,SAT,KRGT,MSAT,SSG,KRG)
      GMW(J,K)=KRW/MUW
      GMO(J,K)=KRO/MUO
      GMG(J,K)=KRG/MUG
 1170 CONTINUE
C
      IF (KIP(J).LT.0) GO TO 105
      IF (KIP(J).NE.1) GO TO 1191
      ITERQ=0
      QDENOM=0.0
      ALPHAO=0.0
      ALPHAW=0.0
      ALPHAG=0.0
      LAY=IQ3+(LAYER(J)-1)
 1172 ITERQ=ITERQ+1
C
      DO 1180 K=IQ3,LAY
      PP=P(IQ1,IQ2,K)
      BPT=PBOT(IQ1,IQ2,K)
      CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PP,BBO)
      CALL INTERP(NTE,PWT,BWT,MPWT,PP,BBW)
      CALL INTERP(NTE,PGT,BGT,MPGT,PP,BBG)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PP,RSO)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
      IF (ITERQ.NE.1) GO TO 1174
      QDENOM=QDENOM+PID(J,K)*GMO(J,K)/BBO
      GMT=GMO(J,K)+GMW(J,K)+GMG(J,K)
      ALPHAO=GMO(J,K)/GMT+ALPHAO
      ALPHAW=GMW(J,K)/GMT+ALPHAW
      ALPHAG=GMG(J,K)/GMT+ALPHAG
      GO TO 1180
C
 1174 IF (QVT(J).EQ.0.0) GO TO 1176
      TOTOR=QVT(J)*ALPHAO/(ALPHAO+ALPHAW+ALPHAG)
      GO TO 1178
 1176 TOTOR=QVO(J)
 1178 QO(IQ1,IQ2,K)=TOTOR*5.615*PID(J,K)*GMO(J,K)
     $/(BBO*QDENOM)
      QW(IQ1,IQ2,K)=QO(IQ1,IQ2,K)*GMW(J,K)*BBO
     $ /(BBW*GMO(J,K))
      QG(IQ1,IQ2,K)=QO(IQ1,IQ2,K)*(GMG(J,K)*BBO
     $ /(BBG*GMO(J,K))+RSO)+RSW*QW(IQ1,IQ2,K)
 1180 CONTINUE
C
      IF (ITERQ.EQ.1) GO TO 1172
      GO TO 105
 1190 CONTINUE
 1191 CONTINUE
C
      LAY=IQ3+(LAYER(J)-1)
      ITERQ=0
      QDENOM=0.
 1192 ITERQ=ITERQ+1
      DO 1200 K=IQ3,LAY
      IF (ITERQ.NE.1) GO TO 1194
      QDENOM=QDENOM+PID(J,K)*(GMO(J,K)+GMW(J,K)+GMG(J,K))
      GO TO 1200
C
 1194 IF (KIP(J).NE.2) GO TO 1196
      QW(IQ1,IQ2,K)=QVW(J)*5.615*PID(J,K)
     $ *(GMO(J,K)+GMW(J,K)+GMG(J,K))/QDENOM
      GO TO 1200
 1196 QG(IQ1,IQ2,K)=QVG(J)*1000.*PID(J,K)
     $ *(GMO(J,K)+GMW(J,K)+GMG(J,K))/QDENOM
 1200 CONTINUE
C
      IF (ITERQ.EQ.1) GO TO 1192
 105  CONTINUE
      DO 1210 J=1,NVQN
      IQ1=IQN1(J)
      IQ2=IQN2(J)
      IQ3=IQN3(J)
      IF (KIP(J).LT.0) GO TO 1210
      LAY=IQ3+(LAYER(J)-1)
C
      DO 1205 K=IQ3,LAY
      PWFC(J,K)=0.0
      IF (PID(J,K).LE.0.0001) GO TO 1205
      PP=P(IQ1,IQ2,K)
      BPT=PBOT(IQ1,IQ2,K)
      CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PP,BBO)
      CALL INTERP(NTE,PWT,BWT,MPWT,PP,BBW)
      CALL INTERP(NTE,PGT,BGT,MPGT,PP,BBG)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PP,RSO)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
      FAC=PID(J,K)*5.615
```

```
C
      IF (KIP(J).EQ.2 .OR. KIP(J).EQ.3) THEN
        IF (KIP(J).EQ.2) THEN
        GMTB=(GMO(J,K)+GMW(J,K)+GMG(J,K))/BBW
        PWFC(J,K)=PP-QW(IQ1,IQ2,K)/(FAC*GMTB)
        ELSE
      GMTB=(GMO(J,K)+GMW(J,K)+GMG(J,K))/BBG
        PWFC(J,K)=PP-QG(IQ1,IQ2,K)/(FAC*GMTB)
      END IF
        ELSE
        GMTB=GMO(J,K)/BBO+GMW(J,K)/BBW+GMG(J,K)/BBG
      SOLN=RSO*QO(IQ1,IQ2,K)+RSW*QW(IQ1,IQ2,K)
      QT=QO(IQ1,IQ2,K)+QW(IQ1,IQ2,K)+QG(IQ1,IQ2,K)
      PWFC(J,K)=PP-(QT-SOLN)/(FAC*GMTB)
      END IF
 1205 CONTINUE
 1210 CONTINUE
C
CCC  ------------------------------------------------------------------
CCC
CCC          PRESSURE CONSTRAINT
CCC
CCC  ------------------------------------------------------------------
C
      DO 1341 J=1,NVQN
      IF (KIP(J).GE.0) GO TO 1341
      IQ1=IQN1(J)
      IQ2=IQN2(J)
      IQ3=IQN3(J)
      LAY=IQ3+(LAYER(J)-1)
      DO 1340 K=IQ3,LAY
      PPN=PN(IQ1,IQ2,K)
      BPT=PBOT(IQ1,IQ2,K)
      CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PPN,BBO)
      CALL INTERP(NTE,PWT,BWT,MPWT,PPN,BBW)
      CALL INTERP(NTE,PGT,BGT,MPGT,PPN,BBG)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PPN,RSO)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PPN,RSW)
C
CCC  ------------------------------------------------------------------
CCC
CCC          PRODUCING WELL
CCC
CCC  ------------------------------------------------------------------
C
      IF (KIP(J).NE.-1) GO TO 1310
      QO(IQ1,IQ2,K)=PID(J,K)*5.615*GMO(J,K)
     $ *(PPN-PWF(J,K))/BBO
      IF (PPN.LE.PWF(J,K)) QO(IQ1,IQ2,K)=0.0
        QW(IQ1,IQ2,K)=PID(J,K)*5.615*GMW(J,K)
     $*(PPN-PWF(J,K))/BBW
        IF (PPN.LE.PWF(J,K)) QW(IQ1,IQ2,K)=0.0
        IF (QO(IQ1,IQ2,K).LE.0.0) GOTO 1305
        QG(IQ1,IQ2,K)=QO(IQ1,IQ2,K)*(GMG(J,K)*BBO
     $/(BBG*GMO(J,K))+RSO)+RSW*QW(IQ1,IQ2,K)
      GO TO 1307

 1305 QG(IQ1,IQ2,K)=PID(J,K)*5.615*GMG(J,K)
     $*(PPN-PWF(J,K))/BBG+RSW*QW(IQ1,IQ2,K)
 1307 CONTINUE
      QG(IQ1,IQ2,K)=QO(IQ1,IQ2,K)*(GMG(J,K)*BBO
     $ /(BBG*GMO(J,K))+RSO)+RSW*QW(IQ1,IQ2,K)
      GO TO 1340
C
CCC  ------------------------------------------------------------------
CCC
CCC          WATER INJECTOR
CCC
CCC  ------------------------------------------------------------------
C
 1310 IF (KIP(J).NE.-2) GO TO 1320
      QW(IQ1,IQ2,K)=PID(J,K)*5.615*(GMO(J,K)
     $ +GMW(J,K)+GMG(J,K))*(PPN-PWF(J,K))/BBW
      IF (PPN.GE.PWF(J,K)) QW(IQ1,IQ2,K)=0.0
      GO TO 1340
C
CCC  ------------------------------------------------------------------
CCC
CCC          GAS INJECTOR
C
CC   ------------------------------------------------------------------
```

```
 1320 IF (KIP(J).NE.-3) GO TO 1340
      QG(IQ1,IQ2,K)=PID(J,K)*5.615*(GMO(J,K)
     * +GMW(J,K)+GMG(J,K))*(PPN-PWF(J,K))/BBG
      IF (PPN.GE.PWF(J,K)) QG(IQ1,IQ2,K)=0.0
 1340 CONTINUE
 1341 CONTINUE
C
      RETURN
      END
C
CC
C ==========================================================================
C
      SUBROUTINE  PRATEI(NVQN)
C
C ==========================================================================
C
      INCLUDE  'COMMB3.FOR'
C
      DO 205 J=1,NVQN
      IF (KIP(J).GE.-10) GO TO 205
      IQ1=IQN1(J)
      IQ2=IQN2(J)
      IQ3=IQN3(J)
      LAY=IQ3+(LAYER(J)-1)
C
      DO 203 K=IQ3,LAY
      P56=PID(J,K)*5.615
      PPN=PN(IQ1,IQ2,K)
      BPT=PBOT(IQ1,IQ2,K)
      CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PPN,BBO)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PPN,RSO)
      CALL INTERP(NTE,PWT,BWT,MPWT,PPN,BBW)
      CALL INTERP(NTE,PGT,BGT,MPGT,PPN,BBG)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PPN,RSW)
C
      CPIO=GMO(J,K)*P56*(BBO-BBG*RSO)/BBO
      CPIW=GMW(J,K)*P56*(BBW-BBG*RSW)/BBW
      CPIG=GMG(J,K)*P56
      CPI=CPIO+CPIW+CPIG
      B(IQ1,IQ2,K)=B(IQ1,IQ2,K)-CPI*PWF(J,K)
      E(IQ1,IQ2,K)=E(IQ1,IQ2,K)-CPI
  203 CONTINUE
  205 CONTINUE
C
      RETURN
      END
C
CC
C ==========================================================================
C
      SUBROUTINE  PRATEO(NVQN)
C
C ==========================================================================
C
      INCLUDE  'COMMB3.FOR'
C
      DO 2059 J=1,NVQN
      IF (KIP(J).GE.-10) GO TO 2059
      IQ1=IQN1(J)
      IQ2=IQN2(J)
      IQ3=IQN3(J)
      LAY=IQ3+(LAYER(J)-1)
C
      DO 2057 K=IQ3,LAY
      PP=P(IQ1,IQ2,K)
      BPT=PBOT(IQ1,IQ2,K)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PPN,RSON)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PP,RSO)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PPN,RSWN)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PP,RSW)
C
CCCCCC ---------------------------------------------------------------------
C
C            AVERAGE RATES OF INJECTION AND PRODUCTION
C
C ---------------------------------------------------------------------
```

```
      RSOAV=0.5*(RSO+RSON)
      RSWAV=0.5*(RSW+RSWN)
      FACTOR=PID(J,K)*5.615*(PP-PWF(J,K))
      IF (KIP(J).EQ.-13) GO TO 2053
      QW(IQ1,IQ2,K)=GMW(J,K)/BW(IQ1,IQ2,K)*FACTOR
      IF (KIP(J).EQ.-12)QW(IQ1,IQ2,K)=
     $ (GMO(J,K)+GMW(J,K)+GMG(J,K))/BW(IQ1,IQ2,K)*FACTOR
      IF (KIP(J).EQ.-12) GO TO 2057
      QO(IQ1,IQ2,K)=GMO(J,K)/BO(IQ1,IQ2,K)*FACTOR
      QG(IQ1,IQ2,K)=GMG(J,K)/BG(IQ1,IQ2,K)*FACTOR
     $ +RSOAV*QO(IQ1,IQ2,K)+RSWAV*QW(IQ1,IQ2,K)
      GO TO 2057
 2053 QG(IQ1,IQ2,K)=(GMO(J,K)+GMW(J,K)+GMG(J,K))
     $ /BG(IQ1,IQ2,K)*FACTOR
 2057 CONTINUE
 2059 CONTINUE
C
      RETURN
      END
CCCCC
CCCCC ==========================================================
CCCCC
      SUBROUTINE  PARM1(II,JJ,KK)
CCCCC ==========================================================
CCCCC
      INCLUDE  'COMMB3.FOR'
C
C          DIMENSION  RPHL(NZ),RKXL(NZ),RKYL(NZ),RKZL(NZ)
      REAL KXC,KYC,KZC
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
CCCCC ----------------------------------------------------------
CCCCC
CCCCC     READ INPUT CODES FOR PHI,KX,KY,KZ
CCCCC
CCCCC ----------------------------------------------------------
CCCCC
      READ(20,*) KPH,KKX,KKY,KKZ
CCCCC ----------------------------------------------------------
CCCCC
CCCCC     ESTABLISH POROSITY (PHI) DISTRIBUTION
CCCCC
CCCCC ----------------------------------------------------------
C
      IF (KPH.GE.0) GO TO 135
      READ(20,*) PHIC
      DO 140 K=1,KK
      DO 140 J=1,JJ
      DO 140 I=1,II
  140 VP(I,J,K)=PHIC
      WRITE(IOCODE,56)
      WRITE(IOCODE,26) PHIC
      GO TO 165
C
  135 IF (KPH.GT.0) GO TO 145
      READ(20,*)(RPHL(K),K=1,KK)
      DO 550 K=1,KK
      DO 550 J=1,JJ
      DO 550 I=1,II
  550 VP(I,J,K)=RPHL(K)
      DO 560 K=1,KK
      WRITE(IOCODE,510) K,RPHL(K)
  560 CONTINUE
      GO TO 165
C
  145 WRITE(IOCODE,39)
      DO 160 K=1,KK
      WRITE(IOCODE,38)K
      DO 155 J=1,JJ
      READ(20,*) (VP(I,J,K),I=1,II)
  155 WRITE(IOCODE,73)(VP(I,J,K),I=1,II)
  160 CONTINUE
  165 CONTINUE
      WRITE(IOCODE,56)
C
```

```
CCCCCC ------------------------------------------------------------------
CCCCCC
CCCCCC          ESTABLISH PERMEABILITY (KX) DISTRIBUTION
CCCCCC
CCCCCC ------------------------------------------------------------------
C
      IF (KKX.GE.0) GO TO 180
      READ(20,*) KXC
      DO 175 K=1,KK
      DO 175 J=1,JJ
      DO 175 I=1,II
 175  KX(I,J,K)=KXC
      WRITE(IOCODE,56)
      WRITE(IOCODE,29)KXC
      GO TO 195
C
 180  IF (KKX.GT.0) GO TO 185
      READ(20,*)(RKXL(K),K=1,KK)
      DO 187 K=1,KK
      DO 187 J=1,JJ
      DO 187 I=1,II
 187  KX(I,J,K)=RKXL(K)
      DO 182 K=1,KK
      WRITE(IOCODE,511)  K,RKXL(K)
 182  CONTINUE
      GO TO 195
C
 185  WRITE(IOCODE,43)
      DO 192 K=1,KK
      WRITE(IOCODE,38) K
      DO 190 J=1,JJ
      READ(20,*)(KX(I,J,K),I=1,II)
      WRITE(IOCODE,72)(KX(I,J,K),I=1,II)
 190  CONTINUE
 192  CONTINUE
 195  CONTINUE
      WRITE(IOCODE,56)
C
CCCCCC ------------------------------------------------------------------
CCCCCC
CCCCCC          ESTABLISH PERMEABILITY (KY) DISTRIBUTION
CCCCCC
CCCCCC ------------------------------------------------------------------
C
      IF (KKY.GE.0) GO TO 200
      READ(20,*) KYC
      DO 202 K=1,KK
      DO 202 J=1,JJ
      DO 202 I=1,II
 202  KY(I,J,K)=KYC
      WRITE(IOCODE,56)
      WRITE(IOCODE,33) KYC
      GO TO 220
C
 200  IF (KKY.GT.0) GO TO 207
      READ(20,*)(RKYL(K),K=1,KK)
      DO 205 K=1,KK
      DO 205 J=1,JJ
      DO 205 I=1,II
 205  KY(I,J,K)=RKYL(K)
      DO 210 K=1,KK
      WRITE(IOCODE,512) K,RKYL(K)
 210  CONTINUE
      GO TO 220
C
 207  WRITE(IOCODE,47)
      DO 212 K=1,KK
      WRITE(IOCODE,38)K
      DO 215 J=1,JJ
      READ(20,*)(KY(I,J,K),I=1,II)
      WRITE(IOCODE,72)(KY(I,J,K),I=1,II)
 215  CONTINUE
 212  CONTINUE
 220  CONTINUE
      WRITE(IOCODE,56)
C
CCCCCC ------------------------------------------------------------------
CCCCCC
CCCCCC          ESTABLISH PERMEABILITY (KZ) DISTRIBUTION
CCCCCC
CCCCCC ------------------------------------------------------------------
C
```

```
      IF (KKZ.GE.0) GO TO 225
      READ(20,*) KZC
      DO 230 K=1,KK
      DO 230 J=1,JJ
      DO 230 I=1,II
  230 KZ(I,J,K)=KZC
      WRITE(IOCODE,56)
      WRITE(IOCODE,36) KZC
      GO TO 245
C
  225 IF (KKZ.GT.0) GO TO 232
      READ(20,*)(RKZL(K),K=1,KK)
      DO 235 K=1,KK
      DO 235 J=1,JJ
      DO 235 I=1,II
  235 KZ(I,J,K)=RKZL(K)
      DO 237 K=1,KK
      WRITE(IOCODE,513) K,RKZL(K)
  237 CONTINUE
      GO TO 245
C
  232 WRITE(IOCODE,48)
      DO 240 K=1,KK
      WRITE(IOCODE,38)K
      DO 242 J=1,JJ
      READ(20,*)(KZ(I,J,K),I=1,II)
      WRITE(IOCODE,72)(KZ(I,J,K),I=1,II)
  242 CONTINUE
  240 CONTINUE
  245 CONTINUE
      WRITE(IOCODE,56)
CC ---------------------------------------------------------------
CC
CC        POROSITY AND PERMEABILITY MODIFICATIONS
CC
CC ---------------------------------------------------------------
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      READ(20,*) NUMP,NUMKX,NUMKY,NUMKZ,IPCODE
      IF (NUMP.EQ.0) GO TO 715
      WRITE(IOCODE,27)
      DO 274 L=1,NUMP
      READ(20,*)I,J,K,VP(I,J,K)
  274 WRITE(IOCODE,32)I,J,K,VP(I,J,K)
      IF (IPCODE.NE.1) GO TO 715
      WRITE(IOCODE,39)
      DO 851 K=1,KK
      WRITE(IOCODE,38)K
      DO 852 J=1,JJ
  852 WRITE(IOCODE,73)(VP(I,J,K),I=1,II)
  851 CONTINUE
  715 CONTINUE
      IF (NUMKX.EQ.0) GO TO 716
      WRITE(IOCODE,31)
      DO 275 L=1,NUMKX
      READ(20,*)I,J,K,KX(I,J,K)
  275 WRITE(IOCODE,32)I,J,K,KX(I,J,K)
      IF (IPCODE.NE.1) GO TO 716
      WRITE(IOCODE,43)
      DO 853 K=1,KK
      WRITE(IOCODE,38)K
      DO 854 J=1,JJ
      WRITE(IOCODE,72)(KX(I,J,K),I=1,II)
  854 CONTINUE
  853 CONTINUE
  716 CONTINUE
      IF (NUMKY.EQ.0) GO TO 717
      WRITE(IOCODE,34)
      DO 276 L=1,NUMKY
      READ(20,*)I,J,K,KY(I,J,K)
      WRITE(IOCODE,32)I,J,K,KY(I,J,K)
  276 CONTINUE
      IF (IPCODE.NE.1) GO TO 717
      WRITE(IOCODE,47)
      DO 855 K=1,KK
      WRITE(IOCODE,38)K
      DO 856 J=1,JJ
      WRITE(IOCODE,72)(KY(I,J,K),I=1,II)
  856 CONTINUE
  855 CONTINUE
  717 CONTINUE
      IF (NUMKZ.EQ.0) GO TO 719
      WRITE(IOCODE,37)
      DO 277 L=1,NUMKZ
      READ(20,*)I,J,K,KZ(I,J,K)
      WRITE(IOCODE,32)I,J,K,KZ(I,J,K)
  277 CONTINUE
      IF (IPCODE.NE.1) GO TO 719
      WRITE(IOCODE,48)
      DO 857 K=1,KK
      WRITE(IOCODE,38) K
      DO 858 J=1,JJ
      WRITE(IOCODE,72)(KZ(I,J,K),I=1,II)
  858 CONTINUE
  857 CONTINUE
  719 CONTINUE
```

```
 69    FORMAT(80A1)
 56    FORMAT(//)
 72    FORMAT(1X,20F6.2)
 73    FORMAT(1X,20F6.4)
 26    FORMAT(15X,'POROSITY (PHI) IS INITIALLY SET AT',F8.4,' FOR ALL',
     $' NODES'//)
 27       FORMAT(//15X,'****POROSITY NODE MODIFICATIONS****',
     $//15X,'      I    J    K          NEW PHI VALUE')
 29    FORMAT(15X,'PERMEABILITY (KX) IS INITIALLY',
     $' SET AT',F10.4,' FOR ALL NODES'//)
 31    FORMAT(//15X,'**********PERMEABILITY (KX) NODE MODIFICATIONS',
     $'**********',//15X,'      I    J    K              NEW KX VALUE')
 32    FORMAT(15X,3I5,5X,F14.4)
 33    FORMAT(15X,'PERMEABILITY (KY) IS INITIALLY',
     $' SET AT',F10.4,' FOR ALL NODES'//)
 34    FORMAT(//15X,'**********PERMEABILITY (KY) NODE MODIFICATIONS',
     $'**********',//15X,'      I    J    K              NEW KY VALUE')
 36    FORMAT(15X,'PERMEABILITY (KZ) IS INITIALLY',
     $' SET AT',F10.4,' FOR ALL NODES'//)
 37    FORMAT(//15X,'**********PERMEABILITY (KZ) NODE MODIFICATIONS',
     $'**********',//15X,'      I    J    K              NEW KZ VALUE')
999    FORMAT(I5)
 38    FORMAT(/1X,'K =',I2/)
 39    FORMAT(//15X,'**********POROSITY DISTRIBUTION FOLLOWS********'/)
390    FORMAT(//15X,'**********NODE MIDPOINT ELEVATIONS **********'/)
 43    FORMAT(//15X,'**********PERMEABILITY (KX) DISTRIBUTION*******'/)
445    FORMAT(13F10.2)
 47    FORMAT(//15X,'**********PERMEABILITY (KY) DISTRIBUTION*******'/)
 48    FORMAT(//15X,'**********PERMEABILITY (KZ) DISTRIBUTION*******'/)
 510   FORMAT(//15X,'POROSITY IN LAYER',I5,' IS INITIALLY SET AT',F8.5,
     $' FOR ALL NODES',/)
 511   FORMAT(15X,'PERMEABILITY (KX) IN LAYER',I5,' IS INITIALLY AT',
     $ F8.2,' FOR ALL NODES',/)
 512   FORMAT(15X,'PERMEABILITY (KY) IN LAYER',I5,' IS INITIALLY AT',
     $ F8.2,' FOR ALL NODES'/)
 513   FORMAT(15X,'PERMEABILITY (KZ) IN LAYER',I5,' IS INITIALLY AT',
     $ F8.2,' FOR ALL NODES'/)
C
       RETURN
       END
C
CC
C  ===================================================================
C
       SUBROUTINE  NODES(NVQN,WRAD,SKIN)
C
C  ===================================================================
C
       INCLUDE  'COMMB3.FOR'
C
       REAL  WRAD,SKIN
C
CCCCC  -------------------------------------------------------------
CCCCC
CCCCC       ESTABLISH RATE-SPECIFIED AND PRESSURE-SPECIFIED WELLS
CCCCC
CCCCC  -------------------------------------------------------------
C
       READ(20,69) (IHEDIN(IH),IH=1,80)
       READ(20,*)NVQN
       IF (NVQN.EQ.0)RETURN
       WRITE(IOCODE,67)
       WRITE(IOCODE,68)
C
       DO 2000 J=1,NVQN
C
CCCCC  -------------------------------------------------------------
CCCCC
CCCCC       INPUT RATE SIGNS BY CONVENTION:
CCCCC            INJECTORS ARE -VE
CCCCC            PRODUCERS ARE +VE
CCCCC
CCCCC  -------------------------------------------------------------
C
       READ(20,*)WELLID(J),IQN1(J),IQN2(J),IQN3(J),LAYER(J),KIP(J),
     $            QVO(J),QVW(J),QVG(J),QVT(J)
```

```
C
      IQ1=IQN1(J)
      IQ2=IQN2(J)
      IQ3=IQN3(J)
      LAY=IQ3+(LAYER(J)-1)
C
      DO 1900 K=IQ3,LAY
      READ(20,*) WRAD,SKIN,PWF(J,K)
C
CCCCC---------------------------------------------------------------
CCCCC
CCCCC       PID CALCULATION NOW DONE WITHIN THE PROGRAM
CCCCC
CCCCC---------------------------------------------------------------
C
      IF (SKIN .GE. 500.)THEN
      PID(J,K)=0.0
      ELSE
      DENOM=ALOG(+.121*SQRT(DX(IQ1,IQ2,K)*DY(IQ1,IQ2,K))/WRAD)+SKIN
      PID(J,K)=.00708*KX(IQ1,IQ2,K)*DZ(IQ1,IQ2,K)/DENOM
      ENDIF
C
      WRITE(IOCODE,70)IQN1(J),IQN2(J),K,QVO(J),QVW(J),QVG(J),
     $                QVT(J),PWF(J,K),PID(J,K),WRAD,SKIN
 1900 CONTINUE
 2000 CONTINUE
C
      WRITE(IOCODE,33)
      DO 2999 J=1,NVQN
      IQ3=IQN3(J)
      LAY=IQ3+(LAYER(J)-1)
      DO 2999 K=IQ3,LAY
      IF (KIP(J).EQ.1) WRITE(IOCODE,2995) IQN1(J),IQN2(J),K
      IF (KIP(J).EQ.2) WRITE(IOCODE,2996) IQN1(J),IQN2(J),K
      IF (KIP(J).EQ.3) WRITE(IOCODE,2997) IQN1(J),IQN2(J),K
      IF (KIP(J).EQ.-1)WRITE(IOCODE,3005) IQN1(J),IQN2(J),K
      IF (KIP(J).EQ.-2)WRITE(IOCODE,3006) IQN1(J),IQN2(J),K
      IF (KIP(J).EQ.-3)WRITE(IOCODE,3007) IQN1(J),IQN2(J),K
      IF(KIP(J).EQ.-11)WRITE(IOCODE,3015) IQN1(J),IQN2(J),K
      IF(KIP(J).EQ.-12)WRITE(IOCODE,3016) IQN1(J),IQN2(J),K
      IF(KIP(J).EQ.-13)WRITE(IOCODE,3017) IQN1(J),IQN2(J),K
 2999 CONTINUE
C
 69    FORMAT(80A1)
 67    FORMAT(//15X,'RESERVOIR CONTAINS THE FOLLOWING RATE NODES'/)
 68    FORMAT(2X,'    NODE    ',3X,'OIL(STB/D)',3X,
     $'WATER(STBD)',3X,'GAS(MCFD)',3X,'TOTAL(RBD)',3X,'BHFP(PSIA)'
     $7X,'PID',2X,'WELL RAD',2X,'SKIN FACTOR')
 70    FORMAT(1X,3I3,3X,F11.2,4F13.2,F10.6,F10.3,E12.4)
 33    FORMAT(/)
 2995 FORMAT(15X,'BLOCK',3I3,
     $ ' CONTAINS A RATE SPECIFIED PRODUCING WELL')
 2996 FORMAT(15X,'BLOCK',3I3,
     $ ' CONTAINS A RATE SPECIFIED WATER INJECTION WELL')
 2997 FORMAT(15X,'BLOCK',3I3,
     $ ' CONTAINS A RATE SPECIFIED GAS  INJECTION WELL')
 3005 FORMAT(15X,'BLOCK',3I3,' CONTAINS AN ',
     $ 'EXPLICIT PRESSURE SPECIFIED PRODUCING WELL')
 3006 FORMAT(15X,'BLOCK',3I3,' CONTAINS AN ',
     $ 'EXPLICIT PRESSURE SPECIFIED WATER INJECTION WELL')
 3007 FORMAT(15X,'BLOCK',3I3,' CONTAINS AN ',
     $ ' EXPLICIT PRESSURE SPECIFIED GAS INJECTION WELL')
 3015 FORMAT(15X,'BLOCK',3I3,' CONTAINS AN ',
     $ ' IMPLICIT PRESSURE SPECIFIED PRODUCING WELL')
 3016 FORMAT(15X,'BLOCK',3I3,' CONTAINS AN ',
     $ ' IMPLICIT PRESSURE SPECIFIED WATER INJECTION WELL')
 3017 FORMAT(15X,'BLOCK',3I3,' CONTAINS AN ',
     $ ' IMPLICIT PRESSURE SPECIFIED GAS INJECTION WELL')
C
      RETURN
      END
C
```

```
C  ==========================================================================
C
      SUBROUTINE  PRTPS(NLOOP,KPI,II,JJ,KK,PAVGO,PAVG,
     $    COP,CWP,CWI,CGP,CGI,MBEO,MBEW,MBEG,DELTO,OPR,
     $    WPR,GPR,WIR,GIR,ETI,CWOR,CGOR,
     $    WOR,GOR,IPMAP,ISOMAP,ISWMAP,ISGMAP,IPBMAP)
C
CC ==========================================================================
C
      INCLUDE  'COMMB3.FOR'
C
      PPM=0.
      SOM=0.
      SWM=0.
      SGM=0.
C
           IF (NLOOP.EQ.1) GOTO 300
           DO 240 K=1,KK
           DO 240 J=1,JJ
           DO 240 I=1,II
           DPO=P(I,J,K)-PN(I,J,K)
           DSO=SO(I,J,K)-SON(I,J,K)
           DSW=SW(I,J,K)-SWN(I,J,K)
           DSG=SG(I,J,K)-SGN(I,J,K)
           IF (ABS(DPO).LE.ABS(PPM)) GOTO 210
           PPM=DPO
           IPM=I
           JPM=J
           KPM=K
210   IF (ABS(DSO).LE.ABS(SOM)) GO TO 220
           SOM=DSO
           IOM=I
           JOM=J
           KOM=K
220   IF (ABS(DSW).LE.ABS(SWM)) GO TO 230
           SWM=DSW
           IWM=I
           JWM=J
           KWM=K
230   IF (ABS(DSG).LE.ABS(SGM)) GO TO 240
           SGM=DSG
           IGM=I
           JGM=J
           KGM=K
240   CONTINUE
           WRITE(IOCODE,5)
           WRITE(IOCODE,105)
           WRITE(IOCODE,6)
           NLM= NLOOP-1
           GORM=GOR
           WRITE(IOCODE,110) ETI,NLM,DELTO,PAVG,PAVGO,IPM,JPM,KPM,PPM,
     $    IOM,JOM,KOM,SOM,IGM,JGM,KGM,SGM,IWM,JWM,KWM,SWM
           WRITE(IOCODE,111) MBEO,MBEG,MBEW,OPR,COP,GPR,CGP,WPR,CWP,
     $    GIR,CGI,WIR,CWI,WOR,CWOR,GORM,CGOR
110   FORMAT(/,1X,'ELAPSED TIME (DAYS)          =',F9.2,3X,
     $    'TIME STEP NUMBER          =',I5,7X,
     $    'TIME STEP (DAYS)          =',F9.2,//,1X,
     $    'CURRENT AVG RES PRESSURE =',F9.1,3X,
     $    'PREVIOUS AVG RES PRESSURE =',F9.1,3X,
     $    'PRESSURE DPMAX(',I3,',',I3,',',I3,') =',F9.1,/,1X,
     $    'OIL DSMAX (',I3,',',I3,',',I3,')      =',F9.5,3X,
     $    'GAS DSMAX (',I3,',',I3,',',I3,')      =',F9.5,3X,
     $    'WATER DSMAX(',I3,',',I3,',',I3,')     =',F9.5,/,1X)
111   FORMAT(' OIL MATERIAL BALANCE (%)    =',F9.6,3X,
     $    'GAS MATERIAL BALANCE (%)          =',E10.4,3X,
     $    'WATER MATERIAL BALANCE (%)    =',E10.4,//,1X,
     $    'OIL PRODUCTION RATE (STB/D)   =',E10.4,3X,
     $    'CUM. OIL PRODUCTION (STB)    =',E10.4,/,1X,
     $    'GAS PRODUCTION RATE (MSCF/D)=',E10.4,3X,
     $    'CUM. GAS PRODUCTION (MSCF)   =',E10.4,/,1X,
     $    'WATER PRODUCTION RATE (STB/D)=',E10.4,3X,
     $    'CUM. WATER PRODUCTION (STB)  =',E10.4,//,1X,
     $    'GAS INJECTION RATE (MSCF/D)  =',E10.4,3X,
     $    'CUM. GAS INJECTION (MSCF)    =',E10.4,/,1X,
     $    'WATER INJECTION RATE (STB/D)=',E10.4,3X,
     $    'CUM. WATER INJECTION (STB)   =',E10.4,//,1X,
     $    'PRODUCING WOR (STB/STB)           =',E10.4,3X,
     $    'CUM. WOR (STB/STB)                =',E10.4,/,1X,
     $    'PRODUCING GOR (SCF/STB)           =',E10.4,3X,
     $    'CUM. GOR (SCF/STB)                =',E10.4,//)
```

```
  300   IF (NLOOP.EQ.1 .AND. KPI.EQ.0) WRITE(IOCODE,304)
  304   FORMAT(//15X,7('*'),'INITIAL ARRAYS ',7('*')//)
        IF (IPMAP.EQ.0 .AND. NLOOP.NE.1) GOTO 315
        WRITE(IOCODE,61)
        DO 310 K=1,KK
        WRITE(IOCODE,51) K
        DO 310 J=1,JJ
        WRITE(IOCODE,41) (P(I,J,K),I=1,II)
  310   CONTINUE
C
  315   IF (ISOMAP.EQ.0 .AND. NLOOP.NE.1) GO TO 422
        WRITE(IOCODE,71)
        DO 420 K=1,KK
        WRITE(IOCODE,51) K
        DO 420 J=1,JJ
        WRITE(IOCODE,101) (SO(I,J,K),I=1,II)
  420   CONTINUE
C
  422   IF (ISWMAP.EQ.0 .AND. NLOOP.NE.1) GO TO 432
        WRITE(IOCODE,81)
        DO 430 K=1,KK
        WRITE(IOCODE,51) K
        DO 430 J=1,JJ
        WRITE(IOCODE,101) (SW(I,J,K),I=1,II)
  430   CONTINUE
C
  432   IF (ISGMAP.EQ.0 .AND. NLOOP.NE.1) GO TO 440
        WRITE(IOCODE,91)
        DO 439 K=1,KK
        WRITE(IOCODE,51) K
        DO 439 J=1,JJ
        WRITE(IOCODE,101) (SG(I,J,K),I=1,II)
  439   CONTINUE
  440   CONTINUE
        IF (IPBMAP.EQ.0) GOTO 450
        WRITE(IOCODE,102)
  102   FORMAT(//15X,'******* BUBBLE POINT PRESSURE DISTRIBUTION ***'/)
        DO 449 K=1,KK
        WRITE(IOCODE,51) K
        DO 449 J=1,JJ
        WRITE(IOCODE,41) (PBOT(I,J,K),I=1,II)
  449   CONTINUE
  450   CONTINUE
        IF (NLOOP.NE.1) WRITE(IOCODE,7)
C
    5         FORMAT(//30X,69('*'),/,30X,'*',67X,'*',/,30X,'*',67X,'*')
    6         FORMAT(30X,'*',67X,'*',/,30X,'*',67X,'*',/,30X,69('*'),///)
    7         FORMAT(///5X,54('*'),'  END OF REPORT  ',54('*'),6(/))
  105   FORMAT(30X,'*',12X,'SUMMARY REPORT: UNSW BOAST (VERSION 2.0)',
       $      15X,'*')
   41         FORMAT(5X,20F6.0)
   51         FORMAT(/1X,'K=',I2/)
   61         FORMAT(//15X,'****RESERVOIR PRESSURE DISTRIBUTION****',/)
   71         FORMAT(//15X,'********OIL SATURATION********'/)
   81         FORMAT(//15X,'********WATER SATURATION********'/)
   91         FORMAT(//15X,'********GAS SATURATION********'/)
  101   FORMAT(5X,20F6.3)
C
        RETURN
        END
C
```

```
C  ==========================================================================
C
      SUBROUTINE  UINIT1(KPI,II,JJ,KK,PWOC,CUMPO,MBEO,
     $                   CUMPW,MBEW,CUMPG,MBEG,SOI,SWI,SGI,
     $                   WOC,GOC,PGOC,CUMIW,CUMIG)
C
C  ==========================================================================
C
      INCLUDE  'COMMB3.FOR'
C
      CUMPO=0.0
      CUMPW=0.0
      CUMPG=0.0
      CUMIW=0.0
      CUMIG=0.0
      MBEO=0.0
      MBEW=0.0
      MBEG=0.0
C
      READ(20,69) (IHEDIN(IH),IH=1,80)
      READ(20,*) KPI,KSI
      IF (KPI.NE.0) GO TO 5601
      READ(20,*) PWOC,PGOC,WOC,GOC
      DO 200 K=1,KK
      DO 200 J=1,JJ
      DO 200 I=1,II
      IF (EL(I,J,K).LT.GOC) GO TO 175
      IF (EL(I,J,K).GT.WOC) GO TO 150
      BPT=PBOT(I,J,K)
      CALL INTPVT(NTE,BPT,BSLOPE,POT,BOT,MPOT,PWOC,BBO)
      CALL INTPVT(NTE,BPT,RSLOPE,POT,RSOT,MPOT,PWOC,RSO)
      RHOO=(RHOSCO+RSO*RHOSCG)/BBO
      PN(I,J,K)=PWOC+RHOO*(EL(I,J,K)-WOC)/144.
      GO TO 200
C
  150 CALL INTERP(NTE,PWT,BWT,MPWT,PWOC,BBW)
      CALL INTERP(NTE,PWT,RSWT,MPWT,PWOC,RSW)
      RHOW=(RHOSCW+RSW*RHOSCG)/BBW
      PN(I,J,K)=PWOC+RHOW*(EL(I,J,K)-WOC)/144.
      GO TO 200
C
  175 CALL INTERP(NTE,PGT,BGT,MPGT,PGOC,BBG)
      RHOG=RHOSCG/BBG
      PN(I,J,K)=PGOC+RHOG*(EL(I,J,K)-GOC)/144.
  200 CONTINUE
      GO TO 3010
C
 5601 DO 3009 K=1,KK
      DO 3005 J=1,JJ
 3005 READ(20,*) (PN(I,J,K),I=1,II)
 3009 CONTINUE
 3010 CONTINUE
C
CC  ------------------------------------------------------------------------
C
C         INITIALIZE N+1 PRESSURE ARRAY
C
CC  ------------------------------------------------------------------------
C
      DO 3012 I=1,II
      DO 3012 J=1,JJ
      DO 3012 K=1,KK
 3012 P(I,J,K)=PN(I,J,K)
      IF (KSI.NE.0) GO TO 5600
C
      READ(20,*) SOI,SWI,SGI
      DO 30 K=1,KK
      DO 30 J=1,JJ
      DO 30 I=1,II
      SON(I,J,K)=SOI
      SWN(I,J,K)=SWI
      SGI=1.0-SOI-SWI
      SGN(I,J,K)=SGI
      SO(I,J,K)=SOI
      SW(I,J,K)=SWI
      SG(I,J,K)=SGI
      IF (SG(I,J,K).LT.0.0) SG(I,J,K)=0.0
   30 CONTINUE
C
```

```
 3015 RETURN
 5600 DO 3011 K=1,KK
      DO 3006 J=1,JJ
 3006 READ(20,*) (SO(I,J,K),I=1,II)
 3011 CONTINUE
      DO 3020 K=1,KK
      DO 3007 J=1,JJ
 3007 READ(20,*) (SW(I,J,K),I=1,II)
 3020 CONTINUE
      DO 3030 K=1,KK
      DO 3030 J=1,JJ
      DO 3030 I=1,II
      SG(I,J,K)=1.0-SO(I,J,K)-SW(I,J,K)
      IF(SG(I,J,K).LT.0.0) SG(I,J,K)=0.0
      SON(I,J,K)=SO(I,J,K)
      SWN(I,J,K)=SW(I,J,K)
      SGN(I,J,K)=SG(I,J,K)
 3030 CONTINUE
C
  69  FORMAT(80A1)
C
      RETURN
      END
C
C  ===================================================================
CC
      SUBROUTINE  MATBAL(II,JJ,KK,STBO,STBOI,STBW,STBWI,
     $     MCFG,MCFGI,MBEO,MBEW,MBEG,DELTO,RESVOL,OP,WP,GP,WI,
     $     GI,PAVGO,PAVG,N,OPR,WPR,GPR,WIR,GIR,D5615,
     $     COP,CWP,CGP,CWI,CGI,MCFG1,MCFGT,CWOR,WOR,CGOR,GOR)
C
CC  ===================================================================
CCC
C
      FACT=D5615*DELTO
      PAVGO=0.0
      PAVG=0.0
      OP=0.0
      WP=0.0
      GP=0.0
      WI=0.0
      GI=0.0
      DO 100 K=1,KK
      DO 100 J=1,JJ
      DO 100 I=1,II
      PAVGO=PAVGO+PN(I,J,K)*VP(I,J,K)
      PAVG=PAVG+P(I,J,K)*VP(I,J,K)
      OP=OP+QO(I,J,K)*FACT
      IF (QW(I,J,K).GT.0.0) WP=WP+QW(I,J,K)*FACT
      IF (QW(I,J,K).LT.0.0) WI=WI+QW(I,J,K)*FACT
      IF (QG(I,J,K).GT.0.0) GP=GP+QG(I,J,K)*DELTO
      IF (QG(I,J,K).LT.0.0) GI=GI+QG(I,J,K)*DELTO
  100 CONTINUE
      COP=COP+OP
      CWP=CWP+WP
      CGP=CGP+GP*.001
      CWI=CWI+WI
      CGI=CGI+GI*.001
C
CC  -------------------------------------------------------------------
CC
CC        CONVERT SCF TO MCF (STANDARD TO THOUSANDS)
CC
CC  -------------------------------------------------------------------
C
      GP=GP*.001
      GI=GI*.001
      DIV=1.0/DELTO
      OPR=OP*DIV
      WPR=WP*DIV
      GPR=GP*DIV
      WIR=WI*DIV
      GIR=GI*DIV
      PAVG=PAVG/RESVOL
      PAVGO=PAVGO/RESVOL
C
      DENOM1=STBOI - OP
      IF (DENOM1.LT.1.E-7) GO TO 200
      MBEO=(STBO/(STBOI-OP) - 1.0)*100.
  200 CONTINUE
      DENOM2=STBWI-WP-WI
      IF (DENOM2.LT.1.E-7) GO TO 201
      MBEW=(STBW/(STBWI-WP-WI) - 1.0)*100.
  201 CONTINUE
      DENOM3=MCFGI-GP-GI
      IF (DENOM3.LT.1.E-7) GO TO 203
      MBEG=(MCFGT/(MCFGI-GP-GI) - 1.0)*100.
  203 CONTINUE
```

```
C
      IF (COP.EQ.0.0) GOR=0.0
      IF (COP.EQ.0.0) WOR=0.0
      IF (COP.EQ.0.0) GO TO 333
        GOR=GP/OP*1000.0
      WOR=WP/OP
  333 IF (COP.EQ.0.0) CGOR=0.0
      IF (COP.EQ.0.0) CWOR=0.0
      IF (COP.EQ.0.0) GO TO 666
      CGOR=CGP/COP*1000.
      CWOR=CWP/COP
  666 CONTINUE
C
      GP=GP*0.001
      GI=GI*0.001
C
      RETURN
      END
C
CC ==========================================================================
C
      SUBROUTINE  INTERP(NTE,X,Y,N,XO,YO)
C
CC ==========================================================================
C
      DIMENSION X(NTE),Y(NTE)
      IF (XO.GE.X(N)) YO=Y(N)
      IF (XO.GE.X(N)) RETURN
      DO 10 I=2,N
      IF (XO.GE.X(I)) GO TO 10
      YO=Y(I-1) +(XO-X(I-1))*(Y(I)-Y(I-1))/(X(I)-X(I-1))
C
      RETURN
   10 CONTINUE
      END
C
CC ==========================================================================
C
      SUBROUTINE  INTPVT(NTE,BPT,RM,X,Y,N,XO,YO)
C
CC ==========================================================================
C
      DIMENSION X(NTE),Y(NTE)
      IF (XO.GT.BPT) GO TO 100
      IF (XO.GE.X(N)) YO=Y(N)
      IF (XO.GE.X(N)) RETURN
      DO 10 I=2,N
      IF (XO.GE.X(I)) GO TO 10
      YO=Y(I-1)+(XO-X(I-1))*(Y(I)-Y(I-1))/(X(I)-X(I-1))
      RETURN
   10 CONTINUE
  100 CONTINUE
      DO 200 I=2,N
      IF (BPT.GE.X(I)) GO TO 200
      YOBP=Y(I-1)+(BPT-X(I-1))*(Y(I)-Y(I-1))/(X(I)-X(I-1))
      YO=YOBP+RM*(XO-BPT)
C
      RETURN
  200 CONTINUE
      END
C
CC ==========================================================================
C
      SUBROUTINE  NTERP1(NTE,X,Y,N,XO,YO)
C
CC ==========================================================================
C
      DIMENSION X(NTE),Y(NTE)
          IF (XO.GE.X(N)) YO=Y(N)
          IF (XO.GE.X(N)) RETURN
          DO 10 I=2,N
          IF (XO.GT.X(I)) GOTO 10
          YO=Y(I)
C
          RETURN
10        CONTINUE
      END
```

REFERENCES

1. Hubbert, M.K.: 'Darcy's Law and the field equations of the flow of underground fluids', JPT(1969), pp222-239.

2. Thomas, G.W.: 'Principles of Hydrocarbon Reservoir Simulation', International Human Resources Development Corporation, Boston (1982).

3. Douglas, J.,Jr., Peaceman, D.W. and Rachford, H.H., Jr: 'A method for calculating multidimensional immiscible displacement', Trans.SPE of AIME (1959), pp297-306.

4. Coats, K.H., Nielsen, R.L., Terhune, M.H. and Weber, A.G.: 'Simulation of three-dimensional, two-phase flow in oil and gas reservoirs', Trans. SPE of AIME (1967), pp240.

5. Coats, K.H.: 'An analysis for simulating reservoir performance under pressure maintenance by gas and/or water injection', SPEJ (1968), No.4, pp331-340.

6. Coats, K.H.: 'Elements of Reservoir Simulation', Lecture Notes, University of Texas, reprinted by Intercomp Resources Development and Engineering Inc., Houston, 1968.

7. Sheffield, M.: 'Three phase-flow including gravitational viscous, and capillary forces', Trans. SPE of AIME (1969) Vol. 246, pp255-269.

8. Breitenback, E.A., Thurnau, D.H. and Van Poolen, H.K.: 'The fluid flow simulation equations', SPEJ (9), No.2 (1969), pp155-169.

9. Varga, R.S.: 'Matrix iterative analysis', Prentice-Hall, Inc., Englewood Cliffs (1962).

10. Watts, J.W.: 'A method for improving line successive overrelaxation in anisotropic problems', SPEJ (1973), pp105-108.

11. Aziz, K. and Settari, A.: 'Petroleum reservoir simulation', Applied Science Publishing Ltd., London, 1979.

12. Crichlow, H.B.: 'Modern reservoir engineering - A simulation approach', Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1977.

13. Peaceman, D.W.: 'Interpretation of well-block pressures in numerical reservoir simulation', SPEJ (1978).

14. Peaceman, D.W.: 'Interpretation of well-block pressures in numerical reservoir simulation with non-square grid blocks and anisotropic permeability', SPEJ (1982).

15. Lecture Notes of Professor W.V. Pinczewski, University of New South Wales, 1988.